Integrating Perceptions of Best Practice: Running an SME Business in the Distinctive COTS Application Software Sector

By

Diccon Vokins

Master of Business Administration, Bachelor of Science

A thesis submitted in fulfilment of the requirements for the degree of

Doctor of Business Administration (DBA)

Macquarie University

Sydney, Australia

May 2010

CERTIFICATION

This thesis is submitted in fulfillment of the requirements of the degree of DBA, in the Graduate School of Management, Macquarie University. This represents the original work and contribution of the author, except as acknowledged by general and specific references.

I hereby certify that this has not been submitted for a higher degree to any other university or institution.

Signed:

Diccon Vokins

Date: 25/05/10

ACKNOWLEDGEMENTS

Two groups of people contributed to the completion of this thesis. On the academic side, my thanks to both my supervisors. Richard Dunford supported and motivated me at every step of the five and a half year journey. Ernie Jordan rigorously challenged the academic presentation and content of my final written thesis. On the personal side, my thanks to my family: Juwanti, Alex, Topan and Bimo. The extended duration of a part-time doctorate while working full-time and raising a young family was a significant challenge. This resulted in a huge sacrifice for my wife Juwanti. She supported and loved me throughout in her own beautiful way. Topan, while you were having your sleeps, I snuck away to spend an hour here and an hour there progressing my DBA. With your energy and zest for life that only a toddler can possess, you have made it all worthwhile.

ABSTRACT

How exactly the business managers of commercial off-the-shelf (COTS) application software small and medium sized enterprises (SMEs) should run their businesses is a particular problem. The market has a number of distinct characteristics and is increasingly competitive. In contrast to their larger counterparts, SMEs often have limited resources and knowledge that can be applied to strategic planning and management activities. A literature review reveals that there is no core body of research, or accepted set of concepts, that specifically address the problem in any great depth. While a wealth of generic business and strategic management reference materials exist, it is unclear exactly what the precise relevance and uptake of these is for COTS application software SME business managers.

This study specifically investigates whether a holistic guiding management framework can be developed for running a COTS application software SME business. A grounded theory paradigm provides the theoretical basis for the research design. This was chosen to foster creativity and ensure the research is not restricted by previous thinking. The results of the research are a new model that provides a single guiding framework for managers running COTS application software SME businesses. This integrates numerous aspects of perceived best practice in a unified and holistic theoretical model. The research contributes to both scholarly and practitioner domains. A more rigorous and scientific approach to software business management theory has been developed. The ability to undertake specialised strategic consulting services for a COTS application software SME business is improved.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 The Business of COTS Application Software

1.1.1 COTS application software products

The reach of commercial off-the-shelf (COTS) application software products and their influence on how business activities are transacted across the globe is ubiquitous. COTS application software products are used extensively by businesses across the developed world and increasingly across the developing world as well (OECD 2008). COTS application software products come in a wide range of different configurations and they can be used by customers to create business value in numerous ways.

COTS application software products can be categorised using industry terminology into three broad groups: enterprise resource planning (ERP), niche/vertical, and shrink-wrapped solutions. ERP software applications provide cross-industry solutions to integrate and optimise multiple business processes across the whole organisation (Jensen & Johnson 1999; Mabert et al. 2001). Standard ERP applications include Finance Control and Accounting, Supply Chain Management (SCM), Human Resources (HR) and Customer Relationship Management (CRM) (SAP 2006a). The provision of specialised products to support the unique business processes of specific vertical industries is also extensive for application software solutions. Financial services, manufacturing and retail industries are just a small sub-set of verticals that are served by a whole swathe of different COTS application software offerings (IDC 2006). Shrink-wrapped COTS products provide generic functional capabilities that have relevance when applied to relatively standard business tasks. PC desktop and office software are typical examples of these product types. A small business accounting package is another example.

COTS application software is one of three different types of COTS software products. The other two are COTS software development tools and COTS system infrastructure software (OECD 2008 p53).

1.1.2 The industry

The COTS application software sector is a large and fragmented international industry (Cusicka et al. 2008). The size of the global COTS application software market was estimated at roughly US$134 billion in 2008 (OECD 2008 p53). This forms a part of the global COTS software market estimated at US$295.8 billion, which in turn is a sub-sector of the Information and Communication Technologies (ICT) industry, which had a total world ICT market value of US$3,434 billion. The global COTS application software market is largely led by suppliers from North America and in particular the United States of America (USA). Figures for the larger global COTS software market split market value with the USA having 40.8% (US$120.6bn), with the European Union and Japan comprise roughly 30% (around US$89bn) of the industry (OECD 2008 p53). Australian constitutes 1.2% (US$3.54bn).

The COTS application software sector is dominated by a number of extremely large vendors. These are commonly referred to as ‘mega-vendors’. Statistics for the larger combined worldwide COTS software and IT services show that in 2007, the top five industry vendors comprised over 30% of the market share of revenues (OECD 2008 p64). The leading COTS application software vendor, SAP, had software and services revenues in 2007 of US$14.0bn and Oracle with total revenues (including COTS software development tools and COTS system infrastructure software) of US$18.0bn (OECD 2008 p64). Other major players in the IT industry, such as Microsoft, IBM, Computer Associates and EDS, do compete in the COTS application software arena, but have their core businesses in the provision of software development tools, COTS system

infrastructure software, IT services, or a mixture of these (Desmond 2005). However, the mega-vendors are not the only players in the COTS application software market.

In addition to the mega-vendors, the COTS application software sector is also characterised by numerous small and medium enterprise (SME) vendors (Fuller et al. 2001; Thörn 2010). Literally tens of thousands of COTS application software vendors supply a vast array of different software products and solutions to a diverse customer base (OECD 2008). While mega-vendors dominate the ERP product spaces (for example, Financials and HR solutions) and generic platform shrink-wrapped products (for example, Microsoft Money), SMEs tend to operate in an assortment of numerous smaller niche markets. Examples of these COTS application software SME markets are qualitative research analysis software, television media management and broadcasting software, and transport regulation enforcement software. Typically, these markets are either too specialised, or too small, for the large vendors to enter. They are generally more attracted to the returns that can be made from competing in mass-markets.

1.1.3 Business challenges

A significant challenge for the business managers of vendors operating in the COTS application software market is how they should compete within an increasingly competitive landscape (Ahmed & Fernando 2007; Aurum et al. 2008; Alic et al. 1991; Igel & Islam 2001). Numerous complexities and uncertainties need to be considered (Boscha & Bosch-Sijtsemab 2010). How to develop a cohesive and robust approach to competing in this market is a significant challenge (Ahmed & Fernando 2010). The COTS application software sector is very dynamic and competitive. There is a mix of visionaries, leaders, challengers and niche players. Factors such as company size, target business sector, product cost, functional richness, geographical market and extent of software services offering often define where a vendor fits in the overall software market landscape. The industry is characterised by mergers and acquisitions (Gaoa & Iyer 2009).

However, while there is much consolidation resulting in a decrease in established vendors, in parallel many new innovative start-ups continue to enter the sector (OECD 2002). In addition, alternative COTS application software business models are emerging and these may considerably change the way software businesses operate in the future (Cusicka et al. 2008; Postmusa et al. 2009).

While there are certain inherent challenges for all players associated with competing in the high-tech COTS application software market, the dominant mega-vendors have carved out substantial businesses for themselves (OECD 2008; Business Insights 2008). SAP, the leading application software vendor, has over 82,000 customers in over 120 countries around the world and its products are used by tens of millions of users (SAP 2008). These large vendors often have numerous customers, extensive product ranges, wide reaching distribution networks, considerable research and development (R&D) funds, and mature business management capabilities. In contrast, many SME vendors are running much more fragile business ventures.

1.2 The SME Problem

How exactly the business managers of COTS application software SMEs should run their businesses is a particular problem (Thörn 2010). In contrast to their larger counterparts, the managers of COTS application software SME vendors often have limited resources and knowledge that can be applied to strategic planning and management activities (Pulkkinen & Forsell 2007; Verlage & Kiesgen 2005). While some managers of COTS application software SME vendors seem to do a good job of running their businesses, from industry reports, others appear to do this badly.

Numerous COTS application software SME vendors are commercially unsuccessful over the long-term (Chapman 2005; Honjo 2000). Often their managers have failed to build robust underlying businesses (Hunt 2010). There is a multitude of reasons for why this

can be the case, but there do appear to be some common patterns (Cusumano 2004a). A lack of new sales is often prevalent (Chapman 2005). A desire to achieve growth frequently leads vendors to take on unjustifiably high risks or undertake little contingency planning for possible downsides. Chief executive officers (CEOs) who have a technical background but limited business management experience and skills can lead to businesses with no foundations (Haider et al. 2009). Many vendors operate in a reactive fire-fighting mode and limp from year to year. Many have ended up going bust.

Two COTS application software SME vendors that have struggled to build robust businesses are profiled below.

> Company A is a twenty-year-old, mid-sized health software solutions vendor with 120 staff. It develops and sells a range of patient administration, primary care and clinical information systems products. The executive management team comprises technical enthusiasts, academics and a number of medical practitioners. None appears to have any business management experience or a track record of running a software business successfully.
>
> Organisationally, the vendor is fragmented. The R&D team is very innovative but has an inward looking focus. Many members have never spoken to a customer. Sales staff fit a stereotypical software salesperson profile: software dreams that do not actually reflect the vendor's capabilities or products are enthusiastically presented to potential customers.
>
> A separate department of legacy technology professionals work on all aspects of the legacy product. The teams have limited respect for each other and largely isolated in their own areas. More than 70% of the company revenues originate from a single legacy product: these comprise support, maintenance and minor enhancements. This product is increasingly cumbersome to operate and maintain, and only a few new customers have been generated from this in the last ten years. The small group of customers that do use this product would like to move to a more modern software solution, but they are locked in and any would not undertake such a move lightly.

The vendor has invested tens of millions of R&D dollars over a five-year period to develop a new state of the art clinical information system. This software product is overly complex technically and its reliability and finish is of a very low quality. Despite significant sales and marketing, no successful sale and implementation of this product has been achieved.

The vendor is aggressively run. A senior management team whips up a culture of frenzy and excitement. Numerous acquisitions have been undertaken in the last five years, but it is unclear exactly how these new capabilities and products will fit together to form a total bigger than the sum of its parts. After an Initial Public Offering (IPO) and the raising of around $50 million dollars in funds, within four years, this had all been spent with no tangible return of any sort to show. During this period the company had a $20 million a year turnover, and over four consecutive years managed to lose $10 million a year. In summary, the vendor appears to limp from one crisis to the next. It appears to be unclear exactly what business it is in and has no strategy or direction for where it should head.

Company B is a small, fifteen-year-old software vendor that develops and sells outdoor media sales and inventory management software. The company's founders originate from the outdoor media market and they know this industry very well. The company has a single product and has roughly a dozen customers spread across Europe, Australia and the USA.

The company used to have fifty staff a number of years ago; it now has less than twenty, as it no longer has the revenues that it had previously when several clients were spending a great deal of money on its software. Although several new customers have been signed over the last five years, the main revenue sources now come from support and minor enhancement work for existing customers.

While the vendor has an established product and operates in a niche target market that currently has few competitors, commercially, the vendor could be described as being in limbo. The technology that the vendor's product was developed on is now dated and as a result, the product is relatively clunky and clumsy to use in comparison to software that is more modern. Having operated at high levels of risk when the company was younger, the vendor does not want to borrow the funds that would be required to develop a commercially more attractive product.

It is not suggested that all COTS application software SME vendors have inherent problems. However, it is argued that the profile of these two vendors is not uncommon in the industry (Chapman 2005; Sink 2006; Thörn 2010).

More formal guidance on how to compete as a SME vendor in the complex COTS application software market is increasingly essential (Ahmed & Fernando 2007). Historically, many COTS application software SME vendors have managed to build reasonable businesses that have been based upon innovative product development and capitalising on immature software markets. Over time, these have been supplemented with professional services and product extensions, and as market segments have become crowded, merger and acquisitions (M&As) have been a common means to attack competitors and achieve growth. However, as the market matures and becomes more competitive, it is questionable if these somewhat opportunistic approaches to running a COTS application software SME business will provide a viable model for the future.

1.3 Problem Context

The problem of how to run a COTS application software SME business is of particular interest as there is uncertainty about what specific guidance for this exists. Managers have pursued an assortment of different approaches and strategies for running COTS application software SME businesses, but a clear, holistic and focused approach on how to run such a business appears elusive (Ahmed & Fernando 2007; Ruokonen 2008). Limited specific reference frameworks and models appear to exist and it is unclear how the managers of successful COTS application software SMEs approach the running of these businesses.

While a wealth of generic business and strategic management reference materials exist, it is unclear exactly what the precise relevance and uptake of these is for COTS application software SME business managers (Ahmed & Fernando 2007). At a first glance, it seems

plausible that the standard models and frameworks for strategy management, innovation, software systems engineering, product development, sales and marketing, and operations management would provide a COTS application software SME business manager with all the tools they need for running their business. However, closer examination reveals that these business management materials are only partially relevant for the problem. While these standard frameworks provide guidance for various aspects of a COTS application software SME business, on aggregate they do not provide a complete toolset for running such an organisation. The shortfall is that collectively these standard frameworks do not address a number of specific COTS application software SME business challenges. Therefore, a holistic and end-to-end business reference model is lacking.

It is the complexity and distinctive characteristics of COTS application software SME sector that lie at the heart of the COTS application software SME problem (Boscha & Bosch-Sijtsemab 2010; Hunt 2010; Pulkkinen & Forsell 2007). The COTS application software SME sector is by no means a unique industry. The main cornerstones of any business, such as having a product that serves a target market, a viable financial model, and functioning sales, distribution and operations capabilities, are still key factors for a COTS application software SME business. However, there are a number of other competitive dynamics, some of which are present in other industries, that come together to make the COTS application software SME distinctive (Cusumano 2004a). These factors include, but are not limited to: dynamic high-tech markets, short product life cycles, high research and product development costs, the culture of IT professionals, the pricing and distribution of information goods, technology hype and band wagon effects, a complex eco-system of market participants and dependents, software upgrades, technology platform standards wars, and an evolving and maturing market. As a result of these distinctive characteristics of the COTS application software SME sector, a

'software business requires a unique approach to strategy and management' (Cusumano 2004a, p3).

There are a number of market specific questions that COTS application software SME businesses should ask themselves (Cusumano 2004a). Should a vendor's prime orientation be product or services? Should the software product or service be a horizontal or vertical market offering? Should the company aim to be a leader, follower or a complementor? Should the software offering be aimed at niche, enterprise or mass-markets? How will a recurring revenue stream be generated so the company can stay in business through buoyant and turbulent times? Standard strategy management concepts are still very relevant for vendors, but in addition, there are other competitive strategy rules that also need to be considered (Anderson & Wood 2002).

There has been limited research into the specifics of running a COTS application software SME business (Ahmed & Fernando 2007; 2010; Thörn 2010). The problem has been largely neglected thus far. Gallaugher and Wang (2002a) suggest that '*despite the importance of software to the world economy and the notion that software markets are different from conventional markets, few studies have attempted to offer a synthesized analysis and empirical examination of the unique forces at work in this industry*' (p304). Previous work over the last thirty years has introduced data and paradigms for technology adoption and demand-side forces. However, there is a lack of truly historic knowledge and supply-side research is scarce (Hopcroft 1987; Mason et al. 1997). I personally have been involved in and around software businesses as a practitioner for nearly twenty years, and in my experience, I have not come across any specific models for running a software business.

1.4 From Problem Identification to Research Objective

While many COTS application software SME vendors struggle to run their businesses and limited specific guidance reference materials exist, an opportunity to address the COTS application software SME business problem is now identified. There are a number of successful robust COTS application software SME businesses; the managers of these organisations possess considerable wisdom on how to build and run successful COTS application software SME businesses. The Australian headquartered COTS application software SME vendors, Technology One, Mincom, Bravura Solutions and Market Boomer, provide examples of organisations that are informally referred to by industry practitioners as well-run, robust businesses. In addition, there are also many business managers who have been involved in running COTS application software SME vendors that have not performed brilliantly, often these individuals have learnt lessons from these episodes and they now possess valuable perceptions and insights into how they would approach this differently next time around.

Collectively, the business managers of COTS application software SME businesses hold a substantial amount of knowledge that has the potential to be developed into a holistic set of business management tools for running a COTS application software SME business. However, a large part of this knowledge and expertise currently lies uncollected and fragmented across a number of business managers who have experience of running COTS application software SMEs. The challenge is how to capture this information and what to do with it when it is collected.

The objective of this research is to integrate perceptions of best practice held by practising software business mangers to create a focal theory. The intention is that this new focal theory will provide business managers of COTS application software SME businesses with an innovative and holistic reference framework to assist them in running

their businesses. If improved guidance on how to run a COTS application software SME business can be developed, this will contribute to filling a gap in existing theory and provide real tangible benefits to practice. To achieve the research objective, data that describes perceptions of best practice need to be located, captured, distilled and consolidated into a robust set of findings. These findings then need to be interpreted and implications understood. An analytical approach that includes creativity is fundamental. A purposeful and sound approach for undertaking the research is an imperative. The following methodology overview in section 1.6 will explain this, but as a starting point, determining exactly what questions the research needs to ask business mangers of COTS application software SME businesses is vital to focusing the research on meeting its objective.

1.5 Research Question

The main research question to be answered by the research is:

- Can a holistic guiding management framework be developed for running a COTS application software SME business and if so, what might it comprise?

A key part of answering this central research question is to illuminate, contextualise and break down the key aspects involved in running a COTS application software SME business. Therefore, the main research question is distilled into a number of subsidiary questions:

- What existing management models are relevant to running a COTS application software SME business?
- What distinctive aspects of the COTS application software SME market require management models beyond those available in general reference materials?
- What mental models have business managers formulated to assist them in running COTS application software SME businesses?

- What commonalties and differences are apparent across different business manager mental models?
- What are the overriding factors that can enable the various mentals models and generic management tools to be integrated into a unified and overarching framework?

1.6 Methodology

A qualitative research method has been chosen to investigate the COTS application software SME business problem. Being direct, intense and using expressive language, this mode of research offers the facility to inductively analyse data that describes the software vendor problem (Creswell 1998). My own epistemological view is that immersion in the problem environment guided by a qualitative paradigm, even allowing for any presuppositions, is the most suitable approach for investigating the software vendor strategy problem (Groenewald 2004). This approach allowed for in-depth, face-to-face interviews with research subjects and supported an open-minded journey of discovery.

A grounded theory paradigm provides the theoretical basis for the research design. The COTS application software SME business problem research design needs to foster creativity where there is little existing theory or previous thinking is been constrained by traditional paradigms. In parallel, it is imperative that the research approach does have a strong foundation in both principles and procedural steps. A key aspect of the research approach is to examine and reflect on perceptions of senior software industry professionals in an attempt to gain an understanding of the main factors associated with running a COTS application software SME business. Grounded theory is differentiated from much other research, as it is explicitly emergent. It does not test a hypothesis. It sets out to find what theory accounts for the research situation as it is (Dick 2005). The

advantage of the method is that it allows new theory to emerge from research data, in contrast to data being forced into predetermined frameworks (Glaser 1992).

My own theoretical lens is orientated to a research approach with foundations in grounded theory. I am a senior IT professional with nearly twenty years commercial experience of supplying and consuming application software. With direct experience in and out of COTS application software SME businesses, none that I have witnessed has used an explicit guiding management framework to run their business. There is a danger that a hypothesis-based, quantitative survey approach based upon current literature may just result in standard industry rhetoric being reiterated with no new insight being provided. In contrast, grounded theory can provide a fresh perspective. I have a strong background, both professionally and in academic knowledge, which provides the background context for analysing the problem. I feel confident in my ability to manage the delicate balance between having an open-minded view while collecting data about the phenomenon, with the need to keep focused on developing new theory that contributes to the body of knowledge about the research question.

1.7 Contributions to Practice and Theory

The research aims to contribute to the body of knowledge by the creation of new understandings that will be of interest to practitioners running COTS application software SME businesses. This will contribute to theory and to how practitioners take informed action.

New frameworks and models that can assist owners and managers running software business have the potential to have significant value. In initial conversations preceding the commencement of the research, I received very positive feedback from several industry participants on the importance of developing the thinking in this space. However, if any contribution to practice is to be well received, it is imperative that it is

perceived as being relevant and having real practical applications. A challenge is to develop theory that is robust enough to be applicable across the variants of the many different software businesses that exist across the industry as a whole. New theory is unlikely to provide a prescriptive set of instructions for running a software business. It is more likely to be point of reference that provides a backdrop for vendors to check against when making decisions involved in running their business.

The apparent domicile for contributions to academic body of knowledge concerning the running of a software business is likely to be Information Systems (IS) research. However, IS research is predominantly orientated towards the engineering of technology products (Ein-Dor & Segev 1993) and the demand-side consumption of technology solutions (Orlikowski & Iacono 2001). Therefore, there may not be a perfect home or taxonomy into which new research into the running of a software business fits.

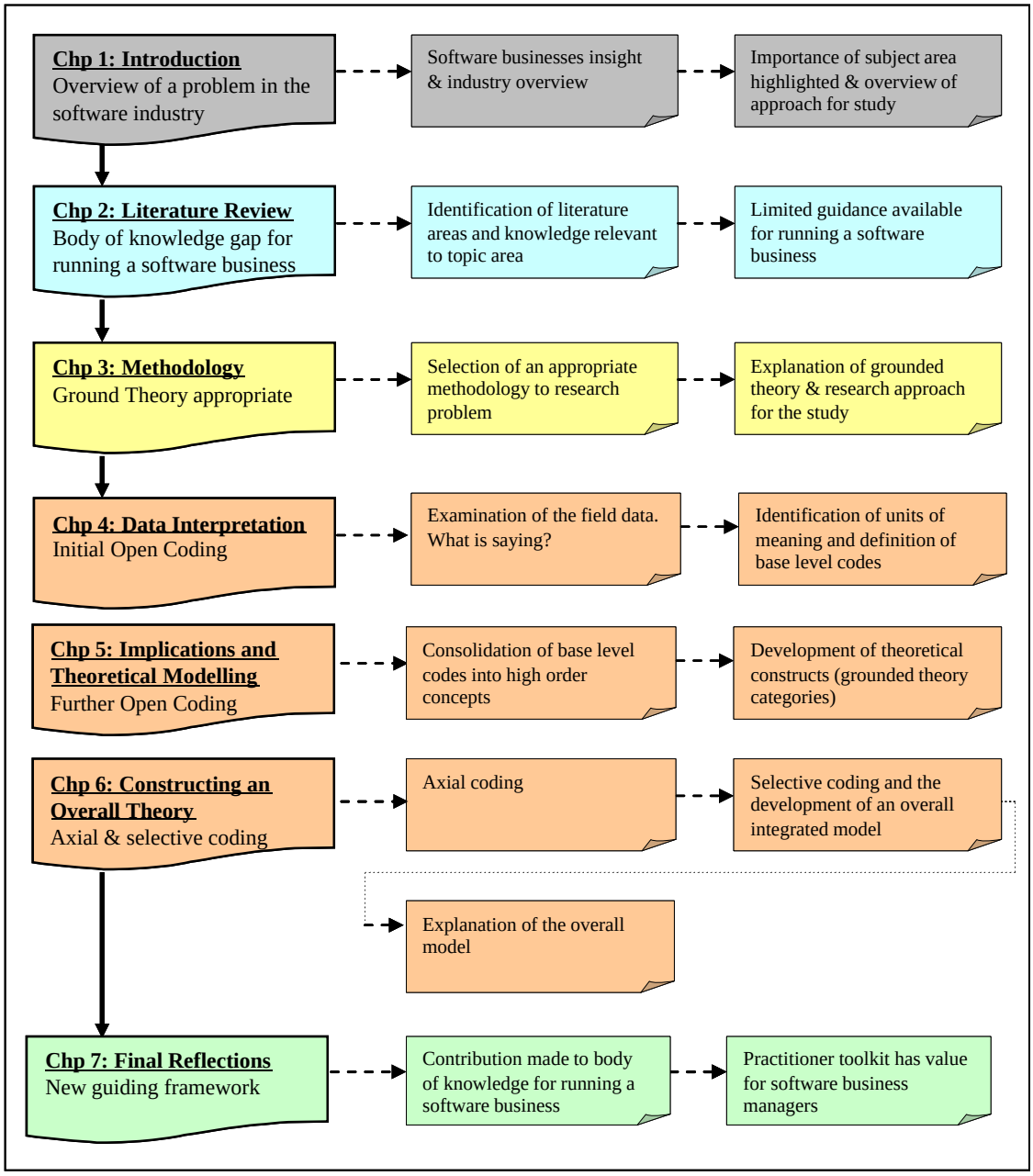
1.8 Structure of Thesis

Figure 1a maps out the narrative structure of the thesis.

Chapter 1 has provided an overview of the research problem, the research questions and an insight into the paradigm for undertaking the research.

Chapter 2 is the literature review. While the IS discipline offers a number of concepts that are deemed relevant to the running of a COTS application software SME business, there is a notable lack of any central premise or theory specifically orientated to providing guidance. Thus, a much wider search of management research disciplines was undertaken. Contributions were taken from the scholarly fields of management science, strategy, innovation, product development/management, marketing and sales, and economics. Concepts were framed and contextualised to provide a literature map and a platform for investigating the problem.

Figure 1a: Thesis Structure and Narrative



Chapter 3 addresses methodology. The selection process and justification of a grounded theory research methodology for guiding the software business study is detailed. The mechanics of the research study are then outlined. An outline of the research design for the data collection component of the study is provided. A systematic description of the data analysis approach for constructing theory to explain the software business is given.

Chapters 4 to 6 constitute the analysis and findings for the study. A COTS application software SME business is a complex entity. Numerous interrelated concepts of relevance to running such a business have been identified. The findings chapters un-stack all these factors, find meaning and order, identify main themes and then progressively build up a total picture. The findings chapters systematically follow the grounded theory data analysis coding steps. Chapter 4 documents the data interpretation process and summarises an initial set of base-level codes that were generated from an initial open coding exercise. Chapter 5 covers the next stage of the analysis, which was to understand the implications of these codes and develop a set of higher-order concepts and categories that helped explain the various aspects of running a software business. Chapter 6 covers the final stage of the analysis. The categories are connected into a single unified model for running a COTS application software SME business. The consequential overall model for running a COTS application software SME business is then outlined.

Chapter 7 discusses the contribution the new overall software business model makes to the current body of knowledge. A number of key insights concerned with running a software business are highlighted. The findings are assessed. The strengths and weaknesses of the new theory are discussed. Recommendations for further research are suggested. Finally, the implications of the findings are summarised.

1.9 Summary

The research aims to develop new theory to provide new and improved guidance to business managers running COTS application software SME businesses. This is important because limited reference literature exists to assist managers running business in the sizeable and significant COTS application software SME market. The intention is to integrate perceptions of best practice held by practising software business mangers to create a focal theory. This focal theory will provide a holistic and overarching framework

of management tools. A grounded theory research paradigm has been chosen to foster creativity and to ensure the research is not restricted by previous thinking.

2 PROBLEM GUIDANCE: LITERATURE REVIEW

2.1 Introduction

The starting point for a detailed literature review was to examine research undertaken within both the Strategy Management and Information Systems (IS) disciplines.

Having studied strategy in some depth prior to commencing this study, my view was that this field would offer a wealth of management frameworks that would provide guidance to a manager running any generic type of business. The unanswered question was what contribution generic models from the strategy management domain would specifically provide for running a COTS application software SME business. Would the generic strategy models be relevant and appropriate? Would the strategy management field contain any models that specifically addressed businesses in the software industry?

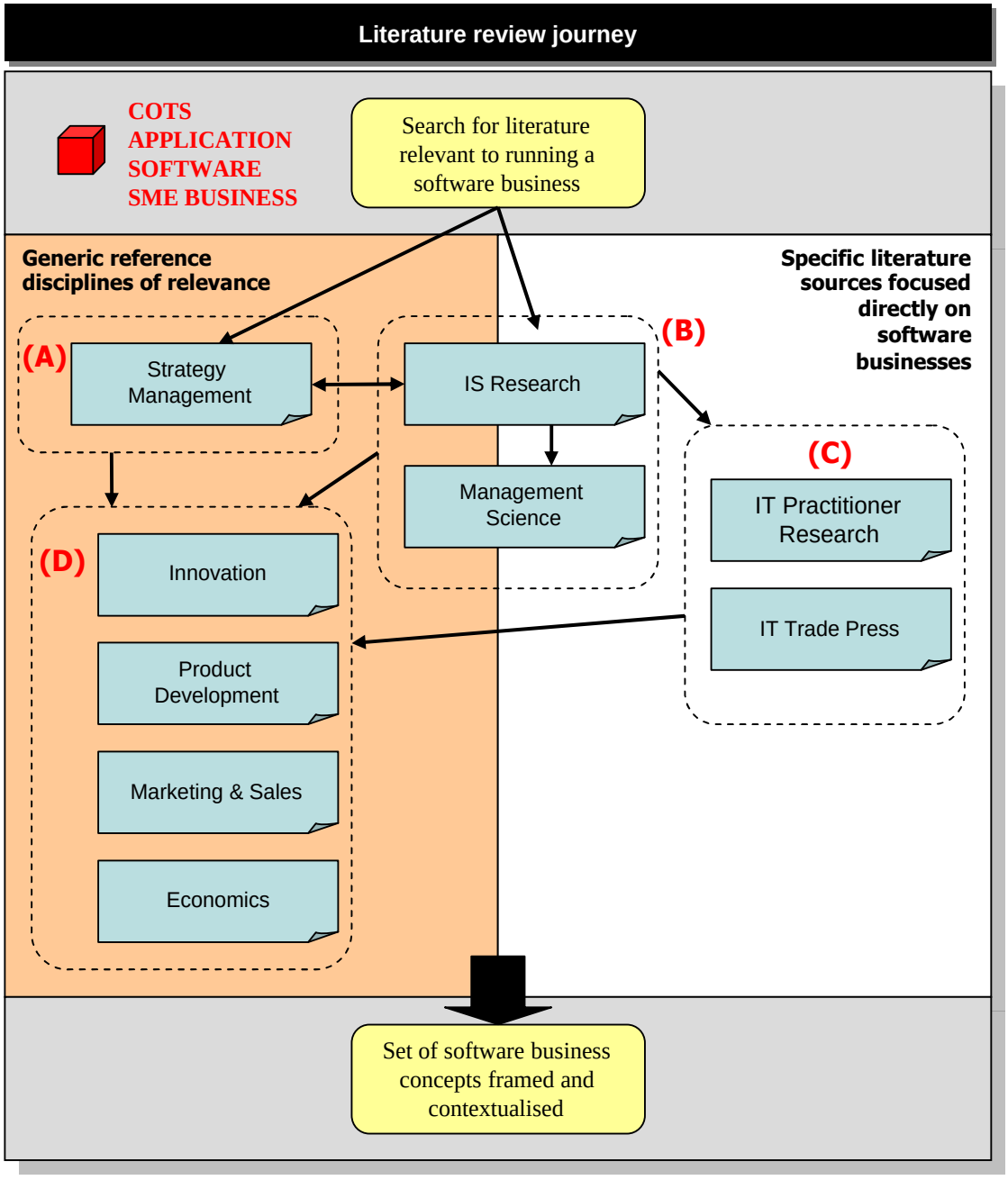
In addition, I expected that the IS discipline would provide a plentiful source of models and concepts to contextualise and frame the research problem. Prior to the detailed literature review, I was aware that IS research contained a significant body of knowledge on the topics of software engineering and the effective application of IS solutions. However, it was unknown exactly what I would discover in terms of specific guidance for how to run a COTS application software SME business.

As the search for guidance on the research problem progressed, it became evident that a wide range of different business concepts and variables outside of both the strategy management and IS disciplines were being identified as relevant to the running of software business. This led to the considerable broadening of the literature review in order to include relevant research.

While a traditional literature review for academic research focuses on ‘drilling to great depth’ in a narrow topic area, when investigating real-life business problems, this is not always the most appropriate strategy. It became apparent that running a software business involved dealing with multiple variables. Consequently, the scholarly fields of Management Science, Innovation, Product Development, Marketing and Sales, and Economics came into scope for the literature review. Relevant contributions from practitioner research and industry analyst reports were also identified. A detailed review of each of these areas would have been very time consuming and questionable in terms of its direct contribution to the problem. However, the identification of a set of high-level concepts of relevance to running a software business contributes value in terms of framing the research problem. Figure 2a depicts the journey that was undertaken in constructing the literature review and outlines the sources included.

The structure of the main body of the chapter mirrors the order in which the literature references were identified. As displayed in Figure 2a, literature sources have been grouped into four clusters. The chapter has five main sections. First, relevant concepts and frameworks from the Strategy Management domain are identified and outlined. Second, contributions from IS and Management Science are discussed. Third, inputs from IT practitioner research and IT trade press are captured. Fourth, generic management models from a range of disciplines are flagged; these include Innovation, Product Development, Marketing and Sales, and Economics. Fifth, all the relevant concepts identified in the literature review are consolidated to provide a frame of reference for investigating how to run a COTS application software SME business.

Figure 2a: Literature Review Narrative



2.2 Strategy Management (Literature Cluster A)

A review of strategy management literature reveals a substantial body of research and much of relevance to vendors running a software business. Four key areas from the strategy domain are highlighted. These are categorised as 'sustainable competitive advantage', the 'strategy development process', 'software business strategy context' and 'software business strategic directions'.

2.2.1 Sustainable competitive advantage

A primary strategic objective for an organisation is to focus on how such a superior relative competitive position can be created (Hedley 1976). A core premise of strategy theory is that an organisation can accomplish superior performance if it can achieve a position of advantage over its competitors (Porter 1991).

Sustainable competitive advantage is linked to an organisation's ability to create a distinctive set of products and services from its underlying portfolio of assets and capabilities (Porter 1996; Teece et al. 1997). Strategy management research assists organisations in this goal by providing guidance and a structured approach for devising strategies to attain competitive advantage and superior and sustainable financial performance. Generating strategies that discover new points of differentiation are at the heart of strategy management (Nelson 1991). These may include speed, reliability, customer service, product features or technological superiority. Intangible factors may be innovation, intellectual property (IP), the organisation's architecture, accumulated experience, reputation or business relationships with industry partners. Two different schools of strategy, a resource-based view (RBV) and market positioning, provide alternative approaches for achieving sustainable competitive advantage.

A RBV of strategy focuses on how an organisation can best exploit its internal competencies to locate a best fit to compete in a given market (Andrews 1971; Barney 2001). Barney (1991) categorises indicators of potential sustained resource-based competitive advantage into four different groups: value, rareness, imitability and substitutability. Tangible attractive resources may be high-quality products, customer loyalty, operational experience and effectiveness, and technological leads (Wernerfelt 1984). Intangible resources can also contribute significant value to achieving sustainable competitive advantage and these should be considered carefully.

A market-based positioning view of strategy focuses on understanding the external environment in which an organisation operates and best exploiting opportunities that exist within it (Porter 1985). Positive performance and commercial success can be achieved if an organisation's strategy and market offering is fully aligned with the environment (Venkatraman & Prescott 1990; Davies & Walters 2004). The greater the strategic fit between an organisation's resources and its external environment, the greater potential for competitive advantage (Zajac et al. 2000).

2.2.2 Strategy development process

The first step in a strategy development process is to define a strategic vision and formulate clear associated objectives. A vision needs to be contextualised within its macro-competitive environment, and defined taking into account a strategic assessment of industry forces and internal strengths and weaknesses (Raynor 1998). It needs to be rational about what is realistically achievable in the given competitive environment (Wilson 1992). A strategic vision should define the strategic position where exactly the organisation wants to compete.

The second step in a strategy development process is a strategic analysis. The dynamics of a competitive environment can be complex and it is important that an assessment of this is multi-dimensional in nature (Shay & Rothaermel 1999). The scope of this strategic analysis will cover the identification of commercial opportunities, qualification of industry threats, 'what if' questioning, and the examination of key competitor businesses (Glaister & Falshaw 1999). Porter's (1980; 1991) five key forces describe competitive dynamics and these can be used to measure the attractiveness of an industry. The forces are: the threat of new entrants, the bargaining power of suppliers, the bargaining power of buyers, the threat of substitute products or services, and the rivalry amongst existing competitors. A study of an organisation's internal resource capabilities will consider both

individual tangible and intangibles attributes, as well as aggregate value chain activities. Internal resources include an organisation's staff, skills, core competencies, cash flow and technological expertise. The strengths and weaknesses of each entity of a business should be determined. Costs, profit contribution and growth potential should be quantified (Hussey 1968). Intangibles resources such as intellectual capital (IC), customer capital and social capital may be an asset or a liability.

The third step in the strategy development process is to determine the strategic directions that an organisation will pursue to achieve its long-term objectives. Strategy management literature offers a number of well-established standard strategic directions that provide a foundation for formulating an organisation's strategy. Over the last thirty years, various strategy management researchers have attempted to identify and classify a definitive set of strategy profiles (Abell 1980; Porter 1980; 1985; Snow & Hrebiniak 1980; Galbraith & Schendel 1983; Herbert & Deresky 1987). If a standard strategy profile can be utilised, commercial benefits can be estimated more accurately, strategy implementation managed more effectively and the chances of achieving sustainable competitive advantage enhanced (Fiegenbaum & Thomas 1995). Standard strategy directions, such as product innovation and development, market segmentation, diversification, and growth into new markets, have been proposed as foundations for creating sustainable competitive advantage. A well-known attempt to establish a set of standard approaches for achieving sustainable competitive advantage is Porter's (1980; 1985) classification of three generic strategies. Porter's seminal work suggested that if an organisation wished to attain sustainable competitive advantage, its strategy would need to be based on cost leadership, differentiation or focus. Overall, cost leadership could be based on, but not limited to: superior engineering processes, low-cost distribution, or economies-of-scale. Advantages from differentiation may come from superior innovative products, strong marketing capabilities, or a unique combination of skills. Advantages from focus may be achieved

from a concentrated effort at serving a particular strategic target better than any competitors do.

2.2.3 Software vendor strategy context

In hypercompetitive high-tech sectors such as the software industry, business strategy and planning often needs to be fast moving. Vendors need to be conscious of changes in technology, products and customer demands. They need to be able to react quickly and change direction if necessary (Matheson & Tarjan 1998). The basis for competitive advantage may be technical superiority, higher quality, greater product capability, short product cycles, the strongest sales and marketing, price leadership, or a combination of these factors (Beard & Easingwood 1992; Igel & Islam 2001). High-tech strategies frequently entail making autonomous offensive moves that incorporate the ability to change position as lessons are learnt along the way (Bahrami & Evans 1989).

High performing software businesses comprise smart people who understand both business and technology and are experts at taking calculated risks (Roeding et al. 1999). Industry experts agree on the importance of these ingredients for success (Yoffie & Cusumano 1999; Keller 2005b). Bill Gates is often cited as a phenomenal CEO as he blends technology vision with an astute business mind (Cusumano & Selby 1998). Hiring the brightest managers and employees comes next. A high-performing company culture is of vital importance.

The importance of strategic positioning is a central point to any software vendor strategy. Building a wide-based alliance within the software market, leveraging opportunities for complementary products and services and proactively taking advantage of network externalities are key possibilities (Yoffie 1996). Determining what its core business is and how it is going to differentiate itself will be a software business decision (Brooks 2005; Vincent 2006).

2.2.4 Software business strategic directions

Software businesses can orientate towards a number of strategic directions in order to achieve competitive advantage. These include a primary focus on customer objectives, a focus on technological superiority, or a highly efficient construction of product (Igel & Islam 2001). A strategic position that focuses on the customer may aim to reinvent and redesign the customer value chain, complement existing products with associated services, or improve the ability to mix and match software products (Vincent 2006). A strategic focus on technology may aim to position the vendor as an expert at: designing, generating, acquiring, modifying, converting or vending specific technology (Sharif 1994). Shapiro and Varian's (1999) research identifies seven critical strategic assets that a software business should consider: innovativeness, first mover advantage, product construction ability, IP, reputation and brand, size of user base and relationships with eco-system partners.

A strategy focusing on market leadership can be a highly effective way of achieving advantage over competitors. Three key pillars of market leadership are building market leadership, sustaining market leadership and breaking the control of existing market leaders (Roeding et al. 1999). Cusumano (2004a) argues that it is important that new software product is compelling and has a prospective market, and that there is a strong indication of customer interest. Brouthers and Kruis' (1997) software vendor differentiation strategy model, adapted from Hall's (1992) generic differential model, provides a useful framework to assist vendors in determining a strategic position that will differentiate themselves from their competitors. Four distinct approaches are offered: a software functional differential; a software positional differential, such as a service-based strategy or different marketing approach; a cultural differential such as a future-orientated strategy; or a regulatory differential, such as an alliance based strategy. A vendor may

attempt to break the market leadership of another when commercial prospects are favourable and barriers to entry are relatively low (Roeding et al. 1999).

International expansion can be a highly effective strategy for a vendor to achieve growth (Ruokonen 2008). A vendor hoping to expand overseas will require distribution, implementation and support capabilities for the geographical market it aims to penetrate (Hasted 2005). The cost of entering a new geographical market is likely to be significant. An established local distributor will allow local knowledge, brand and contacts to be leveraged and risk minimised (Dver 2003). A vendor targeting an overseas market needs to make decisions about the level of internationalisation and/or localisation that is required for their software product (Dver 2003). The localisation maturity model (LMM) describes a set of levels that characterise the phases of maturity a software product progress through as it is tailored for international markets (DePalma 2006). Emerging markets are a hive of activity as both domestic and international vendors scramble to achieve market share in these growing segments. While mega-vendors have mature software offerings that local vendors cannot provide, local suppliers often possess key local business expertise that the global suppliers lack.

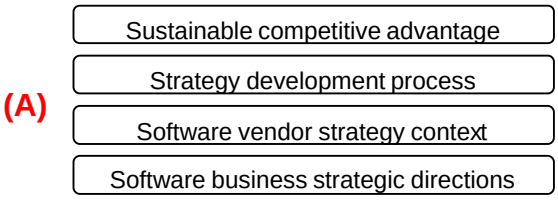
M&A is one of the most common approaches that vendors have used to grow and achieve competitive advantage. Benefits include securing a vendor's position in the market, eliminating competitors and leveraging a monopolistic position with customers. Oracle's approach involves indentifying competitors, acquiring them, discontinuing their products and migrating customers over to the Oracle portfolio of software solutions (Kessler 2003; Babcock & Kontzer 2004). Numerous mergers have in reality turned out to be disappointing and failed to live up to expectations (Gaoa & Iyer 2009). There is now mounting evidence against the common wisdom of M&A (Ghemawat & Ghadar 2000). The majority of software business mergers prove to be considerably more challenging

than predicted. Many turn out to be disappointing or even disastrous for the acquirer (Lay 2005). A common letdown is synergy initiatives failing to achieve hoped for efficiencies and cost savings (Goold & Campbell 1998). M&A can result in distinctive competitive edge being eroded, organisational value being destroyed and future growth opportunities being depressed (Ghemawat & Ghadar 2000; Harding & Rovit 2004). The negative picture of merger mania encompasses giant unresponsive monopolies, market instability, reduced choice and higher prices (Silvers 1992). Lay's (2005) Category Maturity Life Cycle Model distinguishes different types of M&A initiatives by the stage in the industry's evolving maturity. Different M&A strategic objectives include obtaining product IP to achieve a more complete offering, to extending the domain footprint to access bigger markets, to gaining access to new geographies and to taking out a competitor can increase financial leverage.

Partnerships can be an effective vendor strategy for achieving growth. Mega-vendors such as SAP have invested heavily in the partnership model as a strategy for growth. SAP's 'PartnerEdge' programme has constructed a network of hundreds of organisations that possess expertise in implementing and supporting SAP's COTS-AS solutions (SAP 2006d). Partner companies can be motivated and have both direct and indirect incentives to grow a software vendor's solutions footprint (Brooks 2005). Partners need to be trained in the implementation and use of the software product; and the value proposition across core vendor and partners needs to seamless from a customer perspective (Roeding et al. 1999). If a vendor does not share enough information about its software solutions then this may limit its partner ability to enhance the value proposition and facilitate growth. However, if a vendor gives away too much information about its products, there is a danger that a partner may harness this to create a rival value proposition. Managing the risk and reward of a partnership strategy requires a fine balance.

A strategy that a number of vendors are embarking on is to change the value proposition that they provide to their customers. The alternative software business models software-as-a-service (SaaS) and open source software (OSS) are examples of this. Research in outsourcing (Bennett & Timbrell 2000), customer satisfaction and IT adoption (Susarla et al. 2003) provides a foundation for designing robust SaaS business propositions. A major attraction of OSS is that it provides a viable lower cost option to customers who are openly looking for lower cost software options (Berquist 2006).

2.2.5 Summary



A review of strategy management literature reveals a substantial body of research and much of importance to managers running software businesses. While much of this material is applicable, most of this is generic in nature and not directly specific to the software industry or software businesses. As such, an extensive review of this material is not appropriate. However, while no specific guidance or framework for running an overall software business has been discovered, four key factors of importance to running a software business have been identified. First, the generic strategy concept of sustainable competitive advantage provides a driving principle for any commercial business. Second, a strategy development process provides a toolset to assist in this being defined and attained. Third, focusing on the software vendor strategy context, a whole range of distinctive software business considerations will need to be understood. Fourth, a number of prevalent software business strategic directions are available to enable a vendor to achieve its long-term objectives.

2.3 IS Research and Management Science (Literature Cluster B)

The review of the Information Systems (IS) literature revealed that a number of topics in this discipline also appeared to be prevalent in the Management Science field. In the context of the software business problem, it became apparent that IS Research is closely linked to Management Science. Thus, it was decided the review of IS literature would be extended to examine Management Science as well. The Information Systems (IS) and Management Science scholarly disciplines both provide useful reference fields for examining how to run a software business.

Five key areas of relevance to running a software business have been identified across the Information Systems and Management Science disciplines. These are categorised as 'software product', 'software construction', 'software peripherals', the 'dynamic evolving IS industry', and 'software market idiosyncrasies'. In addition, the maturity and focus of the IS discipline in context of the research problem are discussed.

2.3.1 Software product

Software product functionality is a dominant attribute that customers focus on when selecting and evaluating which software to purchase (Hoxmeier 2000). The richness of a product and the provision of features that customers desire and value can directly link to vendors' performance (Hui & Tam 2002). Vendors arguably artificially talk up the value of new product features in order to maximise short-term revenues and profits. The concept of 'featuritis' is endemic in the software sector. Vendors continuously produce new releases of their software products that are packed with new features. Although major product advances do occur, there is often questionable value in customers acquiring every latest version of a product (Negroponte 2004). Rather than pursue a strategy of creating bloated, feature-extravagant systems, it may be more appropriate for vendors to

focus on good software design principles to produce simple, but highly effective and usable software based solutions (Negroponte 2004).

Goodwin (1987) argues that vendors who focus on providing functionality alone are not doing enough and that usability, defined as a measure of how effectively functional capabilities are used, is a more important parameter. The user experience needs to be engaging and pleasing, as well as simple and effective. It is vital that a software product enables a user to undertake whatever business task or objective they wish to achieve as easily as possible (Mirel & Olsen 1998). The collective uptake of a standard set of usability tests is likely to lead to high quality products, which would benefit both vendors and customers in the long-term. The USA-based National Institute of Standards and Technology has made progress to create a well-defined and testable set of industry standards. However, these are not widely adopted by vendors or customers to assess and measure software product usability.

Having the flexibility to be able to customise and tailor a generic software product to the specific requirements of an individual organisation is important. Research has found a statistically significant relationship between this and the customer value achieved from implementing software products with these traits (Nidumolu & Knotts 1998). Customisability provides customers with the option of altering a standard software product, which for example may only have a 90% fit, so that it can fully support an organisation's unique business processes. This in turn can directly influence the end-user organisation's competitive performance.

Software product reliability is an important facet of software quality. Work session interruptions, processing anomalies, accessibility-availability problems and inconsistent response times are all examples of software reliability issues that negatively affect the

reputations of some software vendors (Nickerson 1981; Vollmar 2001). To improve product reliability and user confidence in software, a number of complementary frameworks have been established that provide guidance to vendors attempting to improve product quality in this area. Tian (1999) highlights the importance of tackling the problem at its source by explicitly factoring reliability targets into the software development process.

The length of time a software product exists for is identified as an important business consideration. There is a cost incurred in maintaining an old software system and if these become excessive, this can lead to its replacement with a less costly software product to run (Munson 1998). When a newer and significantly superior software solution becomes available in the market, it makes sense to discard older product and replace it with the newer one (Zvegintzov 1984). However, as software products have matured, the mean life span of these has increased. A study in 1992 placed the average lifespan of software at 9.4 years (Dekleva 1992). This compares to an average of only 4.8 years in 1980 (Lientz & Swanson 1980). Zvegintzov (1998) enthusiastically promotes the advantages of longer-living software. Quality improves as product errors are located and fixed. As a result, customer benefits are maximised and upfront product and implementation investments are surpassed as the system remains in service.

2.3.2 Software construction

Software products are built upon underlying technology architecture. The selection of this architecture can be directly linked to a vendor's long-term competitive advantage (Bhattacharya & Krishnan 2002). While an existing technology may be known to be a proven and viable option, it may be limiting or even become obsolete over time. In comparison, a new, innovative technology may provide opportunities to develop a superior software product that enables a vendor to differentiate itself in the market. However, this prospect needs to be weighed up against the uncertainty and risk of

mastering an as yet unproven technology. Morris and Ferguson's (1993) technology architecture competition model provides a useful reference framework of five sequential phases to assist architectural decisions: commitment, diffusion, lock-in, harvest and obsolescence, and regeneration. As a vendor moves through these stages, a constant challenge is deciding whether to maintain the flexibility of having multiple options, or wholeheartedly commit to a particular technology route (Day & Schoemaker 2000). Vendors should recognise a number of driving architecture determinants (Morris & Ferguson 1993). Successful architectures are proprietary, but also open to others. Good products are not enough; they have to be implemented as well. Special-purpose software solutions need to fit into general-purpose architectures.

Software must be of high quality and highly reliable if it is to be deployed by customers as an enterprise and mission critical business solution (Khoshgoftaar et al. 1999; Harter et al. 2000). The Software Engineering Institute's Capability Maturity Model (CMM) provides an industry standard for software development and quality assurance (QA) (SEI 2006). The CMM model contains five levels of software development process maturity. Level 1 is characterised by competent workers, level 2 by project management, level 3 by engineering process and organisational support, level 4 by product and process quality and level 5 by continuous process improvement (Phan 2001). The basic principle of the CMM philosophy is that a higher maturity process will lead to a more effective and efficient software development process that will directly lead to higher quality software being constructed (Harter & Slaughter 2000). CMM is now widely used by software vendors worldwide to ensure the effective and efficient development of high quality software product (Herbsleb et al. 1997).

Safeguarding the IP of software products and protecting against unauthorised usage is a critical issue for vendors competing in this market. The extent of lost revenue in the

software sector from piracy is considerable. Microsoft calculated that an estimated value of US$2 billion, equivalent to 2% of its revenue at the time, was lost to illegal use of its software products (Rubenstein 2001). At one end of the spectrum is corporate under-licensing, where corporate customers, often through carelessness, have installed more copies of a software product than they have purchased licenses for. At the other end of the scale is the much more difficult issue to tackle, of large-scale organised criminal copying and distribution of software. General approaches for tackling software piracy include: the creation of stronger legal protection, increasing enforcement, improving education and awareness, and setting an example through government leadership. More specifically, software can be legally protected using trade secrets, copyright and patent IP law (Houser 1999) and physically protected using hardware key technology (Stolpe 2000).

Many vendors are attracted to benefits from going overseas to construct their product (Amoribieta et al. 2001). This may be establishing an internal development capability offshore, or by outsourcing to a specialised development house overseas. Off-shoring can be used by software vendors to become cost-effective and gain access to highly skilled resources located overseas (Zatolyuk & Allgood 2004). India and a number of other developing nations are now established as world leaders. However, offshore construction of software introduces additional complexities into the software development process (Saran 2004). Physical distance between customers and development operations, and different cultural ways of working can lead to software products being constructed in a way that does not provide maximum value to end-users. If sufficient management and QA procedures are not put in place, this can cost vendors millions of dollars (Saran 2004). Raval (1999) suggests a set of underlying principles for successful offshore software execution. These include keeping the purpose in focus, seeking relevant information and understanding the overseas environment, effectively understanding and

managing risk, preparing the organisation for the initiative, and preparing the offshore organisation for optimal performance.

2.3.3 Software peripherals

Professional services are often critical peripheral activities that surround a software solution. Often product failure and failed customer projects are strongly linked to a lack of related service expertise, poor execution of technical activities and an overall poor quality of service delivery (Cooper & de Brentani 2002). Professional services can directly influence the business value that is created from a software solution, while also having a direct positive impact on a vendor's long-term product sales, revenue growth, profit levels and customer loyalty (Esposito 2006).

The core focus of implementing professional services should be on getting the software operational so that a customer starts achieving associated business benefits and a software vendor can reap product licence revenues (Downey 2001). There can be a great deal of work involved in understanding the end-user organisation business process and reengineering these so that they can be administered using industry standard software functionality. Implementing standard software can be hugely complex and often involves making modifications to base vendor products.

Software implementation service skills should include business acumen, project management, people skills, technical competency with the product, the ability to understand and empathise with a customer and a pragmatic solution focused approach to problem solving (VanDoren 2000). This work requires clearly understanding a customer's business activities and associated objectives (Stewart et al. 1998; Cooper & de Brentani 2002). This provides a focus and terms of reference for any software solution implementation; while the role involves working through the technical aspects of setting up and configuring a software system. If unique customer requirements exist, this is likely

to include the development of custom software components to run on top of the base software product (Miller 1999). The ability to engage and manage stakeholders at multiple levels in an organisation is also a critical part of implementing a software solution (Skaatesa & Seppänenb 2002).

There are a number of models to assist software solution providers in the construction of implementation services offerings. Chang and Birkett's (2004) professional competence framework offers a model that emphasises organisational context, practitioner skills and activity execution as essential in the design of robust a professional services propositions. Conversely, Stewart et al.'s (1998) framework highlights the degree of customisation, the degree of labour intensity and the degree of customer contact/interaction as key attributes that a vendor should consider.

Once a software system is operational, there is still a substantial cost involved in maintaining it. Over a software system's lifetime, the largest proportion of practitioner effort and cost is expended in maintenance (Glass 1998). The desire to achieve business benefits from software functional enhancements constitutes the main reason for undertaking software maintenance: 64% of the total maintenance effort for ERP systems (Pui Ng 2001). Software system training and end-user support can also comprise a sizable part of software maintenance effort. However, maintenance activity can also be driven by non-business benefits. Enforced upgrades of software by vendors are common. Upgrades that are incompatible with other software solution components can result in frozen functionality and other system defects (Voas 1998). Unforeseen impacts can then trigger a whole host of knock-on and un-planned maintenance activities to get a software system operationally back to where it was previously.

Ensuring that a software maintenance agreement delivers value is critical for both customers and vendors. Multi-dimensional maintenance agreements that can satisfy the needs of all stakeholders are required (Gable et al. 2004; Kaplan 2004). The business benefits of upgrades should always be qualified and the number of times the vendor will be needed to resolve issues should be planned (Patin 2002; Fontana 2005). A value focus is imperative from vendors as customers increasingly become disgruntled with the ever-increasing cost of software maintenance (Krishnan et al. 2004). Microsoft's 'Software Assurance' maintenance and upgrade programme has been criticised as being complex, expense and providing poor quality (Kirk 2006). A slow follow-up to customer-generated issues is a widespread complaint. Many customers are analysing the business value of continuing to pay for software maintenance contracts and often concluding that they would be no worse off without this expense. This should be considered a serious issue by vendors who currently generate sizable proportions of the revenue base from these activities.

There are a number of reference models that vendor's can use to manage and plan their software maintenance and support operations. Although over thirty years old, Swanson's (1976) software maintenance classifications are still useful reference for drawing out the value-adding perspective of software maintenance and support. These classifications comprise of corrective, adaptive and perfective activities. As an alternative, Chapin (2000) offers a lower level of granularity in his software maintenance taxonomy to assist software stakeholders in a better understanding of the exact drivers of these activities. Four broad maintenance categories are subsequently broken down into twelve classifications. Software 'support interface' comprises training, consultative and evaluative activities. 'Documentation' is either reformative or updative. Changes to 'software properties' are groomative, preventative, performance improving or adaptive. Changes to software functionality are reductive, corrective or enhancive to 'business

rules'. In 2005, April et al. (2005) fused together literature on software maintenance with practitioners' experience and international standards to produce a Software Maintenance Maturity Model (SMmm). This model moves beyond just providing a taxonomy of different maintenance tasks and offers a sequential framework of maintenance progression activities that a software solution should undergo to achieve a virtuous cycle of value creation. The SMmm model is based upon and complements the computer software CMM, which has become an industry standard for software quality since its establishment in 1991 (Paulk 1993; SEI 2006). The model also consists of five maturity levels. In the first instance, a newly implemented software system will require 'performed' *ad hoc* maintenance tasks as it seeks to bed-in and achieve stability. The software will then move beyond this to a more 'managed' maintenance process of change requests. As the system matures, the activities progress through the three higher maintenance maturity levels of 'established; 'predictable' and 'optimising' activities.

2.3.4 Dynamic evolving IS industry

The consolidation of the software industry is being driven by technology convergence, standardisation and commoditisation (Sheth & Sisodia 2002). Consolidation occurs as there become too many suppliers to provide industry *de facto* solutions (Faletra 2004). Large, dominant vendors are integrating existing solutions into single integrated product suites, which is edging small specialists out of the market. Most industries move through a sequential life cycle that includes four stages of consolidation (Deans et al. 2002). The first stage begins with start-up organisations; in the second stage, empires emerge as market leaders acquire competitors; thirdly, companies focus on outgrowing rivals; and finally, the remaining organisations concentrate on defending their leading positions. There is some evidence to suggest that the industry is at a point of 'inflection' (Mills 2005), with the market is simultaneously expanding and contracting. While it is probable that certain areas of the sector will consolidate further (Economist 2004; 2005a), it is also likely that new firms, products and services will continue to emerge in parallel (Kim et al.

2000). At the same time as high-profile mergers, there is a continuous influx of new, often unnoticed, innovative software organisations that increase the sector's overall size every year. The combination of established market leaders and new start-ups results in a complex industry landscape that encompasses both oligopolistic and monopolistic characteristics at the same time (Sheth & Sisodia 2002).

As the industry matures and the market becomes saturated, mergers and acquisitions have been a way for software businesses to continue their growth, acquire new customers and enter new markets (Rosenbush 2004; Kerstetter & Hamm 2004). There is debate about whether the exponential market growth followed by consolidation has led to efficiencies where the best technological software solutions prevail, or whether confusion and incompatibility has been the major outcome (Linderholm & Sanborn 2001). M&A is notoriously difficult to execute successfully and hasty attempts to grow can have the reverse effect resulting in value destruction instead of synergy realisation (Sheth & Sisodia 2002). Many software industry executives are perplexed by whether they should be looking for appropriate acquisitions, or instead should be looking for a suitable buyer. There is a real '*danger, as there could be "mass confusion between activity and progress" as firms leap into "awkward combinations" for fear of being left behind*' (Economist 2005b p47).

The software sector as part of the technology industry is a dynamic and disruptive market that is characterised by new entrants and changes in market leaders. Continuous advances in technology fuel ever-shorter product life cycles, which in turn result in widespread changes in offerings to the market (Nault & Vandenbosch 2000). It is difficult to make accurate forecasts on the future of disruptive high-tech markets and market players should constantly reassess their product mix (Borés et al. 2003). To compete in a market with environmental turbulence, it is important to understand as much as possible about

potential factors that drive uncertainty and change. A framework to identify and assess this instability considers richness of technology, speed of change, intensity of competition, customers prejudices, influence from governments and pressures groups, and overall changeability of the market environment as core contributing factors (Ansoff 1987). It is imperative to respond swiftly to technological change (Nault & Vandenbosch 2000) and stay close to customers (Bower and Christensen 1995).

The successful utilisation of software solutions and the creation of customer business benefits directly drive the future growth of the new software products and the industry as a whole (Karahanna et al. 1999). One particular characteristic of the software industry is the software adoption 'herd behaviour', in which customers ignore rational decision and evaluation processes and instead imitate the actions of peers (Prasad & Prasad 2002; Xiaotong 2004). Excitement about the next technology fad and fear of being left behind appear to be the main explanations for this. In this wave of enthusiasm, new information technology can be purchased by many organisations, but subsequently are only implemented by a small proportion of these. This concept is referred to as an assimilation gap phenomenon (Fichman & Kemerer 1999) and needs to be carefully qualified.

A vendor who can reduce software product complexity and provide simplified software solutions has an opportunity to achieve competitive advantage. The landscape that mega-vendors have created is one characterised by excessively complex and often convoluted software products. As vendors have raced to include an ever-increasing number of features into their products, the objective of achieving improvements in efficiency and effectiveness through digitisation of business processes have often become lost (Sadagopan 2006). Although vendors continue to market the benefits of their integrated portfolio COTS-AS solutions, questionable return on investment (ROI) and a loss of business agility is slowly eroding their value proposition to customers (Keller 2006b).

Software businesses will need to be more proactive in addressing this imbalance between their offerings and the viability of these as value propositions to their customer (Keller 2004b).

The software contribution over the last twenty years has been to digitise the back-office functions of organisations to improve their efficiency and effectiveness. The software market then moved further up the value chain to building a 'front-end' business engine for its 'back-office' software products (Mullin 2001). More recently, e-Commerce has enabled many new business interactions with customers and suppliers.

Through the development of new software value propositions, there is an opportunity for vendors to achieve competitive advantage and growth. Boundaries between adjacent markets and related technologies become blurred. The term 'applistructure' has been coined to describe products that merge together software and underlying technology infrastructure to provide a simplified overall software solution (Keller 2005d). The convergence of software products and services is also occurring (Roeding et al. 1999). 'Componentware' is the concept of COTS software services being 'productised' to make the implementation of software solutions simpler, while the 'servicisation of software products' is being taken to market through offerings such as SaaS, on-demand and business process outsourcing (BPO). The threats to traditional software products and services are real, but opportunities to redefine the software competitive landscape also exist (Brooks 2005).

Newly industrialising countries are likely to continue to emerge as players in the IT and software market. The dominance of North American and European vendors is now being challenged by the arrival of Asian and Eastern European software development organisations (Jones 1994). India and China, in particular, have managed to establish

themselves as serious contenders. Having entered the IT sector providing value-for-money professional services and developing custom-built application software, India specifically has increased its footprint in the software product and packages market (Arora et al. 2001).

2.3.5 Software market idiosyncrasies

The review of literature has revealed that the software market is characterised by a number of nuances and idiosyncrasies. While these factors do not fundamentally change what is involved in running a business, it is suggested that they do need to be understood and factored into a manager's perspective of a software business.

Releasing new versions of software product and marketing the benefits and/or superior features of these is a widespread practice in the software industry. Many vendors are motivated by the opportunity to maximise short-term revenues with limited investment in new product. They play on the fear that customers have of being left behind in a dynamic software sector where products are continually changing. Offering customer discounts to upgrade to newer versions of a software product is a low cost way of generating revenue that would otherwise not be realised (Fudenberg & Tirole 1998). However, vendors should be careful not to excessively exploit customers using a product versioning and upgrades strategy. A long-term tactic of providing superfluous software versions and upgrades may lead to negative customer social welfare and market inefficiencies (Ellison & Fudenberg 2000). Although there may be incentives for a vendor to force new software versions and upgrades on customers, this can be very detrimental. From a customer perspective, these are costly to purchase, learn and implement, and often the benefits over a previous version of the software product are minimal. Upsetting customers and creating market inefficiencies, in time, will put a vendor in a disadvantaged competitive position.

One significant problem for vendors is the illegal use or piracy of software products. The unlicensed use of product costs vendors millions of dollars of unrealised revenues (Schelp 2006). License enforcement and the ability to control who uses software product is of the utmost importance. These lost revenues are split between software users who purposely acquire and use unlicensed products and those customers who through carelessness do not accurately track how many copies of a software product they are using. The problem is exacerbated because there is no simple, watertight industry solution for managing software license compliance. A survey revealed that 72% of organisations manually monitor software licence compliance or they do not monitor it at all (SIIA 2006). It is suggested that vendors should incorporate some form of licence enforcement mechanism to control the use of their products. Internet-based software product activation and network licensing are the easiest to introduce (Wohl 2004). More conventional methods such as dongles and serial numbers can be highly effective even if they somewhat cumbersome for customers.

Customer retention through customer lock-in is widespread in the software industry. When a customer has selected and invested in a software solution from a vendor, the costs associated with switching to another vendor's product can be substantial. Customer switching costs may be incurred from the physical transactional of changing products, from learning how to use a new product, or from the overhead of terminating an old and starting a new vendor relationship (Klemperer 1987). Chen and Hitt (2002) offer a cause and effect model that can be used to analyse input variables that influence resulting customer switching and attrition decisions within the software market. Independent input variables include: customer product usage, product quality and breadth, ease of use, maintenance costs, and benefits in preserving the status quo.

A vendor can use customer switching costs and lock-in as a basis for maximising revenues and achieving competitive advantage. In the dynamic software market, it is possible for a first mover to capture early adopters and lock-out other competing technologies and suppliers at an early stage of the market evolution (Arthur 1989). The term 'create-capture-keep' has been coined to describe the strategy of creating new innovative product, capturing customers through the ability of IT to deliver value and keeping them through the deterrent of switching costs (Mata et al. 1995; Fenny & Ives 1990; Clemons 1986). However, a vendor strategy of customer retention through customer lock-in is unlikely to be successful in the long-term. There is a likelihood that customers will anticipate and avoid the risk of being captured by a supplier, and those suppliers that do exploit their customers will only damage their reputation in the process of doing so (Hopper 1990). Malone et al. (1989) extends this further and suggests that companies that try to lock-in customers will actually trigger their defection. Alderman's (1999) view that the ability to maintain competitive advantage from technology advances is now limited and therefore a vendor should lock customers in and squeeze revenues out of them is rejected. Although this approach may work for vendors in the short-term, the lack of value creation for customers is unlikely to make this sustainable over longer periods.

2.3.6 Limitations of IS Research

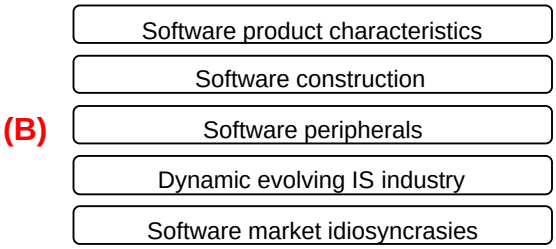
The output of this review was very much in line with a debate about the maturity and focus of the IS discipline that was discovered. This debate by IS academics covered the relevance and contribution of IS research to practice and the field's own scholarly progression. Criticisms of IS research have suggested that it is often fragile and incomplete, IS artefacts have comprised of an assortment of delicate and disconnected components (Orlikowski & Iacono 2001) and that it is too diverse and does not possess any distinguishing theories of its own (King & Lyytinen 2004). This general appraisal of IS research could also be concluded about the IS discipline's contribution to the running

of a COTS application software business. While the IS Research review identified many factors of relevance to managers running software businesses, it became evident that the IS discipline did not appear to offer a big picture insight into running a software business.

Gosain et al. (1997) suggest that a disconnect exists between the topics that IS academics investigate and those that interest IS practitioners. Orlikowski and Iacono (2001) argue that much of our understanding about IS has been proliferated by industry journalists and those with commercial interests. McCubbrey (2003) are of the view that more time invested in interfacing with practitioners and understanding the business perspective of IS should provide a sharper focus on the relevant core issues of the moment. This is supported by Hirschheim and Klein (2003) who suggest that researchers need to concentrate on understanding IS stakeholders challenges and setting IS research priorities accordingly. A persistent focus on investigating the salient issues of the industry could result in the production of rich IS models (Lee & Barua 1997).

An ongoing challenge for IS researchers is how to advance the development of IS-centric theories that account for IS specific phenomena, rather than always just extending existing frameworks from more mature reference disciplines (Benbasat & Zmud 2003; Weber 2003). It is vital that these models can cross-interface and are able to evolve in the dynamic landscape that constitutes the IS industry (Orlikowski & Iacono 2000). The advent of powerful generic IS theories that explain IS phenomena, such as a COTS application software business, could deliver a fundamental improvement in the guidance that is available to industry practitioners.

2.3.7 Summary



A review of Information Systems and Management Science literature reveals much of importance to managers running software businesses. Five key factors of importance to running a software business were identified. First, software products can be complex and opaque. The characteristics of software products that a vendor wishes to take to market are therefore important to understand. Second, the process of 'software construction' involves choosing base technologies, ensuring appropriate QA and protecting IP. Third, there is a whole range of 'software peripherals' beyond pure software products. Peripheral activities such as professional services may be an essential part of delivering customers a value-adding software solution. Fourth, software business operate in a 'dynamic evolving IS industry'. Accommodating forces and trends of a disruptive market into business plans and operations is likely to be important. Fifth, there are a number of software market idiosyncrasies that software business need to familiarise themselves with. Particular nuances include software versioning, illegal software use and customer lock-in.

On commencement of the literature review, I hoped that I would locate specific guidance on how to run a COTS application software SME business. After examination of the Strategy Management, Information Systems and Management Science disciplines, this was not the case. While much of relevance had been identified, specific models and references to contextualise and frame the overall research problem had not been discovered. Instead, the broad range of topics of relevance to running a software business that I had located were leading me towards the examination of a number of other

literature sources. This was aligned with Ahmed and Fernando's (2007; 2010) research into software product businesses that highlighted that this is an inter-disciplinary phenomenon. It was at this stage in the literature review that I decided that a much wider search of management research disciplines was appropriate. Rather than being a narrowly focused literature review that looked in-depth at a small number of very pertinent reference models, the approach would be to cross-fertilise concepts from a range of different sources. This would provide a contextual backdrop that would act as a platform of ideas to investigate the research problem of how to run a COTS application software business.

2.4 Practitioner Reference Sources (Literature Cluster C)

As a software industry professional myself, I was aware that there are numerous practitioner reference sources that discuss the software industry. After a review of the Strategy Management, Information Systems and Management Science disciplines failed to reveal an overarching framework for running a software business, an examination of practitioner reference sources to see what contribution they could make seemed appropriate. IS trade press, industry analyst reports and IS practitioner research, were specifically targeted. These provide a vast repository of publications, reports, news stories, commentary, opinions, discussion and analysis about the software industry and the software businesses that operate within it. It was hoped that these sources would provide valuable insights into the running of a COTS application software business. While the majority of the concepts and themes offered from these sources are not underpinned with the underlying academic rigour associated with research disciplines, nerveless they do represent a cross-section of the latest innovative thinking about the industry.

Four key areas of relevance to running a software business have been identified across practitioner reference sources. These are categorised as 'software business models',

‘software business financials’, ‘software business imperfections’ and ‘alternative software business models’.

2.4.1 Software business models

Traditionally, software vendor business models have fitted into one of three classifications: a software product business, a software services business, or a software solutions hybrid business combining both software product and services (Cusumano 2004a). A software product business can take different forms and there are a number of decisions that a software vendor needs to make about their business and associated business model.

A pure software product business can be commercially attractive and vendors of this type can be highly profitable. Standardising a software product and making high volume sales is a fundamental of such a model (Cusumano 2004a; SMS 2004). Software has a near zero marginal cost of producing an additional product unit and it is an easily distributable information good. Once a break-even point has been reached, each additional sale can be pure profit. The attraction of high rewards fuels the race for market leadership in a sector where there are relatively low barriers to entry (Roeding et al. 1999). However, it is not always possible to create a fully standardised product that is attractive to all customers, and making high volumes of new sales can be elusive. It is critical that a software vendor is not carried away with exciting product R&D. Building up a formidable sales force that actually sells the software product that the software vendor possesses today is a top priority (SMS 2004).

A software services business can offer significant revenue and profit opportunities. Many software vendors orientate their business models to the provision of professional services that bridge a gap between customer requirements and base software products (Esposito 2006). SAP and PeopleSoft are examples of software mega-vendors who have orientated

their businesses away from pure product. The majority of their revenue is now achieved from services and maintenance (Cusumano 2004a). However, there are differing views on the role of professional services as part of a product business (Brooks 2005). One perspective is that services are a necessary evil, in order to ensure that software product is successfully implemented. Another is that service revenues provide an opportunity for software vendors to grow, subsequent to reduced profit margins in the software product business following increased market commoditisation. An alternative view is that services can be leveraged to become an enabler for growing the core product business and be a profitable entity in their own right. Whatever the viewpoint is, it is suggested that software vendors need to be clear about their primary business model.

A software solutions hybrid has a mix of both product specific and services capabilities. A solutions hybrid business can be used by a software vendor to balance its risk profile in the dynamic software market. When a software product business exhausts the most easily achievable sales opportunities, the ability to make new sales of software product becomes increasingly more difficult. At this point, the balance of a company's revenues is likely to shift towards maintenance and related services that are easier to sell (Cusumano 2004a). A variation of a hybrid business involves the use of partnerships with third party organisations. SAP utilise business partners to enable the fast and cost-effective implementation of their software products and to ensure that adequate focus is given to customer satisfaction (SAP 2006d). IBM's PartnerWorld programme assists third party organisations in attaining the capabilities needed to sell, implement and support IBM products (IBM 2006b). The strategy of building up a pool of partners that can provide IBM software solutions helps grow the overall market for IBM software products.

2.4.2 Software business financials

A software business' financial metrics are a key part of a software business model. Dependent on exactly what business a software vendor is in, such as 'product' *versus*

'services', or 'niche' *versus* 'mainstream', these financial dynamics can vary significantly from vendor to vendor. Table 2a provides a point of reference for analysing the variation of financial ratios across different business scenarios. Financial ratios for operating costs, revenues and profits are included. A high and low figure is provided for each metric to demonstrate the variation that exists across different vendors in the same type of business. Separate figures are provided for a software product businesses and software services businesses to illustrate fundamental financial model differences between those different business types.

Table 2a: Software Business Financial Metrics and Ratios

Business Model Metric	**Software Product Business**	**Software Services Business**
Cost of Revenue	5%-20%	40%-80%
Gross Margins	40%-95%	20%-60%
R&D	5%-25%	< 1%
Sales & Marketing	25%-40%	5%-20%
Profit (Pre-Tax)	Up to 40%+	Up to 30%

Source: Adapted from Cusumano (2004a) and Keller (2005b).

A software product business can be highly profitable. Up to 40% pre-tax profits can be achieved, but there is a high fixed cost investment that needs to be made in R&D (up to 25% of total costs) and sales and marketing (up to 40%) before these returns can be realised. Innovative software product business' R&D costs may be even higher. Beyond the break-even threshold, software product businesses have the potential to be much more profitable than corresponding software services businesses. Product businesses can grow revenues without needing to add headcount, which services companies cannot achieve. The percentage of a product business' revenues that come from sales of new software licences can vary dramatically. For software vendors experiencing rapid growth, this can be as much as 99%, but as the product business matures and new sales opportunities are

exhausted, this percentage will start to decline (Cusumano 2004a). Understandably, market analysts tend to place a higher value upon software vendors that realise a high percentage of their revenues from new software licence sales.

A software services business is generally less profitable than a product business. A services business is less easy to scale, but is arguably less risky in terms of securing revenues when new sales are limited (Cusumano 2004a). Growth for services and hybrid companies is likely to be directly related to growing the number of staff. However, the services model does have the advantage of being able to secure recurring revenues. After purchasing a given software product, customers are generally locked into purchasing ongoing related services and maintenance to support their continued use of a software solution.

Financial metrics for both of the different software business types, 'product' and 'services', do also themselves have a large variation of values. The software industry's mega-vendors are generally more efficient and more profitable. Smaller and medium sized software vendors typically spend significantly more on sales and marketing, and general and administrative costs; but achieve lower revenues (Keller 2006a). These inefficiencies present an opportunity and a threat. There is an impetus for smaller and medium sized software vendors to create alternative business models, as there is a chance that they will be absorbed in a wave of continued industry consolidation if they stay still. The revenue streams for a typical software vendor are often proportionally equal across the three areas of new product licences; maintenance and support and related IT services. As an example, in 2005, 35% of Oracle's revenue came from new software licences, 45% came from updates and support and 20% came from associated services (Oracle 2005). However, variations in this model do exist and a software vendor's maturity and market offering are two factors that may alter the make-up of its revenue streams. Investment in

R&D is essential to ensure that software vendors bring new and quality software products to a dynamic market. Vendor expenditure on R&D in the software sector ranges from 2%-17% of total gross business costs (OECD 2002b). At the top-end, Microsoft invested US$6.18 billion, 15.5% of its total expenditure in 2005, into R&D activities (Microsoft 2005).

2.4.3 Software business imperfections

Some industry analysts claim that the historic model for buying and selling software is fundamentally flawed (Keller 2005b; Keller 2006a). Renowned management guru Prahalad (2006 p1) argues that '*most software vendors have gained a reputation of being slow moving and intractable in their business methods*'. There are a number of reasons attributed: customers are not satisfied with the industry software solutions; the customer value proposition no longer stacks up; software licensing and pricing is highly restrictive; and new alternatives are undermining traditional software paradigms. To claim that software vendor business models are broken is probably an overstatement. However, there is no unanimous view as to what future software business models should look like (Keller 2006a). Many established software vendors are attempting to cement their positions through improved efficiency and economies of scale. In parallel, a new generation of software solution providers are attempting to leap frog today's industry leaders by introducing new, innovative, value-driven software business models (Keller 2005b).

Revolutionary new software product ideas of today become the commodities of tomorrow (Keller 2005b). Therefore, the customer business value of a software product is likely to decline over time as the product's use becomes more widespread. Interestingly, very few software vendors and software buyers have changed the way of selling or buying software products as a result of this changing software value proposition.

Market saturation is directly increasing the power of the industry's customers and demands for better value for money are increasing (Brooks 2005). Over the last fifteen years, there has been a constant pipeline of software customers purchasing new software solutions to achieve advantage. This has now reached a point where the software market has become more saturated as most potential customers have implemented a portfolio of back-office software systems. Ten years ago, software vendors were typically able to achieve two-thirds of all revenue from new software product sales; this has now fallen to around a third (Keller 2006a). As a result, many software vendors have turned to maintenance as a major revenue stream. A widely propagated view within the industry is that this software market saturation and struggle for revenues amongst software vendors is just a natural part of the industry maturing. The endpoint would be a consolidated market comprising of a several mega-vendors such as Oracle and SAP, and Microsoft and IBM (Keller 2006a). However, a criticism of these mega-vendors is that their COTS application software products are excessively complex to implement, are too generic in ability to support unique business processes and do not offer value for money. It remains to be seen whether a new generation of software vendors, many of which are currently struggling to reach critical mass, will break free and revolutionise the current software solution landscape and associated supply-side business models.

A number of antiquated business practices in the software market represent opportunities for software vendors to revolutionise the way software is bought and sold (Keller 2005c). It is an almost accepted norm in the software industry that new versions of software products will have quirks and bugs, which will only be fixed in time as further software service packs become available. Customers generally have no recall to software suppliers for the impact or cost that these quality issues cause to their businesses (Keller 2005c). In most industries, there are third party organisations that modify and service industry standard products. With software products, many software vendors invalidate warranties

if the base software product is touched by anyone other than themselves or a certified business partner (Keller 2005c). In most other industries, the concept of forced aggressive product upgrades from suppliers does not exist. In the software market, customers have to endure regular upgrades of new products, pushed upon them by software suppliers (Keller 2005c). In addition, many new software products are also often specifically designed not to work with older software platform products, to coerce customers into upgrading whole portfolios of products (Pollack 1999). In most industries, when you purchase a product you own it outright and are entitled to do what you want with it, including reselling it. This is not the case in the software market.

2.4.4 Alternative software business models

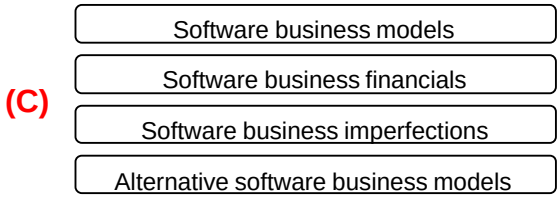
Historically, the software market has been characterised by software vendors pushing their software product and services to market (Keller 2006a). Software customers are required to make large upfront investments, with little recall for product shortfalls or implementation complexities. Any cost for resolving software system implementation problems are generally left to customers to pay for, while software vendors are rewarded with additional revenues for undertaking this 'additional' work. These traditional models have arguably favoured software vendors, while placing customer interests and the creation of customer value as a secondary factor. As the industry continues to evolve, alternative software business models that focus on the customer value proposition are gaining prevalence. Their emergence is a threat to the traditional perpetual licence model used as a basis for many based software businesses.

SaaS provides an attractive value proposition for software customers and a viable business model for software vendors. With SaaS, a customer rents a full software solution as a service or utility for a regular fee (Choudhary et al. 1998). There no large upfront investment required in software, hardware and consultants (Dewire 2000). Customers automatically benefit from new upgraded versions of the application software solutions as

they become available (Smail 2000). The SaaS industry pioneer Salesforce.com continues to lead the way in constructing a new business model for selling software solutions (Montalbano 2005). The long-term acceptance of SaaS in the software market will be closely linked to its viability as an alternative to traditional software business models. To gain significant market share, SaaS will need to prove that it is a better value proposition than the established industry standard software in-house option (Bennett & Timbrell 2000).

OSS turns the software business model upside down. There are numerous OSS application software products available (SourceForge 2006). In contrast to a traditional software business, a vendor does not sell product, instead it charges a fee for providing a range of associated products and services, such as product documentation, consultancy to better configure the product and assistant supporting the product. Advocates of OSS argue that it presents customers with a better value proposition than traditional software product licences and that over time it will replace this as the leading application software solutions model (Dix 2006). However, this is unlikely, as there are always likely to be commercial opportunities for vendors to create and sell software products in specialist areas (Krill 2006).

2.4.5 Summary



A review of Practitioner Reference Sources literature reveals much of importance to managers running software businesses. While practitioner reference sources do not have the rigour and robustness found in academia, the review of this area has proved to be a valuable source of topics and concepts relevant to running a software business. Four key factors of importance to running a software business were identified. First, having a

software business model that provides clarity about what business a vendor is in is important. Second, software business financials will need to be commercially viable. Third, there are numerous software business imperfections that exist in the software market; being aware and accounting for these is critical. Fourth, the traditional software business model is evolving. Alternative software business models may fundamentally change the parameters of running a software business.

In addition to the four specifics identified, the review of the practitioner reference sources also highlighted a number of other factors previously identified by the review of the Strategy Management, Information Systems and Management Science. These included software product characteristics, software product development, software engineering, software implementation, strategy, and mergers and acquisitions. Hence, the relevance of these topics for the research was reinforced. However, while the practitioner reference sources have provided some very specific models, these have been lacking in the conceptual density and sufficient breadth that would be required by a manager running an overall software business.

2.5 Other Generic Reference Disciplines (Literature Cluster D)

The review of literature from Strategy Management, Information Systems, Management Science and Practitioner Reference Sources led to a number topics being identified that are grounded in other generic reference disciplines beyond these domains. Specifically, seven key areas of relevance to running a software business have been identified where substantial generic literature exists. These are categorised as ‘innovation and product development’, ‘innovative [software] product’, ‘market segmentation and positioning’, ‘market communications’, ‘sales and distribution’, ‘licensing and pricing’ and ‘network externalities’. The academic disciplines of Innovation, Product Development, Marketing

and Sales, and Economics provide a range of models and theories for these areas. While these fields are not primarily focused or orientated around software businesses, they do provide insight into a number of aspects of running a software business. Therefore, a broad high-level review of these fields was deemed appropriate. This section sequentially discusses the seven areas listed above.

2.5.1 Innovation and product development

A strategy of innovation and new product development (NPD) is a means for vendors to achieve growth and competitive advantage. Businesses can diversify, adjust and even reposition themselves to match changing market and technical conditions (Brown & Eisenhardt 1995; Schilling & Hill 1998). IBM's business model since the company's inception has been one of innovation (IBM 2004).

As innovation is an intrinsically volatile and dynamic activity (Cheng & van de Ven 1996), researchers have produced a structured procedural approach to manage its inherent uncertainties and facilitate the success of new business ideas (Kim & Mauborgne 2000). In a conventional innovation process, organisations typically search for opportunities, then analyse, design, manufacture, market and distribute these themselves (Chesbrough 2003). With an open innovation model, companies attempt to look beyond the boundaries of their own firm and aim to generate ideas in the overarching environment that constitutes their customer market.

The long-established product life cycle (PLC) identifies four phases of a product's life. Products sequentially move through stages of birth, growth, maturity and finally death (Tellis & Merle 1981). However, in a dynamic and volatile software market, product life cycles have compacted and product markets continue to fragment into ever-smaller niches (Schilling et al. 1998). Guidance for NPD in the high-tech sector is provided by

Bhattacharya and Krishnan's (1998) real-time definition model, which allows a firm to iteratively adjust its product design process to the market and competitive environment.

2.5.2 Innovative software products

Developing innovative software products can be an enabler of commercial success. If a market opportunity can be transformed into a commercially successful product, the rewards can be substantial (Krishnan & Ulrich 2001). Tabrizi and Walleigh (1997) suggest that it is essential that software businesses have a comprehensive understanding of their market, have a clear product roadmap and have a product strategy that is flawless with no weaknesses for competitors to exploit.

There is an important distinction here between 'sustaining' innovation, which makes an existing product perform better, and 'disruptive' innovation, which can create a completely new value curve and market (Christensen & Overdorf 2000; Lyytinen & Rose 2003). A key objective of high-value disruptive innovation is to focus on challenging market norms and create new benefits to customers that previously did not exist (Hamel 1999). Disruptive innovation, although disrupting competitors, should not require customers to restructure their lives. A new idea is more likely to succeed if can be easily integrated into a business' current activities, while offering new value through its simplicity, convenience or lower cost (Christensen 2002). Key attributes for innovation include alertness to novelty and differentiation, an appreciation of different contexts, an awareness of multiple perspectives and an orientation to the current environment (Sternberg 2000). In addition, focus, persistence, listening, blending of ideas and iterative learning are all important foundation practices of IT innovation (Denning 2004).

Downes and Mui (2000) recommend twelve principles for developing new products in the dynamic and uncertain software market. These include: ensuring continuity for the customer not yourself; destroying and reengineering the value chain; cannibalising

existing markets; managing a portfolio of opportunities; and structuring initiatives as joint ventures to harness industry expertise. An alternative perspective comes from Alderman (2000), who challenges the value of large-scale product development as a strategy and suggests that vendors should concentrate on locking-in existing customers, charging them for upgrades regularly and undertaking this very efficiently. Although the logic for short-term revenue and growth makes sense, it is questionable if this approach is sustainable for creating long-term competitive advantage. A more balanced perspective comes from Sommer (2006), who acknowledges the value that a NPD strategy can deliver, but unsympathetically points out that vendors need to move away from a self-absorbed philosophy of product development that has often previously ignored the outside world of commercial realities.

Vendors have had a mixed record with the success of new software products. Many innovative software product initiatives never achieve commercial success in the marketplace. The ability to connect market demand and technology capability is critical (Cooper & Kleinschmidt 1987). Thus, widespread adoption and utilisation of new products is paramount.

2.5.3 Market segmentation and positioning

How effectively a software business segments its market and positions within it can directly influence its level of competitive advantage and long-term commercial success (Geisman 2006a).

Four prime types of software vendor are observed: technology driven, sales driven, market driven and financial driven (Chapman 2005). Chapman (2005) identifies various factors that vendors can get wrong:

- Putting a product to market that does not meet its claimed feature set
- Mistakes in pricing levels
- The inability to conceptualise a software product coherently

- The indirect creation of conflict of software offerings across adjoining market segments

Various software industry experts and scholars (Dver 2003; Hasted 2005; Sink 2006; Walsh 2006) provide guidance on how to identify and target attractive market segments and strategically position to compete within these.

Vendors that are able to combine comprehensive knowledge of customers with the ability to provide customised software solutions are more likely to achieve competitive advantage and leadership in the market (Treacy & Wiersema 1993). The most valued customers should be identified and then these should be relentlessly courted and served by vendors (Roeding et al. 1999; Davenport et al. 2001). To achieve this closeness to customers, vendors need to be able to segment and target niche areas accurately. Rather than focusing on the software product, they should religiously make customer-centricity a focal point of their business (Hagel III & Rayport 1997).

Walsh (2006) recommends that software businesses need to ask themselves who their competitors are and what are the comparative software product strengths and weaknesses. Some competitors will be targeting these segments as the core of their business, while others will be 'accidental' competitors as they have entered the sector as part of a differentiation or integration initiative (Sink 2006). Strategies for competing with each competitor profile need to be tailored appropriately.

2.5.4 Market Communications

A strong, focused marketing strategy that communicates a clear value proposition to potential customers is an imperative (Roeding et al. 1999). A software value offering might emphasise: technical superiority; a low price; appeal for high-end users; a low risk; or a unique application (Beard & Easingwood 1999). The uptake of new products, financial implications, customer feedback and exposure to competitive attack are all

issues that need to be considered as part of the overall software marketing strategy decision making process (Stone 1985). It is important that marketing and advertising does not focus purely on the software product, but instead focuses on how it can create real tangible business value to a customer (Hogan 2006).

Ideally, a vendor will build a strong brand and reputation, cultivate good relations with market analysts and actively push strategically targeted market signals into the software eco-system. A strong brand is valuable asset (Walsh 2006). Research has shown that software customers rank a supplier brand highly in their purchase decision process (Tam & Hui 2001). A well-recognised and well-regarded brand enables a vendor to charge a premium for their software products (Gallaugher & Wang 2002). Above all else, it is critical that a brand is meaningful and easy to understand (Dver 2003). There is a direct correlation between a vendor's reputation and its revenues (Herbig & Milewicz 1996). As a vendor's reputation increases, so does its ability to make sales and achieve competitive advantage (Herbig & Milewicz 1995). Establishing credibility early on is paramount (Hasted 2005). A vendor's ability to keep promises made about future software product functions and capabilities seems to be of highest importance to customers (Hoxmeier 2000). Conversely, a vendor's loss of reputation and credibility will result in lost customers (Cusumano 2004a).

Product preannouncements are widely used in the attempt to strengthen a vendor's product positioning in the software market. Benefits from preannouncements may include the favourable positioning of a software new product, the forestalling of customer purchases of rival products and the signalling of a commitment to new technology (Herbig & Milewicz 1996). Much guidance is available to assist vendors developing market signalling strategies and the effects that a signal will have (Eliashberg & Robertson 1988). Blooma and Reveb (1990) offer a number of different market signalling

strategy objectives and characterisations: blowing the whistle for everyone to stop, redefining the market, quietly fixing existing problems, or full speed ahead. However, many product preannouncements in the software market have left behind a haze of marketing rhetoric where it is unclear what form the resulting product will eventually take (Hoxmeier 2000). Reputation and trust can be damaged when vendors set expectations for unrealistic delivery dates, or worse still when they do not intend to release a product as promised (Foster 1997).

2.5.5 Sales and distribution

There are many cases of vendors who have unsuccessful quality products, as they have not managed to master sales activities (Chapman 2005). A vendor that can excel at marketing and selling their software offering is more likely to achieve competitive advantage and dominance in the software sector (Roeding et al. 1999). A sales plan needs to include those products to be marketed, customers to be targeted and the process of exactly how product will be pushed into the market to maximise sales (Stone 1985).

Chapman (2005) suggests four fundamental software sales strategies. These are characterised as: 'Godzilla', 'sweep the board', 'divide and enter' and 'defender'. All of these sales strategies have the implicit common objective of securing territory in the software market that a vendor can call its own. A persistent focus on how a software offering can generate business value to a customer needs to be a sales foundation (Hasted 2005). Chapman's (2005) psychology on customer-orientated sales translates into a sequential set of sales activities: learn about a customer's mission and objectives; analyse their challenges; and suggest software solutions that solve their problems.

A vendor needs to consider which distribution channels to employ to get its software product to market. Achieving the optimal proximity to its customers is crucial (Chapman 2005). The time and energy a vendor needs invest in creating the ability to sell direct to

customers effectively may be impractical. Alternatively, a two-tier distribution channel model, where a product is sold to a distributor who then sells to reseller, can result in a vendor being too far away from its customers. The traditional middle ground for software distribution is for a vendor to be one-step removed from its customers (Hasted 2005), with vendors generally only becoming directly involved with major sales.

2.5.6 Licensing and pricing

The perpetual license established itself as an industry standard and for many vendors this has been the only way that they have sold their software product (Geisman 2005a). The most common configurations are for software licences to be sold per named user, or named machine, or they may be for a set number of maximum concurrent users (SIIA 2006). The subsequent yearly maintenance component of a software price is usually equivalent to 20% of the perpetual licence. This fee normally entitles a customer to a level of support in using the product and the provision of updates or new versions of the product as they become available (Wohl 2004).

A licensing and pricing model needs to support a critical mass of software product sales that will bring a company a high enough level of revenue to survive (Hasted 2005). A vendor's licensing and pricing model should be simple and have an internal logic that is easy to understand (Sink 2006; Geisman 2006c). The licensing and pricing model should aim to accelerate and increase revenues coming into the vendor (Sainio & Marjakoskia 2009; SoftwarePricing.com 2006). The pricing model needs to be realistic and defendable (Dver 2003). It is critical not to undervalue a software product (Hadley 2003). Discounting, in particular *ad hoc* discounting, should be avoided at all costs (Geisman 2006c). This can destroy the perceived value of a software brand and product and create an irrecoverable position in the market.

There has now been a major shift in how software is priced. The capacity-based model of software pricing, where a customer purchases a set number of perpetual licences, is likely to be replaced by a usage-based model (Graham 2000). A survey of 484 technology executives spread equally across software vendors and end-user organisations in 2006 revealed that while vendors are content with traditional pricing models, 57% of customers are not (SIIA 2006). As an indication of the shift in software pricing by vendors, over 60% have altered their licensing and pricing policies in the two years up to 2006. Whatever the dominant software licensing and pricing model is in the future, demand for increasingly flexibility in how software solutions can be consumed and paid for is likely to be here to stay (Geisman 2005a).

Product bundling provides another dimension to software pricing. This is widely used by vendors in the attempt to maximise profits (Shapiro & Varian 1998; Bakos & Brynjolfsson 1999). Bundling can be used as a useful way to persuade customers to purchase and use more products than they may have originally intended to (Chuang & Sirbu 1999). Customer advantages of bundling include the opportunity to make overall cost savings and the ability to purchase a total software solution as opposed to isolated software components (Wuebker & Simon 2005). However, software product bundling is a price discrimination technique that can be used to leverage revenue potential and maximise profits (Sundararajan 2004). Research has shown that using bundling makes it possible for a company to increase its profits by 10%-40% (Wuebker & Simon 2005). Price bundling approaches include: forced pure bundling of products that can only be purchased together; optional mixed bundling, which incentivises customers to purchase more product; and add-on bundling (Adams & Yellen 1976; Wuebker & Simon 2005).

2.5.7 Network externalities

The concept of interdependent demand, commonly referred to as network externalities, is a phenomena where customer purchase decisions are heavily influenced by those of

fellow consumers (Rohlfs 1974). Exponential increases and decreases in the demand for software products can occur in short succession as the latest craze comes in and goes out of fashion (Gartner 2005). This upward spiral concept is the basis of positive network externalities (Oren & Smith 1981; Katz & Shapiro 1985). A vendor's ability to cultivate and harness positive consumption externalities can be a key determinant of their competitive success (Gallaugher & Wang 2002). Positive consumption externalities that a vendor experiences are likely to be positively correlated to the size of their installed base and their alignment to industry standards (Brynjolfsson & Kemerer 1996). A software product will become more valuable to a customer in the market if it conforms to an industry standard, is compatible with other complementary goods, and its user base expands (Rohlfs 1974; Farrell & Saloner 1985; Church & Gandal 1992). The more customers that a vendor has, the more customers they are likely to achieve in the future (Roeding et al. 1999).

Network externalities can also have negative consequences for a market as well as positive. Microsoft in particular has been accused of anti-competitive practices that exclude competition in a network system. In several markets, to all intense purposes, the choice of a non-Microsoft alternative has been removed completely (Hildebrandt 1999).

2.5.8 Summary

The scholarly disciplines of Innovation, Product Development, Marketing and Sales and Economics contain a well grounded set of principles, concepts and ideas that are relevant

to running a software business. There is a large quantity of research into Innovation and Product development, although specific research on software innovation and software NPD is limited (King et al. 1994; Swanson 1994). The wealth of generic Innovation and Product Development literatures provide useful context for some aspects of running a software vendor business. Some specific contributions about innovative software product have been identified. Marketing and Sales is a mature scholarly discipline with a wealth of theories and models to support its practice in the real world. It is of particular relevance to software businesses, as one of the most significant reasons software business have failed is through a lack of sales. Several specific elements of the Economics discipline are of particular relevance to running a software business. These streams of Economics have a research basis in the demand and supply of information goods, and the concept of market networks that emphases interdependent demand across consumers.

Specifically, seven key factors of importance to running a software business were identified. First, innovation and product development capabilities can be essential in the dynamic software industry. Second, having an innovative software product can be an enabler of commercial success. Third, effective market segmentation and positioning is worth getting right. Fourth, in a market where perception is an important factor, effective market communications are an imperative. Fifth, developing robust sales and distribution channels is essential. Sixth, licensing and pricing of information goods such as application software requires careful consideration. It is an inexact science. Seventh, understanding and proactively managing network externalities can make or break a software business.

2.6 Conclusion

The literature review commenced with the anticipation that the Strategy Management and Information Systems disciplines would provide a repository of relevant reference frameworks and models for running a software business. It quickly became apparent that

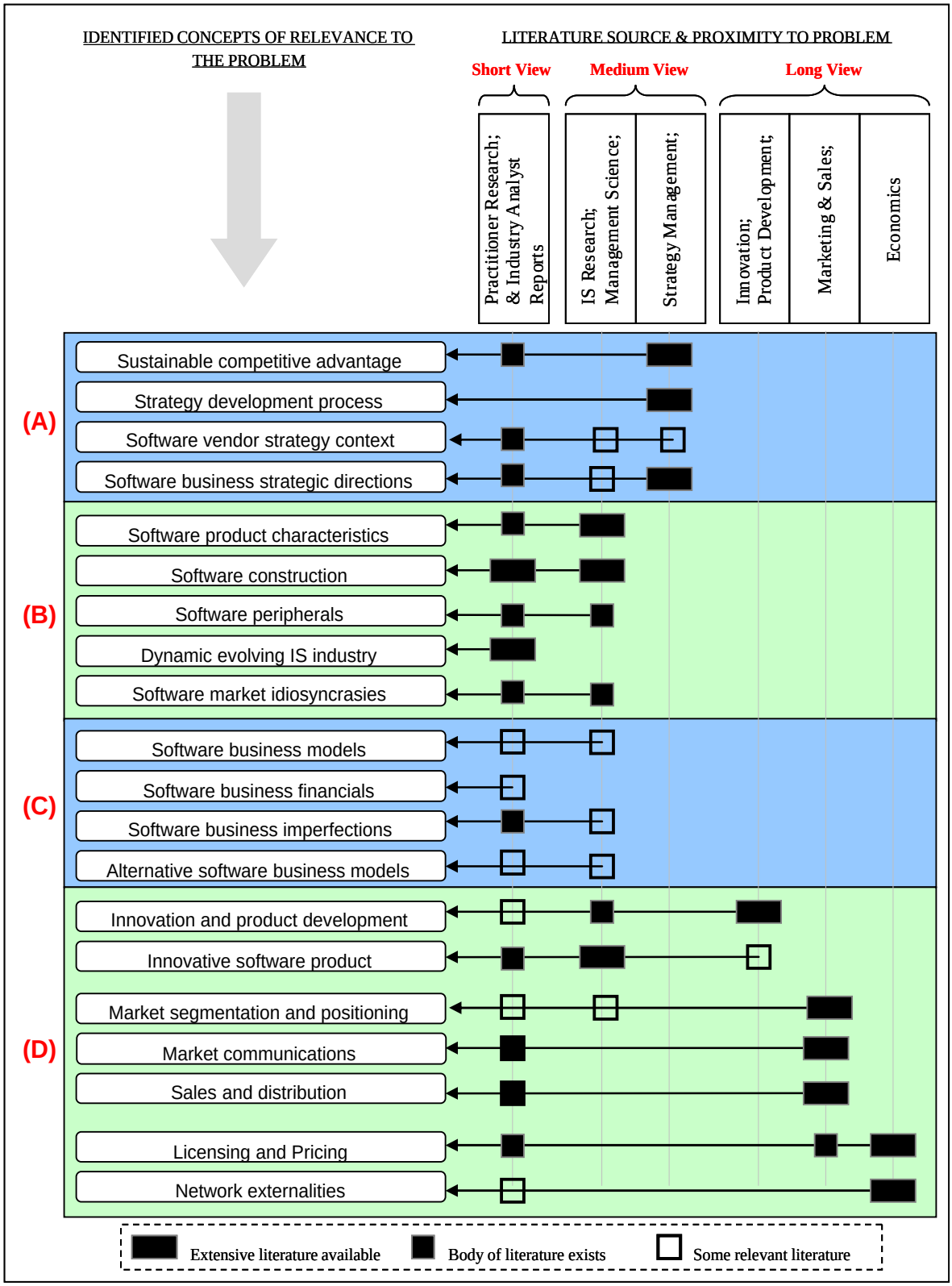
this was not the case. While no central premise or theory specifically orientated around providing guidance on the research problem was discovered, many topics of partial relevance were identified. These topics generally covered one or more component parts of running a software business, but they did not address how to run an overall software business as a whole. In addition, it became apparent that running a software business involved dealing with many different things. This led to the considerable broadening of literature review and a higher-level, sideways looking, open-cut approach to identifying reference materials. Consequently contributions from IS trade press, industry analyst reports, and IS practitioner research sources were examined, while the scholarly fields of Management Science, Innovation, Product Development, Marketing and Sales, and Economics came into scope for the literature review.

The literature review has resulted in the identification of twenty concepts of relevance to running COTS application software SME business. Figure 2b provides a reference literature map, which shows relevant literature concepts and the extent of contributions from different disciplines examined. Contributions have been loosely classified as 'extensive literature available', 'body of literature exist' and 'some relevant literature'. The twenty concepts have been grouped into four clusters based upon the primary contributing literature source for each. This mirrors the structure of this chapter and the order that these concepts were introduced. Additionally, the literature sources have been categorised as providing short, medium or long range perspectives of the research problem.

Literature cluster (A) is primarily informed from the Strategy discipline. Four concepts are a mix of pure strategy and the application of strategy themes to a software business. Literature cluster (B) is primarily informed from IS Research and Management Science. Five concepts are orientated around software product and defining attributes of the

software industry. Literature cluster (C) is primarily informed from practitioner research and industry analyst reports. Four concepts are associated with the commercial centre of a software business.

Figure 2b: Reference Literature Map



Literature cluster (D) provides a long-range view of a software business. Seven generic concepts that exist outside the domain of a software business have been identified as having relevance and application to the investigation of a software business. The cluster contains two concepts informed from the Innovation and Product Development disciplines, three concepts from the Marketing and Sales and two concepts from the Economics domain.

In conclusion, the literature review identifies that there is a considerable gap in knowledge to explain the running of a COTS application software SME business. A search for literature pertinent to software vendor business management reveals that there is no core body of research or accepted set of concepts that specifically address the problem in any great depth. The literature map of twenty concepts relevant to the research problem *is essentially an aggregation of isolated unlinked ideas and reference points of interest. These are not explicitly connected or integrated into a holistic framework or model for running a COTS application software SME business.* However, the results of the literature review do provide a broad set of concepts that are relevant to the study of a software business. In aggregate, these provide a useful set of ideas and reference points that can assist the investigation of the running of a software business. This provides a powerful catalyst and launch pad for the research design.

3 RESEARCH METHODOLOGY

3.1 Introduction and Overview

This chapter outlines the research methodology for the investigation of the thesis research question: can a holistic guiding management framework be developed for running a COTS application software SME business?

As the review of existing literature has not revealed any central premise or theory that comprehensively explains how a software business should be run, no predefined hypothesis or existing theory was committed to prior to the primary research commencing. While the literature map of twenty relevant concepts provides a powerful launch pad and catalyst to initiate an overall investigation of the problem, it was important that these elements should not dictate or constrain the investigation in how a holistic and integrated software business should be run. The intention was very much to undertake the study with an open mind. A key part of answering this central research question is to illuminate and contextualise a software business with a rich tapestry of relevant business factors as defined by practitioners. A key objective is to understand how practitioners conceptualise the key aspects of running such a business.

The philosophy of the study is theory building and as such, grounded theory was selected as an appropriate methodology for the research. The exact breakdown of this methodology and the corresponding structure of this chapter are now introduced.

Section 3.2 selects and justifies an appropriate research paradigm for the study. An overview of the grounded theory methodology is given and prevalent elements of the grounded theory domain discussed.

Section 3.3 provides detail on the approach to the research data collection. Units of analysis and source of data are considered, purposeful sampling discussed, an approach for entering the field and negotiating access to research informants detailed. The process of developing research constructs is outlined. The strengths of interviewing are highlighted, the research questions are broken down into interview questions and ways for refining the data collection process are discussed.

Section 3.4 describes a procedural approach to the data analysis. This begins with data classification and conceptualising. Open coding is used as a means of assigning units of meaning to data. This is followed by axial coding as a process for formulating concepts. A higher-level theorising analysis is then outlined. The grounded theory practice of selective coding is explained and the exercise of theory validation through theoretical sampling offered.

Section 3.5 covers a number of other research methodology considerations. Ethics approval is an imperative. Methodology validation from a process perspective is critical. The quality and worth of the research findings require qualification. Finally, a narrative as a style for writing and communicating the thesis is offered as an effective way of presenting the study.

3.2 Methodology Selection and Justification

This section covers the selection process and justification of a grounded theory inspired research methodology for guiding the software vendor strategy study. A theoretical overview of grounded theory principles, procedures and activities are outlined. The evolution of the paradigm and different scholarly perspectives on this research paradigm are discussed.

3.2.1 Qualitative methods: Field research and interpretivism

Research methods can be classified into broad categories: quantitative research and qualitative research. Quantitative research is typified by the collection of large sample based of specific and data that are statistically analysed to prove or disprove a proposed hypothesis (Collins & Hussey 2003). In contrast, the focus of qualitative methods typically is to generate ideas and theory. Rich qualitative data is collected from small samples.

A qualitative method has been chosen to investigate the software vendor strategy problem. Limited research has been undertaken on specifically running a software business and not much guidance exists for this problem. As the focus of the research is to develop new theory, qualitative methods are appropriate. As the qualitative paradigm has the ability to deal with complexity and provide reflexivity, this makes it is particularly relevant. Being direct, intense and using expressive language, this mode of research offers the facility to inductively analyse data that describes the software vendor problem (Creswell 1998). Conversely, qualitative methods have their limitations. It can be criticised as being unscientific and subjective, skewed by my personal biases and values and generating theories that are not verifiable (Creswell 1998). However, my epistemological view is that immersion in the problem environment guided by a qualitative paradigm, even allowing for any presuppositions, is the most suitable approach for investigating the software vendor strategy problem (Groenewald 2004).

Increasingly, interpretivism methods are being using to investigate IT research problems such as the software business challenge. If novel theories are to be developed to explain phenomena such as the software business problem, it is questionable whether these will be achieved using purely positivistic methods. An advantage of using an interpretative method for this investigation is that it offers contextualisation, abstraction and

generalisation, and dialogical reasoning (Rowlands 2005). New theories can be engendered from the field, where limited previous research has occurred (Benbasat 1987). The use of field research will be advantageous to investigating the software business problem. An appropriate technique for investigating '*a contemporary phenomenon within its real-life context; when the boundaries between the phenomenon and context are not clearly*' (Yin 1984 p23) is the use of field research. Field research is inductive. Variables and concepts can emerge from the qualitative data collected, once these are understood and contextualised; there is a strong chance and of new theory being generated.

3.2.2 Choosing grounded theory: A justification

Grounded theory provides the theoretical basis for the research design. The software business problem research design needs to foster creativity and not be restricted by previous thinking. In parallel, it is imperative that the research approach does have a strong foundation in both principles and procedural steps. For a supplier in the software market, limited guidance is available on how to pull together all the considerations involved in running an overall software business. The challenge of investigating the problem is further exacerbated as the background IS discipline arguably has no strong theoretic core or central foundations that can be built upon (Weber 1999; 2003; Benbasat & Zmud 2003; King & Lyytinen 2004). For this reason, the intention is to commence the study with an open mind about the factors that software businesses need to consider, rather than be driven by elements identified from an examination of existing fragmented literature. The software business problem research design is therefore heavily influenced by grounded theory (Glaser and Strauss 1967).

A key aspect of the research approach is to examine and reflect on perceptions of senior software industry professionals in an attempt to gain an understanding of the main artefacts associated with strategy formulation in the software industry. Grounded theory is differentiated from much other research as it is explicitly emergent. It does not test a

hypothesis. It sets out to find what theory accounts for the research situation as it is (Dick 2005). The advantage of the method is that it allows new theory to emerge from research data, in contrast to data being forced it into predetermined frameworks (Glaser 1992).

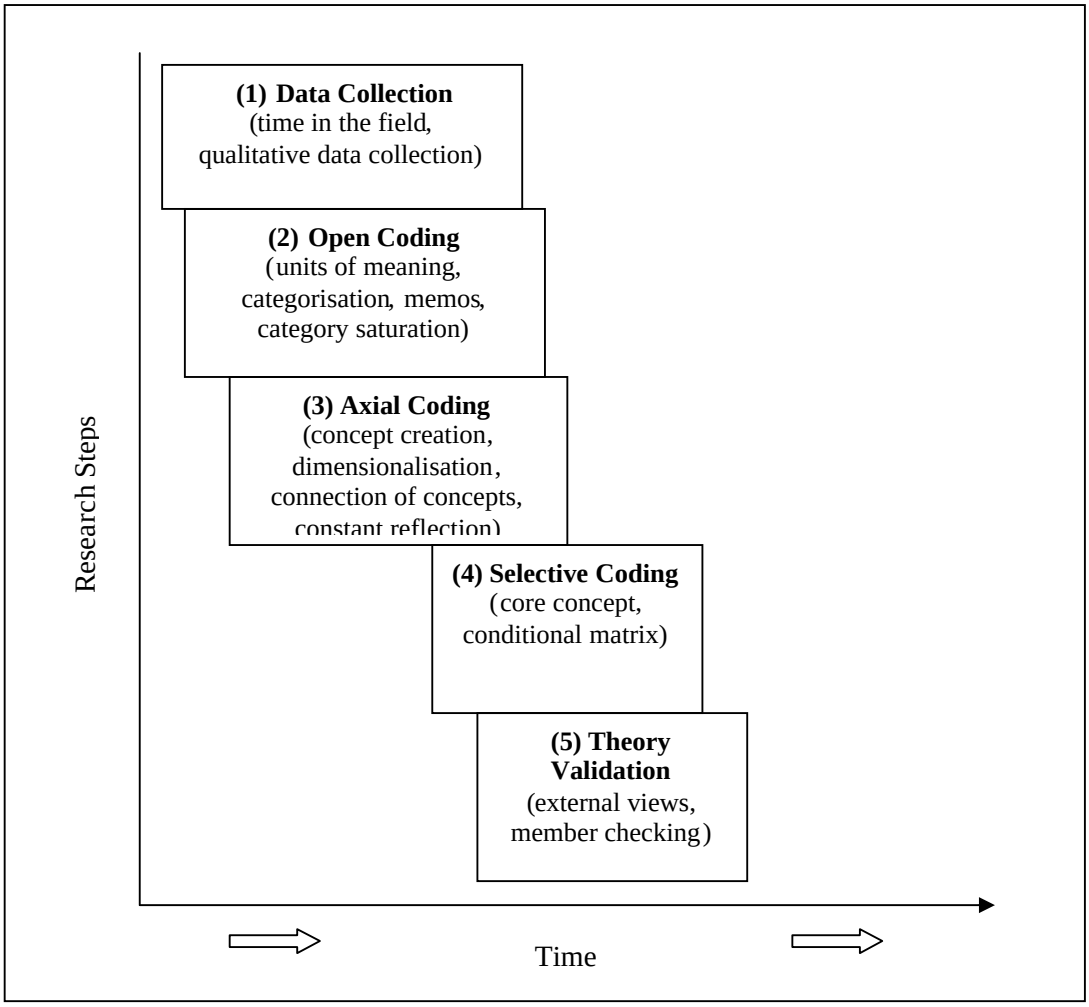
My own theoretical lens is orientated to a research approach with foundations in grounded theory. I am a senior IT professional with nearly twenty years commercial experience of supplying and consuming application software. This includes interacting with application software from a number of different perspectives: developing, selling, implementing and purchasing software; and working with software across a number of different industry sectors. I am of the opinion that there is a danger that a hypothesis-based quantitative survey approach based upon current literature may just result in standard industry rhetoric being reiterated with no new insight being provided. In contrast, grounded theory can provide a fresh perspective. I have a strong background, both professionally and in academic knowledge, which provides the background context for analysing the problem. I felt confident in my ability to manage the delicate balance between having an open-minded view, with the need to keep focused on developing new theory that contributes to the body of knowledge about the research question.

3.2.3 Grounded theory methodology overview

A fundamental characteristic of grounded theory is that multiple iterations of data collection, analysis and theory building are undertaken to ensure that new theory is adequately grounded in data (Babchuk 1996). Strauss and Corbin (1990) emphasise the importance of data collection and analysis being interrelated processes, analysis making constant use of comparisons, and hypotheses about relationships between categories being verified. Grounded theory allows for the development of a theoretical formulation of reality. As theory is generated by the observations rather than being decided before the study, theory is then built that is loyal to and that illuminates the phenomenon being

studied (Turner 1981). The research is 'inductive' and 'applied' (Collins and Hussey 2003).

Figure 3a: Research Method Steps Based Upon Grounded Theory



The methodological approach underpinning the research paradigm translates into five practical steps (Strauss & Corbin 1990). The research commenced with identification of relevant field sites and negotiation of access to collect primary data concerning software vendor business models and strategy considerations. 'Open coding' classifies this raw data: events and actions are grouped together into categories and sub-categories. These are grouped and parameterised as dimensions using 'axial coding': contexts, inter-relationships, and conditions around the categories and sub-categories. A process of 'selective coding' interconnects these dimensions to develop a software vendor strategy paradigm around a central category (or construct). Lastly, the new theory is validated and

verified to ensure that it is adequately grounded. Figure 3a illustrates the five research steps and identifies the sub-elements for these. The diagram shows the relative timing of these research procedures and emphases the extensive overlap between these activities through the end-to-end research process.

Beyond the methodical steps of grounded theory, Strauss and Corbin (1990) stress that: the collection and analysis of data are interrelated processes; the units of analysis are concepts, categories and ideas: these should be constructed and relationships explored; the path a study takes is influenced by the emerging theory; and environmental context should be accounted for and inconsistencies with the core theory explained.

3.2.4 Glaser, Strauss and the progression of grounded theory

Grounded theory was pioneered in the 1960s by Barney Glaser and Anselm Strauss, as a powerful new interpretative research paradigm (Glaser & Strauss 1967). The two academics believed that quantitative hypothesis-based studies were limited and that to generate new theory about phenomena it was imperative to get out and immerse yourself in field. They developed a set of principles and created techniques that allowed for the development of theory that was grounded in data. Grounded theory was revolutionary for its time as it offered the advantage of creativity, while cradling this in logical process. Glaser and Strauss developed techniques for analytically developing theory; this involves interplay between envisaging ideas and structuring these into explanatory models. To enable an objective perspective, it is crucial that the phenomenon is analysed from multiple viewpoints; and that continual comparisons are made between reality and the emerging theory. However, the crux of grounded theory is to understand the intention of procedures and ensure these are achieved, as opposed to blindly embarking upon these techniques in a prescriptive fashion (Strauss & Corbin 1998). Grounded theory is both an art and a science.

As grounded theory evolved and became more widely adopted, Glaser and Strauss diverged in their views about the true essence of grounded theory. Glaser and Strauss' varying beliefs were centred around their incompatible views on epistemological perspective and methodological approaches (Babchuk 1996). Strauss' version of grounded theory emphases the following methodical steps to ensure that a researcher is guided through the study and an endpoint is reached where new theory emerges (Strauss & Corbin 1998). Strauss argued that grounded theory needed to retain elements of scientific rigour to be usable and acceptable (Babchuk 1996), with generalisability and verification of new theory being a requirement for credibility. In comparison, Glaser's version of grounded theory stressed the importance of keeping the paradigm fluid. He argued that theory would emerge from guidance by informants, rather than from following a set of research procedures (Babchuk 1996). Glaser's concern was that an over-focus on method could result in the essence of the phenomenon being missed and field data being forced into preconceived frameworks. As the academics diverged in their views, two schools of thought materialised. Strauss teamed up with Juliet Corbin to continue his work, while Glaser continues to publish primarily on his own.

This research into the software business problem more closely aligns with Strauss and Corbin's (1998) version of grounded theory. Grounded theory researchers need to decide which type of grounded theory they will use: Glaser's, Strauss and Corbin's, or a hybrid. However, it is questionable what level of importance researchers should place on the fundamental differences in opinion between grounded theory's founders (Parker & Roffey 1997). It is suggested that a generic starting point is used to take a balanced view. While possible, it is unlikely that a researcher will adhere to a wholly pure version of either Glaser's, or Strauss and Corbin's, grounded theory. There has been much discussion about the paradigm over the last forty years and it is suggested that a researcher is likely to reference a number of other authors' perspectives on the research

method in arriving at a final methodology (Babchuk 1996). A hybrid method is often likely to eventuate as most qualitative methods combine both inductive and deductive analysis activities (Orton 1998). The research into the software vendor problem will more closely align with the Strauss and Corbin version of grounded theory, but also attempts to defend against Glaser's criticisms of this approach. As the study needs to form the basis of a tightly scoped DBA qualification and I am a novice in respect to undertaking grounded theory, the pursuit of defined procedural research activities are more likely to result in success than would be likely through the adoption of a more fluid approach. However, I was keen to explore new ideas and be guided by informants. I was determined that the research data would not just fall into preconceived frameworks.

3.3 Data Collection Approach

This section outlines the research design for the data collection component of the study. It covers step one of the five research steps discussed in the grounded theory methodology overview section and is illustrated in Figure 3a. Units of analysis and sources of data are defined and identified respectively. A sampling frame is delineated; a purposeful sampling strategy outlined for case selection; and the approach for entering the field described. Interviewing as a data collection medium is discussed; the process of translating the software business paradigm into a problem-focused interview is sketched out; the approach to developing the interview questions are described; and techniques for refining the data collection process as it progresses, suggested. Finally, interview protocols and administrative tasks are explained; and the approach to recording and storing data illustrated.

3.3.1 Unit of analysis and source of data

The unit of analysis is business application software vendors and the source of data is informants who have first-hand experience and knowledge of running these. Determining

an appropriate sampling frame and subsequently identifying and negotiating access with appropriate informants was absolutely critical.

Determining the unit of analysis and sampling frame

The unit of analysis is software businesses and the source of data is individuals from organisations who have specifically have firsthand knowledge and experience of running COTS application software businesses. The population of cases and informants for the research consists of a sub-set of IT industry software vendors and professionals. There are a large number of organisations, individuals and bodies that supply and consume an immense array of COTS application software products and associated services around the globe. To allow the software business problem to be investigated, it is important to delimit the problem so the research is manageable and meaningful conclusions can be drawn within context. Therefore, the population for the research has been reduced significantly to determine a more focused unit of analysis.

The sample frame focuses and limits the research to data collection on Sydney, Australia-based SME vendors, whose core business is both building and selling COTS business application software. Software distributors, software system integrators and software customers were not targeted. Although the software market is global with a high concentration of software vendors in North America, there are also many Australian software businesses. The Australian COTS software sector market value is US$3.54bn and constitutes 1.2% of the total worldwide COTS software sector (OECD 2008 p53). There are currently over 750 suppliers in Australia that provide COTS software. Geographically, New South Wales (NSW) dominates and 365 of the 750 COTS suppliers are based in this location. It is therefore assumed that a research sampling frame of software businesses in Sydney, Australia, will provide valuable insight into the industry as a whole. Mid-sized growing organisations are targeted. This is so the businesses are large enough to be multifaceted, but not so unruly that they are too overly complex to

analyse. A preference for Australian headquartered software businesses is made as these organisations are likely to have executives in Australia who are responsible for core business functions such as strategy, product development, marketing and finance. Executive-level professionals, such as CEO, product director, sales and marketing director, chief financial officer (CFO) and others managers directly responsible for these areas are targeted. Overseas companies who just have sales, distribution and support functions in Australia are unlikely to have Australia-based executives who cover all the aspects of running a software business.

Purposeful sampling and informant selection

A necessary factor for the research to be successful is to have informants who have experience and knowledge about the software business problem (Curtis et al. 2000). Formulating a sampling strategy with qualitative methods can be complex (Tuckett 2004). There are various approaches to sampling (Coyne 1997). The sampling philosophy for this research had three key characteristics (Curtis et al. 2000). First, the sample size was small, but each informant was examined meticulously to generate a rich set of data. Second, the selection of samples was not random or based upon a statistical probability. It is based upon purposeful sampling. Third, the samples were not wholly pre-determined prior to entering the field, rather selection is an ongoing process that occurred throughout study.

The intention at the start of the study was to interview at least thirty software executive-level professionals from at least twelve software related organisations. This quantity of informants and number of cases should provide a sufficient set of qualitative data about the problem for a qualitative doctoral study. This should enable a range of software business instances to be examined and allow for a number of different perspectives on these to be captured. It was envisaged that collection of data from the field would take place over a six to nine month time-span.

A purposeful sampling strategy was chosen for the research data collection. The sampling strategy for the software vendor problem is relevant for the grounded theory paradigm; it should result in the collection of rich information about the phenomena (Miles & Huberman 1994 p34). This sampling strategy blends a mix of sampling types (Patton 1990; Coyne 1997). It is 'criterion-based' sampling in order to identify typical cases of the phenomenon and those who have experience of it in the field. From a feasibility and practicality perspective, it utilises 'convenience' and 'opportunistic' sampling techniques to identify these primary informants (Groenewald 2004).

The strategy for gaining access to these software vendors was first to exploit my own network of contacts and the second to utilise the interview snowballing technique. Snowballing is a technique for expanding a researcher's original sample of informants (Groenewald 2004). I have a sizable network of personal and professional contacts that I leveraged to identify a research sample. As a management consultant in the technology arena, I have various existing relationships with industry professionals. This includes an array of colleagues and ex-colleagues, a range of contacts from software suppliers, a large base of recruitment contacts who are well connected in the Sydney IT market, and a number of friends who also work in the IT/software sector. As a starting point for the research, these contacts include four software vendor CEO/ex-CEOs and around fifteen software vendor executives. In addition, I had another twenty plus software/IT-related senior managers and an opportunity to reach many more potential informants that are one step removed from my existing relationships. All potential informants and organisations were assessed against the sampling frame delimiters, prior to an interview being undertaken, to ensure that they meet the target research units of analysis criteria.

I also had an alternative plan for identifying potential informants for the study if the snowballing technique was unsuccessful in delivering thirty appropriate interviews. An examination of Dun and Bradstreet's (Australia) companies' database identified around 350 software vendors in NSW that are classified with the SIC reference code '7372 – Prepackaged Software'. Narrowing the search criteria results in 290 in software vendors in NSW that have ten or more employees. My backup plan was to approach executives from these companies; fortunately, the snowballing was hugely successful and this alternative approach was not required.

Entering the field: Securing access

Collecting qualitative data about the research phenomenon from software vendor field sites could be problematic if access cannot be successfully negotiated. It was hoped that Sydney-based executives from a number of software vendors would be receptive to the study and would make time available to be interviewed. However, it could not be assumed that this was a *fait accompli* and securing access within the field may have involved overcoming a number of challenges (Bailey 1982). When approaching potential informants, it was important to consider in advance any prospective resistance that may have been encountered. I defined an approach for how problems were to be tackled if they were experienced, to ensure I was as successful as possible in gaining access to interviewees. I devised an approach for handling any potential informants who were nervous or unsure about being interviewed for the study. Potential informants were sent a formal e-mail letter with the Macquarie Graduate School of Management (MGSM) heading that outlined the purpose and objectives of the study, a request for participation in the study by being interviewed and the expected benefits to the participants from contributing to the research (Bailey 1982). Either agreement was gained for the informant to be interviewed and an interview subsequently scheduled, or they were categorised as a non-participant.

I was pleasantly surprised how successful the negotiation of access turned out to be. The high success rate was attributed to two factors: the goodwill of the friends and colleagues of initial informants and the real interest in the research topic from those executives contacted. While I had several back-up plans for achieving thirty relevant interviews, none of these other approaches were necessary.

The purposeful sampling and snowballing was also highly productive. The snowballing technique was deployed from the start of data collection with all interviewees being asked if they could recommend anyone else who would be suitable and willing to take part in the research. As I became aware of new potential informants through my time in the field, my potential informants contacts list was continually appended. My strategy for approaching potential informants was incremental. Only four to six potential informants were contacted at a time, to allow them to either accept or reject the request for them to take part in the study. This allowed me to keep on top of scheduling and actually undertaking interviews. In addition, it ensured that I always focused on those potential informants who were perceived to have the most intimate experience of the running a COTS application software SME business. This was regardless of whether the potential informant was on my list prior to entering the field, or whether they were just added to the list the day before as a result of a snowball contact.

My initial research sample of twenty plus software/IT-related senior managers and extended contacts list successfully managed to generate an additional eighty potential informants for the study. Of these, around forty were rejected as not meeting the appropriate criteria in terms of having been closely involved in running a software business. Over the time in the field, fifty three potential informants were approached. Of these fifty three, thirty successful interviews were conducted. Table 3a outlines the high-level profiles of the thirty interviewees.

Table 3a: Interviewee Characteristics

Interviewee (unique identifying code)	**Characteristics of Interviewees**
I-01	Previous Country MD of global software vendor, 25+ years industry experience
I-02	Executive Chairman, SME COTS vendor, 30 years SME experience
I-03	Product Director, SME COTS vendor, 15 years SME experience
I-04	Professional Services Manager, SME COTS vendor ,15 years SME experience
I-05	Sales Manager, SME COTS vendor, 20 years SME experience
I-06	Consultant, previous COTS vendor manager, 30+ years industry experience
I-07	CIO, SME COTS vendor, 15+ years SME experience
I-08	Consultant, numerous COTS SME vendors, 30+ years industry experience
I-09	Regional Sales Manager, global software vendor, 15+ years industry experience
I-10	Software Sales Manager, SME software vendor, 20 years industry experience
I-11	CEO, SME software vendor, 20 years industry experience
I-12	Founder & CEO, SME software vendor, 25 years industry experience
I-13	Founder & CEO, SME software vendor, 25 years industry experience
I-14	Previous Operations Director, SME COTS vendor, 5 years SME experience
I-15	MD, SME software vendor, 20 years industry experience
I-16	Previous Director, numerous SME software vendors, 35 years experience
I-17	Operations Director, SME COTS vendor, 10 years SME experience
I-18	Software Sales Manager, Large & SME software vendors, 20 years experience
I-19	MD, SME software vendor, 20 years industry experience
I-20	GM, SME software vendor, 25 years industry experience
I-21	Marketing Director, Large & SME software vendors, 25+ years experience
I-22	Software Sales Manager, SME software vendor, 20 years industry experience
I-23	CEO, SME software vendor, 20+ years industry experience
I-24	Consultant, various end users & vendors, 25+ years industry experience
I-25	Director, SME software vendor, 10 years SME experience
I-26	CEO, SME software vendor, 20+ years industry experience
I-27	Previous Country MD of global software vendor, 25+ years industry experience
I-28	CEO, SME software vendor, 25+ years SME experience
I-29	Executive Chairman, SME COTS vendor, 35 years SME experience
I-30	GM Sales, SME software vendor, 25 years industry experience

The thirty interviewees came from a cross-section of twenty five different software vendors. Using a loose classification, these software vendors comprised six large, six medium and thirteen small sized businesses. Nineteen of these were headquartered in Australia. Six were headquartered overseas. The profiles of the thirty informants included

twelve CEOs, seven product development and/or operations executives, and eleven business development executives.

3.3.2 Research construct definition: Designing the interview

Collecting a rich and representative data set was an imperative in terms of setting the research up for success. Interviewing as a method of data collection was selected. A structured approach to designing the interview was undertaken based upon grounded theory principles. The data collection process was significantly refined over a piloting phase of interviewing.

Interviewing as a data collection medium

The data collection approach for the research design was face-to-face, semi-structured interviews. This approach has many advantages that are beneficial to the study (Bailey 1982). It provides flexibility in which questions are asked and in what order, and allows the interviewer to be spontaneous and notice non-verbal signals. A much more complex questioning can be undertaken and I had the ability to ensure answers to all questions are retrieved. However, there are also some shortcomings of interviews that need to be noted (Bailey 1982). A significant issue is the danger that the data collected may incorporate interview bias. Further, from a practicability perspective, interviews take time and can involve accessibility issues.

The questions embrace a number of guiding principles of grounded theory paradigm (Strauss & Corbin 1998). An imperative is that the content of some of the questions enables my existing assumptions, perspectives and biases to be re-examined by the informant. The questions need to recognise what the interviewee believes is occurring with respect to the phenomenon and identify data properties and dimensions that exist in the field are not yet captured. I then evolve the questioning from requesting description, to acquiring explanative data about the phenomenon.

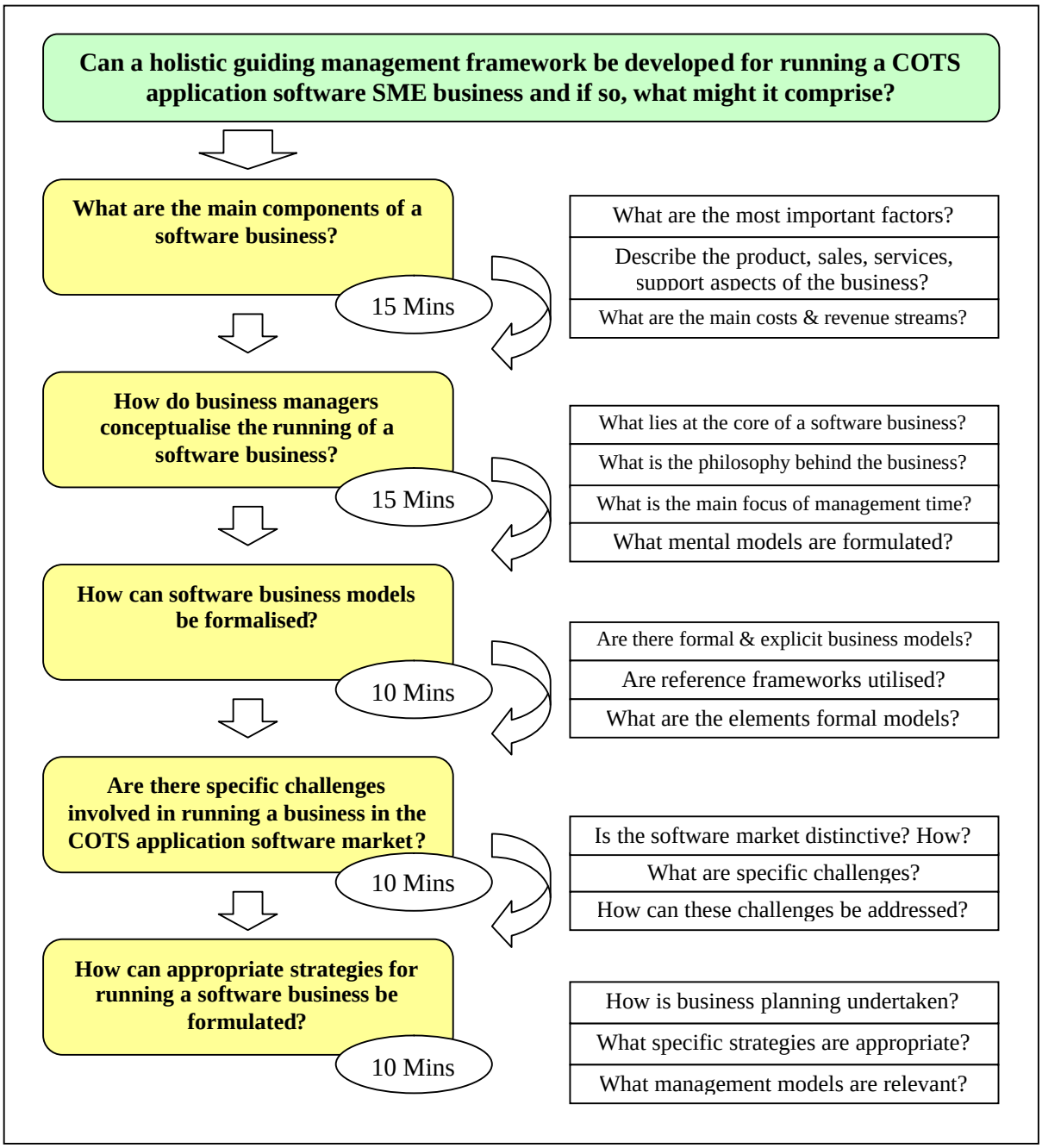
The key driver for the interview design was that it should be problem-centred. Research questions have been developed to understand the phenomenon at the centre a software business (Flick 2002). Contextual questions aim to identify what factors influence the phenomenon, where connections exist between variables and the likelihood of certain outcomes occurring. Careful consideration has been given to ensure informants' answers should illuminate the phenomenon (Glesne 1999).

Interview design

The contextual literature review for the problem, a component of the grounded theory paradigm, has provided a background landscape for investigating the phenomenon. This landscape has assisted in defining an initial scope and preliminary set of questions for investigating the problem.

The overall research question '*can a holistic guiding management framework be developed for running a COTS application software SME business?*' has been broken down into five main underlying research questions. Each of these main questions has in turn been further sub-divided into lower-level research questions. Figure 3b below summarises the framework for the interview design and structure. It illustrates the plan for how the interview was navigated from start to finish, and shows a rough estimation of targeted time to be spent on each area.

Figure 3b: Plan for Research Interview Structure and Content



The five main research questions are essentially 'guiding' questions, also referred to as grand tour questions, which provide a high-level framework to keep the interview on topic (Glesne 1999; Rubin & Rubin 1995). The flow of the interview was been designed to focus on the research problem progressively. The intention is to contextualise the research question firmly, by developing a structured background behind the problem, as

defined by the informant. The main questions also provide a checklist to ensure that responses can be gathered for the pre-defined areas of examination. Below the sub-questions, secondary questions were also constructed to enable the drilling into particular areas of focus that interviewees may flag as important. The intention was that these were available for the interviewer to call upon if necessary, as opposed to forming a mandatory set of questions to be answered.

The interview was designed to focus on the software business problem, while incorporating flexibility within the bounds of an overall maintaining structure. The design incorporates open-ended questions and a degree of flexibility, it has the advantage of being able to evolve and change as new themes and theories emerge. This is in line with the characteristics of the qualitative grounded theory paradigm. The open-ended nature of the main questions for the area of study allows informants to go into depth and construct their answers based on the way they perceive the phenomenon, instead of being confined by a rigid set of sequenced questions (Fontana & Frey 2000). The objective has been to develop questions that resonate to interviewees' lives, rather than using theoretical terminology of my study that may have limited meaning to the informant. If the responses are relevant and insightful to the investigation of the problem, the interview design allows the informant to drive the interview content.

A range of questioning techniques have been employed to ensure that the interaction with the informant was effective as possible in gathering a rich set of relevant information about the phenomenon. Asking questions about an interviewee's opinions, values and behaviour in relation to a phenomenon can be useful for sensitising how the informant constructs a mental picture of the topic in their head. Probes are a distinct type of question with a number of functions (Rubin & Rubin 1995). These can be particularly effective for encouraging an informant to elaborate, describe further or clarify what they have said.

Specifically, my questions can enquire why, when, and how something happened, and query by asking for specifics and examples. To maximise the quality and integrity of the data collected, it was imperative that the interview design avoided leading questions, double barrel statements and confusing lines of enquiry.

A number of guiding principles were followed to ensure the most effective execution of the interview (Glesne 1999). The ability to capture rich quality data was linked to the interviewer's creativeness and aptitude in manoeuvring the interview in response to the informant responses (Fontana & Frey 2000). The interpersonal dynamics of the interview were of particular importance. Awareness to building rapport and establishing trust early on was paramount.

Refining the data collection process

I piloted the interview design to ensure it was as effective as possible for collecting data for the study. The first interviews were undertaken with informants with whom I had an existing relationship. This allowed the pilot interviewees not just to be interviewed, but their feedback to be canvassed for what worked well and what could be improved with the interview format (Glesne 1999).

The inclusion of field notes and follow-up questions was an important part of the data collection process. The writing of memos, a process known as memoing, is where I noted down the ideas and thoughts that came into my mind during the interview or reflections from immediately afterwards (Groenewald 2004). These field notes could be theoretical, methodological or analytical. This can include observational notes on what has been seen or inferred by body language from the respondent. Although, occurring as the data is collected, there is a level of interpretation involved, the memoing process can also be classified as a preliminary data analysis activity. So my thoughts were not forgotten, it was critical to capture these notes on the same day as the interview took place.

Immediately after the interview had finished, while it was fresh in my mind, a list of follow-up questions from the session were written down (Rubin & Rubin 1995). Considering areas of the interview in which themes emerged, or the answers could have gone further, additional questions were prepared that could be factored into subsequent interviews.

There were a number of specific areas of note from the pilot. I had to condition myself to switch off a normal inclination to add my own opinion to the conversation. After several interviews I realised I was being overly structured. As I was so focused on keeping to my interview design and to a time schedule, this did not really although any in-depth exploration or analysis of topics that informants' were raising. Moving to a less structured approach ensured that the conversation was not so rigid and stunted. Completely open-ended and very broad questions did not work well. Often informants did not really understand what was being asked. Refined questions became much more specific and caged in real life context. I transitioned from having multiple pages of explicit questions that were cumbersome to navigate. Instead, a single page of topics and key words were the main instrument I used in the interview. This still provided a reference point to ensure all topics of interest could be covered, but provided the flexibility for these to be covered in any order if the informant took the conversation off into interesting areas of discussion. Having a single sheet of paper allowed me to really focus on what the informant was saying; and meant that I did not become pre-occupied flicking through sheets of questions while attempting to stick to a pre-determined interview structure. This approach worked extremely well and I continued to hone my interviewing and data collection skills as the interviews progressed.

As the interviews progressed, I continued to examine how the interview could be enhanced (Rubin & Rubin 1995). Reviewing the interview data collected, a number of

factors were looked at. Were the main phenomenon variables being identified and problem questions being explained? Were enough examples and references being collected to justify what was being communicated? Critiquing the interview execution considered a number of other considerations. Were the informants knowledgeable about phenomenon and clear about what the research questions were asking? Did the discussion flow or was it too jerky?

While the interview design and focus of questioning will naturally evolve in any new field research, my base interviewing skills did improve over the duration of this study. On commencing the interviews, I was a novice qualitative methods interviewer. After thirty interviews, my skills became well honed. It is therefore suggested that the data collected at the start of the interviews is not as high in quality as the data collected nearer the end. If the study was repeated, it is likely an overall richer and more focused set of data could be collected.

3.3.3 Administration of research instruments and procedures

A structured approach to recording and storing data made certain that at no point was the primary research data compromised.

In setting up the interview, a location and time was chosen that was appropriate for both the informant and myself (Glesne 1999). The location was selected so that the interview could occur where there were no distractions and where there was a degree of privacy and the responses that the informant gave could be heard without background noise. At the start of the interview, the exact length of time the interviewee has available was clarified, the Macquarie University ethics form signed and a check made to confirm that my audio recording device has been switched on. At the end of the questioning, a number of interview closure activities were performed. The interviewee was thanked for their contribution. They were asked if there was anyone else that they knew who could

contribute to the study and would most likely be willing to do this. If an individual was suggested, a request for an introduction to them was requested.

All interviews were recorded using a Panasonic PP-US065 digital voice recorder and then electronically stored in secure location. The interview audio recordings were transcribed using an Australian internet-based voice transcription service called 'eScribe Digital Transcription Services'. The text of the interviews was then made available in a MS Word format. The primary and sole purpose of data is for the achievement of the DBA qualification. The data is to be held for at least five years following completion.

3.4 Data Analysis Approach

This section systematically describes the data analysis approach for the software vendor strategy study. It covers steps two to four and some of step five, of the five research steps discussed in the grounded theory methodology overview section and illustrated in Figure 3.2. First, methods for data description and classification are described. The grounded theory activities of open coding and axial coding are examined in detail. Second, the approach for data interpretation and theorising is outlined. Selective coding and theoretical validation bring the research data analysis to conclusion.

3.4.1 Data classification and conceptualising

This section covers the organising, describing and classifying of the field data and the subsequent building of relationships between these emerging concepts. First, the qualitative data analysis software product NVivo is introduced. This provides a backbone for the end-to-end data analysis and theory development of the study. Second, open coding, as a means of defining units of meaning, is outlined. Principles, procedures and techniques for effective coding are offered. Third, as the most relevant concepts start to develop a level of density; and potential theories that link these together start to emerge.

Data management and analysis software

The QSR International: 'NVivo' qualitative methods software has been used to store, code and analyse the research data. At the turn of the millennium, data analysis software products continued to be plagued by the decade old argument that their rigid functions suppressed the intuition and creativeness that is a vital part of generating new theories (Dembrowski and Hammer-Lloyd 1995). More recently, however, QSR International's qualitative methods software product, NVivo 7, released in 2006, provides the ability for data analysis software to support the creative aspects of explorative research.

NVivo supports the grounded theory paradigm that underpins the research method for this study and was integral to the data analysis phase of this study. First, the software vendor paradigm field data transcripts were loaded into NVivo. These reside in the data 'sources' area of the application. As the line-by-line analysis of the interview was examined, the NVivo coding functionality provided an easy way of linking text within the transcript to an open code. These are called nodes in NVivo. The new NVivo 'models' capability is extremely powerful. A visual perspective of the data, combined with a user modelling tool, enables new levels of theoretical exploration to be undertaken using the software analysis tool. These NVivo 7 functions more than justified the software product's appropriateness and usefulness for the inductive software business research.

Open coding: Units of meaning

The first stage of data analysis process uses 'open coding'. Qualitative coding is different from quantitative coding that involves fitting data into preconceived categories (Goulding 2002). In contrast, open coding is '*the analytic processes through which data are fractured, conceptualised, and integrated to form theory*' (Strauss & Corbin 1998 p3). The process of open coding involves the interpretation and categorisation of field data into distinct 'units of meaning' (Goulding 2002).

A key philosophy of grounded theory qualitative coding is to commence this exercise with an open mind and create categories from scratch. It is important not to rely on preconceived categories and Glaser (1978) warns of the dangers of the analysis being contaminated by prior perceptions. I attempted to use the phenomenological technique of bracketing for the review of the transcripts (Groenewald 2004). This suspends, or brackets out, any presuppositions and previous theorising related to the phenomenon, while new codes are identified and named (Locke 2001). The principles of coding are based in the principles of scientific classification (Chrisman et al. 1988). To provide a level of rigour to the study, I ensured that the generated concepts were defined by relevant and appropriate terminology, and were collectively exhaustive.

The first attempts at coding were a relatively slow exercise. I was perhaps looking a little too deeply into every word that informants had said, essentially not being able to see the wood for the trees. As the coding progressed, I became more adept at scanning transcripts and picking up a whole range of concepts and relationships that emerged from the data. At times, there was a sense of enlightenment about new and different ideas that informants had raised that I had not ever considered.

Patterns in the data were looked for and provisional hypotheses about how concepts interrelate, developed. In establishing the richness in the field data, it was essential to attempt to look at things through the eyes of the research informants and see how these compared with prior assumptions (Ahrens and Dent 1998). Comparing codes with each other occurred in parallel to the naming process. This avoided the overlap of similar defined codes, and allowed for the refinement of code definitions as the same units of meaning were observed multiple times.

The process of writing theoretical ‘memos’ was an important part of undertaking the grounded theory study (Strauss & Corbin 1990). Goulding (2002b) define memos as:

'*notes written immediately after data collection as a means of documenting impressions of the researcher and describing the situation*' (p75). Memoing was used as a means of capturing spontaneous ideas that were generated when immersed in the field (Locke 2001). Theoretical memos were used to summarise and sensitise thoughts about potential theories.

The open coding process encompassed substantiation and modification of concepts, and this process continued until saturation was achieved. Around a hundred codes were identified that were distinct and unrelated units of meaning before this happened. This is not unusual (Goulding 2002). Combining concepts to form categories was a way to distil the volume of units of meaning into a smaller set of higher-level codes (Strauss & Corbin 1998). These categories comprised properties and dimensions, and included hierarchical child sub-categories. A more manageable set of emerging explanatory themes with greater conceptual density was the result. As saturation was starting to occur, I knew that I had a very rich set of data and concepts, although I was unsure what it all meant and how it could all fit together into a holistic and integrated model to explain a software business.

Axial coding: Formulating concepts

Axial coding moves the analysis process beyond open coding; categories generated from open coding are fused together at an axis, to result in a set of linked concepts. This process of abstraction involved asking relevant qualitative questions of the coded data to create theoretical concepts and categories associated with the phenomenon. An essential part of axial coding is to continually reflect, question and test concepts as they emerge.

To undertake axial coding successfully, it was crucial to build connections between the different concepts (Strauss & Corbin 1990). At this point in the analysis, I further educated myself in some of the advance features of NVivo. Specific functionality around

NVivo hierarchies, relationships, sets, memo links and models was particularly useful to furthering the data analysis. Pattern making was a key technique used to facilitate the axial coding process (Ahrens & Dent 1998). Using this technique, it has been possible to illuminate the data and explain multifaceted narratives. This is a highly effective means of complexity reduction. Visual diagrams and frameworks are also a valuable means of drawing out potential relationships between concepts. Mini-theories were used iteratively to explore and explain certain elements of the overall phenomenon. Relational statements or hypotheses that relate categories together in the context of the phenomenon have been particularly useful. The creation of these relationships between categories adds clarifying power and density to emerging theories.

At the later stages of the axial coding, the analysis shifted from identifying new relationships to refining the linkages and properties of the connections between categories. Diagrams were used extensively through the data analysis of the research. While not all the diagrams developed are included in the final software business model, the innovative and iterative process of diagrammatically attempting to explore relationships between concepts was invaluable. It was at this point in the data analysis process that the study arguably started to go beyond previous research. The literature review prior to entering the field concluded that guiding frameworks for software businesses to date consisted largely of theories that lacked a multi-dimensional perspective for understanding the phenomenon. The output of the axial coding process revealed the beginnings of a holistic model to provide guidance for the software business problem.

3.4.2 Theorising

This section covers the latter part of the data analysis. Creativity and inspiration is an important part of theorising. Selective coding is used to distil and solidify emerging

theory. The use of literature in a grounded theory study is atypical to most other research paradigms. The validation of new theory is important.

An introduction to theorising

Theorising is a challenging activity to undertake. Inspiration was required throughout the analysis. There was no simple and guaranteed way for me to be inspirational, but one technique was to pursue a disciplined imagination approach: following thoughts trails and consciously deciding which of these to develop, or not to develop, was effective (Weick 1989). Other techniques for sense-making included using metaphors, causal maps, loosely coupled systems and searching for retrospective rationality (Weick 1976; Orton 1998). I encapsulated the above approaches and ideas for theorising and these were pulled upon in the selective coding, grounded theory phase that has been undertaken in this study.

The analysis attempted to unify lower levels forms of theorising ideas and concepts, and fuse these together at a higher level to form conceptual tools within a grand theory (Llewelyn 2003). However, some of the concepts and categories generated in the earlier coding stages had ambiguous boundaries and influenced the phenomenon at multiple levels (Langley 1999). This complexity made it particularly tricky to figure out where to start the selective coding exercise and guidance for how to approach theorising was sought. It was important that the theorising activities were decoupled from the theory validation, as too early a focus on validation of new theories has the effect of stifling creativeness (Weick 1989).

Selective coding: Discovery of theory

The purpose of selective coding was to integrate and distil the emerging mini-theories into a single refined multi-dimensional software business model. The challenge was how a central category could be selected. For all their differences of opinion about grounded theory principles and methods, Glaser (1998) and Strauss and Corbin (1998) highlight

remarkably similar criteria for choosing and justifying a central category. It was imperative that the core concept was interconnected with other concepts in a meaningful way. It needs to be visible frequently in the data, being noticeable within in almost all of the cases. It should be of sufficient importance that it can be generalised and extendable for research in other areas. This includes conceptual depth and explanatory convinceability, so that the central category stands up to varying conditions and contradictory or alternative cases can be explained.

I was surprised by how long it took to move forward to an overall big picture. There was a great deal of pondering of ideas and possible directions that I had to work through before the overall framework took shape. Over the course of the time in the field and the data analysis stage, four or five top-level models were developed before a final model was settled upon. As the holistic integrated multi-dimensional software business model was taking shape, the theory generation process needed to arrive at a point of conclusion. The theory needed to be delimited and stopped (Locke 2001). When limited new evidence emerging from the data, a point of saturation was felt to have been reached (Goulding 2002); the theoretical framework then solidified and major modifications became fewer and fewer (Locke 2001). The emphasis then changed to checking the fit of data incidents to the newly generated theory. Working through a checklist was a useful way to ensure the theory was sufficiently refined (Strauss & Corbin 1998). Consistency and logic through the theory was an imperative. Unwarranted complexity and surplus detail were removed. Outliers and exceptions needed to be explainable. When these above conditions were attained, the theory building was considered complete and valid. While a level of theoretical saturation that allowed a grounded theory based in empirical evidence to emerge, it is arguable that I could have gone further with the theoretical sampling to further develop and ground a number of the concepts that made their way in

into the final model. In particular, the conceptual breadth and depth of the business levers construct could have been further explored.

The place of literature in grounded theory

In comparison to traditional quantitative research paradigms, in grounded theory, literature directly related to the research topic is not read extensively before primary research commences Glaser (1998). The premise is that knowledge of current theories and pre-conceived ideas, would limit creative and inspirational occurrences that can be a pivotal part of developing new theory. The research data analysis will become contaminated and lose its inductive ability. However, in reality, no researcher can truly remove themselves from their prior knowledge and experience to achieve neutrality and a lack of subjectivity (Parker & Roffey 1997). The challenge was to develop sensitivity to prior literature; this involved utilising it when it is advantageous to the research, but being extremely careful not to let it limit the creative theory building process.

Relevant prior literature has been integrated into the research at appropriate points through the study. At the start of the study, literature was useful basis from which to devise initial questions for the first interviews. McCallin (2003) notes the difference between having unconstrained thoughts, in contrast to having no thoughts at all.

Theory validation through theoretical sampling

Theoretical sampling is a key component of grounded theory; it has been used in this research as a mechanism for validation of the emerging theory. It is an activity that involves gathering additional data concerning specific phenomena characteristics that are emerging through the study so that they can be more closely examined. Through the analysis, surfacing concepts and theories were substantiated or annulled as they arose (Dick 2005). Theoretical sampling was used during open coding to gather properties and dimensions in axial coding to fully understand the dimensional range of a category and

near the end of the analysis in selective coding as a means to facilitate theoretical saturation (Strauss & Corbin 1998). I was particularly surprised by how important this technique was; five or six of the key components of the end software business model originated from this method and these were concepts that I had not considered prior to designing the field research.

3.5 Research Validity

A key part of undertaking quality research is ensuring that due process and rigour is followed. While a grounded theory paradigm allows for more fluidity than a traditional positivistic study, it is still important that the main steps and principles of this methodology are followed. Staying faithful to grounded theory process provides a direct link to the validity and robustness of the end theory generated. It is therefore important that any divergences from the method that occurred in this study are identified and any potential weakness in the research from these qualified. This section covers a number of methodology considerations for the research validity. Ethical considerations for the research are examined. Validation of the research is important and includes both validation and verification of the research process activities and an assessment of the quality and value of the new theory. Whether the research can really be classified as grounded theory is objectively assessed. The approach for writing up the research is discussed.

3.5.1 Ethical considerations

The sole purpose of the interviews was for the Doctor of Business Administration qualification. Final approval for the research was from gained from Macquarie University Ethics Review Committee (Human Research) on 4 May 2007. Important ethical considerations included attaining informed consent from participants, and ensuring that they have a right to privacy and are not harmed in any way as a direct or indirect result of the research (Fontana & Frey 2000).

I was not attempting to sell or purchase any product or services from participants. No favours were being sought. The reason for and purpose of the research was communicated to participants from the outset. The output of the research, the creation of further software business guidance, should be of a direct benefit to participants. No financial incentive or benefit was provided to participants; but they were offered access to the summary findings of the study. The research was 100% self-funded.

A research information statement and consent form was provided to all participants prior to commencing the interview. This included a brief statement of the objectives of the research, my name, my MGSM affiliation and my contact details and the details of my supervisor. Also stated were: a brief outline of what I wished the informant to do and the estimated time involved to do this; the requirement for the interview to be recorded; and a confidentiality paragraph explaining how the interview data would and would not be used.

3.5.2 Research effort and timelines

The research occurred on a part-time basis in the period 2006-2010. I was also in full-time employment in this period. The year 2006 comprised the research proposal and literature review. The first half of 2007 focused on the research design. Nine months were then spent in the field between July 2007 and March 2008. Data analysis was undertaken in parallel from early in the fieldwork up until the second half of 2008. The research thesis was drafted through late 2008 until submission in mid-2010.

3.5.3 Methodology validation and research quality

Validating the research methodology and verifying the quality of the research output are fundamental. The validity and reliability of the research processes are assessed. Criteria are offered for how the quality and value of the new research can be assessed.

Validation and verification of methodology

Quality and rigour in the end-to-end research process is an imperative. The validation of this and the identification of any limitations are important. The need for pragmatism in accessing informants for the study is highlighted. Challenges experience gathering data specifically on the unit analysis are raised. My bias is considered. The characteristics of a good, grounded theorist are emphasised. Issues connected to entering the field and collected data are positioned. The importance of care and attention to detail when classifying data is stressed. The challenge of balancing creativeness with subjectiveness is raised.

While it is possible to specify the perfect software businesses and subjects for the research, in reality this research has been limited by whom I have actually been able to interview. Of the 25 different software businesses represented, only nineteen of these explicitly fell into the categorisation of an SME. The remaining six were large global organisations headquartered overseas but with an Australian presence. However, it is suggested that these types of organisations do share many common challenges of running a software business as SMEs. Most informants representing these organisations had worked at an SME at some stage. In addition, while software CEOs were the preferable subjects for the study as they had a holistic understanding of running a software business, only twelve of the informants had this title. The other eighteen informants were senior executives who were not generally responsible for running all aspects of a software business. It is noted that while many of these eighteen informants contributed some very useful ideas that are in the end model, on balance, the contributions from CEOs tended to be richer in nature.

At the outset of the study, the intention was for the unit of analysis to be SME COTS application software businesses, with the source of data being individuals who have had

firsthand knowledge and experience of running those types of organisations. While the thirty interviewees provided a rich source of data on the software business problem that ultimately led to the creation of new theory, there are arguably some weaknesses in the unit of analysis and research sample. Although at the time of the interview informants were only representing one software business, most talked about their experiences across more than one software business that they had been involved during their careers. Therefore, while the unit of analysis for the study was a software business, it has not really been possible to associate the data collected to specific software vendors explicitly.

It is important to recognise the tendency towards researcher bias, with the best way to handle this being to clarify any bias explicitly from the outset. Grounded theory rejects the use of existing theories from literature and suggests that analysis should be commenced with a clear and open mind. The reality is that as I have worked in the field as a practitioner for nearly twenty years, I am already knowledgeable about the topic and hold existing perspectives on the problem. It was anticipated that my bias and pre-conceived ideas would influence the research in some way (Howe & Eisenhardt 1990). I entered the study with a basic premise that there is a gap in theoretical guidance for how software vendors should run their businesses. My suspicion was that ubiquitous industry rhetoric about innovative products and consolidation through mega-vendors was only part of the picture. I thought that the industry value proposition is questionable and that there is massive opportunity for the software vendors to do better. I was cautious about the market not being as fluid as sometimes suggested in the popular press and a new technology fad is unlikely to change the competitive landscape overnight. To minimise bias, the best approach was to be conscientiously responsive to the data, to search for counter-evidence tirelessly, and to refer to possible biases in the argument explicitly (Dick 2005).

I attempted to attain the characteristics that are required by an effective grounded theorist. To achieve success, a number of key skills are fundamental: the ability to step back and critically analyse situations; the ability to think abstractly; and the ability to be flexible and open to helpful criticism (Strauss & Corbin 1998 p7). Absorption in the field and devotion to grounded theory activities were an essential ingredient for new software business theory to emerge as an output of this research.

The successful collection of rich and thick descriptive data was an imperative. Prolonged time and persistent observation in the field was critical to achieve this. Respondents as individuals came with many complexities, biases and varying professional statures (Lincoln 1995; McKinnon 1988). To keep focus, it was absolutely critical that the research questions drove the data collection, rather than interesting data collection driving the research (Howe & Eisenhardt 1990).

The validity of the base data set and the data classification process was considered. As transcripts were completed by a third party, this ensured that they were wholly complete and an independent representation of the interview that transpired.

Data interpretation and theorising can be highly subjective. It was therefore critical that I could demonstrate and justify where new theoretical frameworks and hypotheses originated from in the data (Strauss & Corbin 1990). Rather than just attempting to justify casual relationships between concepts, attention has been given to focusing on emerging theories within the context of the problem. Selection of the core category was a particular challenge, but through an iterative process of considering a number of categories for this, the most appropriate nucleus for the new theory surfaced.

Quality and value of new theory

The evaluation of a newly generated software business model is essential if it is to be academically credible and pragmatically useful. Validating the goodness of the new theory needs to cover a number of aspects. The plausibility, value, generalisability and robustness of the theoretical framework are key factors to consider (Strauss & Corbin 1990). The plausibility of the new theory is linked to the perception of whether or not it can explain the phenomenon and contribute to improving real life practice (Howe & Eisenhardt 1990; Locke 2001). The new theory needs to demonstrate tangible value. Parameters for measuring this include: significance; innovativeness; usefulness; integrative ability and predictive capability (Spiggle 1994). Ideally, its strength comes from the introduction of new insights, as well as being lucid, logical and parsimonious (Eisenhardt 1989). The greater the analytical generalisability of a theoretical model the greater the range of circumstances it can be applied and the greater its ability to handle a larger number of constructs and variables (Schofield 2000). A robustness theory needs to have a number of characteristics (Strauss & Corbin 1998). Conceptual density must exist: component concepts need to be analytically related and linkages well defined. The theory needs to be able to account for and explain variations. With these characteristics, the theory has a better chance of standing the test of time and becoming established amongst relevant academic and professional groups.

3.5.4 An overall perspective: Was it actually a grounded theory study?

Suddaby (2006) suggests that the many researchers claim to be undertaking a grounded theory study, when actually they are just using this as generic term and not following standard conventions. He observes six common misconceptions of research of this nature. These six misconceptions provide useful considerations for comparing this study against a grounded theory paradigm.

The first consideration is that grounded theory is not an excuse to ignore the literature. This study did comprise a wide literature review, and this in turn provided a very useful platform to develop questions for entering the field. This contextualisation of the problem at a macro-level has provided credibility and an entry point for commencing the investigation of the problem. The period between completing the literature review and commencing interviews, six months, acted as a useful buffer so that concepts identified in literatures could be 'parked' and the field entered with an open mind. As the analysis solidified, scholarly literature has formed a useful secondary source of data to back these up where appropriate, while comparison of new theory to prior literature has revealed where the existing works are erroneous or over simplistic.

The second consideration is that grounded theory is not just the presentation of raw data. In this study, the analysis process included distinct activities of open coding, axial coding and selective coding. These are explicitly evidenced in the NVivo project that was used to analyse the data. In this thesis, Chapters 4 to 6 follow a sequential journey from the raw data of informants' quotations, through interpretation of what the data was saying, to individual implications for practice, and then finally to implications for an overall theoretical framework.

The third consideration is that grounded theory is not theory testing, content analysis, or word counts. In this study, the analysis of field transcripts was purely focused on searching for units of meaning and the assembly of these into aggregated conceptual structures. There is no attempt to make any truth statements about hypotheses. A key reason why the theory building has arguably been successful is that it is has not been constrained by a need to prove evolving ideas at each twist and turn.

The fourth consideration is that grounded theory is not simply the routine application of formulaic technique to data. While a large proportion of the data analysis for this study has involved mechanical exercises of cataloguing and sorting data, creativeness and innovative thinking have played a major part. The attainment of an overall model would not have come to fruition without several eureka moments occurring during the coding activities.

The fifth consideration is that grounded theory is not perfect. While every attempt has been made to adhere to grounded theory ideals, there are perhaps weaknesses in the some of the research steps undertaken in this study. These have been discussed above. Specifics weaknesses in the research findings are subsequently addressed in Chapter 7: Final Reflections.

The sixth consideration is that grounded theory is not easy. It is with some relief that I saw this as a criterion against which my research should be matched. While the final output from choosing a grounded theory paradigm has been hugely satisfying, there were points on the long grounded theory data collection and data analysis journey at which I thought that if I had chosen a positivistic study, this would have been much simpler to complete and a much lower risk option.

In summary, Suddaby's six common grounded theory misconceptions have been taken into consideration in carrying out the research. As well as being appropriate method for the research, the experience of applying grounded theory was in line with what is normally expected. While a number of imperfections have been identified in the research process, it is argued that these do not undermine the core philosophy or rigour of the study. In line with Suddaby's (2006) suggestions that grounded theory is not easy and is not perfect, the discussion of this study reflects those factors. However, by staying true to

grounded theory principles the research has managed to navigate itself to a suitable endpoint. Overall, this research has followed a grounded theory process and the findings of the study are empirically grounded (Strauss & Corbin 1998).

3.5.5 Style of narrative

Writing up a qualitative methods study in a clear and coherent manner that engages the reader and persuades them study is of interest and is contributing something relevant and of value is a challenging task (Alvermann 1996). My approach to planning the text has involved developing a thesis architecture that provided an overall structure that the components of the written manuscript would then fit into (Strauss & Corbin 1998). The order of the writing then followed a logical route through the thesis architecture so that the reader is always aware of where each topic area fits relative to everything else. The author has attempted to tell a story through the thesis and hence the narrative has included orienting the reader with the problem context, taking them through the ups and downs of the research journey, capturing the excitement of the new theory emerging, answering any questions or doubts the reader may have, and bringing the story to conclusion, but also opening directions for further research. The thesis architecture was illustrated in Figure 1.3 in the introduction chapter.

3.6 Summary

This chapter has selected and justified an appropriate research paradigm for the study. As there is limited existing literature that illuminates the problem and I was keen to investigate the running of a software business with an open mind, grounded theory was chosen as a basis for the methodology. The end-to-end approach for collecting data for the study has been detailed. The procedural approach used for the data analysis of open coding, axial coding and selective coding have been explained.

With specific regard to the research objective of searching for answering the question 'can a holistic guiding management framework be developed for running a COTS application software SME business?' the grounded theory approach has demonstrated itself to be a highly effective choice. The field data collected, findings determined and resultant theory constructed have all surpassed my prior expectations.

4 DATA INTERPRETATION

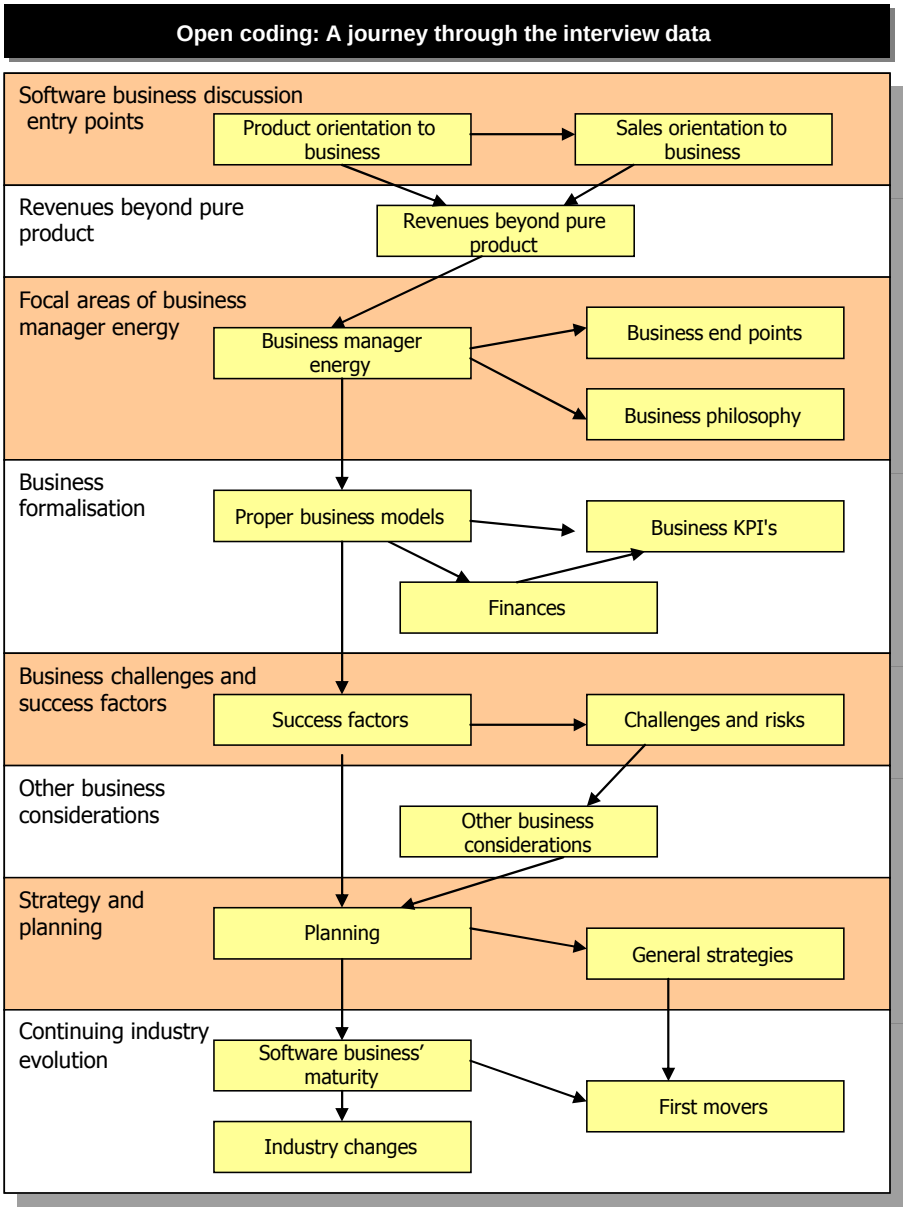
4.1 Introduction

The transcription of the thirty field interviews generated over 205,000 words of text. The first step of the data analysis, data interpretation, involved the scanning of the interview transcripts, the search for units of meaning in informant responses and the generation of base-level open codes. Chapter 4 documents the data interpretation process and summarises the initial set of codes that were created from this exercise. A total of 93 base-level codes were generated from the initial open coding.

While the actual process of coding the interviews was undertaken by sequentially working through the transcripts one-by-one, across the complete set of data, the order that codes were generated, mirrors the general topic areas that the interviews systematically worked through with informants. On aggregate, the interview conversations broadly fell into eight broad topic areas. Loosely following the order that they were discussed, these were: software business discussion entry points; revenues beyond pure product; focal areas of business manger energy; business formalisation; business challenges and success factors; other business considerations; strategy and planning; and continuing industry evolution. Using these eight topic areas, Figure 4a illustrates the journey that was undertaken to interpret over 205,000 words of interview text and transform these into a set of base-level open codes.

Chapter 4 is structured into eight main sections over the next eighty pages, one corresponding to each of the eight broad topic areas. Each section documents the open coding process that occurred and how the data spoke at each step of the analysis. The codes generated within each interview topic area are identified and the associated concept for each described.

Figure 4a: Open Coding Narrative



The main body of this chapter contains quotes made by informants. While there are a number of quotes that underpin each of the codes identified in the open coding, generally only one quote has been chosen to represent this point in the thesis text. The rationale for this is that it will make the chapter flow better; while the consolidation of quotes of a duplicate nature will also make the chapter more concise. The intention is that fewer, but more insightful quotes, will make the chapter much punchier and of higher impact to the reader. Where quotes from informants are provided in the text, a suffix at the end of the quote links back to the anonymous interviewee profiles listed in Table 3a.

4.2 Software Business Discussion Entry Points (Codes 1-12)

The field interviews all commenced with a general conversation that asked informants an open question about being in the business of COTS application software. Most informants appeared to gravitate naturally to talking about software product as an entry point to the discussion. The remainder tended to lean towards software sales an initial talking point.

4.2.1 Product orientation to business

Product R&D (Code-01)

Product R&D capabilities was an area that informants were keen to talk about. Their views were consistent and supported well documented IS research and the industry trade press. Historically, the software industry has been widely criticised for poor quality products that were routinely delivered late. Poor project management, a lack of processes and standards and lacklustre QA standards were cited as explanations. More recently, the software product R&D business has made great gains and matured as an engineering discipline. The importance of developing quality product in a reliable way that follows engineering principles and practices was flagged.

The discussion on product R&D led to the topics of product renewal, the need for product direction, software purpose, fitting customer requirements to product being introduced.

Software capabilities (Code-02)

While vendors tend to always have a long list of new features that can be added to software, various other software capability factors were raised as just as, if not, more important. It is critical that the product is easy to use. Product simplicity is an enabler of this. Product flexibility is attractive for customers as it gives them the ability to configure

the product to the specific nuances of their business. The product needs to be easy to maintain, with product upgrades being seamless to customers. The interoperability of the product with customer's other software rated as having value.

> *There's big slabs of a lot of the product that don't get used … make sure that the ninety per cent of things, that ninety per cent of people are going to want to do, are easy and apparent* (I-12).

The majority of respondents stated that only a limited number of features in a product were actually used.

Product renewal (Code-03)

The short product lifespan of software was continually mentioned. It was suggested that there is a need to re-engineer a software product every six to ten years. However, this can vary from business to business. In the dynamic software industry, if regular improvements in product are not forthcoming, then there can be a danger that the business will slide into decline as its offering becomes less competitive. Therefore, R&D features heavily in short, medium and long-term business plans.

> *No product stands still and if you actually think that you can have a product that you are not going to be pumping investment back into then you are smoking the wrong stuff. If you are not getting investment in your product, I think you will fold pretty quickly* (I-04).

As a software product matures and a business becomes established, it is tempting for a vendor to cut back on R&D and only make small low cost product improvements. If a vendor fails to invest sufficiently in keeping its product current over time, a point can arrive where its ability to evolve the product to the next level becomes limited. However, there are numerous cases of vendors becoming fixated with developing the next product version that they convince themselves is critical to securing the next sale. This often occurs, even though plenty of existing software product is in existence in the vendor's portfolio.

> *Look, that's always an ongoing battle, is how much do you keep improving and how much do you say, 'Hang on, we've just got to sell what we've got* (I-19).

Need product direction (Code-01)

Informants revealed that product roadmaps are often fluid and relatively informal. However, a vendor's strategy and roadmap provide terms of reference for the product direction.

> *Instead of just talk about product releases or product features; you are going to actually talk about the underlying* [roadmap] *and put a spiel on your realisation of why you have done it* (I-27).

Quoting one informant, if a key part of your strategy is going global, you will need to develop a double byte version of your product to support Sinoxenic languages that require Chinese Han characters.

The notion of building the 'killer application' and the customers will come was largely passed over as a fallacy. The involvement of market participants and customers in the process of developing product is much more likely to result in a commercially successful software offering. User community forums and product committees are mechanisms that a vendor can utilise to gain rich inputs on what customers desire in a product and what current shortfalls need to be addressed. Market changes in platform technology can also drive the product roadmap. The latest technology may be required to enable new capabilities within the product.

Software purpose (Code-05)

Software can contribute to the creation of business value in many different ways. Often vendors start by identifying shortfalls in an industry where they believe they can significantly improve specific processes or activities. Automation of a traditionally paper based or *ad hoc* business process is a common area in which software has contributed to

improving the efficiency of a business' operations. A more sophisticated approach involves searching for non-competitive areas of a customer business and developing innovative software product that can assist an organisation to make incremental leaps in their capabilities.

> *All software products are a pre-built solution to a business problem* (I-06).

The market size and number of potential customers for a business solution will vary based on the magnitude and frequency of the business problem that the software business is targeting. Unfortunately, the industry has many examples of vendors ploughing ahead without really understanding the customer demand for their offering. A number of informants stated that they would not introduce a new product line unless there was a definite launch customer.

Fitting customer requirements to product (Code-06)

A common view expressed by informants was that a vendor should aim to provide a base software product that meets a minimum 80% of a customer's requirements. Additional functionality customers require is then met through product configuration or bespoke product customisation. There were mixed opinions about how much customisation should be undertaken, although most informants felt that this was not a good investment from a customer perspective. Configuration that utilises standard templates or profiles was suggested as a preferable option. However, the ability of vendors to run a standardised product business model is generally an exception to the rule. Vendors seem to be caught up in the belief that base products must be tweaked to specific customer requirements before they are usable.

> *Our stuff tends to be easy to deploy … without asking anything of you, it should start doing useful things for you, a smile goes on your face and so it goes on* (I-12).

Level of customer engagement (Code-07)

A relationship between the fit of product to customer requirements and the level of customer engagement with vendors was raised by several informants. There are vendors who have virtually no contract with their end customers; at the other extreme are vendors who spend years customising and configuring product for customers.

A low customer engagement model has the advantage of limiting the challenges and costs of dealing directly with customers. It therefore provides the opportunity to achieve high profit margins. Using a channel model is a common way that vendors disconnect themselves from direct contact with their end customers.

> *The most successful in the IT arena like Microsoft are the ones who arrive to get close to shrink wrap and put millions of copies out there. I think as a goal, it is still probably the ultimate one* (I-14).

Various factors that contribute to whether a vendor has a low or high engagement model were offered by informants. These included product complexity and effort required to implement, the size of market and the size and volume of sales transactions, and the level of support required. However, it was suggested that a lower touch model with customers is now desirable for software businesses.

> *We're actually trying to move away from that* [high engagement] *model where we are trying to have less, the less involvement that we can have with a customer the better* (I-25).

> *We don't do services … We try to avoid that. We see ourselves as you know we sell software in a box … The margins are so much better on software than services* (I-12).

4.2.2 Sales orientation to business

Revenues (Code-08)

Those informants who leant towards sales as an initial topic of conversation, tended to focus the importance of making sales, attaining revenues and being paid by customers as fundamentals. In a software business, new sales can be irregular. Selling product is

instant cash, but selling solutions can involve a six to nine month wait until revenues are realised when a customised product goes live.

> *We're potentially in a precarious revenue position, because it's very hard to forecast revenue … where the revenue is going to come from six months from now, largely we have no idea* (I-12).

Recurring revenue is highly attractive. In contrast to new sales revenues, it is both certain and regular. Informants emphasised that recurring revenue is critical for a business to smooth its peaks and troughs.

> *Certainly recurring revenue is an important thing. Investors value recurring revenue higher than any other sort of revenue … Clearly if you can demonstrate a history of customer retention, the more reliable that recurring revenue is. I think it's a really important measure, an important theme* (I-02).

Support and maintenance fees and professional services were highlighted are key sources of recurring revenues. If a vendor has a big enough mass of customers paying a support and maintenance fee, this can provide a large enough annuity to support the ongoing business. With professional services, although not locked in to the same degree as maintenance fees, selling professionals can be easier to achieve then selling product.

> *If we sell $20M worth of software every year for five years by the time you get to the end of the fifth year, you've got $20M of recurring fees coming from that, and, more importantly the customers will probably stay for another five or ten years* (I-28).

A significant attraction is that every additional recurring maintenance and/or usage fee that is secured will add to a rising cumulative total of recurring revenues. As long as a vendor can grow its product usage and associated maintenance agreements every year, the total financial value of customers paying a recurring to the vendor each year will also rise.

> *The nice thing about maintenance is just the nature of it, that's the snowball, it means that even if you only sell the same amount of software to the same number of people each year, that over time your revenue is going to grow* (I-12).

After discussing the importance of securing revenues, many of the informants then extended the conversation into the broader topic of sales. Software salespeople, the sales pipeline, sales channels and the sales cycle were all flagged as points of importance.

Salespeople (Code-09)

The importance of directing and managing software salespeople was highlighted. A competitive and ruthless salesperson culture has featured at many software vendors. The result has often been a short-term focus on selling whatever can be sold regardless of its long-term profitability or strategic relevance. Selling the wrong thing, or selling software dreams that do not resemble a vendor's offering, is not only unprofitable but also commercially nonsensical.

> *There are a lot of* [sales] *people who are high adrenaline junkie AA type personalities who get into the kind of mind set of this kind of wild ride of selling large scale solutions into large organisations* (I-06).

Small vendors in particular, by virtue of their size, frequently do not have enough salespeople. Good salespeople are priceless.

> *It's the most underestimated key task of any software company, is to have a decent sales force* (I-30).

Sales pipeline (Code-10)

A strong and focused sales pipeline builds predictability to revenues and the entire underlying vendor business.

> [We are] *very strong believers in sales is not the seat of your pants … Salespeople follow very strict methodology … We have again, a very strict Opportunity Review Council. ORC sits down and says: 'do we bid, is it right price, can we win it and can't afford to waste resources bidding on stuff, we either don't think we'll win or don't want to win'* (I-11).

Informants categorised different types of customers that can be targeted. These include: existing customers, new customers and high-volume sales. In some industries, for example health software, which is a relatively immature market, there is predominance to new software sales. However, achieving new sales can be associated with a number of

issues: actually getting the sales over the line, implementation challenge and various other delays in realising the revenue. Different software markets require different levels of sales effort. A highly customisable software offering for instance requires a higher engagement model than that for an off-the-shelf product. A vendor selling an off-the-shelf product for example, may focus only on high volume transactional sales that have a low customer touch-point.

> *We make 50% of our sales currently to existing clients, so we don't have a huge new client target. It's only about, sort of only around 15%, 16% of revenue is new business* (I-11).

The sales recipient at the customer organisation can vary in profile. Selling software to commercial customers is more complex than selling to consumer markets. Vendors often have to sell to multiple audiences in a single customer organisation.

> *We've got quite a different mix of sales propositions, and therefore, potential sales order value categories … We have telesales, we have what you might describe as junior or mid-tier salespeople, we have senior salespeople, and the senior salespeople are supported by pre-sales product and technical specialists* (I-28).

Sales channels (Code 11)

The topic of sales channels did not really get much of a mention in the first half dozen interviews. However, in subsequent interviews, this was offered repeatedly as a key enabler of gaining greater market coverage and making extra product sales. There are number of alternative sales channels that a vendor can develop.

A direct sales model has some advantages. However, it has its limitations in terms of scalability. There is close contact with customers, understanding their needs and problems, and ensuring that they have they have a good overall software solution experience. It is largely dependent upon sales representatives having physical presence geographically in target markets, and it can only be scaled as sales force resources are

recruited. The model can also bring a considerable overhead of client management, which can be a strain on resources.

> *We find we do best with our own salespeople where they don't have an option; you know they sell our product and nothing else* (I-12).

Channel partnerships offer great opportunities to increase market sales. However, these sales channels require significant investment. One example given was of a global vendor that has 96% of its Australian revenue coming from partner channels. Channel development and management has a number of challenges. It takes time to deliver rewards and significant upfront and ongoing investment is required. Training partners on how the product works, how to sell it, how to implement it and how to support it were flagged as important.

> *Probably 50% of our sales now come through partners and the goal over the next probably 12 to 24 months will be 80% to 90% of sales will come through partners* (I-25).

> *You'll only get performance out of a channel if you manage it well and you put time into it and be patient … To keep a channel interested and to keep a channel investing to get that level of expertise is actually has a whole set of challenges in itself* (I-29).

The reseller model offers vendors a more hands-off, lower investment alternative distribution network for their product. The software is sold out of a box, with little or no guidance or support. Product may be sold by physical distributors in local markets or made available electronically through internet software resellers. The benefits of being able to reach a larger market in this way are exponential.

> *We struggle with the reseller concept though because our product is not the kind of boxed product* (I-13).

Vendors of complex and customisable software have often thought that direct contact with customers is required to sell and implement that class of software product successfully. Conversely, vendors of shrink-wrapped software packages have tended to invest more heavily in developing indirect sales channels. However, with the software

market space becoming more global, there has been a big growth in partnering and reselling. The development of indirect sales channels is increasingly an issue for vendors to consider.

> *Generalised* [channel] *enablement and the way they manage channels for me is being the big key* (I-27).

> *I can tell you now that running a channel operation … it is just plain hard work, it is agonising* (I-27).

Sales cycle (Code-12)

The software sales cycle can be anywhere from a couple of months to two years. Its extensiveness and time length is largely driven by the market customers, although this is also linked to a software solution's complexity. This process can involve: definition of software solution requirements, software demonstration, gap/fit analysis, proof-of-concept commercial negotiation and finally an agreement being reached. Closing a software sale can often be challenging; customers often recognise the value that a software product can provide, but they can end up procrastinating and drag out making a final decision on whether to proceed with a vendor. Pre-sales consultancy contributes to doing sales well. This helps provide a balance against making a sale at all costs.

> *Selling to government is a very complex and expensive process, they're very long sales cycles, incredibly expensive processes that we're actually pushed through … You need highly paid people and lots of resources and a lot of patience if you want to go playing that market* (I-29).

How to sell product is an important consideration, vendors need to be careful not to just push product onto customers. Sales discussions based around product features and prices tend to provide a wrong focus. There is a danger that customers can end up purchasing software products from which they receive little value.

> *Ability to communicate the product value to the client in the client's environment. I think that is one strategic and* [second] *fundamental* (I-14).

4.3 Revenues beyond Pure Product (Codes 13-20)

As the interviews evolved beyond the initial entry discussion topics concerning product and associated sales, many informants referred to other parts of a software business that contributed revenues. Support and maintenance, and professional services were generally viewed as important components of an overall software business.

4.3.1 Revenues beyond Pure Product

Support and maintenance potential (Code-13)

The attraction of support and maintenance, as cited by informants, was that it could be used to generate significant revenues.

One informant described support and maintenance revenues as like discovering a gold mine. Historically, many vendors have charged customers a sizable fee, while arguably not providing much in return. Many informants acknowledged this and suggested that this should not be relied upon. A support and business is just one component out of several components in an overall business model. It is supplied and enabled by the product R&D and sales business areas.

> *It's a gold mine but we respect it's a gold mine. One day it mightn't be there and we've got to do a lot of things to make sure we're not relying on it all the time* (I-30).

Informants suggested that support and maintenance as a business line has its limitations and that vendors should not over-invest in this to try to make it into something it will never be. The long-term viability of their support and maintenance models was questioned in line with a general industry trend that is pushing software vendors to demonstrate value across their whole software proposition. A rational and pragmatic approach was suggested when considering what the optimal size and shape of the support and maintenance business should be.

> *If you don't offer value for your money, then, regardless if it's services or support, or sales or whatever, it's irrelevant. People are going to walk away* (I-22).

After highlighting the potential of a software business' support and maintenance activities, further interview probing led to a number of related factors being identified. Support and maintenance levels, the customer value of support and maintenance, and its contribution to an overall software business were further examined.

Support and maintenance levels (Code-14)

A vendor can be positively influence its market perception is by providing competitively superior support and maintenance. If customers want help and assistance with software, and they want to talk to someone about it; it is in a vendor's interest to ensure a customer is helped to achieve their objective.

> *Support can, most definitely, be a big differentiator for us. Because again, most of the criticism from prospects and customers is that, you know, our competitors, the support is either average or pretty poor* (I-26).

Vendors can provide customers with a range of different support service level agreements (SLAs). The higher the support level, the higher the fee a vendor is likely to charge a customer for the privilege. Some vendors will limit support to working around problems with the current release of software product; others may actually make modifications to an existing product to solve a problem at source. Different support capabilities will have different vendor cost implications and different customer value levels.

> *The process that supports support and maintenance is critical – has got to work. If we failed on that basis, our business would fall apart. We could probably tootle along with a nice marketing pitch and win a certain level of customers, but the tail would catch up with us. The brand would fail* (I-26).

Customer value of support and maintenance (Code-15)

Customers with mission critical business processes relying on a vendor's software will be more willing to pay for higher levels of associated support. It is often not affordable or

justifiable for a customer to have an in-house specialist dedicated to supporting a software system. Purchasing a vendor support agreement allows them to have this insurance, but provides the advantage of them only having to pay based on consumption requirements.

> *The percentage of sale that we get of support is increasing, because as clinical usage in the software is increasing the dependency on the system rises and the client is more inclined to contract for 24/7 support, which pushes the fees up* (I-28) .

Charging an annual support and maintenance fee of around 20% of the purchase cost of the software has long been a historic norm for the software industry. The logic has been that customers required ongoing technical support to keep software operational and that they saw value in receiving latest product releases with new capabilities when they became available. In an immature software industry, this was viable. Nowadays, customers now expect mature products that are both capability rich and operationally stable. Customers are now less inclined to see value in paying for something that arguably should not be required in the first place. However, informants noted that limited numbers of customers actually reject annual support and maintenance fees outright.

> *A lot of customers don't see any value in it* [maintenance] *and 50% see a lot of value in it. The issue here is that if you don't upgrade your software, or if you don't add anything to it, if you don't listen to your clients, and if you don't communicate with your clients more than once a year, of course, the client is going to think that it's a rip-off. There's no value in it* (I-22).

Contribution to overall business (Code-16)

Support and maintenance can be easy money for a vendor to attain; it can be a highly profitable with margins twice that of the product sales business. Vendor costs associated with providing support and maintenance are usually low in comparison to the fees charged. Support staff are cheaper than resources for other areas of the business; and often customers consume virtually no support for the fees they pay.

> *They're usually your less expensive employees, and the software by the time it's in support, has been installed and is working, so the amount of work you have to do per dollar received is not too high* (I-28).

The contribution that support and maintenance makes to the overall business varies considerably for different software businesses. Vendors' support and maintenance business are generally the most profitable part of their overall business. Margins are in a wide range from 30%-90%. Percentage of total revenue and total profit also varies considerably: with total revenue figures spanning roughly 15% to 60%. At one end of the spectrum, there are vendors who place little emphasis support and maintenance. Although a convenient revenue stream, management give more head space to the perceived more exciting activities of product R&D and sales. In contrast, there are vendors that fully leverage support and maintenance revenues opportunities, maximisation of these revenues can become addictive.

> *Yeah, well we've got about sixty* [per cent of total revenues coming from support and maintenance], *and I'd like it to be higher, overall across the group, and I think if we could get that up to 90, or 95, I'd be even happier. It's the most profitable part and it serves a number of things* (I-28).

Services as a separate business line (Code-17)

Virtually all but a couple of informants stated that the software businesses they had been involved in undertook professional services work. Most conceptualised these professional services activities as a separate line of business.

A professional services business can be relatively easy to build revenue streams. Small pieces of services work can be sold progressively. The downside of a services business is that unlike a product side of a software business, in times of low revenue, there are still high fixed costs of consultants that have to be covered. Informants had mixed views on the strategic positioning of professional services, but most suggested that a services offering had potential to be structured and developed as a business in its own right.

> *I think it's very tempting to set up and migrate into a services, consulting services, professional services business … it can be very easy to make money … We are taking that now to a more defined strategic step of, I guess, setting up a formal business unit that will be a separate business* (I-26).

> *We won't do systems integration, even if the clients ask us; it's just not the best use of our time and people* (I-11).

Services are not the most lucrative software business line and profit margins usually do not exceed around 20%. Implementation in particular is challenging to run at a profit and is sometimes run as a break-even business line due to enabling profitable product revenues.

> [Services] *that's really just designed as a break-even business … Actually it probably is a profitable part of the business, but it's not the primary driver of the business and the revenue and everything that we're doing is to get that ongoing revenue* (I-25).

The discussions with informants around professional services as a separate business line led to a more detailed questioning of other factors that a software business involved in undertaking of professional services is likely to consider. Several were examined. What software services to offer? How to do services well? Can services supporting a product business?

Software services (Code 18)

Different types of professional services can be provided to complement a software product. Business consulting, implementation and training are common examples.

Consulting services provide customers business direction and guidance on selecting IT solutions. Done well, this can be high value and high margin line of business. It can increase a software vendor's profile; taking it into a broader market than that of just providing software solutions. This can used to open up and secure new product revenue pipelines. However, unless the consulting is very focused and provides high value outcomes quickly, it can be perceived by customers as being high cost and providing a limited contribution.

> *We are now actively in the market, selling a value proposition which is non-technology based, which is more around the process and the people*

> *elements, but the technology still dovetails into those other two concepts* (I-26).

Software implementation can often be complex and time consuming. Challenges include: clearly understanding customer business requirements, configuring the product to meet these and integrating the software into the customers IT environment. Service capabilities require expertise with both the customer's business domain, and the vendor's software product.

> *You know hospitals are going from paper, or very basic bits of IT that have no integration, to a modern integrated system and they need people that understand the hospitals and how they work … you're not going to find local system integration expertise, so we do it ourselves in those places* (I-27).

Training professional services can be a key to ensuring that customers get the most out of vendor software solutions. Services may also include ongoing training or mentoring to ensure that the software product stays a focal part of creating value for a customer business.

> *As a product company you absolutely provide training and mentoring on use of the product … you should be capable of providing this many training courses and that should occupy this much time* (I-13).

Doing services well (Code 19)

Important criteria for doing services well include having staff that have business domain expertise and extensive technical experience with the vendor's software product. Getting a software product working and delivering customer value is the main outcome they should be able to achieve.

> *The smallest software vendors really do struggle on the service side because they are dealing with quite larger organisations that expect a reasonably sophisticated offering … a lot of those* [vendor] *organisations just do not have that level of expertise* (I-27).

A vendor may have a good services business, but if this means that not enough effort is put into having a quality market leading product, then the strength of the overall software business will decline over time. Similarly, a vendor's product offering usually represents

the core of the vendor brand proposition. There is a risk that a move into other lines of business such as services can dilute and confuse the perception of the vendor in the market place.

> *If I spend too much time worrying about my services, I am actually going to lose track of building a leading edge product* (I-06).

Services supporting a product business (Code 20)

Professional services can be an important component of an overall software business. Services provide a bridge between a customer's business objectives and the solution capabilities of a vendor's software product. Informants suggested that they are a vital part of delivering a total software contract to customers.

> *You still need people to go on-site … Well, to date, we have always had a professional services business. That's essential as part of making customers happy* (I-26).

Professional services can lay the foundations for product sales; they can architect customer business environments so that IT solutions can subsequently be slotted in. Implementation of a software product can lead to opportunities to increase existing software product usage through the customer organisation; and extend the product footprint with product extensions and add-ons.

> *Cluster your services around the product … I have seen people try and to do the other way around and fall really badly* (I-27).

As part of a natural maturing and growth progression, many software vendors have built sizable and successful services practices. However, it was flagged that the services always have a limited profit margin as they have a people cost component.

> *If you get the margins that we want, you need to have a licence attached. You know, although we're half and half, say 50% does come from services, we don't consider ourselves a true services company* (I-11).

4.4 Focal Areas of Business Manager Energy (Codes 21-33)

The interview discussions around basic elements of a software business, product, sales and revenue sources, highlighted a number of areas where business managers were focusing their attention and energy. Growth was the biggest theme that emerged. Targeting opportunities for growth overseas was particularly popular.

4.4.1 Business manager energy

Growth (Code-21)

In a dynamic and evolving industry such as the software sector, there was a consistent view from informants that if a business is not growing, then essentially it is going backwards. In a crowded market, competitors can squeeze a stagnating vendor out of business. There are many large predatory vendors on the acquisition trail and there are numerous small start-ups with innovative and revolutionary new offerings.

> *If you don't* [grow] *you stagnate, you fall behind your competitors, and you become nothing really, you have to grow* (I-01).

While in some markets growth opportunities are plentiful, in others, growth can be elusive. Attaining growth often requires investment, the development of new capabilities and may involve a level of associated risk. Pursuit of growth, if not carefully executed, can result in significant loses.

> *It's essential that you keep growing with the times for sure. But it's got to be well directed, examples in Prophecy where we went out on a Unix play and came a cropper and wrote off $3.5M* (I-30).

Opportunities for growth internationally (Code-22)

Many informants were excited by the potential of penetrating overseas markets. The Australian market is very small and the overseas market is very big. In addition, the Australian software market is heavily permeated by overseas vendors.

> *In Australia … 2, 3 or 4% of the global market say in any given environment. If you were in the US and you're sitting on, whatever, 25% of the global market … We don't have the luxury of that, and I think it's mandatory that we think global* (I-02).

Precisely qualifying overseas potential in a given market segment was identified as being important. While the USA is a large uniform market, if a vendor is not first in, it may struggle to penetrate it successfully. In contrast, while Asia is a fragmented market in terms of different languages and many different local requirements, the market is less well served. Overseas expansion also involves significant investment and can be fraught with risk. International growth can be hugely expensive and fraught with risk. There is much disconnection across international markets. There was a concern that vendors often did not fully appreciate what they were getting themselves into before it as too late.

> *International expansion traps, exceptionally expensive to go out yourself overseas. People don't think that's an issue but it is, a major* (I-30).

Emerging markets are now becoming increasingly commercialised, where they were largely unserved by vendors previously. Software products are now more mature and lower cost options are available to lower-end customers. Many vendors are now rushing to tap into potential customers in these markets while a window of opportunity exists.

> *If I was thinking about where I would like to expand at the moment I would want to expand into China or India and places like that which has become rapidly commercialised … there is a vacuum for sales, so therefore if you can create market for your product is going to secure you an annuity or revenues* (I-06).

In the discussions about targeting opportunities for growth overseas, informants identified the potential need for a localised product, and the importance of focused and controlled growth internationally as important factors.

Localisation (Code-23)

Some localisation of a product is likely to be required when a vendor takes a software product to a different region. Foreign language software products may be necessary.

While the underlying technology exists to do this, it can still be a significant overhead. In addition, country specific functionality may need to be developed so that a product is capable of accommodating country specific ways of doing things. Informants advised that business managers need to consider whether the extra effort spent on developing such modifications is commercially justifiable.

> *We have done localised versions, we've done Korean, Chinese, Portuguese, French, German, Dutch, and they were all a complete waste of time. … But the reality is that we haven't sold ISYS to every English speaking person in the yet. So we're better off pursuing them and so we made the decision that we're not going to do any more localised versions* (I-12).

As well as product localisation, vendors need to consider a number of other factors when serving a new international market. Informants indicated that vendors need to have a local presence in a local market if they are to gain traction there. Intricate knowledge of a local market is beneficial. Setting up such a capability is likely to be an expensive and timely exercise with no guarantees of success. An alternative approach is to partners with organisations that already has knowledge of the target market and existing relationships within it.

> *Southeast Asian and even China markets … getting the right partner with the right connections at the government levels who in general is well respected* (I-04).

Focused and controlled growth internationally (Code-24)

Informants made suggestions on how developing an overseas business can be done in a focused, controlled and progressive way. Business and product are first moved into regional markets that are most similar to the home markets. Once a footprint is established in those new markets, the vendor then takes on the next natural extension into the next similar market. This approach provides the benefits or serving new customers, but limits the risk associated with this.

> *Once we get a piece of technology that we feel like we've wrestled into submission, we then take it to New Zealand and once we've worked out how to wrestle it into submission in a semi-foreign country, we then take it to*

> *Asia. We don't have any immediate aspirations to do what we do in any other continent right now* (I-15).

Following on from interview discussions about opportunities for growth overseas, opportunities for growth in adjoining markets and SME markets were also common focus areas.

Adjoining markets (Code-25)

Opportunities exist for vendors to take the core of their products into adjoining markets that have similar characteristics. One informant with an established software business providing a hotel services solutions considered extending their customer base to include hospitals and prisons. Taking a view that prisons were essentially hotels for bad people and hospitals were hotels for sick people; their thinking was the core of their product could be quite easily extended to provide a solution to these new markets. The larger a vendor becomes, the greater potential there is too diversify by re-assembling various parts of their product portfolio into new hybrid solutions for adjoining markets.

> *We start off looking at verticals that are close to what we do. … we've started by looking at markets that leverage from what we have. … but with each of those verticals comes a whole host of processes and supply base issues that we need to re-address for that specific industry* (I-19).

SME customer markets (Code-26)

There is much potential for growth in SME customer markets. Several things have brought the SME customer market into vendors' sights. First, software products have matured and become more commoditised. It is easier shift higher volumes and to spit out low functionality, or 'lite', versions of a more substantial product. Second, there are now lesser new sales opportunities with the larger corporate customers in some spaces. Third, the SME customer market is generally underserved by software vendors. A more complete solution, with limited implementation or customisation, is generally required for SME customers. Additionally, the SME market is more price sensitive. Where larger

customers can be focused on getting the best solution for them, SME purchases are likely to be more driven by minimising costs.

> *I think there is one key trend … a push into the lite market. It seems almost every vendor wants to play in that space now in some way or another* (I-27).

Context for growth strategies (Code-27)

The probing of informants on the subject of growth was rounded off by informants stressing that growth needs to be underpinned by solid foundations. Therefore, it is suggested that the context for growth strategies is important.

A very strong message was that vendors should stick to their core business and concentrate on excelling in that. In addition, there is a real danger to a vendor's business if transactions are not underpinned with a robust commercial model, or if the business has no long-term focus.

> *We said, 'Let's stick to what we know' … because we were successful at it … we walked the walk and talked the talk in this industry, and we're doing pretty well at it, so let's just stay there* (I-19).

Although there is much industry hype about industry M&As, organic growth still offers many opportunities to expand for many vendors. Looking after existing customers and serving them well can be lucrative in terms of generating follow on business. Strong organic growth can also be achieved by expanding existing product into adjoining customer markets, or taking it into new geographical areas.

> *I have already got a very strong customer base hence the new sales will grow organically. I am simply making sure I continue to service those customers that I have* (I-06).

Continual increases in sales were flagged as being a crucial part of staying in business. As well as funding the operational elements of the business, spare capital is required to invest in growth. With regard to profits, many vendors have pursued opportunities to grow

revenues at the expense of profit. It was suggested that vendors need to be clear on what is more important at that point in time, revenues or profit margins.

> *It's the old story, are you better off, with say $4M revenue and make $1M, or $20M revenue and make $1M* (I-20).

The energy that informants demonstrated for the topic of business growth, led to a search for references points for this to be grounded within. The interview questioning then probed for informants to explain what they perceived as business endpoints and driving business philosophy.

4.4.2 Business endpoints

Long-term objectives (Code-28)

Knowing what business you are in and having clearly defined business goals were identified as imperative for a software vendor. A set of goals will provide a logical and meaningful path for the vendor to develop the over time. An example standard set of goals could be: build a competitive product, sell the product to a target market, progressively scale the business, go international. However, it also became apparent that many software businesses just evolve over time and do not have clear objectives. One informant's response captured a theme echoed by several informants who were business owners. He presented a picture of happily drifting along doing the things that he enjoyed. They seemed clearer about what they were strategically opposed to, rather than having clarity about what they were actually focusing on.

> *I think you've got to decide what are we in this business for … we've said, we're now in the business of acquiring businesses that have their own IP, software IP, that is globally competitive … We will nurture it through the early stages and then we'll sell the business to somebody who's bigger, much bigger than us, who can use their marketing muscle to make zillions of dollars which we can't do, because we're not big enough* (I-30).

While growth was the dominant long-term objective stated by informants, future take over, lifestyle companies and continued existence were also featured.

Future take-over (Code-29)

A goal of selling the company at a future point in time was a clear objective for a number of informants. Reasons for this included realising a financial return on an investment, recognising a necessity in the competitive market dominated by larger vendors, and a personal interest in starting up companies as opposed to operational running of a mature business.

> *The business was always going to be designed and run and developed with the ability to exit back out of the business* (I-25).

> *We have a five year goal to achieve a certain amount of revenue, a certain level of profitability … and have a profile where you're going to be noticed by a multinational* (I-13).

Lifestyle companies and continued existence (Code-30)

Informants suggested that not many SMEs are at an embryonic stage in the business cycle and therefore just had a very short-term focus. They needed to run their business conservatively to ensure their own survival. A few informants also described examples of life-style companies. These are privately owned SMEs whose owners had a nice little business working with clients they liked, product they were proud of and a close group of likable colleagues. They did not want to risk losing all this by taking on any ambitious changes for their organisation.

> *Some businesses I believe are set up as what I would call a lifestyle type business. Someone is running a business because they enjoy doing it and they want to build a lifestyle around it and I think that comes with its own challenges and goals and expectations* (I-25).

4.4.3 Business philosophy

Vertical and niche markets (Code-31)

Informants' comments on software business philosophy were varied. They ranged from having virtually no conceptual picture of what a software business was all about, to those

that had a very focused perspective on exactly what business they were in. Key themes that emerged were vertical and niche markets, domain leadership and IP management.

There are still many vertical and niche markets where there is great potential for growth. Informants qualified that these markets can often be less crowded. However, there is a threat from mega-vendors who are increasingly moving into vertical markets. ERP vendors, for instance, are progressively developing more specialised functionality in their systems. The ERP attraction is that a customer can just use one system as opposed to multiple products, while the attraction of a specialised vendor is likely to be that their product provides a richer business solution in that one specific area.

> *I think over time, all software vendors have recognised … you need to be very niche and focus in verticals* (I-29).

Domain leadership (Code-32)

Dominating and leading a domain segment is an approach for achieving significant competitive advantage and attaining business growth. Aspiring to be a tier one global niche vendor was cited by a number of informants as their long-term objective.

> *So we aim to be the largest supplier of back-office software for financial services in the world outside the US; that's our three-year aim. We don't do front office, we don't do trading systems; we really just do the back end record keeping systems* (I-11).

Identifying and defining a unique market segment that a vendor can serve and ultimately own was stated being a key objective. Locating a unique and attractive segment to serve is likely to involve working out where no one else is competing. Being a market leader is likely to include a number of things: having the biggest majority share; having a superior and/or differentiated customer value proposition; being the most well known in that segment; and being close to customers.

> *We've carved out this niche in hospitality and it would be very hard for someone to hit us in hospitality in this space, because we're now known in a lot of places as the people that do this* (I-19).

Intellectual property (Code-33)

A vendor can have the best software product in the world, but if it does not control the product IP, this can be a big risk to its business. This can include domain, product and client knowledge. Products copied illegally and counterfeit inferior competitor products may lead to lost revenues. Software that is altered illegitimately may become unstable and unsupportable, damaging a vendor's reputation. Informants were unanimously in favour of protecting IP. The preferred approach was to ensure that software businesses owned their IP wherever possible.

> *Whatever contract we go into, we would be positioning ourselves that the IP is ours, no matter what* (I-17).

Capturing and retaining IP can be challenging when IP is wrapped up in people's heads.

> *Most of the IP's up here, in these guys' heads … it's a dilemma for us about how do we retain that knowledge … So, we're certainly looking at tools to be able to capture that domain* (I-23).

4.5 Business Formalisation (Codes 34-50)

At around the mid-point in most interviews, when there had been a considerable discussion on a wide range of factors involved in running a software business, the questioning moved into understanding what formalisation of software businesses occurred. Informants were asked specifically about what formal business models, financial models and key performance indicators (KPIs) were utilised by software business managers.

4.5.1 Proper business models

Business model existence (Code-34)

Informants suggested that having a formal business model is increasingly fundamental. It provides foundations and explicit terms of reference for a vendor's goals, directions and performance. However, some software vendor models are crude and often naïve. Indicative costs and revenues metrics for selling, implementing and servicing customers are always not grounded in any commercial reality. Business models tended to evolve and refine over time. As informants were unaware of reference materials in this area, software business models and metrics are largely based upon personal experiences.

> *Need to have essential business models that assure the organisation's fundamentals and goals* [are] *being met* (I-14).

The discussion on software business led into questioning about what was the centre of a software business.

Software business centre (Code-35)

There was a mixture of views on what comprised the centre of a software business. With slightly different informant perspectives, this subject turned out to be slightly more complex than first envisaged. It was argued that a software applications business should have a software product at its centre. With all other parts of the business need to be traceable and linked backed to this software product in some way. However, IP is also a critical ingredient that sits behind a physical software product. It is the IP that exists in a software business that can enable the product to be effectively deployed and used by its customers. If IP is managed and leveraged successfully, a vendor can use this to differentiate itself from its competitors and achieve competitive advantage in its market.

> *If you are in software vending, it is primarily around the product* (I-27).

> *I invest considerable sums of money in developing a piece of IP, Intellectual Property, which can give me a franchise to sustain, and in fact grow the company, and then earn more revenue in order to develop more versions, or better versions, or greater versions of the software* (I-21).

While many companies have great intentions to explicitly capture and codify information about their products, processes, customers and markets, more often than not this information resides in the heads of the company's people. It is the knowledge and experience wrapped up in people, that can often enable a vendor to leverage opportunities and do the right thing at the right time just when it can really make a difference.

> *Success for a software vendor has more to do with their people than it does with their product … It's all about having the right people and understanding vertical industries, that's absolutely the key* (I-29).

Product, IP and people may be key aspects of a software business core, but it was suggested that they cannot exist in isolation. The linkage between a product and a market is the fundamental component that underpins the core of a software business. Informants articulated that a software business core should constitute a product that solves business problems. Specifically, having a close linkage between a software application and vertical markets was emphasised. A vendor's business focus and associated financial models can use this as a foundation and as a point of reference.

> *We've very much focused on what are the business issues of that market place and* [we] *address them with product* (I-05).

Management accounting (Code-36)

Software business financials turned out to be a big theme. Exploration and probing in this area stimulated informants to bring up the topics of management accounting, capital costs, operational costs, profit, profits and cost centres and cost models. As software business financial is an area in which limited specific reference literatures have been found, a considerable amount of time was spent discussing this with informants.

Profit centres encompass product sales, professional services and support, and maintenance. Cost centres include product R&D, and general and administrative. The

relative sizes of each of these profit and cost centres, and how these interact together provide the dynamics of a vendor's financial model. This total financial model can be broken down into: revenue management; cost management; and profit centre management. Each of these areas has their own metrics.

> *Our model is basically: what's it going to cost to build? What's it going to cost to maintain and then, what's our margin going to be? … There's not a lot of variable cost in what we do* (I-23).

> *Let's say 20% of that* [costs] *is R&D which you should write off every year, not capitalise and you might need a sales force of 25%. So you should be able to make 50%* [margin] *on the licence. Definitely should be able to make 50%* [margin] *on consulting. Definitely should be able to make 50%* [margin] *on services, on support, maintenance* (I-30).

Appropriately matching costs and revenues was stated as an imperative. Mismatches between revenues and costs have resulted in the downfall of numerous software vendors. Ensuring that there is enough cash to pay the bills is critical. While revenues can follow a curve of peaks and troughs, the costs of running a software business are reasonably even.

> *Cash is king, that's such a statement that a lot of people just forget about. You must ask yourself monthly, 'can we pay our debts as they come through', and that doesn't happen in a lot of places* (I-30).

4.5.2 Finances

Capital costs (Code-37)

It was noted that mid-sized companies operating in this disruptive industry sometimes struggle to make the long-term financial commitments necessary to stay competitive.

> *I think you've always got to have money available. You never know when the right opportunity would come up to maybe buy a competitor … You never know when you have to get a large deal where you need to develop particular add-ons to your existing products.* (I-22)

Capital investments, such as product R&D, are essentially about speculating what the market needs/requires and predicting potential ROI. Capital costs may be a high proportion of total costs when a vendor is upgrading or developing new product; in

contrast, they may be very low when minimal modifications to a vendor's product portfolio are being made.

Operational costs (Code-38)

Software business fixed operational costs are relatively high in comparison to other industries. Staff costs can be over 80% of the total costs. Unless a vendor utilising contractors and outsources parts of their business, the ability to rapidly reduce overall staff costs quickly can be limited.

> *Salaries of people are 81% of our costs so* [it is] *pretty tightly linked between number of people and revenue* (I-11).

Informants had a range of opinions about an appropriate level for cost of sales. A software sale may take place over a lengthy period and involve a significant amount of free of charge pre-consultancy work. Some suggested that sales costs could be as high as 25% of revenue. Others were adamant that the direct cost of sales should be no more than 10% of revenues. An important consideration is the lifetime value of a sale. A customer that uses a software product for a ten-year period for example, will make certain payments over that period, while the vendor will incur a number costs as well.

> *We have a policy of saying, 'okay, how much of the annual licence fee is maintenance', and it varies quite frankly from the old legacy products, it's 90% is licence renewal, because we only need 10% to maintain the product, or in Basis2, it's about 50,50* (I-30).

Profit (Code-39)

The profit potential of the different areas of a software business varies considerably. Support and maintenance is the most profitable area of a software business, followed by software product licences fees, and then professional services. Implementations are the least lucrative. Support and maintenance has high margins as once the software is implemented and running, limited vendor effort is required to keep it operational. In addition, the individuals working in that area are usually the least expensive employees.

In contrast, selling product, while delivering high revenues, has high costs associated with it due to the requirement for expensive salespeople. Professional services is the least profitable as there is a high fixed costs of paying staff regardless of whether they are billing or not; and in the case of implementation there is often a risk of engagement blow outs. On the overheads side of the business, the level of spend on product R&D can have a significant influence on the overall profit level of the company.

> *Really the swing factor* [in overall profit level] *is how much you spend on R&D … So R&D spend is probably slightly higher but about 25% at the moment, so that knocks 5% off your margin* (I-11).

What constitutes a viable and acceptable gross margin for a software business generated lively discussion amongst participants. On balance, a target gross profit margin in the range of 20%-30% for the overall business appeared to cover the broad spectrum of feedback. As a vendor establishes itself and matures, there is the opportunity for profit margins to move upwards as well.

> *We look for margins 30% and above … We'll draw the line at about 20%, but there's not too much that we run that's not making 30% or more* (I-23).

Profit and cost centres (Code-40)

Informants quoted an industry rule-of-thumb known as the 'rule of thirds'. A software business should aim to achieve a third of its revenues from new product sales, a third from product support and maintenance, and a third from related professional services. With revenues apportioned in this manner, a vendor will have a balanced revenue model that allows for revenue growth, but also has a robust foundation of income to fund the core business functions.

> *I think it's one of the valid but simple KPIs in our industry … there's 1/3 from, if you like, maintenance revenue, 1/3 from new business, and 1/3 from services. I actually find that that model holds up pretty good* (I-02).

Informants provided examples of mature software businesses that had recurring revenues in the range 50%-70% of their total revenues. Recurring revenues can take a long period

to build up. Substantial time and energy is required to achieve a position where vendors have a critical mass of customers using their robust software products. However, several informants were sceptical about whether software business could achieve high recurring revenues from support and maintenance in the long-term.

> *The business model moving forward is as an ongoing revenue model, and that's really to increase the value of the business. If we ever were in a position where we wanted to sell or exit the business, if we've got ongoing revenues attached to contracts, that will be worth a lot more* (I-25).

> *I'm sceptical that the market will allow that* [maintenance fees] *to continue and therefore I don't want to base my business basically on the assumption that that maintenance profit stream will be there* (I-13).

Informants advised that the cost of product R&D is a discretionary figure. Set too low, the business may not be able to stay current with innovative products; set too high, the overall business becomes unviable. R&D level of 20% of revenue is a logical percentage taking into account the other business components. The upper product R&D limit was restricted by achieving a base overall profit margin, while the lower limit was restricted by having an investment high enough so that product advances could be achieved.

> *40%* [overheads] *should be a comfortable model to reinvest in R&D, keep your people trained and have a few salesmen that don't work, because you're going to have that* (I-30).

Cost models (Code-41)

Product R&D and general and administrative costs are the main activities that are not directly supported by revenue generating functions. In the early days of growing a software business, securing investment and revenues from key market customers is critical. The development of an annuity stream of recurring maintenance revenue is a common approach used by vendors to support their business overheads and R&D. Organic growth is often internally funded. Growth through acquisition tends to be funded by external capital.

> *We're supporting our own growth with the revenue that we're building* (I-19).

A number of high-level business variables influence a vendor's cost model. A high customer engagement model for instance will involve high costs on pre-sales, implementation and support. Internal provision of all capabilities will result in high fixed costs, whereas the use of external sourcing of capabilities, for example distribution channels, will lower fixed costs relative to variable costs.

> *Depends on your business model* [and] *whether it is high engagement or low engagement. If you focus at the high engagement then you will have a lot of funding sales and support people* (I-14).

4.5.3 Business KPIs

Metrics for marketing and sales (Code-42)

Discussions with informants on formal business models and financials logically evolved into the areas of business KPIs. Similar to the software business financials subject area, software business KPIs is an area in which limited specific academic reference literatures have been found. Therefore, a considerable amount of time was invested in this with informants. Informants identified KPIs across a full software business spectrum. Topics included metrics for sales and marketing, metrics for financial variables, metrics for professional services, metrics for customer satisfaction, holistic performance management frameworks, metrics for support and maintenance, metrics for product R&D and metrics for strategic assets.

Informants identified a wide range of marketing and sales KPIs. Progress against revenue sales targets are the most closely watched KPI. Each individual salesperson has a target of revenue he or she needs to attain. The aggregate these make up the total sales targets for the overall business. Targets are generally set for a financial year, although to ensure adequate cash flow for the business, salespeople are also likely to have monthly and quarterly targets to ensure continuity. While some vendors have KPIs on the number of

prospects identified and number sales calls made, the focus appears to be on results, as opposed to inputs to the sales process.

> *Sales is solely about performance of a salesman against quota* (I-02).

Measuring the number leads and estimating their potential can quite easily and regularly tracked. However, measuring brand value is much more difficult. Most informants suggested a subjective call would be made for this, based on anecdotal evidence.

> *Marketing is based on lead generation and so we have KPIs about the number of qualified leads that are generated through different campaign types* (I-12).

A vendor's business pipeline and qualification of the strength of potential business at the various stages of prospects through a sales funnel are key business metrics. Measures for number and size of unqualified leads, qualified opportunities, proposals and finally closed deal, then follow. Metrics for the number of potential customers entering the pipeline and the conversion rate at each stage of the funnel can assist in managing and administering a sales pipeline affectively.

> *Our real measure is the health of various stages of the funnel. Be that pure unqualified opportunity right the way down to the ten or twelve deals that are in a state of close right now* (I-15).

Vendors frequently measure the cost of a sale as a percentage of the revenue that the sale will generate. There was variation amongst respondents as to what an acceptable cost of sale should be. The range went from 10% up to 40%. Vendors that are in the business of tailoring their products significantly to fit customer business provided the examples of those with the higher cost of sale. Those operating more commoditised product business would be at the other end of the spectrum.

The inclusion of non-revenue targets are starting to feature in performance criteria of salespeople at some vendors. These include customer relationship and satisfaction based measures. The logic in measuring and tracking these as a KPI is that a strong relationship

with a satisfied customer over the long-term, has the potential to produce greater returns than an opportunistic salesperson exploiting a customer in the short-term. One informant mentioned a software business where they purposefully have a mix of revenue and non-revenue metrics. They call their non-revenue measures 'commitment-based incentives'. However, although informants acknowledged the sense in this practice, most appear to be primarily focused on revenues.

Metrics for financial variables (Code-43)

Financial business metrics consistently surfaced. Specifically, revenue and profit were identified as the two most important measures of a business' current health. In addition, an extensive range of financial figures can be calculated to measure and track the performance of a vendor business. Cost metrics of importance include: the types of costs that exist; product development and support costs; staff costs; costs of sales; and sources of funding. Revenues metrics covered: different revenue sources; recurring revenues; sources of recurring revenue; and the lifetime value of support and maintenance fees. Accounting and profit metrics include: revenue splits; cost allocations; recurring revenue's contribution to total revenue; margins for each business area, overall profit margin, and the long-term vendor ROI.

> *Profitability and your sales revenue; I think that's always going to be the guts of it* (I-10).

Metrics for professional services (Code-44)

Utilisation levels of professional services staff stand out as the primary and often sole measure of a services business operation. Target utilisation rates stated by informants ranged from 75% up to about 95%. Value-adding activities that services staff could partake in when they were not billing clients, include IP creation and business development.

> *Services are solely about utilisation. Can you, you know get 75% utilisation that is billable time from your services people* (I-02).

Target profit margins for software related professional services extended a wide array. The spectrum ranged from a zero to 50% margin. At the bottom end, those vendors who saw services as just an enabler of product revenues stated that if implementation project broke-even, then that was an acceptable return in its own right. An acceptable range for general IT services of 20-30% was suggested. While specialised and boutique consulting could command premiums of 30%-50%.

An often-overlooked indicator is a measure of a services engagement success. A successful engagement is one that comes in on time and budget, and is hence profitable; combined with a satisfied customer that will embark on a long-term relationship with the vendor. In the first instance, completing a project usually triggers payments for services rendered and activates licence fees for operation of the live software system. Over the longer term, a satisfied customer will be likely to purchase further product and services over a period of many years.

> *The primary KPI is the time it has taken to implement a solution … the shorter we can basically, you know, finish off an implementation, the sooner we get paid, and the sooner we can take on something else* (I-05).

Metrics for customer satisfaction (Code-45)

Having transparency of customer experiences, good and bad, was deemed important. Providing external visibility to aggregate customer satisfaction levels helps develop a culture of openness with customers and provides vendors an incentive to improve satisfaction levels. Customer satisfaction can be used to provide a barometer of what a customer feels and experiences when dealing with a software vendor and consuming their products.

> *Customer satisfaction, to me, is the main driver of what makes us, or what will make this business successful* (I-26).

While historically customer satisfaction tends to have been measured on an informal basis, vendors now appear to be becoming more formal and scientific in the techniques that they use to qualify this important business indicator. Vendors can send a questionnaire for completion by customers, or to be more effective, actually go out and undertake a structured face-to-face interview. Questions cover all aspects of the customer experience with purchasing and consuming the software, and although customers to raise issues or shortfalls that are important to them. Regular contact with customers to capture any change in satisfaction level is identified. Views on the frequency for this varied from quarterly to yearly.

> *Getting senior management to actually stand in front and actually shake customers' hands, and to actually listen to them, rather than hiding behind their army of troops* .(I-22)

> *We are structuring a more quantitative survey approach … going out with a clipboard and a form, basically, and interviewing the customers … The objective is going to be to capture evidence and examples of what's good, what's not good, but also get down to a benchmark* (I-26).

One hard quantitative metric for customer satisfaction is the customer churn rate. This is relatively easy to measure and provides an implicit guide to how the vendor's offering stacks up against its competitors.

Holistic performance management framework (Code-46)

The value of having an overall performance management framework was raised with informants.

An overriding framework of some sort is required to integrate and bind together all business metrics in a meaningful way. Each business area should be aligned with the vendor's goals and explicitly contribute to the business' internal value chain. This should include a hierarchy of metrics and indicators.

> *The organisation is broken up into sales and marketing, finance and admin, customer support, engineering which is R&D, our professional services group, and each of those groups have KPIs and metrics that they work against* (I-29).

Several software business specific performance management frameworks were suggested. One perspective was that a vendor's KPIs should be based upon 70% financial metrics and 30% on customer satisfaction measures. A second offered three top-level business indicators of: monthly financial reporting, staff appraisals and staff climate surveys. While a third proposed that a software business' performance can ultimately be simplified into the three fundamental KPIs of product sales, staff utilisation and product roadmap progression. It was generally suggested that a more comprehensive performance framework of KPIs is required to provide an accurate and more holistic perspective of a vendor's overall strength and its long-term performance potential.

> *So in simplistic terms, our business is about road map, utilisation, and sales quota, and they're the dominant. I think you can build up other, all sorts of KPIs but they'll all be driven by those three simple measures* (I-02).

Metrics for support and maintenance (Code-47)

The main KPIs for support and maintenance raised by informants were: support and maintenance fee, revenue as a per cent of total revenues, support and maintenance margin, support capacity and resourcing levels, issue resolution times, performance against customer SLAs and the total number of support issues. It was noted that some companies have no explicit support and maintenance KPIs; however, most some form of performance measure for this business area.

The fee charged for software support and maintenance is set at a standard rate across the industry. A charge of around 20% of the initial software licence cost is billed to customers yearly. The figures quoted from the field ranged from 18% to 21%; although there was one suggestion that rates should be up to 25%.

> *We're still on that old fashioned model where you buy the thing outright and you pay 20% a year for as long as you want to keep it alive* (I-13).

Some informants suggested that support and maintenance revenues should be as high a per cent of total revenues as possible. A figure of 90% was offered. An opposing view was that these should not make up more than about a third of total revenues. The argument was that although this may be easy money, it is not a sustainable business in itself as customers are receiving questionable value.

> *We are trying to increase our maintenance and support revenue, and I think we, we need to get that round about 60-70%, we're about 52% at the moment* (I-20).

The profit margin for support and maintenance generally tends to be very high. Most informants quoted profit margins for this area of the business of 60%-90%. While support and maintenance fees remain static, the cost of providing this service can vary year by year. When vendors release new product versions to the market, or when they automatically make regulatory updates, this can push their costs up and margins down. Similarly, in years where customers operating mature stable products, margins can be high.

> *Support and maintenance can run at 60%, 65% margin in years for us when there's no legislation changes. When there's legislation change, depending how big they are, probably comes down to thirty* (I-11) .

Calculating the correct resource levels required to provide the necessary support capacity for a vendor, falls into the contact centre best practice domain. Erlang theory is just one example of a framework that calculates how many people you need to run a support centre based on input metrics such as the number of support calls, time waiting, and resolution times you have.

> *We have a whole lot of ratios that we use, as far as how many people we'll put on a help desk* (I-29).

Measuring how the support operation performs from a customer perspective was deemed an important indicator. Metrics on the total number of issues raised over time, their severity and the average resolution time were all raised.

> *We started tracking now how long, how much time we spend supporting particular customers and from that we will start to generate some KPIs* (I-25).

Metrics for product R&D (Code-48)

The main KPIs for product R&D identified by informants were: customer product feedback, the number of bugs and faults, operational uptime of product, product production and delivery times and costs, and the contribution of R&D to product roadmap.

The robustness of the product and its ability to provide customer business solutions were highlighted. Key product quality factors include the number of product bugs, the severity of these and the time that is taken to fix them. The amount of uptime when the product is fully operational is also a good qualifier.

> *The customer reaction to product from a satisfaction perspective* … [that is] *the key thing around the product* (I-28).

Product delivery and production can be measured. This is often done by using standard project management completeness metrics of quality, time and cost. The closeness of a vendor's actual delivery dates to individual customers can compared to their estimated times.

There was some variation amongst informants for the suggested percentage of total revenue invested in product R&D for established software business. An industry ballpark of roughly 20% was accepted as hitting the mark; a natural level that hit an equilibrium between too high and too low. Most informants quoted figures that were in between the range 15%-25%. It was noted that R&D investment levels frequently peak and trough with business demand.

> *Last year we put 26% of our revenues into R&D. The industry standard is about 16-17%* (I-29).

Some software businesses do not undertake any formal measuring of KPIs for their product R&D department. Some cited an informal tracking of product R&D, a 'does it feel okay' subjective gauge.

> *The honest answer is, we don't have KPIs around the R&D aspect of what we do* (I-26).

The discussion on metrics for product R&D led into a conversation on software product ROI and associated funding.

Roadmap ROI and funding (Code-49)

The need to attain funding to finance product R&D can put significant pressures on a software business. These are exacerbated for smaller software vendors. Developing new products and/or re-engineering existing software generally requires considerable funds in comparison to revenues a software business will achieve in a year. Large investments, often totalling more than several years revenues, can be required to develop the generation of product. High expectations in product quality levels and the investments required to put the capabilities in place so that engineer software rigorously, are making it harder and harder for smaller vendor to compete.

It is questionable whether vendors know the ROI from their R&D investments. Informants advised that there were limited vendor software ROI models in existence.

> *Is our product getting out of hand? What are we spending our R&D dollars on? What is the return? And that'd be a pretty good study, one day to come up with how you do that, because it's not me, I think pretty rough* (I-30).

Product R&D investment at vendors can fluctuate considerably from year-to-year. Explanations for this can include discretionary spending to re-engineer a product platform or a marked decision to invest with the business objective of achieving future growth. Informants revealed that vendors with relatively conservative business targets appear to

keep R&D investment levels the same every year. The product R&D function does what it can with the funds it is given. Often there is no explicit link to where there software is in its overall product life span, or to speculative growth opportunities that mat present themselves.

> *We just think the industry at the moment is taking off, so we're spending, in advance … we try to see, work with industry, where it wants to go and try and have it* [products] *available when they want it, and that way again, we're first to market, you know, we can hopefully get some advantage* (I-11).

Metrics for strategic assets (Code-50)

Informants identified a collection of strategic business metrics that provide indicators about a software business' overall strength. The strength of innovative R&D capability and the strength of distribution channel were repeatedly flagged. Leadership strength, quality of people, domain expertise, employee satisfaction, capability sourcing appropriateness and strength, and operational effectiveness and efficiency were all identified as key strategic assets. The customer value proposition and ROI, CRM, and market/monopolistic power strategic assets originated from conversations around customers and a market perspective. The strength of software business core, appropriateness and viability of customer engagement model, risk profile and risk management capabilities, ability to scale the business and appropriateness of business ownership model were inferred from discussions about the strength and weaknesses of a software business.

> *How do you measure how the company's going? Use market power* (I-30).

The ability if measure these KPIs is often limited. Firstly, many of these business attributes are highly intangible and working out exactly how to measure them is often uncharted territory. Secondly, most of these attributes are related more to the long-term business performance and even with the best of intentions, time is not made available to examine them in more depth.

4.6 Business Challenges and Success Factors (Codes 51-63)

Following the interview conversations on how informants viewed the formalisation of software businesses, an examination of software business challenges and success factors was a next logical discussion area. Several key success factors that informants referred to were the importance of vendors possessing: talented individuals, domain experience, strong customer relationships and operational simplicity, agility and flexibility.

4.6.1 Success factors

Talented individuals (Code-51)

A quirk of the industry is the type of people that it attracts are visionary technologists and software engineers, business evangelists and super salespeople. However, these individuals do not always possess complementary attributes. The combination of exceptionally bright individuals and strong personalities can often be more destructive than constructive. Getting individuals with the right component skill requirements and getting them to function as a unified team can be very challenging.

> *Finding good people and then retaining good people is another major part of the business* (I-23).

Behind software products, vendor businesses are all about IP and IC. IP and IC often have a tacit form and are locked up within specific individuals' minds. A key factor for software business sustainability is therefore retention of talented staff.

> *I mean, the guy who basically invented JBuilder went off to Microsoft. That is a big problem because not just him leaving but there is a whole market impact* (I-18).

Domain experience (Code-52)

Domain expertise brings a vendor closer to its customers' world. From a product development perspective, it suggested that it critical factor in truly understanding customer challenges and problems. In addition, providing solutions is not just about providing good product, with a good understanding of the customer domain and environment, the sales and implementation of software is likely to go much smoother.

> *Everything that we consistently do, from a software development, or development of our service, comes back to making sure that it is what customers need* (I-26).

Not all vendors claimed that they had domain expertise. One informant stated that having an appreciation of a customer domain, as opposed to possessing domain expertise, was sufficient. However, an overwhelming majority stressed domain expertise as a critical success factor.

Customer relationships (Code-53)

A customer-focused approach was identified as being paramount. Listening to customers, focusing on their problems, talking in their language and providing solutions that contribute value to their business are fundamental. The upside of a strong relationship and a satisfied customer can be significant: an ease to do business with, profit potential and follow-on revenue opportunities. The downside of a poor customer relationship can be dismal: a disproportionate time and energy expended, limited if any ROI, damage to reputation in the market. A number of informants suggested that many software vendors failed to focus enough on their customers and as a result often had poor customer relationships.

> *Very few customers in any walks of life will change supplier if they feel they're getting looked after, they're getting respected, they have a good experience* (I-02).

Although most informants agreed CRM was a fundamental, they were less certain about exactly how CRM should be practised and whether it was possible to measure the strength of a vendor's customer relationships effectively.

Attaining a close proximity to customers was identified as being important. This ensures that vendors have a clear picture about what their customers are experiencing and thinking. One advantage of a vendor providing consulting and implementation services is that it enables them to get inside a customer's business effectively, develop personal relationships and gauge exactly what a customer is feeling.

Informants thought that customer trust appears to have a direct influence on the successfulness of a vendor software business. Building trust with customers can take time and it comprises a number of components. These include credibility, experience, integrity, and a feeling of confidence. A strong trust level is likely to be based upon a customer-centric perspective that combines with some real underlying value that the vendor can provide to a customer's business.

> [Customers] *feel comfortable they can do business with me, they can trust me and they think I am confident and it does not go much past there* (I-27).

Customer loyalty can translate directly into repeat business and additional revenues for vendors. Loyalty is most likely to come from customers having an overall positive experience with a vendor and its software product. Having loyal customers provides vendor the opportunity to mine existing customer relationships for additional revenues openings.

> *Our revenue model is very much based on mining that customer relationship … We have an assumption that our projects will lead onto multi-year engagements* (I-13).

In addition, customer education was identified as an important and ongoing vendor activity. The better a customer understands a product, the more value they are likely to

derive value from it and the more likely they are to continue to engage with the software vendor.

Customer perspective (Code-54)

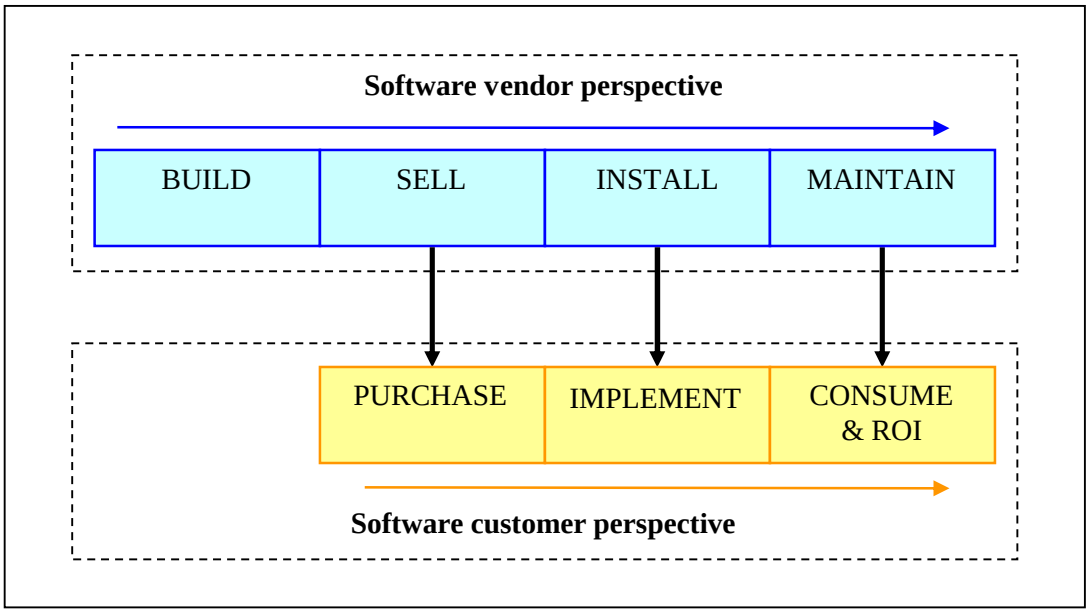
A key factor identified in relation to customer relationships, was for vendors to have a clear customer perspective. Vendors typically look at their operations from their own perspective of building, selling, installing and maintaining software. However, customers do not build, sell, install and maintain software. Customers purchase, implement and consume software. They then expect a tangible return on this investment. Figure 4b illustrates both the vendor and customer perspectives of software.

When customers purchase a software product, they are aiming to acquire a solution to solve a customer problem. The closer they can get to acquiring a total solution to their problem, as opposed to just purchasing individual components that could be assembled to solve their problem, the better. A base software product will generally require some human effort to configure a software solution in a way that provides a tailored solution for a specific client. The rollout of the solution is also likely to involve more human effort as the software is introduced to the people in the customer organisation who will encounter it. These are all activities that a customer is unlikely to be an expert in and they will often expect a software solution provider to assist in this area. However, customer patience for having to spend nine months setting up a software solution is diminishing. Pre-configured and out of the box software that can be used straight away is becoming increasingly desirable.

> *We actually sell a better way of managing your whole procurement … that's what we sell … that's not their core business; it is our core business* (I-19).

> *The key is to provide business facing value for that underlying very sophisticated technology* (I-13).

Figure 4b: Customer Experience Life Cycle



Customers want to get to a point where they are actually using the software and it is contributing value to their business as soon as possible. Until this point, customers are not receiving any return on their investment. Other factors that are worth vendor consideration from a customer value perspective are: the support and maintenance proposition user training opportunities; whether the product stays up-to-date; is there customer value with product upgrades; and whether the overall value proposition is in line with competitors.

Operational simplicity, agility and flexibility (Code-55)]

Customers' tolerance of overly complex and costly software experiences is waning. Substantial functionality and the need to seamless integrate business processes can result in software vendor operations also becoming very complex. The ease of producing, implementing and supporting software product all need careful consideration.

> *Looking at an enterprise-wide application, you have to put a lot of features in there … Yes, they are complex* (I-01).

The lack of business agility and flexibility can be a big threat to vendors competing in the dynamic software sector. Both product development and new business activities can

fluctuate considerably through peaks and troughs. If a vendor has the flexibility to change the emphasis and capacity it has on its different component businesses functions, then it can be one step ahead of its competitors. The ability to be respond to changing business demands is one area in which smaller firms can have a higher level of agility. In contrast, the formal structure and processes of a larger firm can be a limiting for these companies.

> *The successful business vendors do have a level of fluidity, so to speak, so they can respond to the market* (I-14).

4.6.2 Challenges and risks

Specific software business risks (Code-56)

Informants went on to raise a range of challenges that form part of running a software business. These included managing specific software business risks, operating within the constraints of certain business ownership models and having the ability to scale in size.

R&D and technology decisions can make or break a software vendor. If a vendor does not upgrade its technology, its product may end up being inferior in its market. If it picks a new technology that does take off, it may have to abandon this later at great cost and switch to the new market standard.

> *Risk management in our game is important, and one of the reasons it is, is that you can't, if you're an IP vendor, you can't shift ground overnight … Our industry has been characterised by fairly dramatic shifts in technology platforms* (I-02).

Customers being unsatisfied with a vendor's products and services can be a big risk to a software vendor's business. Customers want to feel that when they deal with a vendor this is a low risk interaction. If a vendor does not have a track record and demonstrable experience in a particular market there is a real risk that customers will have a lack of confidence in its ability.

> *The biggest risk in any business is if your client is not satisfied with your product for whatever reason* (I-14).

A high area of exposure for vendors is the implementation of their software products. Implementations can often end up taking much longer and being much more costly than anticipated. Inadequate pre-sales consultancy and not having the right people with the right skills tend to be the main reasons.

> *There's very high failure rate, the financial services back-office implementations, globally. I mean particularly the UK is horrific. There's failed project after failed project after failed project* (I-11).

A significant risk is that a software business does not have a large enough flow of revenues. As most costs are generally spread reasonably evenly throughout the year, a continuous and reliable flow of revenues is important. Underpinning this revenue risk is the question of whether it can repeatedly sell its product and services. While new product sales can be inherently volatile, selling services in a way that the peaks and troughs associated with new product sales can be smoothed out. Maintenance revenues can be particularly attractive as when these are secured, revenue still comes in even when no new product or services sales are made.

> *Recurring revenue becomes essential. I mean* [for] *the sustainability of the company* (I-04).

Informants emphasised the importance of renewal, growth and sustainability in the dynamic and evolving software industry. It was argued that the long-term success of a software business is directly linked to a robust growth or renewal strategy. Informants have suggested that the topic of sustainability has not been given enough focus by vendors to date.

> *I think the biggest risk at this stage is to tune the sustainable growth model to try to grow fast enough to make the demands and have a financial business* (I-07).

While there are often great opportunities to achieve exponential growth in new software segments, the investment levels required and the risks levels that need to be taken on

provide no second chance if the planned growth is not achieved or turns out to be unsustainable.

> *I mean, okay it sounds like a good idea opening up an office in Perth, but what's the risk involved and what's the investment time before you get a return on your investment?* (I-01).

The discussion on software specific business risks led to the identification of a number of inherent risks of the software industry.

Inherent risk (Code-57)

The software industry continues to be dynamic and volatile. Software products, services and market segments are continuously evolving. Operating as a business in this sector encompasses a whole swathe of inherent risks. While some small software businesses were identified that had risk adverse owner managers, many others were found to have taken on large risks. Commercial risks have often been pushed aside as vendor entrepreneurs have raised and invested large capital amounts in developing their products and markets. However, as the software market is maturing, the level of business risks taken on by vendor businesses appears to be declining overall.

> *If you're risk averse, you'll make decisions and you'll run a business in a certain way and if you're prepared to take on a high degree of risk, you'll run the business in a different way* (I-25).

Risk profile (Code-58)

Pulling all the risks of running a software business together, informants raised the importance of profiling an overall business risk. Approaches to setting business risk profiles and managing risks varied considerably amongst the research informants. At one end of the spectrum was an almost *laissez-faire* approach to big picture risk management: almost trivialising the linkage between which business activities were focused on and an overall business risk profile. In contrast, some of the more developed views suggested that a formal and sophisticated approach to business risk management and governance

needs to be an imperative. It is the aggregate of all the individual risks and the overriding risk management processes that will define a software business overall risk profile.

Target growth rate (Code-59)

While discussing risk in some depth, this looped back to some of the discussion about vendor growth aspirations. Informants drew a direct link between risk levels and target growth levels.

A growth target too high and a vendor may not be able to match targets with its internal capabilities. A too low growth target and a vendor may miss a window of opportunity that will not reappear.

> *We're in a bit of a dilemma now about where we go to next. You know, yes, there's organic growth here, but how quickly do we want to head overseas is my next dilemma in trying to convince the board we need to take that next step* (I-23).

A number of informants have been approached by advisers and financers and told that they should borrow significant sums with the intent of pursuing exponential growth. Several aired caution about risking the position that their business held today. Where a business held competitive advantage in its market, executives appeared not to feel any compelling reason to take on a high degree of risk. Others had simply evaluated the risks and upside benefits of pursing substantial growth and essentially decided to go for it.

Balanced, sustainable and controlled growth was preferred by most informants. There is often a fine balance between pursuing potential market opportunities and gauging this against what a vendor can realistically deliver at a given point in time. Growth will only be sustainable if the necessary capabilities required to underpin the enlarged commercial operation as actually developed. Growth controlled include making measured invests in the business, developing robust plans and using these to guide the business to challenges as they arise.

> *Technology One I think's an example … it's a software company, publicly listed, stuck to their knitting, they didn't go overseas quickly, they really made sure the product was settled, they dominated the Australian market and now they're putting their toe into the UK … They haven't gone to America yet, so they're being sensible about it. Public company and making a zillion* (I-30).

Business ownership (Code-60)

The topic of business ownership surfaced repeatedly from informants. Whether a vendor is privately or publicly owned can influence on how it structures its operations and how it performs.

It was suggested that private ownership offers greater flexibility in business management as it does not have the pressures of institutional investors and shareholders. It was argued that as software product development required long-term capital investments, this caused peaks and troughs in internal activities that did not fit well with markets expectations that a PLC deliver consistent quarterly returns.

Informants also suggested that there were numerous small software businesses that, although they were public, were not big enough to justify the overhead of operating as a PLC. A market capitalisation of under $50 million was offered as a threshold for staying private. However, on the downside, private ownership means that some software businesses may not have sufficient controls in place. Shallow and/or very weak boards were cited as one shortfall of some privately owned businesses.

> *Marketboomer is 100% privately owned, so we don't have pressures of an institutional or big shareholders trying to dictate what we do* (I-19).

While the administration associated with being publicly listed is an overhead that needs to be built into the business model, some informants viewed this as a constraint. Others suggested the enforced discipline upon public companies was a good thing.

> *When you've got a big gorilla of a shareholder who has a lot of strategic influence, or a lot of influence over decisions that are made, sometimes, you know, you can limit some of the things that would otherwise help you grow* (I-19).

> *Exploit being a public company I think. I don't agree with to stay private and cuddly* (I-30).

The discussions on business ownership led into a wider conversation about the executive leadership of software business. Vendor leadership quirks, and leadership roles and functions were identified as factors that business managers needed to understand.

Vendor leadership traits (Code-61)

A technical person running a software business was identified as a big problem. Products can become over engineered or fail to provide real customer advantage. Sales can be poor and business finances troubled. These businesses can often get stuck in rut and lurch from crisis to crisis without realising their full potential.

Founders and owners are also not always the best people to run software businesses. Some CEO founders know their organisation intimately, but treat it more like a hobby than a commercial enterprise. Energies may be focused on things that they enjoy, as opposed things that are essential to leading and running a business. Founders/owners are often not growth accelerators of going concerns. As a vendor expands, founders and owners can become progressively out of the depth and increasingly a constraint upon the success of the business.

> *Founders can get very blinkered. I think there's a crossover point that owners, founders should leave and hand over and they invariably all hang on too long* (I-11).

Having a leader with business acumen was identified by most informants as critical. When big changes in business direction are made, they need to be thought through and justified. When major initiatives and activities are undertaken, they need to be planned and run professionally.

Leadership roles and functions (Code-62)

A vendor's CEO and board combine to provide the leadership for a software business. Several informants stressed that the CEO is the number one most important individual in a software business. They are the single person who integrates all the different elements of the software business into one single picture and ensures that what needs to happen does happen. Ideally, the CEO should be both a commercial businessperson and well as being skilled in business management and administration. However, it was noted that the CEO can't run a vendor business themselves; they need to create a structure for the organisation that ensures that all necessary bases are covered. Motivations and incentives for the CEO were stated as key factors to get right. This will ensure that the CEO is accountable for their actions and performance.

> *You must have a good CEO to start with. A business manager* (I-30).

> *The CEO has got to be accountable. 'You said you'd do that, you haven't done it. Why?'"* (I-30).

A board of directors can provide an overall perspective to a software business and ensure a level of governance exists over the CEO. In particular, a vendor's finances were highlighted as an area for scrutiny. A board can be essential to steering and guiding a CEO, as well as protecting stakeholder and shareholder interests. Having a strong and active board of directors was considered important. Ideally the Board will understand the key elements of the business and be involved in the company's strategic decision making process.

> *Where private companies sometimes go down the tubes because they don't have strong enough boards* (I-30).

> *We involve a lot of people in the decision-making process to move forward, especially the Board* (I-23).

> *Need to have a board of directors that are actively involved in the day-to-day running of the company* (I-16).

Scalability (Code-63)

Informants identified the ability of a software business to scale its operations as an important business model consideration. Some elements of a software business are easily scalable, while other elements are more difficult and more costly. A pure product reseller model using partners for sales, implementation and support should be easy to scale. However, there are elements of some software businesses that can still be very dependent on people and knowledge. It takes time and cost to expand these areas.

> *Scalability is all about us being able to grow and continue to grow fast … Each growth step you hit means that various things in the business have to change, either the people or the processes, or the vision* (I-26).

With software demand often fluctuating, a business that can respond to peaks and troughs of demand would be very attractive.

Informants communicated that for a business to scale effectively, a critical mass of capabilities and business size needs to be in place. It was suggested that as a company has established product, processes, people and customers, it becomes easier to extend and duplicate these. Prior to this, a vendor's functional capabilities may not be mature enough to be extended.

> *We've got a big enough business now and enough penetration and people and customers, if we're ever going to be able to scale it that will occur in the next four or five years for this business* (I-28).

Partnerships and in particular the development of distribution channels can be used as an effective means of scaling up quickly. The investment in a channel model is a way to increase market share without dramatically increasing overheads. Although a vendor potentially sacrifices some profit potential when entering into a partnerships arrangement, informants suggested that this was generally outweighed by the upside potential of being able to grow business revenues.

> *You don't actually have to massively upscale the business internally. If you think of your channel partners as an extension of your salespeople, as an*

extension of your technical people, as an extension of your marketing people (I-22).

4.7 Other Business Considerations (Codes 64-72)

The interview discussion on software business challenges and success factors stimulated a wide range of other business considerations being brought up by informants. While these were raised at different points throughout the interviews, this section captures a number of these miscellaneous topics. These include a business' critical mass, candidate functions for external outsourcing and market opportunities.

4.7.1 Other Business Considerations

Critical mass (Code-64)

Software business can benefit from a critical mass and economies of scale as they reach a certain size. Product R&D, sales and marketing, services, and support and maintenance functions can all benefit from economies of scale as grow in size. Similarly, a business' strength also improves with the number of customers. A vendor is able to take a more balanced approach to developing a market offering as opposed to being at the mercy of one critical customer.

> *I mean once you achieve a certain, I guess number of clients … then you have got more leverage in the market* (I-07).

Multiple products and versions (Code-65)

One particular aspect of a software business that was identified as having a significant effect on operational efficiency was the number of software products and versions a vendor had in its portfolio. The ongoing development of innovative new products, combined with the provision of bespoke customer solutions, can result in numerous software products and associated versions that a vendor needs to manage and support.

> *We don't have multiple versions. So yeah, we're very careful about ensuring that anything we build is reusable. We specifically tend not to do business where it is so client specific that we can't see any reuse for it and it really just becomes a professional services project* (I-11).

Informants offered a number of approaches for managing this area. Software is often structured as a base product with optional modules. The core product is built and provides revenue from having a wide market demand. Add-on modules can then be evolved as and when specific customers are willing to contribute funds to their development.

> W*e've categorised those solutions in our marketing blurbs about these seven areas … we have a single underlying product that supports those solutions across those different areas* (I-13).

Candidate functions for external sourcing (Code-66)

A wide range on vendor business functions ranging from sales to delivery and from build to support can be provided by partners or outsourced to external organisations.

The QA of software product is an area in which there are tangible benefits from outsourcing. Handing this responsibility over to a specialised third party there is an opportunity to lift quality levels.

> *We're getting quite able to understand the outsourcing option regarding R&D. It's now, in my opinion, desirable to look at outsourcing software development, and there's a blend of model now that works* (I-02).

The upside of developing external distribution channels can be an enormous uplift in sales revenues. On the downside, managing channel partners successfully can be challenging and the expected benefits do not always materialise.

> *Large value piece of software, you're talking about half a million a million dollars that type of thing, it's probably better to directly market it yourself. The medium range where the cost of sale, in other words, the effort put into the sale is only worth about $50,000, then you're probably best to look at, looking at resellers, the reseller market. It really depends on the product* (I-01).

Informants suggested that implementation is arguably the hardest part of the business to outsource. Vendors are often keen for third parties to undertake the implementation work, although there are often reservations about third parties ability to do this successfully. As a result, vendors at times develop an in-house capability in preference to entrusting this to a third party.

> *The biggest problem for us is basically the implementation, is actually finding someone who can go out there and implement it, without coming back to us every five minutes … Someone who knows the market that they're operating in, you know, has contacts, is established, is able to, you know, to hit the streets running* (I-05).

Some support and maintenance activities are also appropriate candidates for outsourcing. Functions such as first and second line support may be candidates for outsourcing has they have a transactional nature and do not require specialised skills. Efficiencies may be able to be gained from these being met by a third party customer contact centre or managed services arrangement.

However, the one component of a software business that informants stressed should not be outsourced is the creation and management of IP. IP is such a fundamental of a software business that protecting this from third parties is paramount.

> *To me the* [intellectual] *property is the number one issue. The rest of the stuff I can buy off the shelf I can go and buy it. Marketing people, I can buy brand management. I can buy product positioning. I can but sales and I can buy superior sales* (I-06).

External sourcing of business capabilities was a big and important topic for informants. Further probing in this area covered capability sourcing considerations and capability sourcing drivers.

Capability sourcing considerations (Code-67)

Informants identified a number of factors as important concerning outsourcing or partner relationships. Specialist expertise in the domain of the software vendor's customers is

arguably fundamental. Credibility with customers is vital and should hopefully have a presence in the vendors target market already. At the heart of any commercial arrangement needs to be trust and a mutual motivation for the relationship to be successful. Some vendors are nervous about handing over responsibility for certain aspects of the business to third parties. Keeping key functions in-house allows a vendor to keep control and be closer to key areas of its business.

> *I like to be responsible for everything we do. So the project goes bad, they kick us or the client kick us; when it's good they congratulate us. Don't like to be beholden to somebody else* (I-11).

There is a management overhead for vendors when sourcing business capabilities and functions externally. Close monitoring and management is likely to be necessary. Although there is often a perceived direct cost savings when using external parties, this benefit can evaporate when a vendor factors in additional costs and time required for QA and management of the third party.

> *Overhead of project managing those external resources I think adds another layer of cost that doesn't make it as dramatically interesting as some people make out* (I-29).

Capability sourcing drivers (Code-68)

A vendor can scale business capabilities easily and quickly when they use external sourcing for business functions. In contrast, when a vendor relies on in-house resources, there are limits to its flexibility in changing the size of their business operation. A vendor may need to rapidly scale up for a number of reasons including increased business and resourcing one-off big deals. While scaling down may result from a fall off in sales and business demand.

Developing partnerships can provide a vendor with access to specialised skills that they may not be able to justify or afford to provide in-house. A vendor may use external

partnerships to access specific vertical domain experience that is a key part of deploying a customer solution, but not a core part of the vendor's business.

> *We're not particularly interested in being storage experts. So we'll just get in a partner who'll do that … We can kind of contain our responsibilities, focus on doing those supremely well and make someone else own the bits that just aren't our core business* (I-15).

Sourcing capabilities externally provides a way for vendors to mitigate the risk of a business downturn. In the event of reduced demand, a vendor has the flexibility of being able to reduce its capacity. With external parties a vendor usually only pays for the services it consumes. With contractors, these can be easily terminated.

> *We use them* [contractors] *as hedges against risk. They're the guys that we can say thanks very much, that deal's now done, we didn't win the next one, you're out of here* (I-15)/

Market opportunities (Code-69)

Several informants were very critical of vendors under investing and falling short in the area of market analysis. Often it is the case that the founder of a software vendor will have a background in a given industry; they see a market gap and a business opportunity that their domain knowledge can be applied. The process of being focused can take a number of years and is often an evolving and natural progression.

> *They don't have marketing people that are strategists, or people that have had the experience. They bring a young marketing person who has no idea of the customer and who has no idea of the technology; and I find that amazing* (I-16).

Often opportunities exist to create value in inefficient industries from the introduction of a software solution. If products are too simple and easily replicable by others, market attractiveness may be limited. Positioning in a market with few competitors is an ideal.

> *Being successful in business is to be where your competitors are not … the trick is to find a market where there is genuine need and a level of maturity that you can actually go and sell too, but not to take on* [competitor] *vendors in the own heartland* .(I-29)

While informants were primarily focused on talking about developing specific market opportunities, this provided a conduit for wider conversations about the topic of marketing for a COTS application software SME business. A number of informants logically reversed through the standard marketing topics of product positioning, market segments and market segmentation.

Product positioning (Code-70)

A simple software product positioned in a market with a large enough customer base can be the basis for a successful software business.

> *There are lots of true vendors out there that have some simple products that target the mid-market and have enough functionality in there to get their market share and keep their businesses going … despite its technical imperfections and its functional limitations, there are enough users out there, it is good enough for a lot of businesses, and it still sells* (I-27).

The market for software products can often appear very crowded, but it is not always as busy as it may first appear. However, all software products are not the same. There are often only several software packages that will meet the requirements of a customer with specific needs.

> *You need to do records management in a way that complies with government regulations, and there is very few products that do it … at the moment Sharepoint is being touted as an enterprise contact management product. It's not … it's a collaboration portal, and just because Microsoft say it's ECM, it doesn't mean it is* (I-29).

Market segments (Code-71)

A limited number of mega-vendors serve well-defined segments such as ERP. Conversely, numerous smaller companies serve a diverse set of other less well-defined segments. The main market segments referred to by the informants included vertical and niche markets, mainstream markets, ERP markets, mature markets, emerging markets and SME markets. However, the market does not comprise mutually exclusive segments and these different sub-markets are often overlapping in defining criteria. As a result, the competitive landscape of different software market segments can vary considerably.

Market segmentation (Code-72)

Vertical segmentation of software markets was arguably the most common delineation that informants described for how COTS application software SME businesses target their markets. Many have constructed businesses based on products that provide a solution for a single vertical business domain. Alternatively, a vendor may choose to develop a highly complex and feature rich product that only large enterprise customers can afford. The inverse is a simple cost effective product offering for a SME volume based market. Another commonly dimension to segment on is geographic region. There may be possibilities for a vendor to extend a base product and make it suitable for a particular overseas region or country. A leading vendor that competes globally is very active in assessing the revenue potential of different regional markets. Countries from similar economic and demographic profiles are clustered together and then their individual markets compared.

4.8 Strategy and planning (Codes 73-81)

As each interview moved towards its later stage, there was an attempt to channel the wide range of software business topics discussed into some unifying point. The wide range of topics identified, together with earlier discussion around business formalisation, was used as a platform for questioning on COTS application software SME business strategy and planning. The strategy and planning conversations with informants highlighted a number of factors. These included reference points for strategic guidance, strategic gaps, external reference industries and a number of general strategies of relevance.

4.8.1 Planning

Reference points for strategic guidance (Code-73)

Informants identified various reference points that can be use for guidance about how to run their software business. While some informants said they consume a significant

volume of IT research, they questioned whether this actually provided them with direct guidance about how to better run their businesses. Informants gave a lukewarm reception to the value of examining industry success stories concerning other software companies. Success stories were often viewed as remarkable exceptions to the industry norm, as opposed to providing repeatable blueprints for SME success. At best, they might provide ideas, but not much substance on actual execution. Sometimes, software business stories were viewed as examples of what not to do, rather than the other way round.

> *Previous success stories probably give a lead, but you will never be able to repeat them. Like you will never be able to repeat the success* (I-08).

The strategy management discipline is applicable for vendors. Most informants propositioned that a software business was foremost a business like any other business; and as such, strategy management theory should be able to contribute a great deal of a value and guidance for vendor businesses. Informants suggested that the strategy field should be scanned for those reference models that are of most relevance to running a software business.

The discussion on reference points for software business strategic guidance led to informants questioning strategic planning prevalence and highlighted occurrences of software business failures.

Strategic planning prevalence (Code-74)

A widespread, but not universal view, amongst informants was that vendors generally undertake limited strategic thinking and planning. While some informants had informally thought through what their businesses were about, very few appeared to have formal business plans. Fundamental strategic analysis activities, such as clarifying a target market and developing a competitive product to serve, were generally lacking. Various explanations for the lack of planning were offered. The explosive industry growth has hidden, or more accurately provide an excuse for developing a proper strategy. Many of

the software vendor CEOs are typified by being visionary and entrepreneurial, these personalities are often less interested in the protocols of strategy and business management.

> *We've been for a long time in an industry that's just had this explosive growth, and it's been too easy, too easy for either cowboys, or for that matter, poor management to survive* (I-02).

> *They* [software CEOs] *are visionary people who just have sheer driving passion that makes even a bad idea sometimes happen* (I-03).

Informants flagged that planning requires investment and businesses needs the capacity to run formal business models and undertake strategic planning activities. As might be expected in any sector, the larger software businesses tend to have a greater maturity in their strategic planning, and more discipline and structure in the running of their businesses, than the SME vendors.

> *A big organisation they have a lot of people making up the pointy end and you need to be pretty clear on what your value proposition is; hence the IBM and Microsoft are pretty clear on what they are into and what they are not* (I-14).

Access to executives that have proven business experience in previous similar organisations was deemed of importance. This can provide solid guidance for the long-term running of a software business.

> *We've got some very good people on the board that work closely with our own CEO, that have got very strong industry experience* (I-29).

> *We've got good people working for us, they're close to the market, they're close to the customers, they're to technology, I guess we just figure it out ourselves* (I-12).

It is likely that the use of external assistance in developing software vendor strategy will grow as the industry matures and individual vendor businesses grow.

Failed businesses (Code-75)

Informants implied that many software businesses are badly managed. Desperation while in start-up mode, precarious commercial endeavours, a lack of basic business

management skills and cash flow problems were repeatedly quoted. Technical people running companies was mentioned numerous times, as a common reason for software business failures. A focus on technology and product, as opposed to sales and commercials, being the underlying reason.

> *I find that technical people have a dream. They have this great product, they have no idea of how to sell, and how to start up and run a business, that's where it fails, on the business side* (I-01).

Informants stated that flawed strategies are not uncommon amongst software vendors. Many vendors operate continuously in reactive mode, convinced that their business will magically grow in time into a sustainable business. Having an inward product focus and limited perspective on selling a vendor's actual software offering to a target market is frequent. Another flawed approach is that vendors have an overly aggressive sales culture that is focused on making transactional sales. Little attention is given to creating long-term customer value. A strategy of doing everything also features. Some vendors try to focus on innovation, on extending their market, and on mergers and acquisitions all at the same time. Without a solid foundation, this sort of approach is questionable. The software industry has at times been almost obsessed with mergers and acquisitions. Many acquisition strategies have been non-value-adding. Lastly, many vendors have become fixated with IPOs and becoming publicly listed at an early stage. It was questioned whether the software business cycle, with peaks and troughs of R&D activity, is suited to the linear growth and stable financial performance expected of listed companies.

> *I don't think a lot of them* [vendors] *even think about it* [strategy] … *I mean they're in there every day, 'oh shit, that customer's a problem, what do we do?' They've got to sit back at least and do some planning* (I-30).

Strategy gaps (Code-76)

Overall, informants were of the opinion that a comprehensive set of strategy and management tools that gave specific guidance to running a holistic software business

were missing. They stated that it unclear exactly where they should go to source advice and reference materials.

> *There is no clear mantra as to exactly how to do that* [run a software business] (I-24).

Investment levels in product R&D were described by some as bearing little relation to the subsequent return on that capital. Vendors could benefit from a specific set up tools to assist them in managing the commercial potential of their software product families. Evolving technology and technology upgrades plays a large part in the software industry. How vendors can smoothly transition between product platforms, the next big versions of their product, was cited numerous times. Software has been characterised as an opaque product and this has often caused a misalignment of customer and vendor expectations about what a product will and will not do. How to deal with the opaque nature of software product and implementations and providing clarification from a buyer perspective is an unanswered question. Increasingly, vendors are attempting to move into overseas markets. They would benefit from guidance on how to effectively build distribution networks overseas. Software licensing and pricing models are starting to change with the advent of SaaS and other value based price. Exactly how to price software in way that is meaningful to customers and commercially viable for a vendor still need to be better thought through. Advice was sought on to how run a consulting business effectively and efficiently.

> *Is our product getting out of hand? What are we spending our R&D dollars on? What is the return? And that'd be a pretty good study, one day to come up with how you do that* (I-30).

> *Implementation, development, support and sales; all of those areas I'm sure that we could get guidance and people could advise us on better and more efficient ways to do those things* (I-25).

Market distinctiveness (Code-77)

A key link was suggested between software business strategy gaps and the distinctiveness of the COTS application software SME market.

Software is an information good. While there is a high cost for producing the first unit of product, each additional unit has a very low or near zero margin cost associated with its production. As a result, the pricing and pricing strategies of information goods are generally a very different from physical products. There is a significant incentive for suppliers to make volume sales as profit can then grow exponentially. Another important difference for suppliers of information goods is that they do not have to have stock. There are advantages in product distribution as information goods can usually be shipped large distance very quickly at very low costs.

> *I mean the nice thing about software is you're not carrying stock, you know, that's very nice, so you can actually control your product overheads quite well* (I-05).

Vendors have arguably made a significant amount of money in the past from customers who did not fully understand exactly what they were buying. The intricacy of software product is often characterised by both complexity of its installation and the operation of all its functionality and features. Software can be described as being opaque. It is an intangible product that cannot easily be touched or felt like a physical product. At first glance, it is difficult to fathom exactly what a product does and how it does it. This opaqueness can result in mismatches of expectation between vendors and customers on what capabilities a product does and does not have.

> *Quite often you bring these people in who are used to touching and feeling the product and when you talk about software, which is virtual, they can't get their heads around it, they just don't understand it* (I-01).

There was a mixture of opinions on whether the software business is unique or not. While a number of distinct characteristics for software business were identified, most supported the view that a software business is like any other business at the end of the day.

> *We still struggle with the notion that somehow we're still unique … It's schizophrenic. I can't help myself but achieve a mindset where I can justify that thought and then the next morning I can wake up and step outside and say, you idiot, clearly these are some common characteristics* (I-13).

External reference industries (Code-78)

A number of industries were identified as being able to provide a point of reference from which software product R&D operations could learn.

> *I think we've got a lot to learn from mature industries who've, we often think we're pioneering, we're not, I mean we're just trying to catch up with people who've been doing this for a long time right* (I-02).

Engineering, and building and construction were identified as industries that were strong in project management. They have well engineered processes and good track records of delivering on quality product on time and on budget. It was suggested that the software industry should look closer at the standards and methods in these sectors. The pharmaceutical sector was also raised as a reference point that could be of interest for software business to examine more closely. Pharmaceuticals, similar to software, have an extremely large effort and cost to produce the first unit of product. Each subsequent unit has a minimal cost and is generally less complex than to produce than the first.

> *I don't see project management in our space as any different to the construction space* (I-02).

Software vendors can learn from numerous industries in terms of marketing, sales and distribution. Channel management was singled out as a key area in which vendors often have limited capability. Industries with mature distribution channels include the pharmaceutical and automotive sectors. Setting up effective distributors and managing these relationships well is increasingly important for software vendors. Informants suggested that other industries seemed to be more sophisticated in their approach to partner management. As the software market continues to mature and consolidate, marketing and brand management will become more significant. Pharmaceuticals is an industry, amongst others, where global brand marketing is paramount to establishing a product into a wide market and customer base.

> *I don't see channel management in our space as any different to the channel management in the car industry, or the pharmaceutical industry* (I-02).

Car dealer servicing businesses in the automotive industry were identified as a point of reference for software support and maintenance operations. As software more and more becomes an integral part of customer businesses, the need to provide supporting software products and services reliably becomes more critical.

4.8.2 General strategies

Product leadership (Code-79)

The general strategies of product leadership and customer focus were identified by informants.

Informants stated that in the dynamic and innovative software market, product leadership is a common strategy. Having a leading product will not just have superior functionality and features, it will need to be easy to integrate and be interoperable. A strategy of attaining growth from product leadership is not just about building a competitive product. If a vendor is going to grow their product sales, then they will need to develop their associated sales and marketing capabilities to support this targeted increased.

> *We're well in front of the competition; and even if the competition wanted to start now, they're years behind … My viewpoint of it is that we have innovative world-class, world-leader products* (I-01).

> *You need to make sure that you are offering integrates in the client's increasingly complex environment* (I-14).

Customer focus (Code-80)

A number of informants suggested that an established and loyal customer base can provide significant competitor advantage. Serving customers well can result in high satisfaction levels and solid retention levels. Follow-on and new business should then flow.

> *I personally believe in the paramount importance of customer driven businesses and I think it's a cultural thing in businesses, I think that many, many businesses are deficient in that context* (I-02).

If customers are happy, they are unlikely to move to a competitor unless there is an absolutely compelling reason to do so. In contrast to other industries where it can be easy to swap between supplier products, vendors can benefit from the barrier to entry from competitors poaching their customers. Customer focus needs to be much more than just a customer-focused product. Customer relationship and account management were identified as a key to achieving high customer satisfaction and retention levels.

> *The whole thing has to translate into a positive experience for a customer … Two things, one, you'll keep customers for a long and time, and two, you'll get references and references are the best way to sell* (I-02).

The challenge for a vendor is how to achieve satisfied customers in a commercially viable way that supports growth. If too much time is devoted serving customers, the overhead of doing this can actually end up constraining growth as opposed to enabling it. The ability to balance customer focus against maximising opportunity for business growth is likely to be crucial.

> *All my businesses in the past have been customer driven, but they've also limited, limited your growth* (I-02).

Discussions with informants on customer focus progressed into a wider discussion about COTS application software customers.

Customers (Code-81)

Informants highlighted that software customers have matured considerably over the last twenty years. Previously, many customers have made uneducated and unqualified decisions when purchasing software; however, customers are now knowledgeable about software and how it can be used to solve business problems. They understand the challenges involved in implementing, operating and supporting software. They are more likely to seek verification of vendor claims independently. However, there are a range of

different maturity kevels across different markets and customer profiles. Large corporate customers are generally more experienced with the in and outs of software solutions, than SME or emerging country customers.

> *I think the customers are much more savvy about the business problems that they're trying to fix, and whether the software is legitimately the right one to enable that transformation* (I-21).

Customers essentially desire straightforward low-cost solutions to their business problems. Customers have had enough of dealing with all the difficulties involved in adopting software solutions. They want interoperability and flexibility, the ability to make a software products work with their other software products and the ability to change to a competitor product easily in the future is they so desire.

Customer value demands are driving a different approach to the running of a software business. Combined with more choice, customers are now negotiating with vendors about what they do and do not want. High quality bug free products, fewer upgrades, simplicity and ease of install, and reduced lock-in are all key items. There is less appetite to replace legacy systems and move to the leading edge technology unless there is real business benefit from doing so. Customers are now more vocal and they are gaining more power over the suppliers who previously set the agenda for the industry. Many vendors, including some established vendors, are struggling with philosophical change that this brings to the way they operate their businesses. It is a completely different mindset to change from looking at the building, selling, implementing and supporting of software product, to a customer perspective of purchasing, implementing and operating a solution to a business problem.

> *You're really always trying to bring innovation … you're really trying to challenge, I guess, the status quo, that is going to make a customer sit up a little bit straighter and go, 'Actually, that's quite interesting. Tell me more'* (I-26).

4.9 Continuing Industry Evolution (Codes 82-93)

Additional topics that rose out of the interview discussions about strategy included the maturity of the software sector, a number of changes that were occurring within the market and the advantages that first movers could experience. The codes generated from these three topics, covered in this section, wrap up the definitive list of base-level codes generated from the analysis of the transcripts.

4.9.1 Software business' maturity

Variations in maturity across the industry (Code-82)

Informants suggested that while some parts of the software sector demonstrate characteristics of a mature industry, in other areas it is still an immature and volatile industry.

> *It's a maturing market, it's certainly not mature … the larger ones obviously are more established than the smaller* (I-05).

In some software segments, prices are coming down and vendors are focused on reducing costs and improving efficiencies. In others parts of the software industry, there is still a great deal of change and unpredictability, where the repeatability and predictability of a mature market is not present. This can result in disruptions and a degree of inefficiency for both vendors and customers.

> *In a mature industry, you would expect things to be fairly predictable … and the work is fairly repetitive and predictable and that is just not the case at all in the software industry … not even close … The software industry because it is still driven by these bright spark ideas* (I-18).

It was noted that while customers in North America and Europe are familiar and relatively savvy about software solutions, customers in South America and Africa are relatively immature and unrepresented by software products.

Functional perspective of vendor maturities (Code-83)

Informants extended the discussion about market maturity with the observation that different levels of maturity exist for different functional parts of a software business.

The activity of vendor product development, software engineering, has improved and matured considerably. In the early days, there was more interest and excitement in the innovative and creative side of developing product; with the industry have a poor record with regard to product quality. Nowadays, many vendors now have embraced structured ways of working, project management standards, engineering principles, rigorous testing and proven QA processes.

> *We do our R&D to write a product, okay. I think we're getting better at being able to do that, and do that in a reliable way. It's an engineering job in other words* (I-02).

Sales and distribution is starting to mature in the software industry. Vendors now not only recognise the importance of sales and distribution, but they are putting more focus into these areas and upgrading their capabilities in this space.

> *The traditional IT building dreams have taken a bit of a back seat, right? And it is more like what is the brand that I am getting for my expenditure* (I-24).

Consolidation of vendors (Code-84)

Informants noted that the reduction and consolidation of vendors in the application software market is one very visible sign of a maturing in the industry. In the early days of the industry, many entrepreneurs entered the software market motivated by the excitement of technology innovation or by an opportunity to make lots money. Many of these businesses are no longer trading. The software market is now reduced to a smaller set of larger, stronger and more competitive businesses.

> *I think a lot of people saw it as a quick game that they could make money and get out of it fairly quickly … if you really look at the number of vendors,*

> *for a country of this GDP size, there's far too many competitors. Every man and his dog is here* (I-16).

Merger and acquisitions (Code-85)

The link between market maturity and industry mergers and acquisitions was a lively topic with informants.

There are certainly opportunities to grow through M&A. It is an easy way to increase the size of the business, customer numbers and market share. M&A is also arguably an approach for achieving very aggressive revenue targets that are unlikely to be reachable through just organic growth. However, informants stressed that M&A decisions need to be measured and controlled. M&A should be about shareholder return.

> *We have a very aggressive sales target for 2010; that can only be achieved by just acquisitions … we're in the process of looking at three or four players … that fit into a domain, and fits into our business products* (I-10).

Informants highlighted that determining what IP an acquisition has and how this can be commercialised within the context of a vendor's current business should form the basis of an associated M&A decision.

> *We're now in the business of acquiring businesses that have their own IP, software IP, that is globally competitive … We're looking for companies that are well managed* (I-30).

While M&A is still popular and has many supporters, some of the informants argued that there was limited competitive advantage to be gained from such ventures. Often the due diligence on the ROI and commercial benefits of acquisitions has often been negated in the excitement that can go along with being involved in M&A activity. An M&A deal needs to be profitable, the financials need to stack up and the returns need to be actually achievable.

4.9.2 Industry changes

Product, technology, dynamism and market imperfections (Code-86)

Informants highlighted a number of specific changes that were occurring within the software market. These included product, technology, dynamism and market imperfections, the customer value proposition, and software delivery and consumption models.

Many software products are now functionally rich, well-engineered solutions to business problems. This is a step forward from fifteen years ago when software products essentially comprised a box of components that had to be assembled and customised differently for every customer. While there was evidence of products maturing, most informants believed there was still huge opportunity for new innovative products in the market.

> *I think it* [software] *is maturing in that how many new applications have you seen in terms accounting applications, how many new applications are there?* (I-14).

Whereas in the past, both vendors and customers may have been consumed by the pretence that they would be left behind if they did not embrace every new technology, nowadays there is more level-headed thinking as this view is not necessarily justifiable. Technology is changing, but generally not as quickly as the industry press and media often portray.

> *This is one of the unique things about the computer industry that the technology platform does change all the time. If you do not reinvent yourself somehow, somewhere, some new person will* (I-03).

The market could be improved for customers in numerous ways. More robust and easier to implement products, lower costs and less ongoing investment required once solutions are operational and a reduced lock-in with current vendor. However, a common view was that as the market is functioning and sales are taking place, then the market is not broken.

Informants did not see any real driver or necessity to fundamentality operate in a different way. Some were almost dismissive about the topic.

> *SAP for years was accused of being fantastically expensive to put in. The projects were all out of control … SAP is now a multi-billion dollar organisation, so some of it must have gone okay. The customers must have been satisfied, at least a large chunk of them* (27).

Customer value proposition (Code-87)

Informants suggested that software vendor sales personnel have often controlled the software purchase agenda. Sales of new product versions have been achieved based on a compelling story of better functionality and new technical capabilities. As customers have become better educated in respect of software, the ability of vendors to create and demonstrate customer value is now even more important. Convincing customers of the value of purchasing a new version of product underpinned with a supporting business case is increasingly challenging.

> *Our challenge always is, how do we put across our value proposition, or our elevator pitch in the simplest possible way that makes it unique and differentiable, and people go, 'Oh, I get it' … We're constantly challenged by trying to bring it back to basic, simple language … in terms of the problems that we can solve* (I-26).

A software product usually enables some business function to be computerised or automated. Business value is then created through improved business effectiveness, improved efficiency, reduced risk and/or cost. The time it takes to get the software product working and creating customer value is an important value proposition consideration.

> *I believe now, we can get ROI down to sort of 18 months to two years, versus I think it was sort of three to five years, five years ago* (I-11).

Informants felt strongly that vendors understand their customer businesses and can discuss how benefits can be achieved from fitting their software solution into their business. A concentration with product features and IT terminology has often resulted in

a gap between vendors and their customers. Customer executives have often felt blindsided and left with a perception of vendor representatives as a bunch of techies that they cannot really relate to or understand. Although vendors ultimately have an inbuilt bias to their products, a vendor can still attain a position where they can provide direction to customers and perceived impartial advice through a level of transparency.

> [Vendors] *have limited understanding of how people actually use their products and the value that they get out of it* (I-27).

Delivery and consumption models (Code-88)

Software delivery and consumption has its legacy in the perpetual licence fee model. It comprises a one-off software purchase cost and an annual maintenance fee. The maintenance fee is usually around 20% of the one of cost. However, informants warned that there is risk that maintenance revenues will fall off if customers cannot be convinced of any real value from this fee.

> *I think as a software business where most of your revenue comes from one-time license fees, you're really in a fairly precarious position* (I-12).

Transactional and subscription based licences models comprise of ongoing customer payments as they consume software product. These models tend to have a charge based upon a business unit of value. Customers pay for each business transaction or they subscribe to an appropriate level of software usage. The fact that a customer can account for the software as OPEX cost, instead of a CAPEX that a skews their P&L, can also be an added advantage for then that flows trough to vendors.

> *We face a challenge of reorienting ourselves from the perpetual licence plus maintenance stream approach versus the subscription one where it's not quite as clear what that profit is in your subscription type model* (I-13).

Enterprise licence agreements (ELAs) can be appropriate for some vendors and some markets. Usually, this is where a product, or suite of products, is being used very heavily by a customer's organisation. They provide a level of consistency to financial software

budgets, provide efficiencies in procurement processes and offer a level of protection against medium term growth in software requirements. However, ELAs are not a panacea and vendors need to be comfortable with the effectiveness of ELA for both themselves and their customers. While vendors can become blindsided by the incoming revenue, customers can become disillusioned with their financial outlay and unsure about the value they are receiving.

> *More and more companies are opting for enterprise license agreements, ELAs, all you can eat for $5M bucks* (I-21).

SaaS is a relatively new software delivery model that is changing the way customers pay for the consumption of software. With a SaaS model, all the software, data and associated hardware is hosted at the vendor's site. Customers pay for the software as it is consumed, in a similar fashion to paying for utility such as gas or water. This is attractive, as they have no upfront capital expenditure on software licences, hardware, or system installation. SaaS is also attractive to vendors as once a customer is signed up, these provides a regular and recurring revenue stream.

> *I don't think it* [SaaS] *necessarily makes sense for sticky, complex, back-office enterprise applications … maybe everything will be software as a service one day, but it's certainly not something we're seeing our customers drive us toward* (I-29).

Other software delivery models do exist and new ways of consuming and paying for software are likely to continue to emerge. OSS is another example of a software delivery model. The base software is free and vendors create a business out of providing software expertise, installation, support and maintenance for a fee.

4.9.3 First movers

Market share and market perception (Code-89)

Interview questioning on whether first mover advantage was deemed significant or not, led to informants highlighting the importance of market share and market perception for a

software business. This then channelled the conversation into talking about brand value and monopolistic power.

Market share can be an important factor that influences a vendor's ability to attract and successfully make new sales. Customers want to know that somebody else is already successfully using the using the product. In an industry littered with software customer horror stories, they do not want to the first person to be trialling a product that has not already been proven elsewhere.

> *Market shares make a difference in the software. Just because people feel confident about buying something that someone else buys … I think in Australia particularly, market shares are a huge thing* (I-03).

The experience that customers have with a vendor can provide substantial input into the perception the overall market has about a particular software business. A customer's experience with purchasing, implementing and consuming a software solution, can all make a significant and sometimes disproportionate impression upon this market perception.

Brand value (Code-90)

The importance of brand varies dependent upon the market in which a software vendor operates. One perspective is that companies with a well-understood and respected brand are likely to be more successful than those without. It is suggested that generating a perception of quality, trust and a sense of longevity, will positively assist a vendor in securing sales. However, several informants suggested that for niche application software vendors, the customer value proposition is considerably more important than brand.

> *At the end of the day, having a brand that is understood that is well-entrenched in a particular sector I think aids the selling process* (I-04).

Monopolistic powers (Code-91)

Informants' views on monopolistic powers were divided. One section of the informants suggested that a monopolistic position is desirable and that vendors should aim to achieve this where possible. Another group argued that while monopolistic powers had advantages, these were not based on generating customer value and likely to be short-term, and therefore should not be a core focus for a vendor.

> *You got a monopoly and you can charge what you like for the service and software. So, I think it is fairly natural behaviour to try and lock clients into your* (I-14).

Vendors being fully aware of a customer's reliance upon them and have often sought to minimise upfront costs and then when a customers is tied to a product, hit them with subsequent services, and maintenance fees. Forced product upgrades are common in the software industry. Some vendors argue that forced upgrades are a necessary evil. They help keep products standard and keep support/overhead costs down.

> *I lock in an organisation. I upgrade my product and tell me them I am no longer supporting the old version of the product is a cynical strategy for gouging my market. And so in that situation the concept of customer value is irrelevant and there is reasonable chances that the customer will not get value and so therefore disentangle themselves and move onto another supplier* (I-06).

Different degrees of customer lock-in were identified in the field. Outright exploitation at one end of the spectrum, with an acceptable leverage of a commercial opportunity at the other end. A monopoly strategy may now be coming less viable as the market is starting to offer viable alternatives to the mega-vendor offerings.

> *Oracle is finest for its exploitative pricing structures and yet they persist and continue to do it. So how do they do it? I think they manage to because the sail so close to the wind without actually abusing the client* (I-06).

Two areas where monopolistic power can have a significant influence on how a software business goes to market are software pricing and product bundling.

Pricing (Code-92)

Informants' opinions were split about the effectiveness and efficiency of the current software pricing landscape. One perspective was that the traditional model is a 'dinosaur' and is too expensive. The suggestion was that customers are fed up with having to purchase a whole product suite in a way that does not link to value creation, while also being locked into an inflexible agreement over many years. An opposing view from some vendors was that the current model works just fine and is not going to change anytime soon.

There are forces driving change in software licensing and pricing. SME vendors are likely to be under pressures to overhaul the way they charge for their solutions. Overtime, one-sided pricing models in favour of the vendor may diminish. Software pricing is likely to be driven down as customers increasingly become more demanding about how they want to purchase and consume software.

> *One of the struggles of taking the pure product approach is that you're often steps removed from the realisation of that* [customer] *value* (I-13).

Flexibility in pricing models is becoming more prevalent. Customers are demanding it and vendors need to respond to stay competitive.

> *Our strength … is the strength and flexibility in the way that we can deliver software in any way that a customer prefers to buy. And I think that flexibility is very powerful for us* (I-26).

Product bundles (Code-93)

The practice of bundling has been widely used by vendors to increase sales revenues. Due to the near zero marginal cost of providing extra software units to customers, there is an incentive to convince customers to buy more product when they are making a software purchase. Adding additional products into a software bundle at large discounts has been a way to achieve this. Often a core product is sold at a base price with optional or add-on

modules that can be bundled in. However, the practice can be somewhat dubious from a customer perspective. A bundle of software products may not map to a bundle of customer requirements. Often customers end up purchasing product that they do not actually use.

> *As much as vendors talk about solutions and talk about verticalisation, very few software vendors actually do it. They really bundle product together and call it solutions as opposed to truly understand an industry* (I-29).

4.10 Summary

The initial open coding of the interview transcripts has resulted in the generation of 93 base-level open codes. These codes highlight that there are a very broad range of factors that managers of software businesses need to consider as part of running their businesses. At this relatively early point in the data analysis, these base-level open codes should be viewed as a set of autonomous concepts. While Chapter 4 has been structured into eight broad interview topic areas, this has just provided a narrative for how the codes were identified, it should not be viewed as a higher-level grouping for how the concepts might fit together as part of model for running a software business. Table 4a list the full set of 93 codes. The codes have been sorted alphabetically and re-numbered accordingly. While many of these codes correspond to well-grounded concepts previously identified in the literature review (for example, product R&D, software purpose and market segmentation), it is also noted that the coding exercise has also generated a number of other factors that had not been considered prior to entering the field (for example, business ownership, capability sourcing drivers and metrics for strategic assets).

Table 4a: Base-level Open Codes

Codes 01-35	
01	Adjoining markets
02	Brand value
03	Business model existence

Codes 36-70	
36	Localisation
37	Long-term objectives
38	Management accounting

Codes 71-93	
71	Roadmap ROI and funding
72	Sales channels
73	Sales cycle

04	Business ownership
05	Candidate functions for external sourcing
06	Capability sourcing considerations
07	Capability sourcing drivers
08	Capital costs
09	Consolidation of vendors
10	Context of growth strategies
11	Contribution to overall business
12	Cost models
13	Critical mass
14	Customer focus
15	Customer perspective
16	Customer relationships
17	Customer value of support and maintenance
18	Customer value proposition
19	Customers
20	Delivery and consumption models
21	Doing services well
22	Domain experience
23	Domain leadership
24	External reference industries
25	Fitting customer requirements to product
26	Focused and controlled growth internationally
27	Functional perspective of vendor maturities
28	Future take-over
29	Growth objectives
30	Holistic performance management framework

39	Market distinctiveness
40	Market opportunities
41	Market segmentation
42	Market segments
43	Market share and market perception
44	Merger and acquisition
45	Metrics for customer satisfaction
46	Metrics for financial variables
47	Metrics for product R&D
48	Metrics for professional services
49	Metrics for marketing and sales
50	Metrics for support and maintenance
51	Metrics for strategic assets
52	Monopolistic powers
53	Multiple products and versions
54	Need product direction
55	Occurrences of failed businesses
56	Operational costs
57	Operational simplicity, agility and flexibility
58	Opportunities for growth internationally
59	Pricing
60	Product bundles
61	Product leadership
62	Product positioning
63	Product R&D
64	Product renewal
65	Product, technology, dynamism & market imperfections

74	Salespeople
75	Sales pipeline
76	Scalability
77	Services as a separate business line
78	Services supporting a product business
79	SME customer markets
80	Software business centre
81	Software capabilities
82	Software purpose
83	Software services
84	Specific software business risks
85	Strategic planning prevalence
86	Strategy gaps
87	Support & maintenance potential
88	Support and maintenance levels
89	Talented individuals
90	Target growth rate
91	Variations in maturity across the industry
92	Vendor leadership quirks
93	Vertical and niche markets

31	Inherent risk		66	Profit			
32	Intellectual property management		67	Profit and cost centres			
33	Leadership roles and functions		68	Reference points for strategic guidance			
34	Level of customer engagement		69	Revenues			
35	Lifestyle companies and continued existence		70	Risk profile			

The data interpretation exercise has resulted in a very rich set of concepts that describe many aspects associated with running a software business. However, 93 open is codes is a very large number of individual concepts to work with. It is infeasible to the construct an overall theory for running a software business, when so many different elements all need to be interrelated to each other and connected. A smaller set of higher-order concepts needs to be constructed so that the theory building can progress. Chapter 5 describes the intermediate analysis process that was undertaken to achieve this goal.

5 IMPLICATIONS AND THEORETICAL MODELLING

Having interpreted the field data and identified 93 base-level codes, the next stage of the analysis was to understand the implications of these codes and develop a set of categories that helped explain the various aspects of running a software business. This objective was achieved by following a two-step process. First, the 93 base-level codes were reduced into a set of higher-order concepts. Second, these higher-order concepts were then in turn fused into a small set of theoretical constructs. The structure of Chapter 5 mirrors this two-step process.

5.1 Higher-Order Concepts

The systematic process of creating higher-order concepts comprised of two parallel activities. First, the analysis went beyond the data interpretation of the initial open coding and moved into considering what the practical implications of each base-level code were for managers running COTS application software SME business. Second, the meaning and implication of each individual base-level code was compared with all the other base-level codes. Where these codes naturally fitted together as elements of a higher-order concept, they were amalgamated. The analytical process of developing a set of higher-order concepts was iterative. Numerous passes were undertaken to assess the implications of base-level codes and determine whether there were opportunities to join these into higher-order concepts.

The following sub-sections detail one-by-one the higher-order concepts that were developed. The reasoning for why each higher-level concept was created is described and an outline of the conceptual implications for a manager running a software business is provided. The order that the higher-level concepts are presented does not follow any particular theoretical structure. Rather, the sequence of concepts just loosely mirrors the

order that the analysis was undertaken and that higher-order concepts were created. For neatness, the higher-order concepts are structured with ten contained in each sub-section.

5.1.1 Higher-order concepts 1-10

Product development

The base-level codes of product R&D, software purpose and software capabilities have been consolidated into the higher-level concept of product development. While informants highlighted that it is important to have a robust product R&D process, having a point of reference point for controlling these activities is paramount. Clarity about a software product's purpose and clear logic around its capabilities provide the context for controlling overall product development activities.

Pinpointing exactly where a software product fits into a market is essential. Software product needs to solve a business problem. Potential market demand requires clarification. Ensuring that the 'right' software product is developed is likely to be critical. Software requirements need to include functionality, usability, flexibility, interoperable and maintainability considerations. Getting the right balance of the product capabilities is challenging. Product features need to be very focused on what provides value to a customer. They should not be overly complex or cumbersome to use. In particular, developing bloated products that have limited customer value is a risk. The actual process of constricting software product has now matured considerably as an engineering discipline. Many product development activities are now run as mature and disciplined engineering operations. However, developing a robust product development outfit requires significant investment. Product development economies-of-scale can be realised as a software business grows in size, but very smaller vendors can find themselves competitively disadvantaged.

Product management

The base-level codes product positioning, fitting customer requirements to product, product renewal and multiple products and versions have been consolidated into the higher-level concept of product management. While there are similarities to developing new product, discussed above, the concept of product management is more focused on the management of existing product. A key product management objective is likely to be on making more sales. In the dynamic application software market, refinements on how a product is positioned and how customer requirements meld against particular software products are ongoing product management focus areas. These are framed within the context of two other product management activities. The short-life cycles of the software market put a particular emphasis on product renewal. While having a continually changing product portfolio can result in the overhead of having multiple software products and versions to be managed.

Product positioning and differentiation are extremely important. Having a market that a vendor can call its own is likely to be a competitive advantage. This will be underpinned be being able to convince a customer why a product is the right one for them. This can involve implicitly or explicitly, illustrating why a competitor offering falls short of the mark. The ability to provide customers a shrink-wrapped software product can also be a competitive differentiator. One product management approach is to develop COTS software product as a set of flexible building blocks that customers can then assemble these in various different ways into a business solution. Providing customers with the ability to purchase optional pre-configured functional blocks can provide customers with immediate value. In addition, the more standard product that a vendor can sell, the higher potential profit.

Informants stressed that product renewal life cycles should be built into the business plans and product development activities. The optimal frequency for updating their software products will need to be determined. This requires a gauge on how long their product will be viable in a given market. With a product's capabilities continually evaluated in comparison to competitor products. Functional features, look and feel all, and maintainability are all likely to play a part. The right choices on underlying technology platforms, the base material for their products, are required. If a software product is constructed on a non-standard platform, it may fail to find buyers in the market. In addition, for those vendors that have multi-product business, product management activities are likely to encompass portfolio management. This would include standard practices such as exploiting 'cash cows', sun-setting 'dogs' and invested in future 'stars'.

There are benefits to be realised from a software business limiting the number of software products and versions. A consistent and reliable product will be easier to implement and easier to support. Leveraging a base product so it can to fit multiple customers and/or markets is a technique that can be used to limit a vendor's number of products and versions. A useful framework to assist vendors in this task is to consider software versions as a pyramid. The bottom layer of the pyramid resembles the main core software product. The next layer includes any regional or domain specific elements. The top layer includes any client specific capabilities. However, for a software business to operation a limited number of software products, all customers will need to be running these product versions. Upgrading to the latest product and version would need to be as seamless as possible, with the incentives for a customer to this being unequivocal.

Product roadmap

The base-level codes of product direction, and roadmap ROI and funding have been consolidated into the higher-level concept of product roadmap. A software product roadmap provides a strategy to drive significant product development work and a frame

of reference any product management activities. A product roadmap is likely to be driven by a software business' overall strategy. A roadmap requires explicit and clear definition. It will need to be justifiable in terms of its ability to provide a ROI and be commercial affordable.

Keeping a software product current requires a continuous development focus, which is important to stay competitive. There are likely to be a number of inputs for a product roadmap: the market; customers, competitors and an internal visionary. People who understand the market and understand new technologies are invaluable. Developing innovative new products that make it easier for customers to run their businesses are likely to drive a product roadmap. Competing product roadmap inputs are evaluated against the vendor's strategic objectives and then synthesised. A clear direction and description of the product roadmap will be depicted. A vendor's software product strategy is likely to state 'what', 'where', 'how much' and 'when' product developments will occur. However, the roadmap does not have to be cast in stone. Within reason, it can be quite dynamic if sensible ideas materialise for the addition of new capabilities.

Product roadmaps that are focused, justified and affordable are likely to be the most useful. Software is an investment intensive business and determining the exact financials associated with product developments is a key business consideration. However, informants stated that many software businesses did not consciously and/or proactively develop a long-term ROI model so that decisions on the level and focus of product investment could be justified quantitatively. While focused on market opportunities and vendor strategic objectives, how a roadmap fits with the vendor's risk profile and ability to finance R&D aspirations is likely to be critical.

Intellectual property management

The base-level code IP has been retained as a higher-order concept. The rational is that it is distinct and relevant enough in its own right. The implication for a software business of IP is that IP management becomes a discipline of relevance for running a software business.

A vendor can have the best software product in the world, but if it does not control its product IP then there can be a big risk to its business. Product copied illegally and counterfeit inferior competitor products may lead to lost revenues. Software that is altered illegitimately may become unstable and unsupportable, damaging a vendor's reputation. As IP is often wrapped up in people's heads, ways of explicitly capturing this are important considerations. Ensuring legally binding contracts with customers and suppliers are a mechanism for formally protecting the software business IP. Conversely, incorporating security features into products is a way to stop the illegal usage of software products.

Market positioning

The base-level codes of market opportunities, market segmentation and brand value have been consolidated into the higher-level concept of market positioning. These three activities all assist a software vendor to position its business and associated product/s in a market that will be attractive to serve. This positioning on a target market provides a focus that is likely to drive many other business directions and decisions. A vendor that is clear on where it wants to get to and how it will get there is likely to stand a better chance of achieving this than one that does not know what it is about. Concentrating energies on an attractive core business is important, as is not becoming distracted with other peripheral market opportunities. It was noted by informants that the process of a software business getting focus can take time and is often an evolving process.

Market positioning is all about getting focus. Identifying and qualifying opportunities are a key part of undertaking a detailed market analysis. Ideally, commercially attractive gaps in the market will be identified. Gauging which products are, or are not suitable should be a central marketing activity. Marketers will benefit from having strong market and strategic analysis skills. Being extremely market savvy is advantageous. Market segmentation is as useful technique a software business can use to become focused. Whatever dimensions a vendor uses to segment a market, the main principle that they are breaking down their market in specific chunks whose own specific nuances are understood. Informants suggested that the relative importance of a software business' brand in its given market segment has varying degree of importance. Where brand development investment is appropriate, the importance of this being clearly justifiable with tangible ROI metrics is worth careful consideration.

Market offering

The base-level codes customer value proposition, delivery and consumption models, pricing and product bundles have been consolidated into the higher-level concept of market offering. Historically, the core of a vendor's market offering and customer value proposition has generally been their software product and the value this creates for a customer business. However, increasingly a software business' market offering has become more multi-dimensional. There are new models for how software is delivered and consumed. Innovative pricing paradigms are being developed. There are incentives to customer from purchasing additional software that is bundled together.

Ideally, a customer value proposition will be a commercial differentiator for a vendor in its market. The challenge is how this can be attractive and value-adding from a customer perspective, but also commercial viable and financially profitable for an underlying software business. How software product solves a business problem and delivering customer ROI requires clarification. Aspects of the product that create customer value

provide a focus. Superfluous product features would ideally be removed. Getting the software product ‘working’ and creating customer value quickly and easily is important. Implementations that take a year or two to complete and only deliver ROI after five years or more are not really viable anymore. Flexibility and agility are a key vendor capability. While possessing the skills to integrate a software product into a customer IS environment are likely to be a key enabler of customer value creation. The customer total cost of ownership (TCO) is becoming increasingly important. While it may be common in the industry to charge additional and ongoing costs associated with a software solution, this is becoming less viable for vendors. Many software business are now thinking beyond a purely product mindset and now designing software solutions that consider the whole end-to-end customer experience. Many vendors are also attempting to position themselves higher up the customer value chain. Rather than being a provider of unseen products, they aiming to become an integral part of the customer business. Through the ongoing creation of value from its products and services, and the building of confidence over a period, it is possible to reach the standing of a trusted advisor.

The software industry is increasingly offering customers a range of options for how they pay for and consume software. The traditional of charging for a perpetual licence with annual licence fee is now being supplemented with models that are geared more towards customers paying for what they consume and the value they receive. Transactional or subscription pricing, ELA and SaaS models come with various pros and cons for both customers and vendors. Vendors can address customers are demands for increased flexibility by offering multiple options for how they pay for and consume software. However, it is important that as well as delivering customer value, a delivery and pricing model needs to be viable for a software business over the long-term as well.

To deliver software in a way that customers prefer to consume and to charge in a way that they prefer to pay, can provide a vendor significant competitive advantage. Linkage between software pricing models and the generation of customer value is a key principle. Pricing model inputs will include customer value, consumption levels, simplicity and the variant of time. Gauging market sensitivities are important, as is leveraging customer value based pricing opportunities. Financial loyalty schemes can be offered to repeat customers. It is worth review pricing models frequently.

Software product bundling is likely to be value-adding for a customer if software products are grouped by together into business solution bundles. These are essentially amalgamations of customer needs, not just bundles of software product. Those vendors that have product bundling right have managed to achieve multiple customer bundles from a definitive set of underlying products. Multiple permutations of product capabilities can be mapped to provide numerous specific customer solutions.

Sales

The base-level codes of salespeople, sales channels, sales cycle and sales pipeline have been consolidated into the higher-level concept of sales. A software vendor's success or failure is often directly linked to whether it achieves new sales or not. Due to the nature of the software business, the revenue from these is often inherently unpredictable. A strong sales capability is invaluable and an imperative. The attainment of sales will directly link to a vendor's ultimate success or failure. Have good salespeople and effective sales channels are key sales enablers. Understanding the software sales cycle and developing a robust sales pipeline are fundamental to actually achieving consistent sales. As well as making regular sales, doing the right sort of business is also critical. Sales centred on a vendor's core offering are important. They should be profitable and should ideally contribute towards its long-term success. Selling should be continuous and ongoing activity.

Good salespeople are invaluable. However, salespeople need to be given direction and closely managed. They need to sell what the business wants them to sell. Salespeople need to sell a vendor's market offering and they need to sell it to target customers in a vendor's target market. Sales rewards will vary from area to area and will ideally reflect profitability, strategic importance and risk levels. The ability to close a sale is everything. However, consideration of which sales are actually commercially sensible for the software business requires attention. Salespeople need to do more than just sell. Evaluating whether the vendor's offering is actually suitable for a prospect is just as important. Closing a software sale is likely to involve pre-sales consultants. Their knowledge will include a customer's business domain, the vendor's software product and experience of implementations in similar organisations. It is important to know to whom a vendor is selling at a customer's organisation and address this person, or persons, appropriately. Ensuring that a strong message for how their software can create business value reaches influential customer business executives can assist in closing a deal.

Effective software distribution is critical. Both direct and indirect sales models can be important. Deciding which are relevant and will be adopted is a key sales decision. Distribution networks have huge potential. A vendor that develops effective distribution channels can reap the results they deliver. There are many opportunities for vendors to learn from non-software industries that have well established and effective distribution channels in place.

A key activity of both marketing and salespeople is to generate sales leads. Qualifying the likelihood of individual opportunities being converted into a sale is an imperative. Having an understanding of customer businesses and how software products will fit into those

environments is a prerequisite. A sales pitch will clearly communicate exactly how a vendor software offering can create value in this context.

A structured approach to sales pipeline development is sensible. This will require qualification of opportunities and selection of the most attractive to target. Revenue and EBIT targets for the year need to broken down into a funnel of what the pipeline needs to look like. Targeting the right business is an imperative. Sales opportunities need to be qualified and the pipeline should be selective. Prospects that have a high rate of success should be targeted. Strategic customers that will provide ongoing and repeat sales are likely to be the focus. Maximising revenue opportunities from existing accounts are sensible. Sales to existing customers are generally easier to achieve.

Software professional services

The base-level code of software services has been retained as a higher-order concept. The implication for a software business is that software professional services become a discipline of relevance for running a software business. A software business can develop various professional services offerings to support and contribute to their overall software business. Professional services can generate customer value by providing a bridge between a customer's business objectives and the solution capabilities of a vendor's software product. Professional services can include pre-sales consulting, business consulting, software implementation and training. Having some form of direct linkage between professional services and a vendor's software product is deemed important. In addition, services need to provide tangible customer value and be commercially viable from a vendor business perspective.

Software implementation/integration services are vital part of moving a customer to a point where a vendor software product is up and running and delivering business value These services are therefore of particular importance. Aligning customer expectation with

what is actually achievable with a software product is an imperative. Implementation personnel need to be very clear and what a software product can and cannot be used to achieve. A customers business requirements need to be clearly and explicitly detailed. Getting the product on the customer's IT platforms and environment may require specialised knowledge.

Software support and maintenance

The base-level codes of customer value of support and maintenance, and support and maintenance levels have been consolidated into the higher-level concept of software support and maintenance. This amalgamation provides a more holistic picture of the key elements of software support and maintenance. Software support maintenance has the potential to move from being a questionable value proposition to being a win-win asset for both software customer and vendors.

Clear definition of what a vendor's support and maintenance value proposition is will provide clarity to customers on how they can benefit from its consumption. If a customer is paying for something, they need to feel they are actually getting something. Support and maintenance can help assure business continuity for customers whose business operations are increasingly dependent on software systems. However, it can be much more than just an insurance policy. Support and maintenance can also be about assisting customers in the use of product. Maximising the business value that can be generated from a deployed software product benefits both customers and software vendor. Satisfied customers provide an opportunity to secure long-term revenue streams. Proactively listening to customers who are using the software product is an imperative. Customer support and maintenance activities can be viewed as supporting a partner. A customer's success utilising a vendor's software product is success for both the customer and the vendor.

Vendors need to understand what levels of support and maintenance customers actually require and then optimise their support and maintenance operation around this. A software vendor can provide customers with various forms of software support and maintenance. To provide customers with a quality service that meets their requirements, and do this cost effectively is a delicate balance. Offering different tiers of service levels is worth consideration. Customer value should be maximised as cost effectively as possible.

Professional services business and contribution

The base-level codes of services supporting a product business and services as a separate business line have been consolidated into the higher-level concept of professional services business and contribution.

There was a very explicit view stated by the majority of informants about the role professional services should play within an overall software business. This was that software professional services should primarily exist to support a vendor's software product business. Product should drive and focus services creation. Services should explicitly compliment a vendor's software product; caution is advised about involvement in unrelated services. Services should be focused on creating customer value through the enablement and utilisation of the underlying software product. Professional services should support product sales and drive overall business profitability. The more product that can be successfully implemented, the more customer value is created, the more revenue and profit that can achieved from the underlying product licences. Regardless of what services work a vendor is undertaking for a customer, professional services consultants should always be looking for new business opportunities. Fully understanding a customer's business needs and pain points and being able to translate these into value-add professional services engagements can be a win-win for both parties. The further

entrenched a vendor becomes into a customer organisation, the greater cross-selling opportunities are likely.

Professional services work should be all about enabling profitable product revenue streams. The customer value-add from vendor professional services needs to be very clear. Services should not just be a additional mandatory cost of questionable benefit. Definition around what a vendor will do and what they will not do for a customer provides clarity for all. Avoiding services work that does not fit a vendor's services model is equally important. It is critical that professional services consultants possess the necessary breadth and depth of business and technical experience required to undertake the work well. Having a critical mass of highly skilled professionals who have the reach to serve all customers in a target market is important. A sensible and pragmatic way to build a services business is to grow through a partnership model. A vendor does not have invest directly in building and covering the cost and risk of this business; although on the downside they well have less control and quality could be an issue.

While professional services profit margins are not as high as those that are achievable for product, a services offering can provide a significant contribution to the overall software business. It can also be a secondary business line in its own right. A software business' professional services offering may be set up as a separate business line with its own P&L, or embedded within another business unit. Either model can be viable. However, how a professional services business is positioned within its overall software business can be critical to its success. Care is required to ensure that a move into providing professional services does not erode the strengths of a product business. A high customer touch point engagement model may deflect from ensuring sufficient focus is given to software vendor's core product business. Similarly, protecting against volatile services revenues

that may leave it with the overhead of consultants that are not billing requires consideration.

5.1.2 Higher-order concepts 11-20

Support and maintenance business and contribution

The base-level codes of contribution to overall business and support and maintenance potential have been consolidated into the higher-level concept of support and maintenance business and contribution. Informants felt that a vendor's support and maintenance business should make a net positive contribution to the overall software business. Revenues are relatively easy to attain. They can provide a vendor with a recurring revenue stream and profits can be high. However, it is important that short-medium term opportunities to realise support revenues are not abused. Exploitation of a profitable support business can destroy the overall software business. Vendors, who charge customers when they are not actually providing anything, are likely to find that fed up customers will disappear over time. Ideally, revenue and profit levels will be justifiable against the customer value generated.

There are a number of vendor considerations for optimally structuring a support and maintenance business. Vendor resourcing requirements need to be determined to ensure that customers are provided with required support levels. Providing a high quality support and maintenance offering requires critical mass. Getting the right balance between customer closeness and economies of scale from centralisation is a particular challenge. Determining the physical location of the support and maintenance business will have certain operational and cost implications.

Business leadership

The base-level codes of vendor leadership traits and leadership roles and functions have been consolidated into the higher-level concept of business leadership. A particular quirk

of the sector is that there can be a mismatch between the types of individuals that run software businesses and the skills that are required to be a business manger. Many vendors do not have strong enough leadership and consequently, software business can have inherent weaknesses that limit their success. Determining a pragmatic a model for effective business leadership of software business is an important consideration.

Strong leadership is critical for software vendors. Having a leader who has business management skills and can run a software vendor business as a business is a pre-requisite. Proper business controls and business management are very important. Most informants felt strongly that technical people should not head software businesses. In addition, they felt that there often comes a natural point in a software vendor's existence when it makes sense for a CEO owner and/or founder to exit and hand over the responsibility for taking the business to its next point. If owner/founders are going to continue to lead a vendor, it is likely to become increasingly important that they possess the skills and clarity to take the business forward. A new leader with a more rounded business skill-set is often more suitable as a growth accelerator to take the business to its next level.

The combination of a CEO and board of directors at the head of a software business provides a robust total leadership model. A talented business manager as a CEO is an absolute imperative. The CEO would then be fully accountable to the board and shareholders, with the strong and active board of directors providing appropriate checks and balances for the talented CEO to operate within. In addition, as well as ensuring effective governance, the board has a role to provide business discipline and focus for the organisation.

Human resources

The base-level codes of domain experience and talented individuals have been consolidated into the higher-level concept of human resources. Behind the software

product, the people who work in software businesses can play a big role in whether or not a vendor is successful or not. Having talented staff that understand customer businesses is an imperative.

Robust vendor human resources can provide a backbone for a software vendor business. Finding and recruiting highly talented staff, both strong technologists and strong businesspeople, is a critical part of running a software business. This requires careful consideration and needs to be proactively managed. Retaining highly talented resources is a big part of running a software business. Staff attrition is big issue in the industry and there can be significant fallout for vendors when key individuals leave. Proactively developing HR polices and appropriate employment packages to retain good staff are important.

Possessing domain expertise of customer businesses is critical. Customers want to feel comfortable that a vendor fully understands their operations, has a quality software product and a proven track record of implementing and supporting it. Being in tune with customers, having knowledgeable about their pain points builds credibility. Being able to provide value-adding business solutions helps build trust. Vendors can consider partnering from domain expert organisations if they need to make quick and significant uplifts in their capabilities in this area.

Customer engagement model

The base-level code level of customer engagement has been retained as a higher-order concept. The implication for a software business is that a customer engagement model has particular relevance for running a software business. The design of an appropriate and attractive engagement model will address all vendor sales, implementation and support interactions with customers. The engagement model for interacting with customers can have a large affect on it resourcing requirements and its financial viability. While the

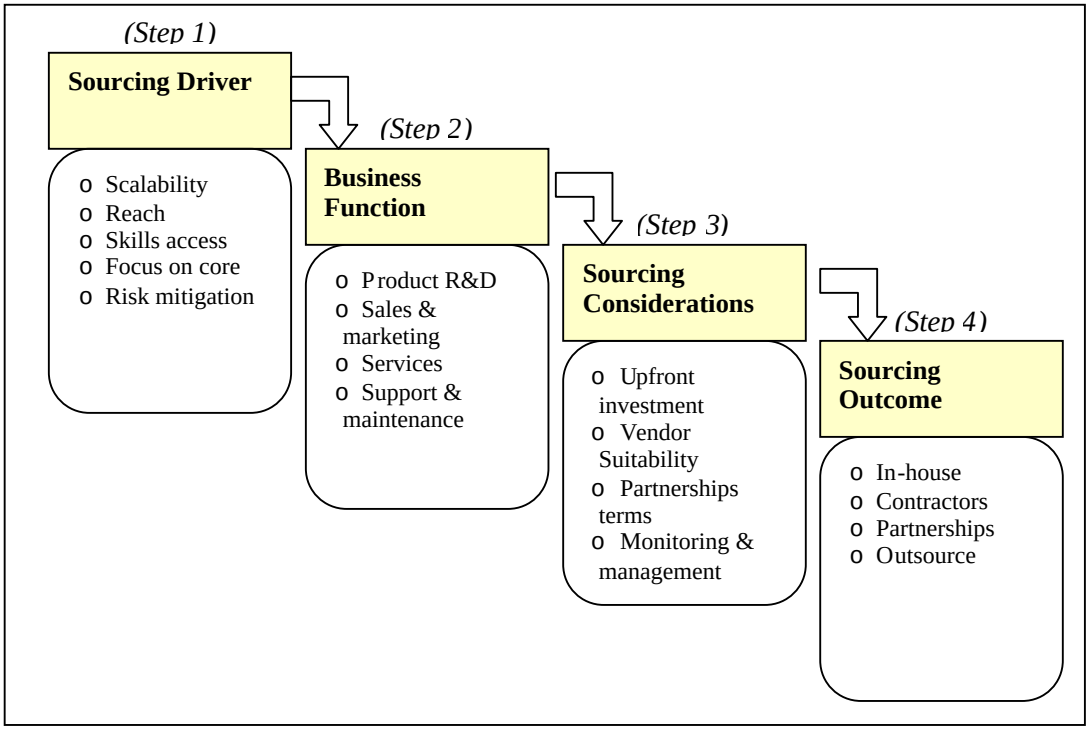
engagement model needs to ensure sufficient contact to meet customer needs, it is desirable for vendors to achieve a relatively low touch model with customers to maximise profitability. However, some vendors have high touch engagements with customer when they are implementing their software. Their logic is that these costs can be offset against tangible returns that can be realised over the total customer life.

Capability sourcing model

The base-level codes of capability sourcing drivers, candidate functions for external sourcing, and capability sourcing considerations have been consolidated into the higher-level concept of capability sourcing model. Many elements of a software business can be sourced from external providers and there can be various advantages from leveraging these opportunities. However, there can be downsides as well. A capability sourcing model provides a framework for sourcing capabilities in an appropriate and thoughtful way. The model displayed in Figure 5a splits the decision making process into four sequential steps. These encompass clarification of the overall business drivers for sourcing, analysis of different business functions suitability for external sourcing, assessment of sourcing considerations and finally determination of appropriate sourcing outcomes.

Sourcing resources externally can provide the ability to easily scale, gain reach in a market, and provide access to specific specialised skills. External sourcing also allows a vendor to hedge its risk as it not committing to a fixed overhead. Software vendors can formally scan their business functions and activities to identify areas where they can achieve a benefit from sourcing capabilities externally. Where the pros outweigh the cons, it makes sense for vendors to leverage opportunities for long-term net positive benefits. Activities where a third party can do a better job of providing a capability than the vendor are obvious candidates.

Figure 5a: Capability Sourcing Decision Model



The business functions of product development, sales, professional services, and support and maintenance all provide candidates areas where advantage may be achievable from sourcing capabilities externally. Quality levels of products may be improved from outsourcing software QA and testing activities. A considerable uplift in sales may be achievable from the development of distribution channels. Utilisation of third parties for implementation can be beneficial. However, implementation partners will need to have the necessary skills and ability to deliver software with limited support from a vendor. Transactional and non-specialised functions of support and maintenance that are non-value-adding for a software vendor should be considered for outsourcing. Support and maintenance is often outsourced. Being a transactional and non-specialised function it is sometimes viewed as being non-core business for a vendor.

The right capability sourcing model for a software vendor is dependent on a number of factors. If a vendor is going to leverage external parties, ensuring that it is still retains an

appropriate level of control over the key functions of its business requires consideration. Although vendors can leverage the advantages of sourcing from external parties, they may lose some control over the software solution that that a customer finally chooses. Large and important long-term customers are likely to require direct contact. Less important and transactional customers may be more appropriate candidates for servicing through external parties.

Carefully targeting and evaluating suitable external parties is a key part of putting in place a partnership and/or outsourcing arrangement. There needs to be a clear and realistic ROI model that accounts for all costs, risks and benefits over the long-term. Domain expertise and credibility of these third parties in the market is an imperative. A shared long-term interest and commitment to the sourcing arrangement is critical. The overhead of monitoring and managing third party suppliers is significant. Cost and quality factors should be explicitly accounted for in a benefits realisation model prior to committing a vendor to such a partnership.

Operational capabilities

The base-level codes of business size and operational efficiency and operational simplicity, agility and flexibility have been consolidated into the higher-level concept operational capabilities. This concept highlights several operational capabilities that are of specific relevance to a running a software business. It is noted that while these potentially fall within the broader topic of operations management, other elements of this wider domain was not raised by informants.

Several operational capabilities are identified as being important to a software business' efficiency and effectiveness. Having a critical mass of operational capabilities to run the various functions of a software business can affect the overall effectiveness and efficiency of a vendor. Informants referred to a definite point of critical mass that needed

to be attained. In particular, small business can struggle to deliver the diverse range of capabilities that are required to be competitive. However, as a software business grows, economies of scale can be achieved relatively quickly. A level of simplicity in a software offering can be attractive to customers and provide efficiencies to a vendor's operations. Ideally, software production, implementation and support activities will be relatively straightforward for both parties. Business alibility and flexibility to meet volatile and fluctuating business demands in the evolving software industry is important. The ability to be responsive and alter the focus of energies across the business can provide competitive advantage. Agility can be achieved in software product portfolio by ensuring the technical design is modularised allowing for easy modification and plug in of new capabilities. Flexibility in capacity can be achieved by having a cross-skilled resource pool. For instance, if resources can be moved between product R&D and professional services functions, this allows a vendor to change its business focus very quickly without changing its make-up.

Scalability model

The base-level code scalability has been retained as a higher-order concept. The implication for a software business is that a scalability model has particular relevance for running a software business. How to scale the business without taking on unacceptable business risks is a conundrum. A software business can benefit from having an effective scalability model. Ideally, a vendor will have a business model that can scale, and de-scale, relatively easily. Designed the right way, a software business should be able to take advantage of growth opportunities through its ability to scale up and down its operations. Understanding the she scalability potential of all the business functional activities of software development, sales, implementation and support is a pre-requisite. Ideally, it will have flexibility in all aspects of developing, selling, implementing and supporting its products so that it can respond to changes in market demand. However, a software business will require an established critical mass of product, processes, people and

customers before it attempts to scale its operations. In particular, partnerships can be considered as an effective means of scaling up quickly and managing risk.

Ownership model

The base-level code business ownership has been retained as a higher-order concept. The implication for a software business is that an ownership model has particular relevance for running a software business. Selecting the most appropriate ownership model for a software business is likely to be a key decision. There are likely to be specific pros and cons of private and public ownership models for different software businesses. Private ownership allows greater flexibility in business management, although vendors will want to be careful not to sacrifice business discipline as a consequence. In particular, private ownership may be suitable for smaller software businesses with a market capitalisation under $50 million. However, advantages of public ownership are that it can enforce discipline and enable growth through access to capital. Vendors with an established and sizable software business are likely to assess whether a listing as a public organisation will be appropriate for them as they move into the next stage of their evolution. Embracing this model may be big enabler of business growth.

Risk management

The base-level codes of inherent risk, specific software business risks and risk profile have been consolidated into the higher-level concept of risk management. A range software business risk related factors have been identified. The implication for running a software business is how a vendor can determine what their overall level of overall risk is and how this can be effectively managed.

A formal and rigorous approach to software business risk management is likely to be an imperative. As the software market continues to mature, a vendor's risk profiles is increasingly becoming a key business variable that requires close management. The

ability to assess and manage business risks in the dynamic software industry is critical. Vendors need to understand their business threats, decide the company risk level and profile, and track and manage accordingly. While qualifying individual risks is a key activity, determining the aggregate risk for the business as a whole is likely to be more critical. Spreading risks across different areas are a common approach. If one particular risk eventuates, then the business does not have all its eggs in one basket. A portfolio management approach to risk management is useful. A vendor can have visibility of individual risks, but can also balance the aggregate for the business as a whole.

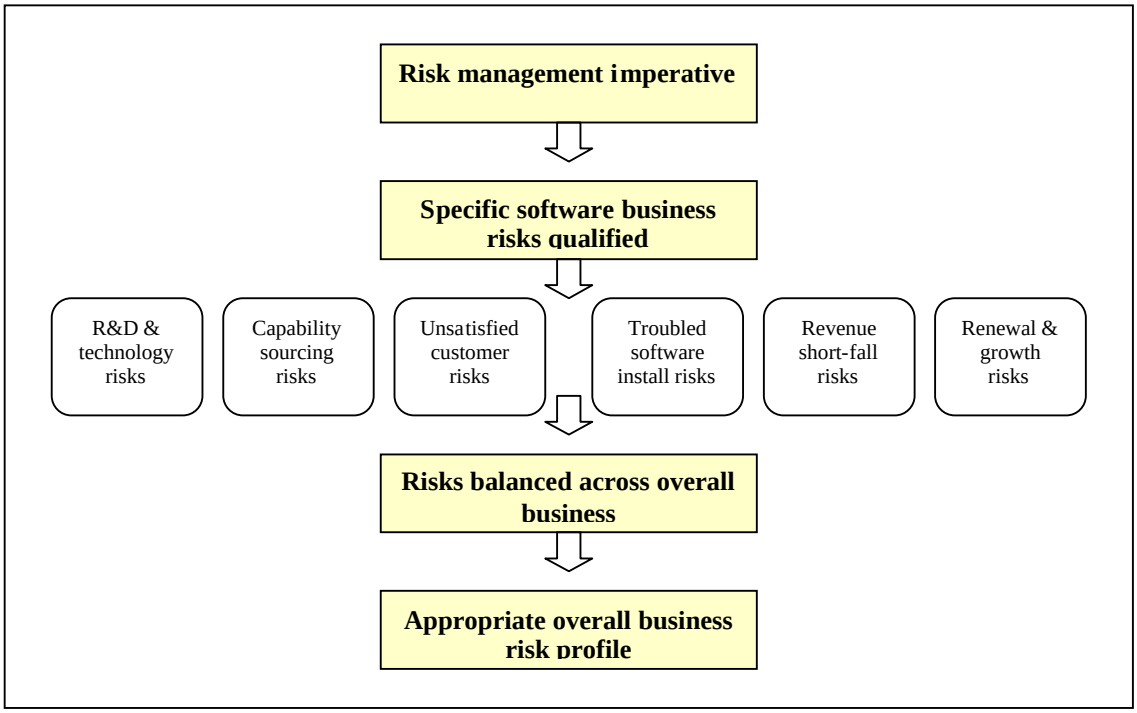
Having business flexibility is also advantageous. A vendor that can change the focus and effort it places on its different software businesses components will be able to manoeuvre as its overall risk profile evolves over time.

A software business risk management model provides a framework to assist vendors in developing a holistic and balanced approach to managing their commercial business risks. This model, displayed in Figure 5b, starts with a risk management imperative, qualifies individual software business risks, distributes risks across different business areas and aims to achieve an overall risk profile that is appropriate.

There are a number of specific risks related to functional aspects of a software business. R&D and technology decisions require careful consideration. Costs, benefits and risks should all be assessed. Risks associated with software implementations need to be limited. While a taking on a level of liability for an implementation is attractive to customers, effectively managing and containing a vendor's commercial exposure is important. Good pre-sales and ensuring that the right people with the right skills undertake the actual implementation work can lower implementation risks. There can be big risks associated with their capability sourcing decisions. Pros and cons of different

capability sourcing decisions require qualification. Calculated and balanced risk talking is likely to be appropriate. The outcomes of different resulting scenarios require factoring into the business plans. Managing the risk of software business revenue streams drying up is an imperative. Developing a mix of three core revenue streams comprising product, services and maintenance, is an important risk mitigation consideration. Revenue risks can be minimised through a solid annual sales cycle that brings in regular new sales, openings for services opportunities and the ability to progressively build up of recurring maintenance revenues. While it may seem tempting to target exponential growth opportunities, the upsides of this require balance against the potential downsides that could eventuate.

Figure 5b: Software Business Risk Management model



Sustainable and progressive growth, with a containable risk profile, is likely to be a preferable option for many software businesses. Sustainability equates to staying focused on a known market where a vendor has a strong position.

Customer retention

The base-level code of customer relationships has been retained as a higher-order concept. The implication for a software business is that customer retention has particular relevance for running a software business. Attaining a portfolio of satisfied customers who will continue a long-term commercial relationship with their software business for many years into the future is laudable aim.

Effective CRM capabilities can be a vital part of customer retaining customers. Developing closeness to customers is a pre-requisite to attaining trust and loyalty. An effective mechanism for achieving this is through direct interactions and personal relationships with customer personnel. Ideally, vendors should aim to generate a feeling of confidence in their customers. Underlying this, needs to be a convincing ability that a vendor can add value to a customer business and has complete integrity in the way they operate. Customers need to understand fully a vendor software value proposition and how the business value this generates can be maximised. Therefore, the benefits from continually educating customers can be considerable. Customer education is likely to be multifaceted and involve multiple customer representatives. The stronger the relationship that a vendor has with a customer the greater opportunities there are likely to be for attaining additional revenues.

5.1.3 Higher-order concepts 21-30

Customer experience

The base-level code of customer perspective has been retained as a higher-order concept. The implication for a software business is that customer experience that is had with a vendor is of particular relevance for running a software business. Understanding how customers look at things enables vendors to have a clear picture of what their customers are experiencing and thinking. The customer experience software covers the purchase,

implementation and consumption of software. It is desirable that customers will have positive experiences at all stages of this life cycle.

Customer centricity is vital. While a vendor offering may comprise a mix of product, content and services, this is not the best context and terminology to frame conversations with customers. Talking in language that refers to their problems and how these will be solved and the business value that will be created is preferable. A pre-requisite is having capable human resources that have domain expertise of customers' businesses. These people will be able to talk the language of the customers and their businesses. Ideally, implementations should be straightforward. With time and expense involved being justifiable. Services will compliment the underlying product and be focused on deriving the most value for a customer organisation. The aim is for customers to have an unambiguous and clear tangible return on their software investment.

Market strength

The base-level codes of market share and market perception and monopolistic powers have been consolidated into the higher-level concept of market strength. Attaining strength in a market segment can lead to a virtuous circle of making new sales and signing up new customers. The inverse is also true. If potential customers are unaware of a vendor market presence, its ability to make new sales may be severely constrained. Building up a market presence and a critical mass in the sector is likely to be an important business objective.

There may come a point when a vendor's strength in a market puts in a position where it has monopolistic powers. However, it is important not lose sight of the need to provide value for money to their customers. Forced customer software product upgrades and extended product lock-ins may be attainable, but they are unlikely to be in a vendor's long-term interests. While it makes sense to leverage positions of strength, outright

exploitation is questionable practice and likely to be unviable over the long-term. The long-term damage from exploiting customers is likely to be greater than the long-term revenue opportunities from satisfied customers. Retaining customers through the ongoing provision of high-value customer value propositions is likely to be a more appropriate long-term approach.

Cost model

The base-level codes of operational costs, capital costs and cost models have been consolidated into the higher-level concept of cost model. A wide range of costs elements and associated complexities has been identified as part of running a software business. An implication of having to make decisions around all these individual cost factors, is determining how all the fit together into an overall cost model. Cost management and control will be a fundamental component of a vendor's financials. Deciding what sort of cost model a vendor will adopt and how business costs will be proportioned across different business areas and functions requires careful thought.

A software vendor costs can be broken down from a direct *versus* indirect (overhead) cost perspective and from a variable *versus* fixed cost viewpoint. The main overheads of a software business are product R&D, followed by general and administrative activities. Direct costs encompass cost of sales, marketing, services and support. The main fixed cost is staff costs. Conversely, variable costs often comprise a low percentage of a vendor's total business. The ongoing and future costs of a software business needed to be funded. Appropriate funding sources need to be determined. Variables for direct *versus* indirect costs; and fixed *versus* variables costs should be factored into the overall cost model. The cost of product development and product support needs to be funded appropriately. The magnitude of total staff costs within the business needs to be understood. Having the flexibility to alter costs as there are changes in business demand can be invaluable. The direct cost associated with making a sale will need to be

quantifiable and justifiable. The ability to access to investment capital is important. This may be needed to leverage opportunities, as well as to fend off business threats to stay competitive.

Revenue model

The base-level code of revenues has been retained as a higher-order concept. The implication for a software business is that a revenue model has particular relevance for running a software business. Developing a robust overall revenue model that contributes revenues from multiple component business areas will both maximise revenue opportunities and help balance financial risks.

Revenues can come from a mix of new product sales, recurring revenue such as support and maintenance or usage fees, and professional services. Developing recurring revenue streams as the majority source of their overall revenues is particularly attractive. If there is a net increase in the number of customers using a vendor's software products every year, then this will flow on to increases in recurring revenues as well. However, building up a large recurring revenue stream requires a significant investment in time and energy. Whatever licensing or business model a vendor decides upon, it is critical that where recurring revenues will come is explicitly identified and the contribution they will make to defined. Maximising the length time customers use a product can lead to increases in lifetime support and maintenance revenues from customers. Ideally, customers will use software products for at least ten to fifteen years. Targeting increases every year in the percentage of total revenue that is recurring as a common aim.

Profit model

The base-level codes of profit and cost centres, profit and management accounting have been consolidated into the higher-level concept of profit model. A formal and robust financial ROI profit model will assist in providing a blueprint for a software business'

long-term success. A 25% plus gross profit margin is attainable by a fully established software business that has matured in terms of products, services and customer base.

Effective profit centre and cash flow management is fundamental for balancing overall revenues with overall costs. The overall costs of running a software business need to be determined for each of the vendor's business functions. The target contribution and margin of product sales, professional services, and support and maintenance profit centres require definition. Overheads for product R&D, and general and administration cost centres need to be appropriate. All costs need to be appropriately allocated across the business as a whole. Determining an affordable and appropriate level of investment in product R&D is likely to be one of the main considerations. This fixed cost metric is likely to have a large influence on the vendor's overall financial model and ultimate profitability. Cash flow in particular was highlighted by informants as an area that needs to be managed very closely. A management accounting function that embraces all financial aspects of the software vendor business is likely to be an imperative. Management accounting can assist in forecasting and tracking financial metrics across and down through all individual business and departments in the organisation. The overall financial health and profit level of the software business will come from an aggregate view.

A hierarchy of profit margins is likely to be key component of profit model. While profitable areas such as maintenance are likely to be leveraged, having a balance between the hierarchy of business areas by profitability *versus* the role and contribution they make to the long-term business strength is important. As a software business grows and matures, there is an opportunity to increase profit margins. Reducing costs and improving the ease of increasing each new revenue stream are a way for this to be achieved.

Software business core

The base-level codes of software business centre and business model existence have been consolidated into the higher-level concept of software business core. While each of the functional business areas has its own set attributes and variables models associated with it, it was not initially apparent how it connects into and contributes to an overall software business.

> *You can have great sales and marketing, you can have great software developers, you can have a really good support model. But all of those things MUST come together* (I-26).

Having a formal and explicit business model underpins an overall software business is important is likely to be pivotal. There will be a number of key factors are identified that connect and bind an overall software vendor business together. Although, having a commercially viable software product that solves business problems is at the core of a vendor software business.

KPIs for primary business functions

The base-level codes of metrics for product R&D, metrics for sales and marketing, metrics for professional services and metrics for support and maintenance have been consolidated into the higher-level concept of KPIs for primary business functions. It makes sense to group these business metrics together into a single group of KPIs, as collectively they cover the end-to-end primary functional activities that a software vendor undertakes. These KPIs generally are quantitative values of tangible business activities that can be explicitly measured.

There are a number of important product R&D KPIs. Quantifying and qualifying product quality is important. Capturing customer feedback on software product provides a useful perspective. Measuring and tracking the number of bugs a product has prior to release is important to minimise any criticism customers might make about the product's quality to

other market members. A certain percentage of revenues will be invested back into in product R&D. This needs to be adequate to keep the product current, but also affordable and financially justifiable by the business. Production and delivery effectiveness of software product are worth measuring. The contribution of product R&D to a vendor's overall product roadmap should be calculated.

KPIs for marketing and sales are primarily financial. The contribution of marketing can be measured with KPIs for qualified leads generated. Some effort is worth making to understand the value of the vendor's brand. Revenues and sales are generally very closely tracked against targets. These are broken down into new business, repeat business and recurring revenues, month by month for a financial year. The business pipeline health is tracked. Vendors can use their historic sales conversion rates to project future sales based on the number of potential customers entering their pipeline. The relationship between their cost of sale and customer profitability is important. Non-financial sales targets are also worth serious consideration. This will incentivise salespeople to take a long-term perspective about the lifetime value of a vendor's customer.

KPIs for professional services are often over simplified. Achieving the right balance between services margins and services utilisation, is a key business decision for maximising total revenues and total profitability. Tracking actual utilisation and ensuring it is as close to target utilisation is a key performance management activity within a professional services business, but it is not be everything. A professional services business' target profit margins are ideally driven by its purpose and contribution to a vendor's overall business. Capturing whether a services engagement was a success and delivered customer value is important not to overlook.

KPIs for support and maintenance can provide an insight into a number of underlying areas of a software business. The per cent of total revenues will vary from business to business. The appropriate figure can be qualified by whether it is viable and sustainable. High support and maintenance margins can be viable long-term if customer feels they are getting value. This value may be being generated on an ongoing basis, or be view as being owed to the vendor for the effort they invested in building and implementing the software solution. A maintenance margin of well over 60% may be obtainable in some cases. Support centre resourcing levels are calculated based on inputs detailing the capacity requirements required to support the product customer base. Formal and agreed customer SLAs generally include performance metrics on customer issue resolution times, customer SLAs and total support calls.

KPIs for the overall business

The base-level codes of metrics for financial variables, metrics for customer satisfaction and metrics for strategic assets have been consolidated into the higher-level concept of KPIs for the overall business. While many KPIs have been identified for individual business functional and associated activities, having a single set of KPIs that measure the strength and performance of the overall business is arguably even more important. The base-level codes consolidated in this higher-level concept contribute to achieving this aim. However, it is noted these KPIs are less well defined than the functional area KPIs. This is largely due to their intangibility and the subjectivity involved in measuring them. In addition, there is limited explicit data from the field describing these overall business KPIs. In a number of cases the KPIs identified in this section, have instead originated from important software business concepts that have emerged from the analysis.

The financial variables and measures that make up the software business' financial model provide the basis for a set of important financial KPIs that should be included in a vendor's overall set of business KPIs. The sub-sections above describing cost model,

revenue model and profit model, all have implied KPIs that supplement those explicitly detailed in the financial variables base-level code.

KPIs for customer satisfaction are increasingly recognised as being paramount. Formally and regularly measuring customer satisfaction provides and important KPI for a software business. If a vendor has high satisfaction levels, this can be a commercial differentiator. If satisfaction levels are low then the long-term viability of the business may be at risk. A customer satisfaction thermometer will capture the main aspects of customer experiences dealing with a vendor and using its products. The customer satisfaction indicator is likely to include both qualitative and quantitative attributes. Measures would combine formal customer survey data as well as feedback received from informal conversations with customers. Customer satisfaction measures should be updated at least half-yearly, or more regularly if appropriate and feasible.

KPIs for the strategic assets of a software vendor are currently very immature. This area would benefit from significant attention. Qualifying and quantifying KPIs for a software business' strategic assets is a high value activity. Understanding the strengths and weaknesses of a vendor's strategic assets can be directly linked to a vendor strategic focus and its long-term success or failure. While there is limited guidance available for exactly how to measure a software business' strategic assets, understanding the strategic strengths of a business is worth a vendor's investment of time and energy. This would involve clarifying what a software business' strategic assets actually are, attempting to measure strategic factors that are currently not measured and actively using qualitative measures as well as just tangible quantitative indicators. While informants did not always make explicit mention of a business KPI, the importance of measuring a strategic business attribute was often implicit in their identification of that business element as being important.

Total business scorecard

The base-level code of holistic performance management framework has been retained as a higher-order concept. The implication for a software business is that having a total business scorecard is of particular relevance for running a software business. Developing and using an appropriate and effective overall performance management framework for a software business is imperative for a software business' long-term success. While numerous individual business metrics and indicators have been identified and described, a holistic picture of the strength of a software business still requires clarity. Ideally, a vendor needs a relatively simple picture of its overall strength and performance potential. The challenge is how a diverse and complex set of business KPIs for multiple software business functions and activities can be integrated into a single holistic model. The overall performance management framework, the top-level representation of business KPIs, can take various forms. There is no one right way. Pulling all the vendor metrics together into a total business scorecard is a common approach to setting up an overall performance management framework. The classic balanced scorecard based upon Kaplan and Norton's (2004) model suggests grouping metrics into financial, customers, internal, and learning and development perspectives.

Strategy development

The base-level codes of occurrences of failed businesses and strategic planning prevalence have been consolidated into the higher-level concept of strategy development. Developing a strong business strategy is an imperative. Business failure is common occurrence in the software industry. The trap of having a fundamentally flawed strategy should be avoided. Ideally, all software business should have a strategy and a documented strategic plan. This would clarify a target market, a competitive software offering and an approach to developing and operating a sustainable business. Just because a vendor is small, or a new business, this is not viewed as a reason for not having a formal strategy

and associated plans. The process of developing and formulating strategy will need to be robust and based upon sound data.

A minimum three-year strategy horizon is common for a strategic plan. This time length is long enough to set strategic direction, but also short enough to provide solid linkage into the business' current operations. It is therefore meaningful and value-adding to the running of the business. In some cases it may be appropriate to plan further out than three years, the key factor is to ensure that strategic planning is undertaken prior to each next big cross-roads in the business' future.

A vendor's management team will pull all elements of the strategic plan together and ensure it is realistic and can be successfully executed. All angles of the strategy development will be covered off and appropriate participants involved in the strategy development process. Having access to executives who have proven business experience in previous similar organisations is a fundamental. The board may provide the primary vehicle for this. Leveraging the knowledge and experience of internal resources, C-level executives or not, is an opportunity to be maximised. External consultants can be engaged, but it is important that these are experienced in business management and it is important that they take the time to fully understand a vendor's software business. Strategy inputs and validation will need to cover all aspects of the business, including sales and marketing, product development and management, operations and financials.

5.1.4 Higher-order concepts 31-34

Strategy tools

The base-level codes of strategy gaps, reference points for strategic guidance and external reference industries have been consolidated into the higher-level concept of strategy tools. While informants suggested that there are number of strategy gaps in the thinking

for running a software business, the research has also revealed that if managers are resourceful there actually a number of reference points that they can use to assist in developing strategy. As well as the more obvious strategy management discipline and industry specific research, a number of external reference industries were identified as being able to provide a contribution to the strategy tools that a software business has at its disposal.

While there is a lack of a standard repository of advice on how to run a software business, it is recognised that there are a number of other sources from which vendors can learn. Making use of relevant industry research and being aware of topics in the industry press is a starting point. While this does not always provide the most relevant advice for running a software business, the topics from these sources are factors that a software vendor could benefit from being up-to-date with. The strategy management discipline is extremely relevant. Identifying and using those general strategy reference models that are of most relevance to running a vendor's individual software business is key. However, unrelated industries also have much to offer in terms of guidance. These can be scanned to look for better ways of undertaking the various facets of their business operations. Construction and engineering type industries are particularly relevant for software product R&D operations. Product development quality and timeliness are areas where these external best practices can help drive in improvements in software businesses. While pharmaceutical and automotive sectors can be used as good reference industries with mature capabilities in the effective management of distribution channels, partnerships and brand marketing. With regard to providing guidance on undertaking professional services and support activities, although the findings did not identify a specific reference industry, it is suggested that one is likely to exist and vendors should attempt to locate one.

Clear business purpose

The base-level codes of long-term objectives, growth objectives, target growth rate, future take-over and lifestyle companies and continued existence have been consolidated into the higher-level concept of clear business purpose. Having clarity of long-term business objectives emerged as an impetrative out of the research. While vendor's can vary in terms of their business objectives and their business philosophy, being meticulous in clearly defining the business it is in and its strategic goals is critical. Clarity on the product, clarity on the target market, and clarity how the vendor's offering differentiates itself amongst competitors are all standard strategic planning concepts all need to be articulated.

Growth is the dominant and primary objective for many software vendors. This is likely to have importance both from the ability to prosper as a long-term player in a target market, but also from a survival perspective as well. Software vendor management teams will ensure sufficient focus is given to working out how the business should be structured and set up to achieve long-term sustainable growth. An appropriate target growth level will vary from vendor to vendor and this will largely be determined by the market opportunities and aspirations of the business owners. Steady controlled incremental sustainable growth is likely to be the preferable option for many businesses. This is likely to include capitalising on available opportunities and not risking a window of opportunity being lost. Investment in the development of new capabilities required to support growth and an enlarged business. While this will be balanced by evaluating risks and ensuring that if expected growth is not achieved, any losses from this are containable.

Two alternative software business objectives to growth are setting a company up for a future sale and operating a lifestyle business. Strategy and the formal articulation of business goals is just as important for these vendors with alternative objectives as it is for

those focused on pure growth. If the long-term objective is to sell the business, working out when it is it most appropriate to do this needs to be determined. Similarly, while life-style companies may not have purely commercial objectives, having realistic and sustainable objectives in their given software sector segment is important if the vendor wishes to have longevity.

Target market analysis

The base-level codes of market segments, market distinctiveness consolidation of vendors, variations in maturity across the industry, product, technology, dynamism and market imperfections, functional perspective of vendor maturities and customers have all been consolidated into the higher-level concept of target market analysis. A comprehensive market analysis of a vendor's competitive landscape is a key first step in the process of formulating a strategy for a software business. While strategy and marketing domains provide extensive reference points for how to undertake a market analysis, specific considerations for the COTS application software market include understanding distinct software market characteristics, accounting for a market that is still maturing and serving increasing savvy customers well. While there are many market factors that business managers can consider, as demonstrated by the seven base-level codes that have been consolidated into this single higher-level concept, the output of a target market analysis will be a clear picture of the target market that a software business intends to compete within.

The software market is as a maturing industry, but it is not a mature industry. There are elements of the industry that demonstrate characteristics of mature sectors, but there also other components of the software market that are still relatively immature. There has been extensive vendor consolidation, particularly so in the ERP space, but many specialised and emerging market segments still have numerous suppliers. Increasing product maturity, technology advancements, continued market dynamism and an imperfect

market for customers are all features that define the competitive landscape. Improving vendor capabilities and higher performance expectations are observed in both product development, and sales and distribution functional business areas.

Specifics of the market that a software business competes will have a direct affect on its business strategy. Its approaches for how it addresses the various threats and opportunities of its market landscape will require direct and explicit attention in its plans. The maturity and sophistication of product offerings in given market segment varies. Less mature segments will offer opportunities to provide improved products to customers. Segments already served by mature offerings are likely to have higher barriers to entry and hence will not be as attractive to compete in. The impact and importance of technology developments requires carefully qualification. Technology advances and the technology hype cycle can result in strong competitive forces. Appropriate business measures in response to these may be necessary. There are many imperfections in the software market that can be improved. Vendors can proactively work towards improving the value proposition for customers and functioning of their businesses. Although change is occurring relatively slowly, it is well worth vendors being ahead of the curve if this does change in the future. High quality software product is now the expected norm. Embrace engineering principles and rigorous QA processes to guarantee quality product are now a necessity. Value based selling is critical to staying competitive. Developing appropriate partnerships and building distribution channels are increasingly vital for growth. Threats and opportunities related to further market consolidation require understanding. This is particularly so for software markets that are still very fractured.

Serving software customers well is becoming an imperative. Customers are now much savvier and the proactive management of customer relationships is increasing in importance. Ideally, interactions with customers will be in an informative, professional

and credible way. While developing relationships with both customer business and IT contacts is generally required. These interactions need to be based on content as well as personal relationships. Listening to and addressing the things that really matter to customers is vital; as is the provision of straightforward, low-cost and flexible solutions to customer business. Legacy practices of aggressively pushing product onto unsuspecting customers are best avoided. All touch points with the customers: build, selling, implementing and supporting product, will ideally provide value from a customer perspective.

Growth strategy

The base-level codes of context for growth strategies, product leadership, customer focus, domain leadership, opportunities for growth internationally, focused and controlled growth internationally, localisation, vertical and niche markets, adjoining markets, SME markets and M&As have all been consolidated into the higher-level concept of growth strategy. An implication for a vendor of having many growth alternatives is that it is important to have a clear a focused and growth strategy. A purposeful rational behind consolidating so many (eleven) base-level codes into a single level higher-level concept, intends to move a software business away from a thought process where it might have numerous approaches to growth that are being pursued simultaneously. While many software businesses appear to have been almost overwhelmed with the growth alternatives open to them, a driving principle of strategy is that it should be focused.

The first place to look for growth is organically. The potential of organic growth opportunities is a good starting point for consideration. These natural extensions of the business are advantageous as they will not involve a vendor taking on so much business risk.

Market leadership is another strategy available to vendors intending to achieve competitive advantage and growth. Product leadership, customer focus and domain leadership are three slightly different types of market leadership that software vendors can pursue. Developing a competitive product can provide the advantage needed to grow a software business. There should be a straightforward reason why a software product provides a superior customer value proposition and this should be easily communicable. While this approach has been common to date, it is not likely to be the most appropriate approach for growth in a mature and saturated market segments. A relentless customer focus can also provide the competitive advantage needed to grow a software business. The customer perspective is cultural mindset. Management of the customer relationship at all points is an imperative. A loyal customer base that creates growth opportunities takes time to build. However, customer value generation will need to be balanced against the commercial viability of providing a software offering to the market. A focus on domain leadership is another alterative that can provide the competitive advantage needed to grow a software business. A unique market to be served will need to be identified. How exactly a vendor can be a leader in that domain needs to be articulated. Focusing specifically on that domain and sticking will be an imperative.

There are often huge opportunities in growing internationally. The potential and importance of serving overseas markets is an important consideration as a software business grows. Due diligence is required when investigating which particular regional markets are the most attractive to serve. A focused, controlled and progressive approach to overseas expansion is likely to most attractive. Investment levels and risks can then be contained. An overseas growth strategy that targets the most penetrable countries first often makes sense. Expenditure in developed localised product for overseas markets will need to be carefully considered and commercially justifiable. When considering product localisation, staying faithful to the core of a software product is a sensible approach.

Product localisation, such as language additions or country specific functionality, ideally will only be minor additions on the peripheral of a product. These additions should not require a disproportionate about time and energy that detracts away from the main customer value generating capabilities of the product. A local presence and local relationships can be a necessity for penetrating a local market. Engaging with partners who already have a presence in a local market instead of setting up this capability from scratch is a good approach. However, selecting the right partners with the right knowledge and connections is critical. The untapped opportunities offered by emerging markets can make them a particularly attractive proposition for many vendors targeting growth overseas. Evaluating the potential of these markets and acting accordingly is a key business challenge.

Similar markets to the market that a vendor operates in currently can offer potential for further growth. Opportunities frequently exist to move into adjoining markets. These often comprise making slight variations to a software product set so that it can be redeployed to a whole new set of customers. Caution is advised on how much modification should be made to an existing product core; this needs to be commercially viable and the ROI should net positive. Serving SME markets is now very popular and these can offer commercial potential. While these markets are generally underserved by software solutions, rushing in without due diligence can be very dangerous. An offering to an SME market is likely to be slightly different to a vendor's normal offering. Assessing what can be provided at what cost and how this generates customer value is likely to provide will be essential to arriving a commercially viable model.

The ability to achieve significant growth by M&A still offers much potential and many vendors are still considering this as a growth option. M&A is theoretically driven by the ability to create and deliver business and shareholder return. To ensure this is the case, a

due diligence process will need to articulate exactly what the business benefits of such a venture are. The aim is for the ROI and financial profit gained from undertaking M&A to deliver tangible commercial returns. Gains in IP, customer base and market share need to be justifiable with hard business performance metrics.

5.1.5 Summary

Further examination and analysis of the 93 base-level codes has resulted in these being consolidated into a smaller set of 34 higher-order concepts. These are listed alphabetically in Table 5a.

Table 5a: List of Higher-order Concepts

Higher-order Concepts 01-17	
01	Business Leadership
02	Capability sourcing model
03	Clear business purpose
04	Cost model
05	Customer engagement model
06	Customer experience
07	Customer retention
08	Growth strategy
09	Human resources
10	Intellectual property management
11	KPIs for the overall business
12	KPIs for primary business functions
13	Market offering
14	Market positioning
15	Market strength
16	Operational capabilities
17	Ownership model

Higher-order Concepts Codes 18-34	
18	Product development
19	Product management
20	Product roadmap
21	Professional services business and contribution
22	Profit model
23	Revenues
24	Risk management
25	Sales
26	Scalability model
27	Software business core
28	Software professional services
29	Software support and maintenance
30	Strategy development
31	Strategy tools
32	Support and maintenance business and contribution
33	Target market analysis
34	Total business scorecard

The exercise to create a set of the higher-order concepts has allowed for an element of balance to be introduced to the emerging concepts. While 93 base-level codes were generated, not all these codes were deemed to have the same weighting or level of

importance in relation to running a software business. In particular, nine of the 93 base-level codes have been elevated to higher-order concepts in the own right. While at the other end of the spectrum, the new higher-order concepts target market analysis and growth strategy have been formed from the consolidation of eighteen underling base-level codes.

While 34 higher-order concepts are abstractly more manageable, it is still a sizable number of individual concepts to work with. An objective of grounded theory open coding is to arrive at a point where the implications of the data are captured in a relative small set of conceptual categories. Section 5.3 describes the next step of the analysis process that was undertaken to achieve this goal.

5.2 Theoretical Constructs (Open Coding Categories)

Having established a definitive set of 34 concepts related to running a software business, the analysis moved into a process of fusing together these concepts into a set of categories. While the creation of higher-order concepts from base-level codes involved merging together units of meaning into higher-order entities, the focus for this part of the analysis was on identifying relationships and establishing connections between the set of already established concepts. In grounded theory, these are referred to as 'open coding categories'. The result of this exercise is the creation of a relatively small set of theoretical constructs. Individually, each theoretical constructs provides a category that explains the factors involved in a particular component of running a COTS application software SME business works. In aggregate, the theoretical constructs provide a definitive (but unintegrated) set of categories that cover all aspects of running a COTS application software SME business.

In total thirteen theoretical constructs were created. The following sub-sections detail one-by-one the theoretical constructs that were developed. The reasoning behind the

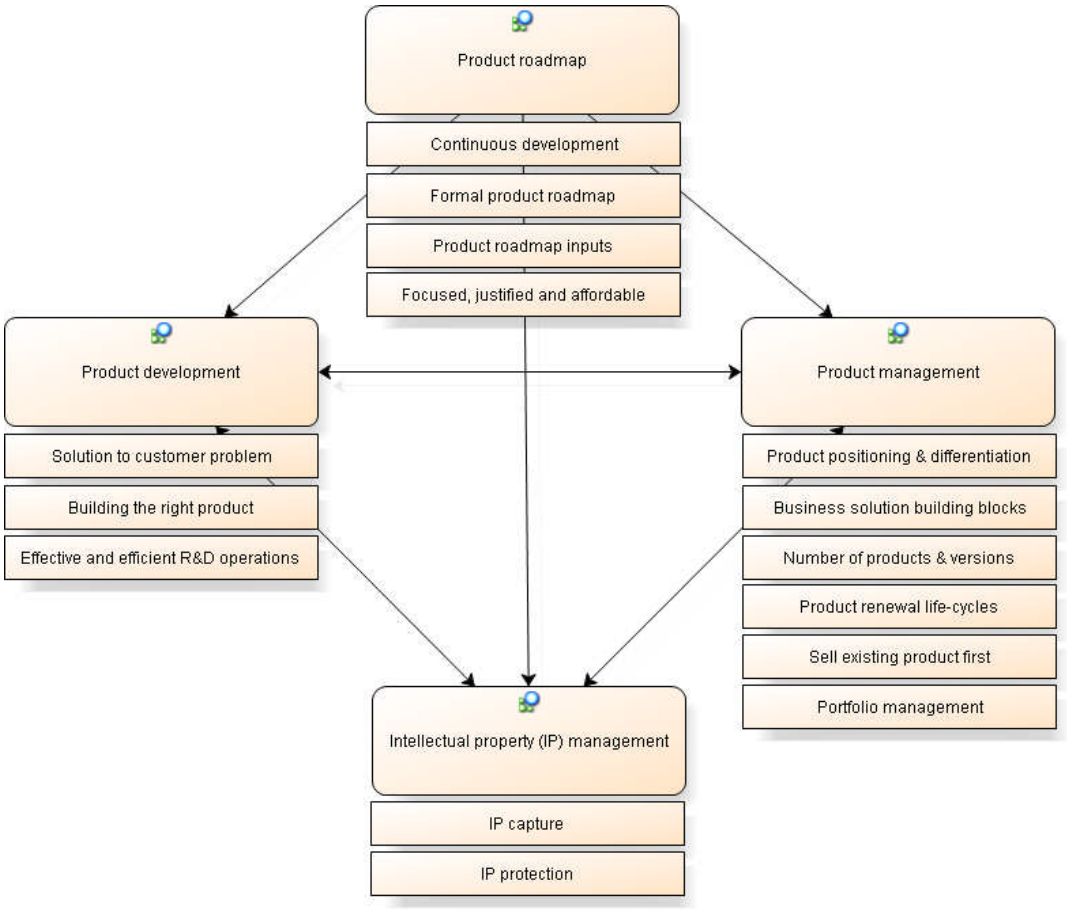
creation of each category is described, the connections between higher-order concepts outlined and implications for running that particular component of a COTS application software SME business offered. The order that the categories are presented loosely mirrors the order that order that the analysis was undertaken and that categories were created.

5.2.1 Product R&D business

Software product is a focal part of any software business. The concepts of building product, managing product, product direction, and IP management are factors that are all concentrated around software product. Rather than being mutually exclusive, these four concepts actually overlap and are closely interrelated. While the creation of product starts with product development activities, product development is also an ongoing activity. However, this further product development will need to be balanced between long product road aspirations and a range of standard product management considerations. IP management provides a level of control and risk management that underpins all these activities. The theoretical construct product R&D business has been created from all these product related concepts. This construct demonstrates how all these factors interact and work together as a component of running a software business. This construct is illustrated in Figure 5c.

The product R&D business is an absolutely fundamental component of the overall software business. Ultimately, having a good quality product has a direct influence on a whether a software business can achieve strong revenue streams and satisfied customers. It is likely to be integral to a vendor’s long-term success or failure. All other functional areas of the vendor are related to it some way, shape or form.

Figure 5c: Product R&D Business



The product R&D business is often a very high profile function within the software vendor. It can be a dynamic and exciting place work in; and offer the opportunity to work with innovative and highly intelligent software professionals. It generally receives solid strong management support and company funding. However, it is sometimes an area that does not perform particularly well. A focused approach to running the product R&D business is an imperative to maximise its efficiency and effectiveness.

Running a software R&D business will involve understanding how software can solve a customer business problem; managing the complex nature that software product can take; determining the degree of fit between customer requirements and product capabilities; dealing with a high frequency of product renewal; managing the intricacies of positioning and differentiating product in a crowded market; the product management and

administration of multiple products and versions; developing a robust product roadmap; sourcing roadmap funding and determining ROI; and capturing and protecting competitive important product IP.

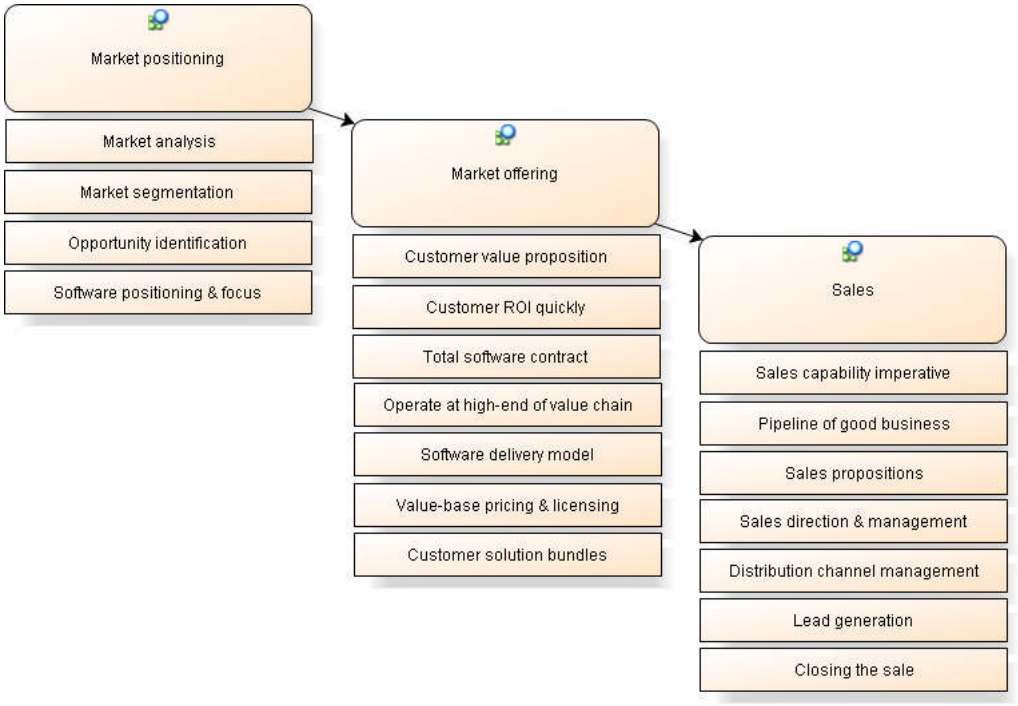
5.2.2 Marketing and sales business

Making sales is a fundamental component of any software business. As identified in the literature review and supported by informant comments, one of the biggest reasons why software vendors fail is because of a lack of sales. There are many cases of software vendors who have quality products, but are unsuccessful as they do not manage this part of the business. However, the attainment of a sale is essentially a final outcome. A whole bunch of prior activities will have laid the groundwork to make this achievable. The functional concepts of market positioning, market offering and sales, combine together to provide an interconnected and sequential set of activities that enable a software business to make sales successfully. The theoretical construct marketing and sales business has been created from the fusing together these concepts together. This outlines the specific marketing and sales aspects of running a COTS application software SME business. This construct is illustrated in Figure 5d.

Marketing and sales is all about being seen in a crowded and messy software market. Running a software marketing and sales business will involve identifying opportunities and segmenting the market. Rigour in the market assessing potential market opportunities will be important. Clarity about the vendor's market position is required. Its brand may also be of importance. A suitable market offering then needs to be developed to serve this target market. This includes developing an attractive customer value proposition, determining a delivery and consumption model, and formulating a viable pricing model. Finally, with an attractive market offering and clear target market the vendor's sales function will have a strong platform from which to make sales. Actually making sales

includes ensuring an active sales pipeline, recruiting high quality software salespeople, navigating the software sales cycle, and developing effective sales channels.

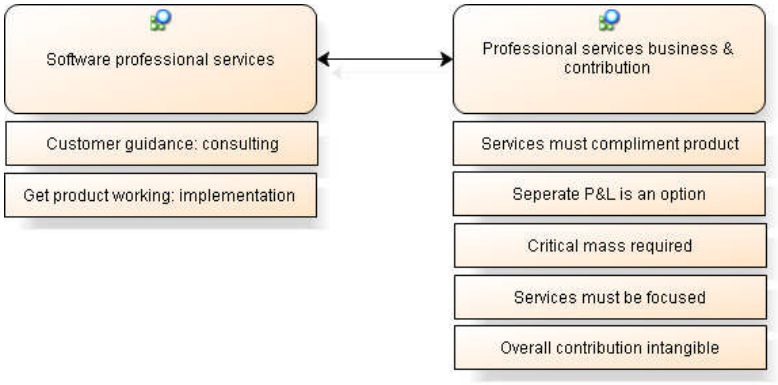
Figure 5d: Sales and Marketing Business



5.2.3 Professional services business

Professional services can form a key part in turning a software product into a value-adding software solution for a customer. Software services exist in various shapes and sizes. The most common are professional services to implement a product, although there is also a range of other business and technologies services that a vendor can provide. However, how exactly a professional services operation fits into an overall software business and the exact contribution it makes, require scrutiny. The theoretical construct professional services business has been created from the concepts of software professional services and professional services business and contribution. This construct demonstrates how these two factors balance each other as part of running a software business. This construct is illustrated in Figure 5e.

Figure 5e: Professional Services Business



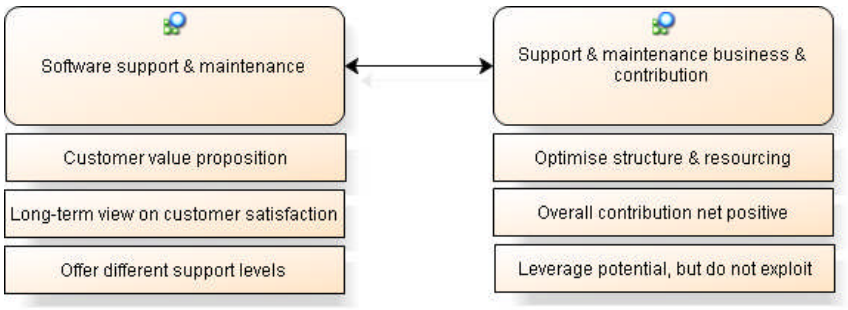
Designing and structuring a professional services business so that the strategic contribution it makes is maximised is preferable. The relative importance of a services practice can vary considerably in a software vendor's overall business model. Various different types of professional services can create customer and business value in different ways. However, providing software services that help get a product working and enable customer value are generally a focus. The key is the linkage between a vendor's professional services and their software product. If product and services fit closely together, synergies across the overall business can be realised. A professional services business will ideally support product sales and drive overall business revenues and profitability. However, building and running an effective professional services business will require investment and the right expertise.

5.2.4 Support and maintenance business

Support and maintenance is an important part of the overall software business. It helps assure continuity of business operations for customers using software product and it can be a highly profitable area of business for software vendors. While the provision of ongoing support and maintenance to customers is generally the norm, the customer value of this is often questionable. What support and maintenance functions to provide and how these fit into an overall software business require needs to be determined. The theoretical construct support and maintenance business has been created from the concepts of

software support and maintenance and support and maintenance business and contribution. This construct demonstrates how these two factors balance each other as part of running a software business. This construct is illustrated in Figure 5f.

Figure 5f: Support and Maintenance Business



A support and maintenance business does have the potential to be a win-win proposition for both software customer and software vendors. However, the long-term commercial viability of a support and maintenance business will need to be linked to its value proposition. How the business is structured and run is likely to be a critical decision. Vendors may be able to offer customer support and maintenance levels that are tailored specifically to their specific needs. Keeping a software product operational and providing a customer an insurance policy in the way they want it can be commercially attractive. Running a support and maintenance business will involve ensuring that there is clear linkage between how support and maintenance activities are provided and the customer value that they generate. While high profit levels area achievable, exploitation of customers is dangerous. In particular, becoming reliant on revenues that come from activities that do not provide customers is likely to be detrimental in the long-term. Customers are likely to leave.

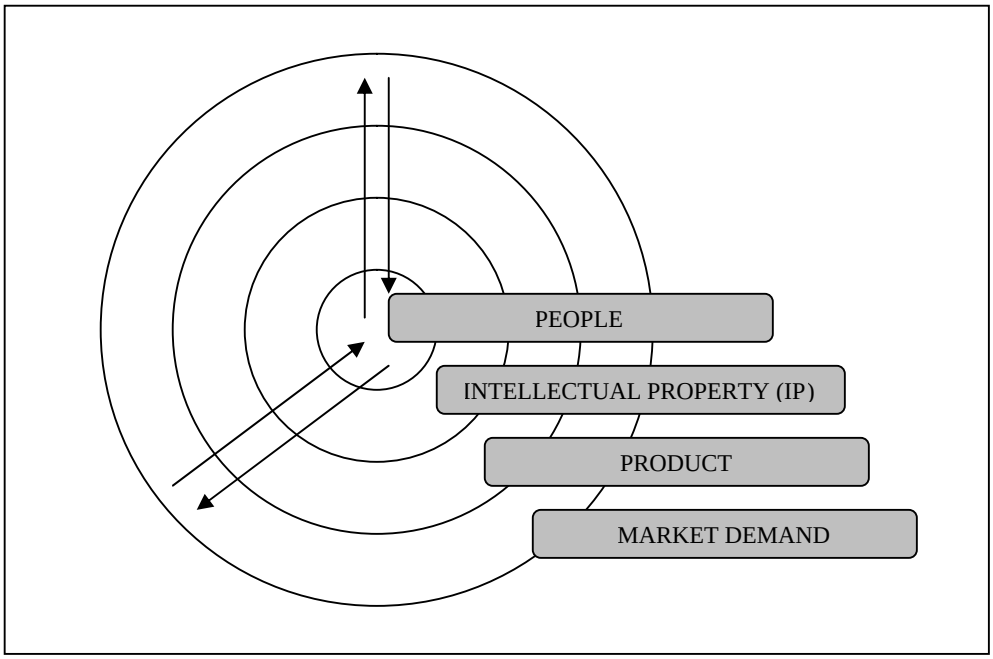
5.2.5 Software business core

A commercially viable software product that solves business problems is at the centre of a vendor software business. After further consideration of the higher-order concept software business core, the significance of this as a key concept involved in running a

COTS application software business SME continues to increase. It appears to pull together a whole range of other factors that have been informants have referred to when conceptualising how they run their business. Therefore, the theoretical construct software business core has been created directly from this individual concept. Figure 5g depicts a model representing s this software business core construct.

The software business core has been identified as having four constituent parts that work together to form the heart of a vendor business. These comprise a mix of product, IP, people and market demand. The challenge for a vendor is to define and link together these attributes together in way that optimises the overall strength and potential of the business. In simple terms, business market drives demand, product is linked to IP and IP originates from people.

Figure 5g: Software Business Core



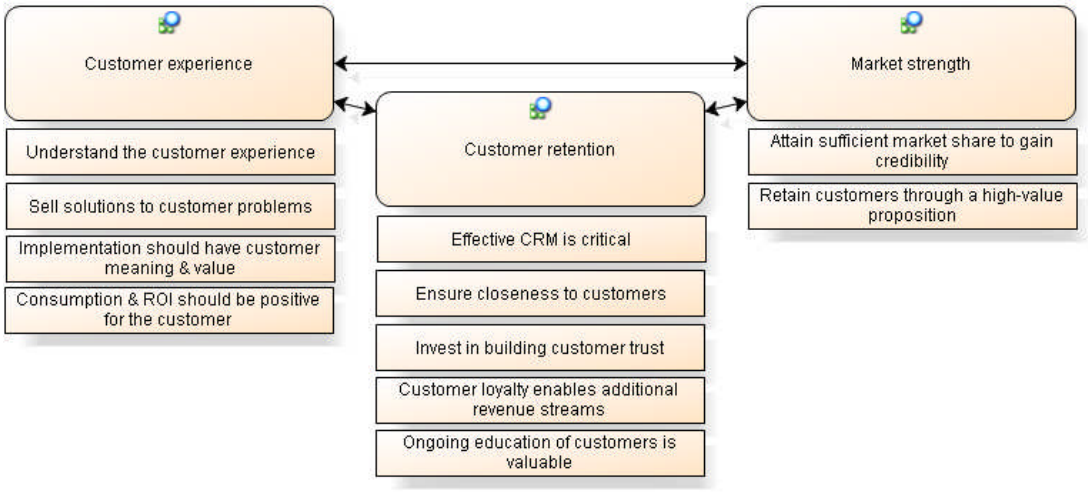
The four components of the software business core need to work together. The tangible centre of a software business will be a physical product. This needs to be of high quality and have sufficient market demand. The IP at the core of a software business needs to be understood and explicitly captured. IP can be a key strategic asset for a vendor, but IP

will need to be well managed and continually developed to stay ahead. The third part of a software business core is the vendor's people. People hold the real IP and know how that allows a vendor to run smoothly. Harnessing and managing the people within a vendor organisation, and the tangible and intangible knowledge and skills they possess, is an absolutely fundamental component that sits right at the heart of a software vendor business. The software business assets of product, IP and people cannot just exist together, the contextualisation and positioning of these within a specific market sector is an imperative. Ensure that there is a good and appropriate fit between these internal aspects of their business and having sufficient and attainable market demand makes the business a viable proposition.

5.2.6 Market centricity

Understanding how a software business is perceived by its customers and by the overall market, will provide a critical insight into what it is doing well and what it is doing badly. Operating with a market centric orientation is likely to be a necessity if a vendor wants to attain a competitively advantageous position. At the forefront will be ensuring that all elements of a customer experience with a vendor are positive. In the background, this be influenced by proactive customer retention activities and underpinned by the strength that the vendor has it its market. The theoretical construct market centricity has been created from the concepts of customer experience, customer retention, and market strength. This construct combines together to provide an interconnected set of activities that give a software business focus in its market. This construct is illustrated in Figure 5f.

Figure 5h: Market Centricity



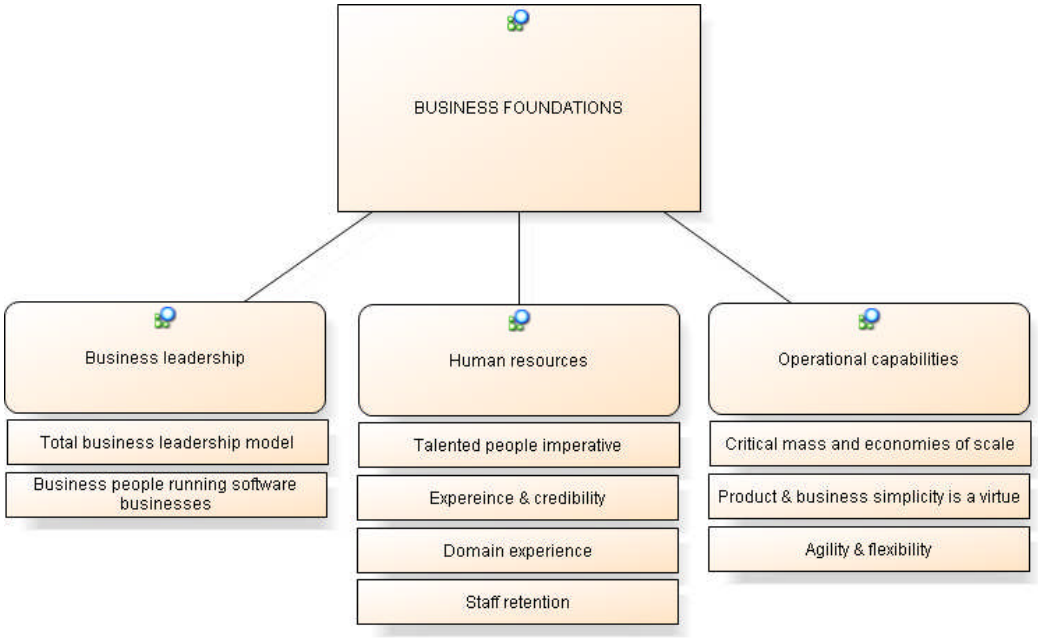
Running a software business that has market centricity will involve understanding a customer's lifetime perspective of dealing with and consuming a vendor's software product and services. This will include the purchase, implementation and consumption of software through to it delivering value-adding ROI. The objective will be to developing strong and deep relationships with customers. To achieve this, CRM activities will in involve managing customer proximity, providing education, gaining trust and building loyalty. A vendor that manages to achieve these attributes is likely to be able to attract new sales and grow its market share. Developing market strength and power can then lead to a virtuous circle of increasing competitive advantage.

5.2.7 Business foundations

A successful software business will have solid foundations that underpin its prime functional activities. Whilst these business foundations are not always explicitly visible and/or tangible, in the context of managing a software business, they are no less important than any other aspects of a software business. The theoretical construct business foundations has been created from the concepts of business leadership, human resources and operational capabilities. This construct combines a set of activities that give

a software vendor a strong platform to operate its business. This construct is illustrated in Figure 5i.

Figure 5i: Business Foundations



While the three business foundations identified may not provide a definitive list, they are the prevalent factors that emerged from the research interviews, In particular the topics of 'business leadership' and 'human resources' were raised over and over again by informants. Each of these foundations will be addressed proactively and formally as part of overall business management activities. Developing a software vendor's business foundations will involve ensuring that vendors are led by capable and experienced business managers; that vendor staff have domain expertise; and that operational simplicity, agility and flexibility exists where possible.

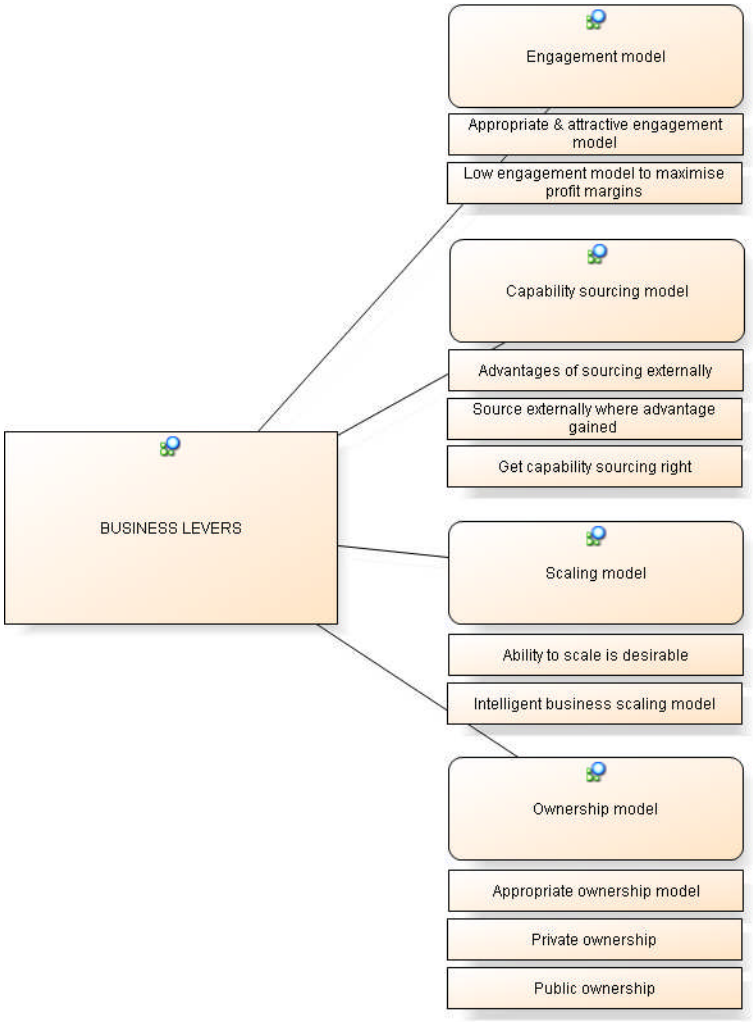
5.2.8 Business levers

A software vendor has a number of tools available that it can use to leverage its business. The level of contact they have with their customers; whether they outsource certain business functions; the ability to scale their operations; and how the ownership model of the business is structured are identified as variables in this respect. While these factors

generally do not directly affect the primary activities of a software vendor, they can have a considerable effect on the successful of the business as a whole. Conceptually, all these factors have been clustered together in the overall software model as business levers that a vendor can arrange in different ways. The theoretical construct business foundations has been created from the concepts of engagement model, capability sourcing model, scaling model and ownership model. This construct provides a software vendor a set of levers that give it the ability to increase or decrease the size and focus of its business. This construct is illustrated in Figure 5j.

There is significant flexibility and variation in how a software business configures its business levers. There is a range of opportunities available to leverage its business. A low engagement model can enable business focus and offer high profit margins. A vendor's capability sourcing model can enable growth and allow risk to be managed. The ability to scale is important and a vendor with established product, processes, people and customers is likely to be able to extend and duplicate these. Deciding whether a vendor is privately or publicly owned can have significant influence on how it is run and performs. However, as with all business decisions there will be pros and cons of each choice. Understanding consequences and having a plan for managing alternative scenarios will also be important. Understanding the aggregate profile of all the business levers is worth consideration. While the four business levers identified may not provide a definitive list, they are the prevalent factors that emerged from the research interviews. In particular, the topics of capability sourcing, scalability and business ownership were raised repeatedly by informants.

Figure 5j: Business Levers



5.2.9 Commercial governance

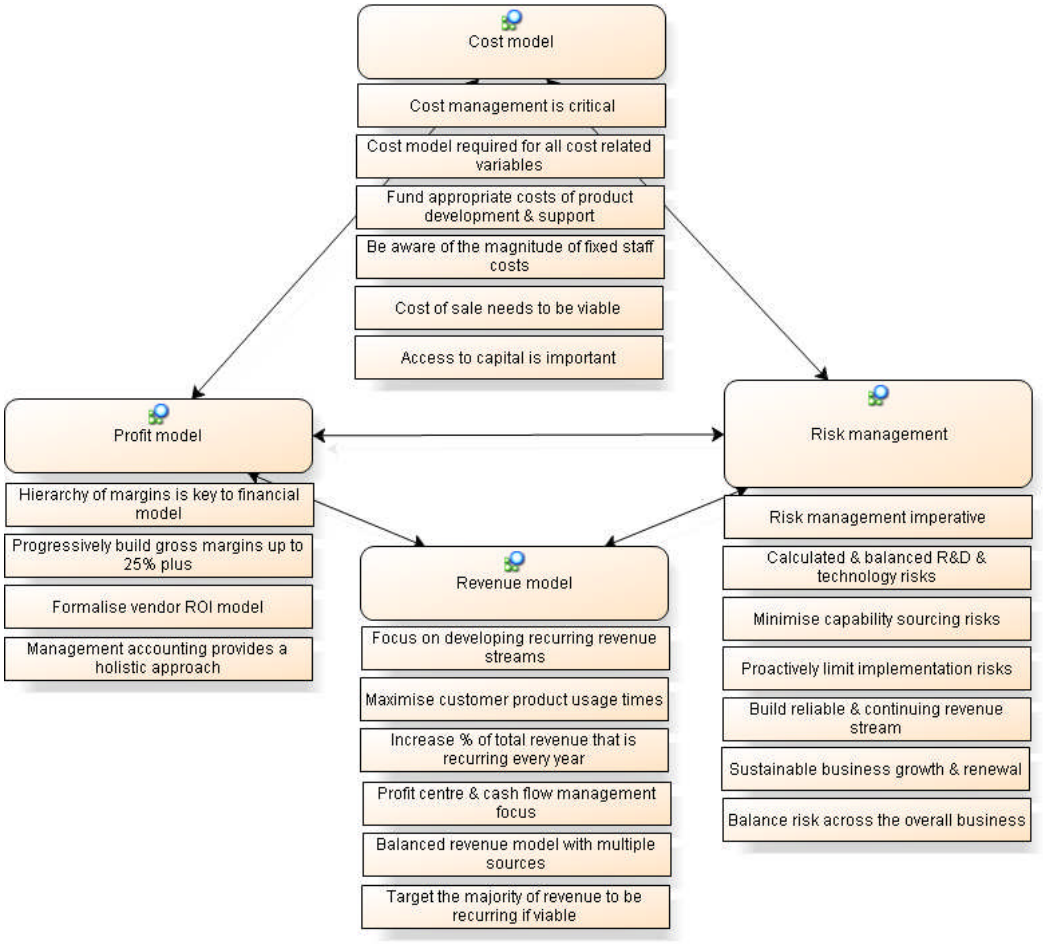
Having a robust model and governance mechanism that bridges the overall finances and risks of running a software business as a whole is critical. Software businesses involve developing and managing a range of diverse activities. The inherent dynamism of the software sector means that the spread of a software business' activities, costs, revenues, risks and profits, may be fluid from year to year. If a vendor does not have a way of pulling together all these factors into a single holistic picture, its ability to operate its business successfully may be severely restricted. The theoretical construct commercial governance has been created from the concepts of cost model, revenue model, profit models and risk management. This construct brings together the main commercial factors

that sit across an overall software business. This provides a commercial governance framework that can be used as a controlling mechanism for running that aspect of an overall software business. This construct is illustrated in Figure 5k. It was noted that some of the smaller vendors discussed by informants did not have well-defined commercial models or forecasts. In contrast, the medium and larger firms appeared to display a higher level of maturity in their approach to commercial planning and management.

Configuring the commercial governance component of a software business will involve a range of factors and considerations. A balanced cost model will consolidate costs revenue multiple component business areas. The main operational costs include staff, cost of sales and product development. There are a number of business variables that can affect a vendor's cost model, while timely injections of capital can be required for a vendor to stay competitive. A balanced revenue model is attractive, as are recurring revenues. The irregular frequency of new sales can lead to revenue peaks and troughs that need to be managed. Hence, securing recurring revenue streams from services and support is often a counter-balancing activity.

Appropriately matching costs and revenues often does not receive enough attention. An effective model and mechanism for balancing overall revenues with overall costs is crucial. Effective cash management is imperative. A target gross profit margin will need to be determined. While investment levels in R&D can have disproportionate effective on a vendor's finances, support and maintenance are generally the most profitable area. An overall gross profit margin in the range of 20%-30% ought to be attainable for the business as a whole. A formal and holistic approach to risk management is an imperative. A vendor's risk management model will be used to assist software businesses in developing a holistic and balanced approach to managing their commercial business risks.

Figure 5k: Model for Commercial Governance



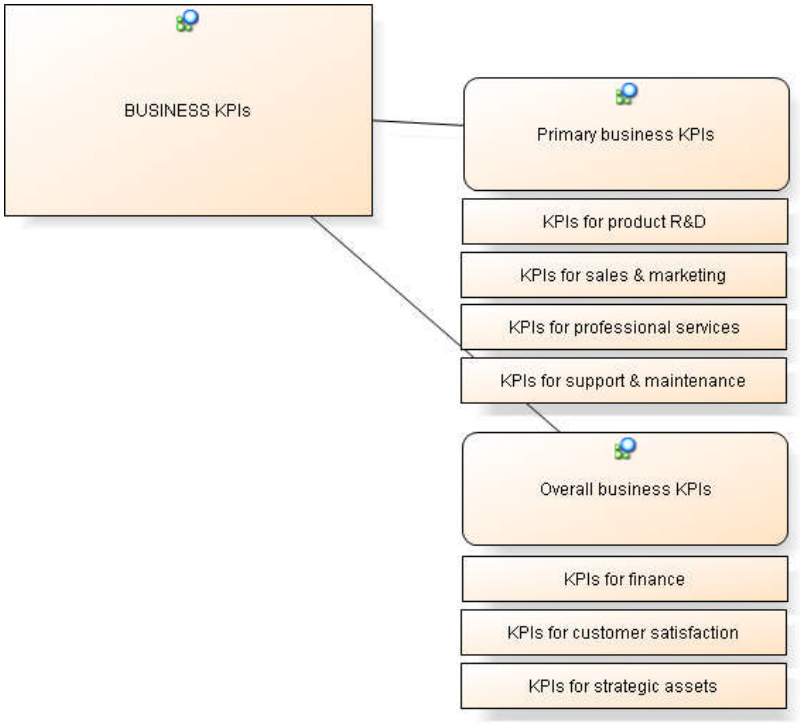
5.2.10 Business KPIs

Clearly identifying and measuring key business KPIs is a part of clarifying the strength of a software business. A set of KPIs can provide a powerful reference point for effectively managing a vendor business going forward. Informants have provided a rich set of metrics for how a software business can set up and measure the performance of their business functions. Individual business metrics and performance indicators for each of the primary functional areas of a software business have been outlined above. At an overall business level there are also a range of other important business metrics that relate to the longer-term future of the business that have been identified. The theoretical construct business KPIs has been created from the concepts of KPIs for primary business and KPIs for the overall business. This construct provides an interconnected set of activities that

provide an extensive set of metrics for a software business to measure its strength and performance. This construct is illustrated in Figure 5l.

While a large set of software business KPIs have been identified, those deemed the most relevant metrics for a software business are highlighted. Product R&D metrics encompass: customer product feedback, product quality, product production; and the business contribution of R&D. Sales metrics include revenue targets, sales performance against quota, new and repeat business levels, brand value, the sales pipeline, sales conversion rates, the cost of sales, and the ROI from each customer. Services metrics include utilisation level, profit margin on billable rates, and engagement successfulness. Support and maintenance metrics include support and maintenance fee and margin, revenue as a per cent of total revenues, support capacity and resourcing levels, and performance against customer SLAs. There are a wide range of very important financial metrics that cover various aspects of cost, revenue and profit. Customer satisfaction metrics are of increasing importance; however, there are challenges measuring these, as they are largely qualitative and subjective. A set of strategic asset metrics include innovative R&D capability strength, the robustness of the software business core, customer value proposition strength, leadership strength, the quality of people, the level of domain expertise, customer relationships strength, the market power, distribution channels strength; the level of operational effectiveness and efficiency, and the business risk profile.

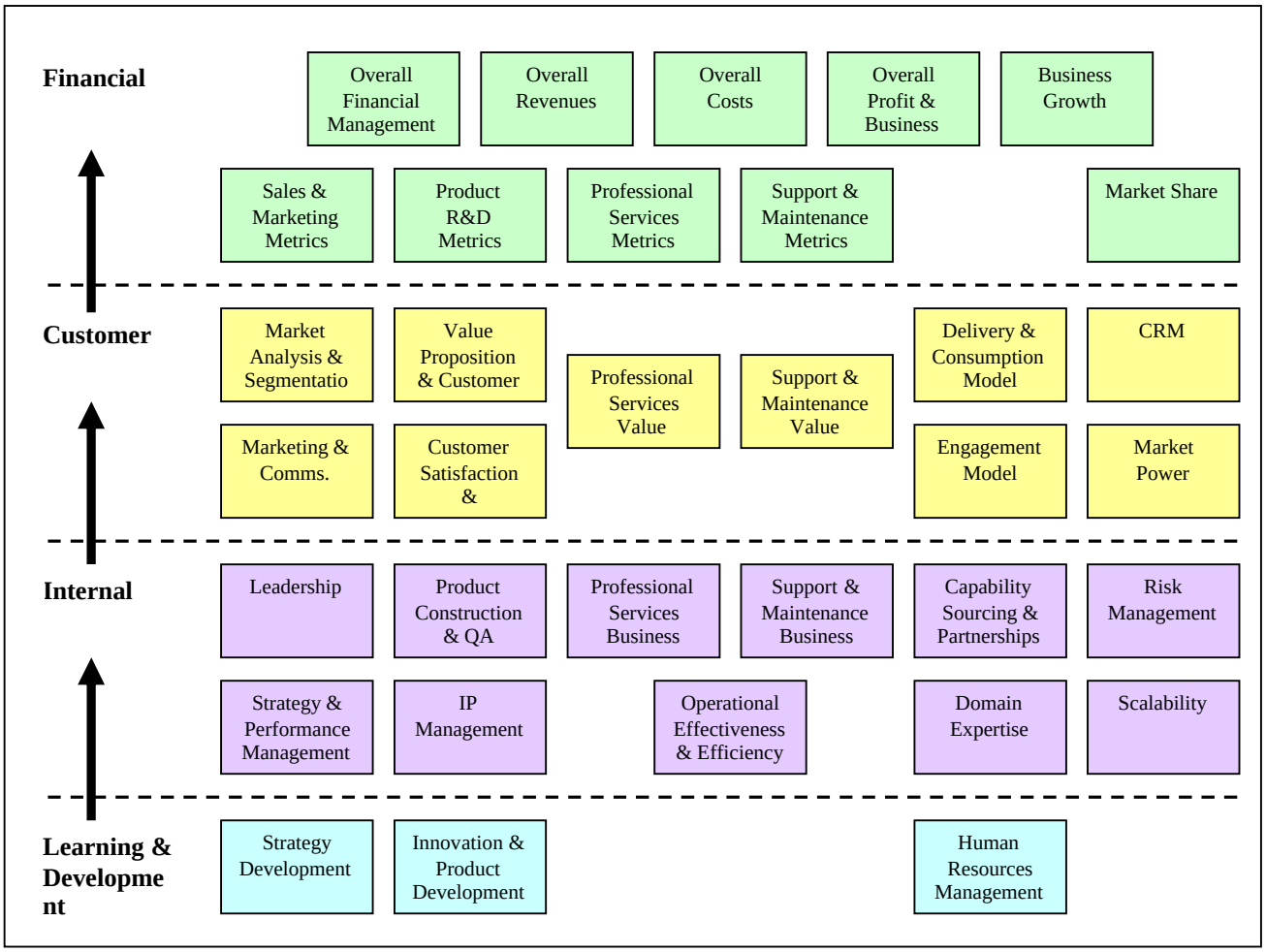
Figure 5l: Model for Business KPIs



5.2.11 Total business scorecard

A comprehensive performance framework of software business specific metrics can provide a software vendor with a powerful and holistic perspective of a vendor's long-term strength and future performance potential. After further consideration of the higher-order concept total business scorecard, the ability of this concept to pull together and integrate a whole range of disparate elements of a software business is considerable. The theoretical construct total business scorecard has therefore been created directly from this individual concept. The KPIs from the business KPIs theoretical construct have been mapped onto a single canvas and logically connected using the Norton and Kaplan (2004) scorecard paradigm as a point of reference. Figure 5m below illustrates the resulting total business scorecard.

Figure 5m: Total Business Scorecard



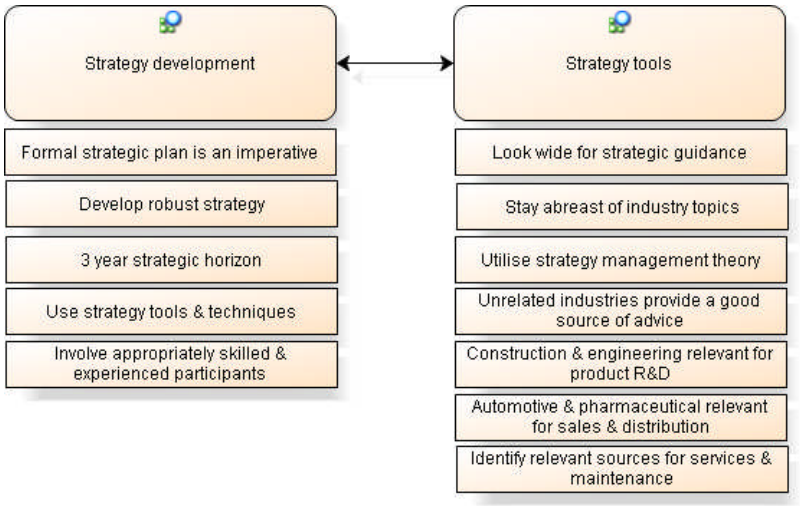
The classic balanced scored based upon Kaplan and Norton (2004) generic model groups metrics into financial, customers, internal, and learning and development perspectives. Where a business attribute, as opposed to a previous explicit KPI is included in the total business scorecard, this will also require explicit qualification and/or quantification. The result is a powerful holistic business performance management tool that is grounded in COTS application software SME business specific concepts and structured in an established strategic management structural framework.

5.2.12 Strategic planning

Investing time and energy in formulating a detailed strategy is essential. A vendor's long-term success is likely to be linked to how seriously and rigorously they approach strategic planning. Due to the lack of a consolidated source of guidance on how to run a software

business, being resourceful in how advice and relevant frameworks are sourced for the development of their strategies can be critical to achieving a clear and focused strategy. The theoretical construct strategic planning has been created from the concepts of strategy development and strategy tools. This construct provides a set of processes and resources associated with formulating a strategic plan for a software vendor. This construct is illustrated in Figure 5n.

Figure 5n: Strategic Planning



Strategic planning for a software business will involve a range of factors and considerations. Generic strategy management theory is very relevant. Although, there are a number of specific areas involved in running a software business have been identified, where business managers would benefit from improved strategic guidance. These include product quality, marketing, sales and distribution. However, a number of alternative sources of guidance have been identified for these areas. Engineering, construction, pharmaceutical and automotive industries are suggested as industries that software businesses could learn from.

5.2.13 Vendor strategy

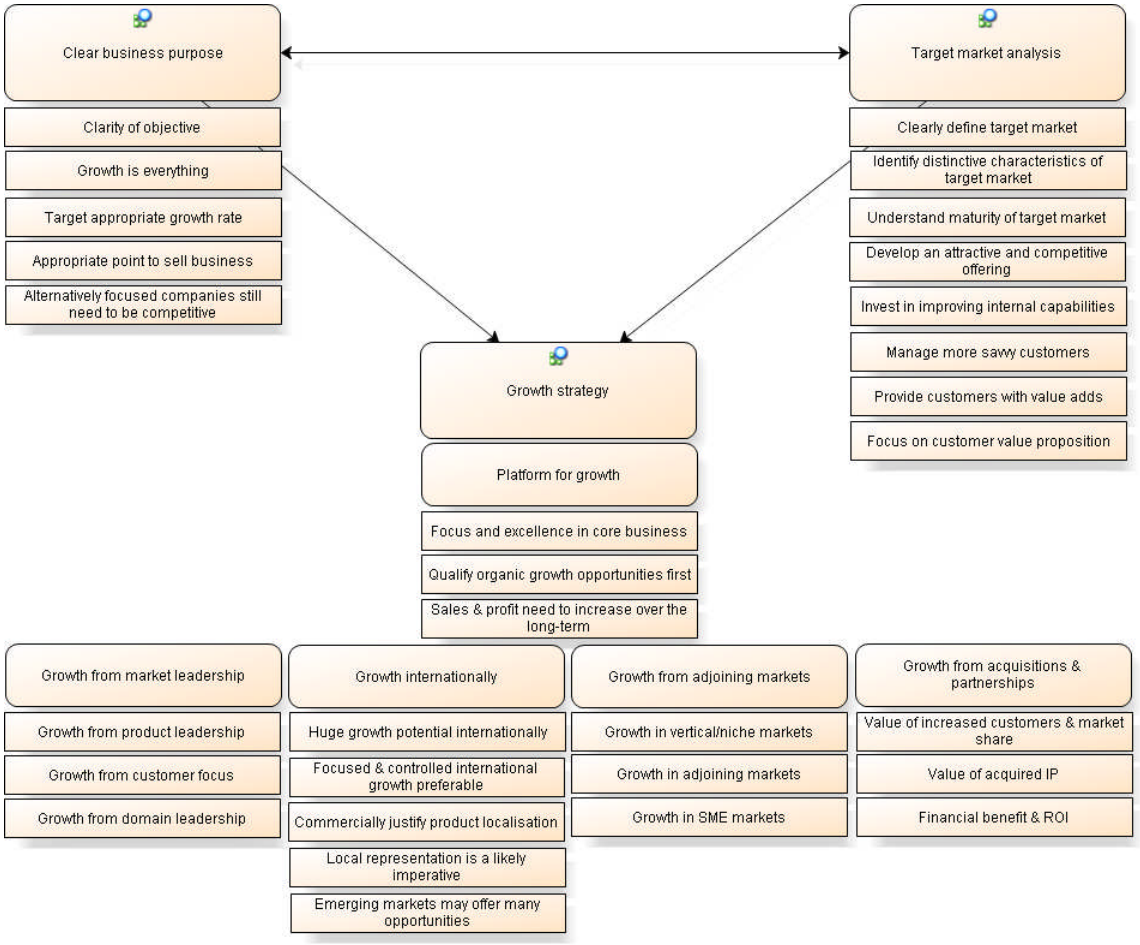
A vendor's strategy will include a number of key attributes. Knowing what business a vendor is in and having clearly defined business goals is an imperative. Clearly defined

business purposes and associated business objectives are necessary. A vendor's strategy will need to be appropriate and realistic in relation to the forces at work in that given market. A comprehensive strategic analysis of a vendor's competitive landscape is an absolute imperative as part of the process of formulating a vendor strategy. A vendor's growth strategy is often likely to be its most important direction and focus. Managing growth is likely to be a prime activity for software business mangers. The theoretical construct vendor strategy has been created from the concepts of clear business purpose, target market analysis and growth strategy. This construct combines an interconnected set of concepts, processes and options that in aggregate form the base content for a vendor's strategy. This construct is illustrated in Figure 5o.

Fully understand and defining a vendor's target market is a pre-requisite to formulating a clearly defined business strategy. Software markets have a number of distinctive characteristics. In particular, software is an intricate, intangible and often opaque product. While in addition, the COTS application software market is a dynamic and evolving market. Undertaking a detailed target market analysis is a means of building a picture of the landscape that a vendor intends to compete. Undertaking a target market analysis for a software business will need to consider a large range of factors. Software is an intricate, intangible and often opaque product. Ensuring 100% clarity about exactly what a product does will be important. Market segments are not mutually exclusive segments. The consolidation of software vendors is very apparent. The software market is maturing, but it is not mature. There is still immaturity and volatility in large parts of the software industry. Software products are stabilising, maturing and becoming more sophisticated. Technology developments continue to fuel the software market dynamism. Market imperfections exist and the market could be improved for customers in numerous ways. Product development and software engineering has improved considerably and vendor

sales and distribution is starting to mature. Software customers are now more savvy and knowledgeable about software and the business problems they wish to solve.

Figure 5o: Vendor Strategy



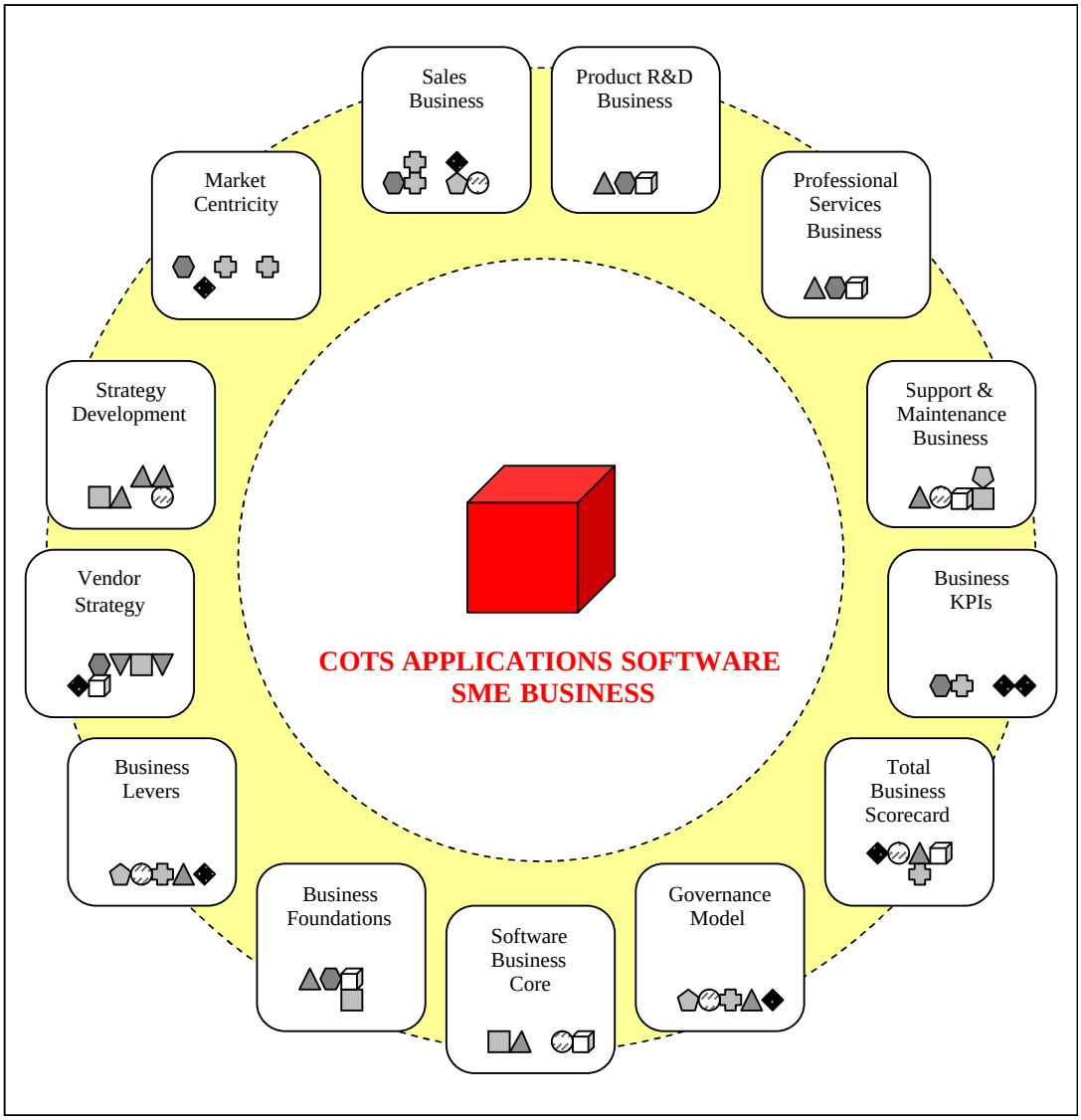
Increases in sales, revenues and profitability are key components associated with growth. There are a number of alternatives open to vendors who wish to pursue an objective of growth. Four broad business approaches for enabling growth include following a traditional strategy of market leadership, pursing explosive international growth opportunities, moving into similar and adjoining markets, and embracing much-hyped M&As. However, informants advised that it is important for a software vendor to stick to what they are good at and focus on their core business. Concentrating on being an IP-based product business that serves a clearly defined target market can provide a core model that underpins any associated growth strategy.

The determination of the most appropriate growth strategy will require a range of specific factors to be carefully considered. Sustainable software business growth requires focus. Organic growth still offers many opportunities to expand. Continual increases in sales are a crucial part of staying in business. A competitive product can provide competitive advantage and attract new sales. Serving customers well can result in high satisfaction levels, which drive additional sales. Dominating and leading a domain segment is a robust strategy for growth. International markets offer the potential of reaching a huge additional set of customers. Product localisation may be required for a different region. A local presence in a local region is often required to gain traction in that location. Many vertical and niche markets still have the potential high levels of new sales. Software products can be taken into adjoining markets with similar business requirements. SME markets are receiving a great deal of focus from software vendors. M&A continues to be a common way for vendors to increase their size, customer numbers and market share.

5.3 Summary

Chapter 5 has documented the intermediate data analysis that has been undertaken. Using a two-step process, the 93 base-level codes from the data interpretation stage, have been consolidated into a set of thirteen theoretical constructs. First, the 93 base-level codes were reduced into a set of 34 higher-order concepts. Second, connections between higher-order concepts were established and the higher-order concepts were fused into a small set of theoretical constructs (grounded theory open coding categories). Figure 5p illustrates the resulting thirteen theoretical constructs that the analysis has determined are relevant to running a COTS application software SME business.

Figure 5p: Theoretical Constructs Relevant to Running a Software Business



In aggregate, these theoretical constructs provide a definitive set of categories that cover all aspects of running a COTS application software SME business. However, these constructs are limited by just being a set of standalone concepts. They do not provide an integrated and holistic model for running a software business. Chapter 6 describes the final analysis work that was undertaken to achieve this goal.

6 CONSTRUCTING AN OVERALL THEORY

Having developed a set of thirteen theoretical constructs that helped explain the various aspects of running a software business, the next stage of the analysis was to connect all these together into a single unified model for running a COTS application software SME business. This objective was achieved by following a two-step process. First, points where the thirteen theoretical constructs intersect were identified. A small set of resulting 'axes' were classified. Second, the axes were connected to provide a single model that integrates together all the theoretical aspects of the research findings. The structure of Chapter 6 mirrors this two-step process. In addition, a final section outlines the consequential overall model for running a COTS application software SME business.

6.1 Axial Coding

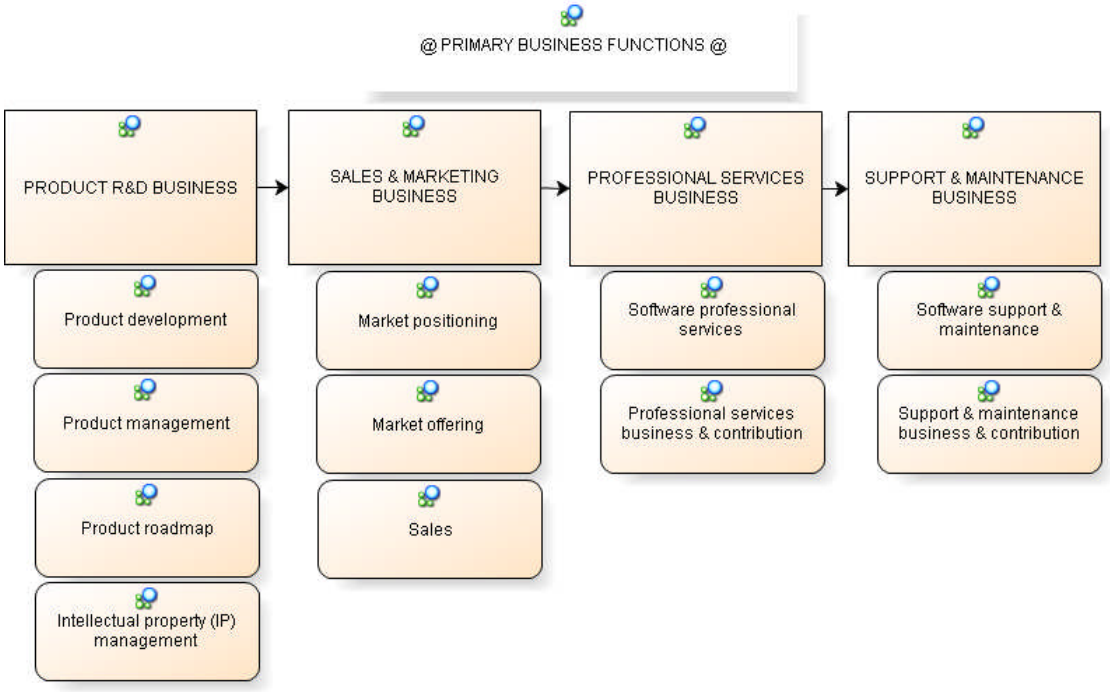
The axes created as a result of the grounded theory axial coding are outlined in this section. The creation of relationships between theoretical constructs, that form the basis of the axes, has added clarifying power and density to emerging theories.

6.1.1 Axis A: Primary business functions

The first axis created connects the theoretical constructs product R&D business, marketing and sales business, professional services business, and support and maintenance business. All of these components can be characterised as being functional activities that form a part of running a software business' base operations. The name 'primary business functions' has been chosen for this theoretical axis. Following a chronological journey through a vendor's primary businesses functions, first, software product is built, existing product managed, the direction product is clarified and any competitive IP protected. Second, a vendor's market is identified, a software market offering determined, and sales attained. Third, various software services may be necessary to get the product working for a customer. Fourth, some support and maintenance is likely to be required to keep the product running. In simplified form, the

product R&D business will drive the marketing and sales business, this in turn will drive a professional services business, which will finally drive the support and maintenance business. In addition, that is also a reverse flow, professional services can contribute to sales, and marketing provides input into product R&D. Figure 6a graphically shows the sequential linkage between four primary business functions of a COTS application software vendor SME business.

Figure 6a: Axis A: Primary Business Functions



A key observation of many software vendor businesses is that their primary business functions can appear to operate in silos. Software vendors often structure their organisations into corresponding separate departments for each of the primary business areas. Function managers do not always appreciate their relative context within the vendor's overall business picture. Conversely, sometimes they are simply not motivated to proactively contribute to the greater good of the business whole. Weak or almost non-existent links between business areas can result in a dysfunctional mode of operation in parts of the organisation. This can result in inefficient and ineffective outcomes for its

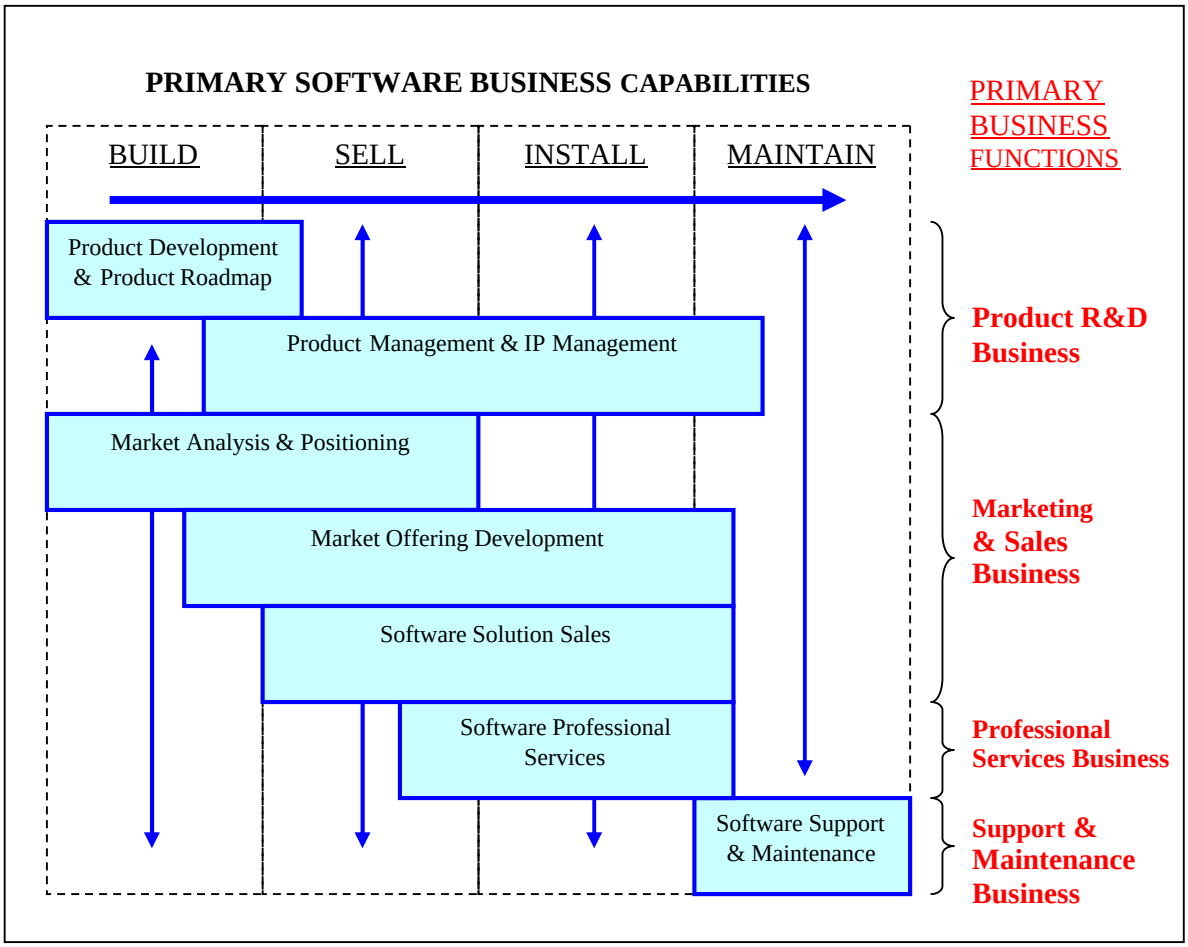
customers; and is value destroying for a software business. Ensuring that all the primary functions of a software business are explicitly connected is a fundamental.

In addition to the sequential linkage of primary business functions, there are also a number of other links between the four components theoretical constructs. The vendor software product life cycle is used as an integrating framework to connect the primary functions of a software business in a more comprehensive manner. This provides context and boundaries to illustrate the primary business functions that are required to operate a business based upon providing software solutions. Essentially software product needs to be ‘built’, ‘sold’, ‘implemented’ and ‘maintained’, and all these activities are related to each other. Loosely, there is a one-to-one mapping between each of these life cycle stages and the prime business functions described above: product R&D maps to ‘build’; sales and marketing maps to ‘sell’; ‘professional services’ maps to ‘implement’; and support and maintenance maps to ‘maintain’. Figure 6b graphically illustrates the main primary business function capabilities a software business will provide and how these are distributed across the vendor product life cycle stages.

On the right hand side of Figure 6b, a grouping of the required business capabilities for each of four primary business functions are displayed. It can be seen that there is not a perfect mapping between product life cycle stage and prime software business functional areas. Whereas some capabilities petty much map one-to-one between product life cycle stage and software business functions, others clearly span multiple functional boundaries. In actuality, there are capabilities that vendors need to provide from more a combination business functional areas. This may appear confusing, but it is this blurring of boundaries that actually helps pinpoint where otherwise functional silos are connected in a larger business landscape. If the vendor activities of product R&D, marketing and sales,

professional services, and support and maintenance all assist a customer in solving a problem, this will have tangible value for a software business.

Figure 6b: Primary Business Functions and the Software Life Cycle Model

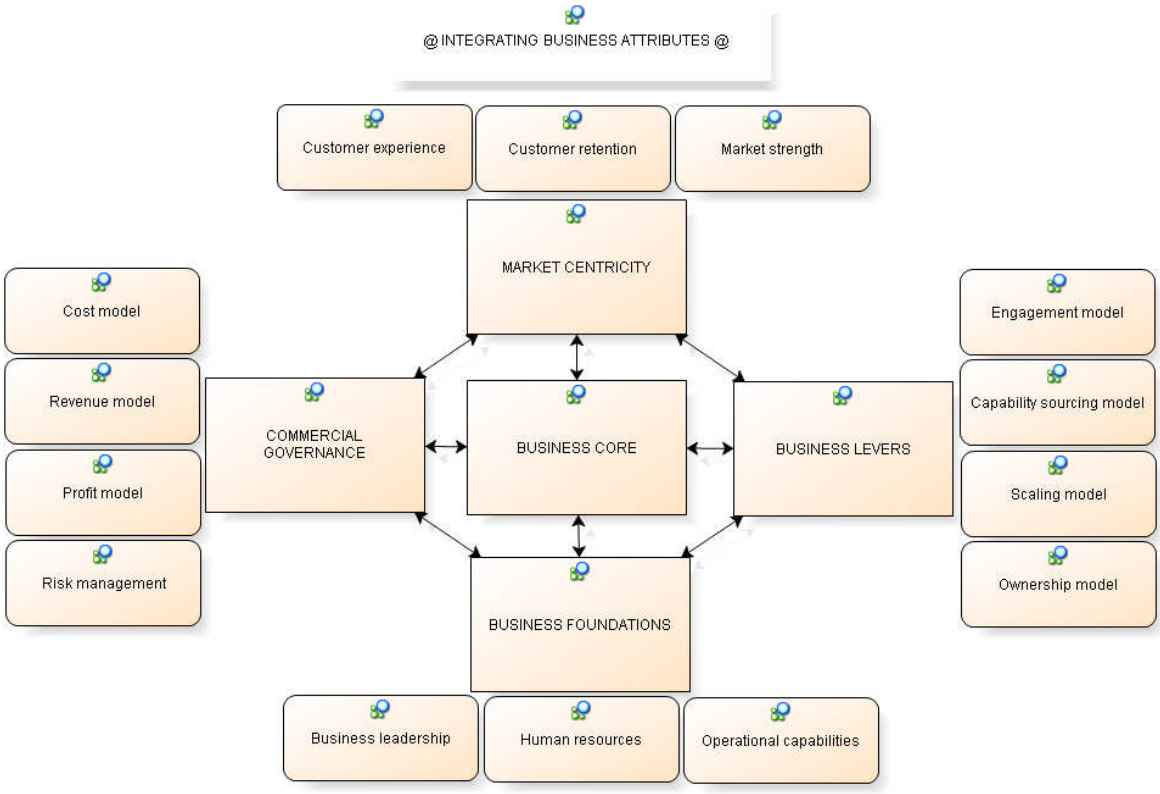


Ensuring that vendor business is structured in a way so that all the primary business functions contribute to the overall software business is likely to be a key objective a software vendor. The primary business functions of the software business can still be distinct, but they will also be part of a larger whole. Ideally, the four theoretical constructs in the model will be seamlessly interconnected and operate as a single cohesive unit. While the primary business functions need to be in place for a COTS application software business to be exist, these functions on their own do not form a software business. A software business needs to be held together by more than these alone.

6.1.2 Axis B: Integrating business attributes

The second axis created connects the theoretical constructs business core, commercial rules, market centricity, business foundations and business levers. While not as visible or prevalent as the primary business functions of a software business, these constructs are of no less importance. The power of this axis comes from its ability to connect and bind together a range of business factors into explicit terms of reference that underpin a COTS application software SME business. The name ‘integrating business attributes’ has been chosen for this theoretical axis. While at first glance, the five theoretical constructs may seem like a diverse set of business components, when integrated they can provide a clear focus for running many aspects of a software business. Figure 6c graphically shows the linkage between five integrating business attributes of a COTS application software vendor SME business.

Figure 6c: Axis B: Integration Business Attributes



The integrating business attributes axis helps solidify a bigger picture, beyond primary business functions, of what a holistic COTS application software SME business is and

what it should be about. The software business core pinpoints exactly what business a software vendor is actually in. This, as implied by its title, sits right at the heart of the axis and everything else is connected to it. However, as a software business is a business, it will require an explicit commercial model and set of governing commercial rules from which to operate within. It is important that whatever commercial governance model a vendor come up with, that this is grounded an inherently linked to the business core. If it this link is missing, there is likely to be a mismatch between the business that the vendor thinks it is in and its commercial realities. The business core and commercial governance model will be continually tuned and refined by the need for market centricity. While in the opposite direction, the need for commercial sensibility will provide balance to over serving customers with a non-viable proposition. Business foundations provide a generic platform of effective and efficient capabilities for running a healthy software business. The level of investment in developing these business foundations will also be appropriated by the parameters of the software business core, commercial governance model and the need to have market centricity. The fifth construct in the axis provides a set of business levers can be throttled back and forth dependent on how vigorously the business is to be run. The development and use of these business levers is also inherently linked to the other four constructs in the integrating business attributes axis.

6.1.3 Axis C: Strategy

The third axis created connects the theoretical constructs strategic planning and vendor strategy. While these constructs make a clear distinction between those activities involved in formulating strategy and the actual resultant strategy, there is likely to be some linkage link between the time and energy invested in strategy development and the robustness of a software business' approach for achieving long-term success. Looking beyond the short-medium term aspects of running a software business, there is a whole range of business directions a vendor can pursue and a number of specific strategic business decisions that will need to made. The name 'strategy' has been chosen for this theoretical

axis. Figure 6d illustrates the strategy axis for a COTS application software SME business.

Figure 6d: Axis C: Strategy

The importance of an undertaking strategic thinking and planning is emphasised. The strategy axis connects together all the factors that are involved in running a software business over a long period. This provides a framework for formulating and determining strategy specifically for a COTS application software SME business. Strategic guidance is available beyond the immediate sphere of the software industry and vendors who are resourceful in the search for this are likely to benefit. Having an overall business rationale and purpose is flagged. Clarity around what business a vendor is in and having clearly defined business objectives are an imperative. The software market is complex and multifaceted in nature. Various subtleties of the software market require consideration as a prerequisite to defining a strategic focus. Growth is likely to be a necessity for long-term survival. Different approaches to achieving growth include growth from market leadership, growth internationally, growth in similar markets and growth by M&A. While

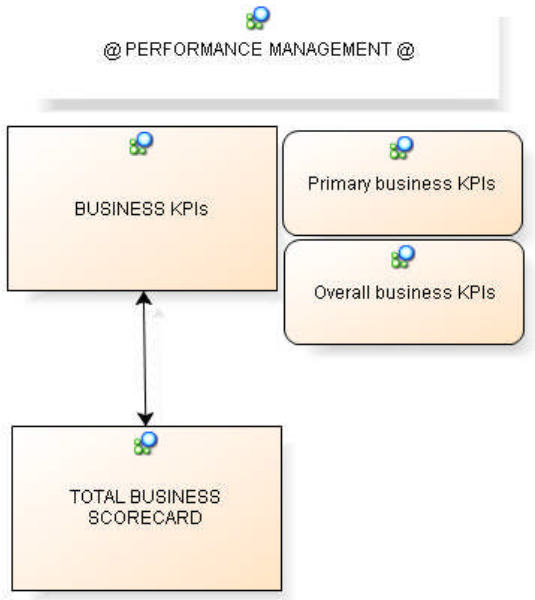
there are numerous areas that growth can be pursued, a focused and controlled is likely to be most sustainable.

6.1.4 Axis D: Performance management

The fourth axis created connects the theoretical constructs business KPIs and a total business scorecard. Identifying and measuring all KPIs of a software business plays a key part in providing a vendor the necessary information it needs to manage its business. However, just having a set of isolated KPIs for different parts of the business will only provide limit control. Ideally, a vendor will have an overriding management framework that connects all the KPIs together and provides a tool for tracking performance and supporting key management decisions.

The name 'performance management' has been chosen for this theoretical axis. Figure 6e illustrates the performance management axis for a COTS application software SME business.

Figure 6e: Axis D: Performance Management



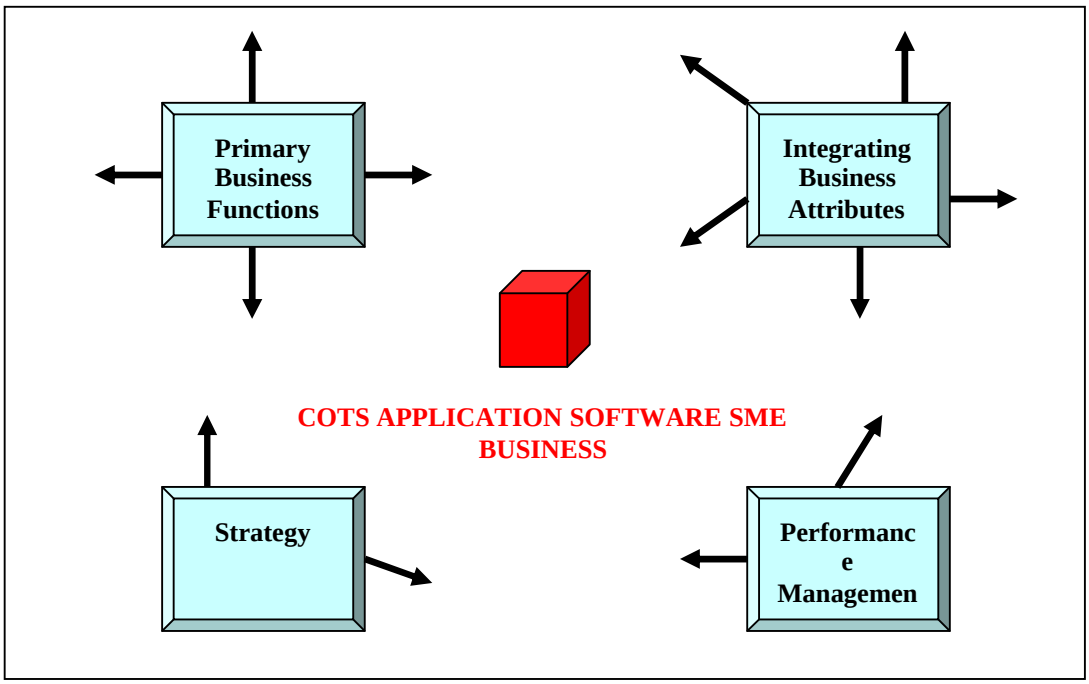
The performance management framework provides a set of quantitative rules, within which a COTS application software SME business should operate. The business KPIs and

total business scorecard theoretical constructs are intrinsically linked. The set of business KPIs provide the contents that goes into the total business scorecard; while the total business scorecard provides a framework for identifying the business KPIs that need to be contained within it. Continual refinement of the business KPIs set and how these are arranged within a total business scorecard will allow a software business to ensure that its performance management framework is as focused as possible.

6.1.5 Summary

The axial processing has been a highly effective means of reducing the complexity associated with the multiple theoretical constructs associated with running a COTS application software business. Four axes were created from the thirteen theoretical constructs as result of the axial coding. Figure 6f illustrates the resulting four axes.

Figure 6f: Theoretical Axis Relevant to Running a Software Business



The remaining challenge is to figure out how the four axes can be connected.

6.2 Integrating the Axes

This section covers the final stage of the analysis, the grounded theory selective coding. The objective was identifying a central construct that would connect all the other theoretical constructs and axis into a single framework.

How a COTS application software SME business can be integrated and bound together into one robust and commanding overall model was always going to be the crux of the research. Right back at the interview stage, I made a specific point of quizzing informants about how they thought this could be conceptualised. The literature review identified many topics of relevance for running a COTS application software SME business, but there was no cohesive framework that pulled all these aspects together. Similarly, while informants also raised numerous topics associated with running a software business, there was very little offered in terms of integrating concepts that connected all these concepts together. However, informants did state that there needs to be something more than just the primary functions that holds and connects a software business together. They highlighted the CEO as a critical in integrating the different parts of the software business.

Informants thought that the financial model is likely to be a big driver of overall business model. Several suggested the use of a value chain is model to identify how each component business area fits into an overall software business. A slight variant on this was to engineer all vendor activities so that they have a focus on adding value to the customer. Simplicity was discussed as a key concept for a holistic vendor business. Keeping software product simple and a vendor's associated business model simple, allows focus and can ensure that everyone knows what they working on and how that achieves value.

> *Everybody in an organisation, I think, really has to understand that they are working for the end customer. And that's where, in so many businesses I've been in, it fails* (I-23).
>
> *We can deliver, sell, deliver and supporting, in a consistent way. That's what we're trying to achieve* (I-28).

Moving forward to final part of the analysis, the research had progressed a long way from the interview discussions about how a software business integrates into a single entity. Thirteen conceptually rich theoretical constructs have been created and four powerful associated axes have been formed. A key part of grounded theory selective coding is the selection criteria for choosing the core category (theoretical construct) for a new theory. However, selection of the central construct for the overall software business model was not initially obvious. Although during the analysis axial and selective coding activities were progressing well, I was struggling to identify a core construct that met the criteria that both Glaser (1998) and Strauss and Corbin (1998) had set for this central concept. The main axes of the model, the primary business functions and integrating business attributes were taking shape, but no obvious concept was standing out that explained a larger proportion of the model than other key constructs. I was overly focused on looking for a large construct, as opposed to focusing on some of the more subtle criteria required for a central construct.

The central construct finally surfaced after reference to Strauss and Corbin's (1998 p147) guidance that the central construct should grow in depth and explanatory power as it analytically integrated. The breakthrough came when diagrammatically modelling how all the constructs in the model fitted together. The software business core construct was set in the middle of software business model as the central construct. While selecting this construct might have seemed like an obvious choice, it had been discarded previously as I had been searching for a larger construct that represented a greater amount of functional activity that occurred within a software business. However, although the software

business core construct is relatively small in terms of dimensions and concepts, it acts as a lynch pin to everything else within the overall model. While the relationship between product and target market is a basic management concept, the relationship between software people, IP, product and target market in an industry of information products is slightly more abstract and intangible. However, it is imperative that these four factors are in alignment not just at the centre of a software business, but also as a backdrop across all of the other twelve theoretical constructs of the overall model. All the primary business functions (axis A) are explicitly driven by the business core. The integrating business attributes (axis B) need to be configured around this as central terms of reference point, as does the performance management (axis D) framework. While from a strategy (axis C) perspective, it is the software business core that is the intersection between a resourced based view and a market positioning perspective.

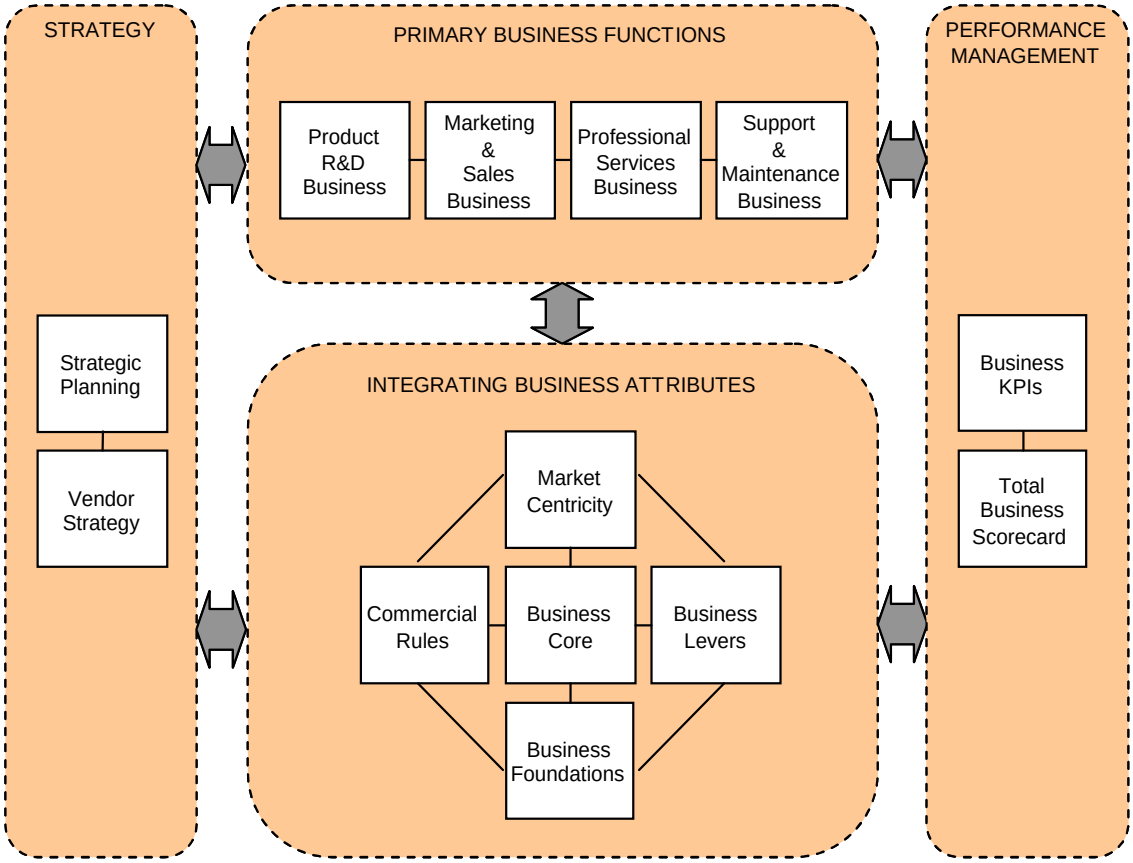
6.3 An Overall Model for Running a Software Business

The culmination of all the findings is an innovative overall model that provides guidance for running a COTS application software SME business. This model, demonstrated in Figure 6g, has four high-level software business components and thirteen separate theoretical constructs that work together in an integrated and holistic model. The arrows and lines within the model illustrate how all these constructs and components are linked together.

The primary business functions component of the model provides a framework for the functional operation of a COTS application software SME business. The integrating business attributes component provides a platform for a software vendor's primary business functions. The amalgamation of the two pulls together both functional activities and a number of other diverse business variables that are required to form a holistic vendor business. The result is a framework used to explain a bigger picture view of managing a software business. The addition of the final smaller two components, strategy

and performance management, contributes a long-term strategic management framework. These complete the overall holistic model for running COTS application software SME business.

Figure 6g: Overall Model for Running a COTS Application Software SME Business



The primary business functions component interacts directly with the other three components of the overall model. How the primary business functions are managed and operated will be driven by largely controlled by the integrated business attributes in the medium term and by the vendor's strategy in the long-term. The performance management framework will provide a check and balance mechanism to ensure that the right primary business functions KPIs are being measured and that the contribution of each area is understood. The primary business functions component includes a product R&D business, a marketing and sales business, a professional services business and a support and maintenance business. The flow of vendor activities associated with the software product being built, sold/purchased, implemented, and kept operational, is

systematically mirrored through the activities of these four primary business areas. A software vendor requires a competitive software product that can solve a business problem. Identifying and targeting a market where the software offering can differentiate itself and create customer value will be a prerequisite to attaining new sales. Getting a software product operational and maximising the benefit it can generate for customers is likely to require some professional services and a level of ongoing support and maintenance.

The integrating business attributes component interacts directly with the other three components of the overall model. There is a bidirectional relationship with all of these. The exact definition and configuration of the integrating business attributes will be heavily influenced by the strength of a vendor's primary business functions and its strategy aspirations. The integrating business attributes will in turn continually be refined and re-calibrated to provide the most appropriate focus for the development of the primary business functions and clarity for the vendor's strategy. The performance management framework will provide a check and balance mechanism to ensure that the right integrating business attribute KPIs are being measured and controlled. However, in contrast to how this works for the primary business functions, the KPIs and associated framework are likely to be intangible and subjective in nature. The integrating business attributes component includes a software business core, market centricity, business foundations, business levers and commercial governance. The software business core pinpoints exactly what business a vendor is actually in. This is further qualified by its market centricity and a set of governing commercial rules that a vendor operates within. Business foundations provide a generic platform for a healthy business. Conversely, a set of business levers can be throttled back and forth dependent on how vigorously the business is to be run.

The strategy component interacts directly with the primary business functions and the integrating business attributes as already explained above. The strategy component also indirectly interacts with the performance management framework. A key part of successfully executing strategy is measuring and controlling key strategic initiatives. The strategy component includes strategic planning and vendor strategy. Taking time to reflect and plan the overall business purpose is a prerequisite to formulating a focused and robust strategy. This needs to drive all other aspects of the business in the overall model. There is a range of appropriate strategy tools that can be used if vendors are resourceful in locating them. Having clear business objectives form the basis of a strategy. Demystifying the dynamic, evolving and distinctive characteristics of the software industry is an imperative. Growth is a topic of particular importance for software business mangers. Growth alternatives include market leadership, similar markets, international markets, and acquisitions and partnerships.

The performance management component interacts directly with the primary business functions and the integrating business attributes, and indirectly with the strategy component as already explained above. The performance management component includes business KPIs and a total business scorecard. While there are numerous business KPIs that can be used to design how a software business is run and measure its ongoing performance, to have an effective performance management framework these KPIs need to fit and link together in an overall business scorecard.

6.4 Summary

Chapter 6 has documented the final stage of the research data analysis. This has built on the two preceding analysis chapters. Chapter 4 progressively unstacked the main attributes and themes associated with running a software business. Ninety-three units of meaning (base-level codes) were created from the 205,000 words of interview transcripts. Chapter 5 then evolved and consolidated the base-level codes into thirteen theoretical

constructs (open coding categories). Due to the volume and complexity of the base-level codes identified, an attempt to assemble an overall model for running a software business directly from 93 codes at one time would have been disorderly and unmanageable. The intermediary step of creating thirteen theoretical constructs allowed main themes to be established, which would subsequently enable a total picture to be developed.

Chapter 6 has completed a systematic grounded theory analysis process. Using grounded theory axial coding, points where the thirteen theoretical constructs intersect were identified. Four resulting axes were classified: primary business functions, integrating business attributes, strategy, and performance management. Next, incorporating grounded theory selective coding, the axes were connected to provide a single model that integrates together all the theoretical aspects of the research findings into running a software business. This included continual refinement and fine-tuning of the model components and in particular re-examining relationships and inter-connections between theoretical constructs. The underlying theme throughout the whole theory construction process was the intent to build a holistic model and ensure that all components of this were fully integrated and connected.

The output of Chapter 6 and the research findings as a whole is the creation of a holistic and integrated software business model that comprises of four main interconnected components. This model provides a guiding management framework for business mangers' running COTS application software SME businesses. The resultant model is discussed and conclusions made in the following chapter.

7 FINAL REFLECTIONS

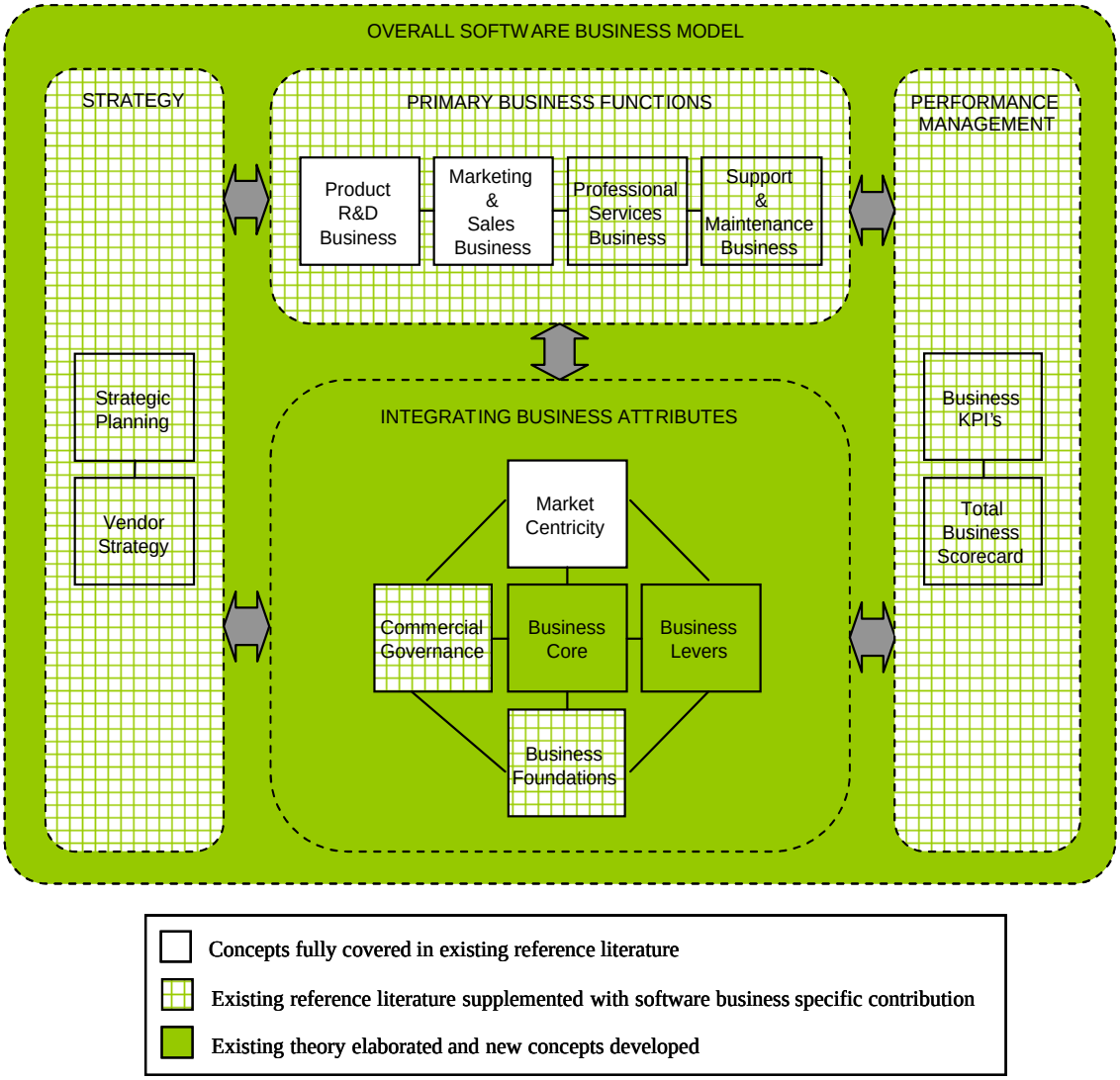
This chapter provides a reflective look at the thesis as a whole. Existing reference literature is re-examined in context of the findings. An assessment of the research findings is made. Strengths and weaknesses of the new overall software business model are identified. The contribution of the study is questioned, implications for both theory and management practice suggested. Recommendations for further research are made and a conclusion is stated.

7.1 Findings in the Context of the Existing Reference Literature

The new overall software business model makes a number of contributions to the current body of knowledge. In this section, the four components and thirteen constructs of the model are compared to existing literatures and new contributions are explicitly highlighted. Figure 7a summarises the constructs of the model where there are new contributions and those that are well covered by existing literature.

The majority of the constructs in the overall software business model are not new when looked at in isolation. It is the way that these have been integrated into a single unified model that contributes to existing literature. The constructs that underpin the primary business functions, performance and strategy components of the new model are largely grounded in existing literature that was identified in the pre-research literature review. However, these constructs have been enriched with software business distinctive characteristics. In contrast, the integrating business attributes component makes a new contribution to the existing body of knowledge. The power of this component of the model comes from it being the glue that integrates all the other disparate pieces of a software business.

Figure 7a Contribution of overall software business model



The remainder of this section examines each of the four model components; a comparison is made between each construct and the context of reference literatures.

7.1.1 Software business primary business functions

The constructs that comprise the primary business functions component of the overall model are largely a consolidation of literature covered in the thesis literature review. Table 7a provides a summary assessment of the contribution this component makes to the existing body of knowledge. The model component is broken down and assessment is provided of whether it is covered in existing literatures, whether a software business

specific contribution to existing literature has been made and whether a specific new theoretical contribution has been achieved.

Table 7a: Contribution of 'Primary Business Functions'

Model Component and Constructs	*Covered in literature review*	*Additional existing literature identified*	*Software business specific contribution to existing literature*	*Specific new contribution*
Primary business functions	No		Yes	Yes
Product R&D business	Yes			
Product development	Yes			
Product management	Yes		Yes	
Product roadmap	Partial		Yes	
IP management	Partial	Yes		
Marketing & sales business	Partial			
Market positioning	Yes			
Market offering	Yes		Yes	
Sales	Partial	Yes		
Professional services business	Partial		Yes	
Software professional services	Partial		Yes	
Professional services business contribution	No		Yes	Yes
Support & maintenance business	Partial		Yes	
Software support & maintenance	Partial		Yes	
Software support & maintenance business contribution	No		Yes	Yes

The main contribution is the aggregation of all the primary business function constructs into an interconnected and integrated framework. These ideas are implicitly presented across other reference literatures. Cusumano (2004a), Hasted (2005) and Roeding et al. (1999) do discuss most of these concepts. However, these are generally handled as individual and discrete activities. This study contributes by formally and explicitly pulling these concepts together in a higher-level construct. This forms a key foundation and point of reference for the rest of the overall software business model.

The individual primary business function constructs are generally covered in existing literatures. The core aspects of running a software product R&D operation are covered widely in IS research and management science, with the reference domains of innovation, product development and product management providing further references. However, the research does supplement existing literature by emphasising COTS software as building blocks of business solutions and reemphasising the importance of real commercial discipline in running a product R&D business in a similar vein to Krishnan and Ulrich (2001). The software sales and marketing business construct draws reference frameworks from both practitioner research and generic marketing and sales disciplines. The software market offering concept unifies a number of distinctive attributes specific to the software industry. This complements Hogan's (2006) principle that software should create real tangible business value for a customer. Two other observations are that distribution and IP management are under-represented in the literature review. In retrospect, it is suggested that both these reference literatures from these domains could further elaborate the overall software business model.

Two areas of interest for the primary business functions model component are the constructs of professional services business contribution and the support and maintenance business contribution' While existing literature discusses software professional sales and software support and maintenance functions, the new findings provide explicit contextualisation and linkage of these into an overall software business. This builds on Esposito's (2006) argument that professional services contribution should have a direct positive impact on the overall business.

7.1.2 Software business integrating business attributes

The integrating business attributes component of the overall model provides the main contribution to theory. The five integrating business attributes draw very little from the literature review that was undertaken prior to entering the field. Table 7b provides a

summary assessment of the contribution this component makes to the existing body of knowledge.

Table 7b: Contribution of 'Integrating Business Attributes'

Model Component and Constructs	*Covered in literature review*	*Additional existing literature identified*	*Software business specific contribution to existing literature*	*Specific new contribution*
Integrating business attributes	No			Yes
Business core	No			Yes
Market centricity	Partial		Yes	
Customer experience	No	Yes	Yes	
Customer retention	No	Yes	Yes	
Market power	Yes			
Business foundations	No		Yes	
Business leadership	Partial	Yes	Yes	
Human resources	Partial	Yes	Yes	
Operational capabilities	No	Yes		
Business levers	No			Yes
Engagement model	No			Yes
Capability sourcing model	No	Yes	Yes	
Ownership model	No	Yes	Yes	
Commercial governance	Partial		Yes	
Cost model	Partial	Yes	Yes	
Revenue model	Partial	Yes	Yes	
Profit model	Partial	Yes	Yes	
Risk profile	Partial	Yes	Yes	

The construct of a business core represented in concentric circles, comprising the importance of people, IP and product capabilities all being aligned with a clearly defined target market is at the heart of a software business. This simple construct provides a useful starting point for any vendor wishing to assess the strength of its software business. While there is considerable literature on these four attributes in isolation, I am unaware of them being linked together in quite this way in the context of a software business.

The market centricity construct identifies the relevance of customer satisfaction and marketing literature for running a software business. These sources were not explicitly examined in the literature review. Although various IS research scholars, such as Hui and Tam (2002) and Negroponte (2004), were referenced as stressing the importance of software having desirable capabilities and being effective. Its inclusion is highlighted in particular, as this often is not actually present in many parts of the industry,

The notion that a software business needs to have a number of prerequisites, or foundations, to ensure that it has a solid basis for success, was a recurring message from the data. In the same way that secondary business activities in Porter's value chain (1985) contribute value across primary activities, the same is the case in the overall software business model. However, it is less clear exactly what such a list of business foundations should be. Therefore, it is suggested that business leadership, human resources and operational capabilities are more an initial set of such foundations, rather than an exhaustive list. It is noted that reference literatures for leadership, human resources and operations management could further inform this area.

Another recurring theme was informants referring to a set of tools that they had at their disposal to re-configure the size and momentum of a software business. These were conceptualised through the analysis as a business levers construct. The literature review prior to the research did not discuss such a concept, although reference literatures on outsourcing, off-shoring and company ownership have some commonalities with this idea. It is questionable whether these disciplines would fully capture the essence of a business having a set of multiple levers that can used to essentially change gear with the their operations. Similar to the business foundations construct, the final set of attributes for the business levers construct, may not solely comprise the list of attributes identified in this research. There could well be others that could be added to this construct.

The commercial governance construct in this the model makes no new contribution in terms of generic concepts. Financial models and risk management are well covered in their associated scholarly domains. However, prior to the research very little information was examined on these areas within the specific context of a software business. A few references to software business financial ratios and metrics were discussed by industry analyst Keller (2005b) and by Cusumano in his software business book (2004a). However, these were not substantial or empirically based frameworks. Topics that highlight the inherent risks involved in running a software business are touched upon in IS research and management science. Nault and Vandenbosch (2000) stress there are continuing widespread changes in the software market and the Economist (2005b p47) highlighted the dangers associated with running a business in such an environment. The new model take these concepts one step further and attempts to codify these explicitly against a risk management framework. Reference literature in risk management could be consulted to inform the development of this aspect of the model further.

While the integrating business attributes in the overall model are to be found in various disciplines, these five constructs interrelate and provide a cohesive set of concepts that pull the whole software business together as a single entity. This specific integrating capability provides the major contribution of the research findings.

7.1.3 Software business strategy

The strategy component of the overall software business model is in line with generic theories and associated frameworks from the strategy management field. Practitioner research, industry analyst reports and IS research domains provide many illustrations of the evolving software business landscape. Sheth and Sisodia (2002) highlighted industry consolidation. Dver (2003) and Hasted (2005) identify international growth as a business

tactics for growth. Table 7c provides a summary assessment of the contribution this component makes to the existing body of knowledge.

Table 7c: Contribution of Strategy

Model Component and Constructs	*Covered in literature review*	*Additional existing literature identified*	*Software business specific contribution to existing literature*	*Specific new contribution*
Strategy	Yes		Yes	
Strategic planning	Yes		Yes	
Strategy development	Yes		Yes	
Strategy tools	Yes	Yes	Yes	Yes
Vendor strategy	Yes		Yes	
Clear business purpose	Yes		Yes	
Target market analysis	Yes		Yes	
Growth strategy	Yes		Yes	

It is suggested that for all the background theory and deliberation in the software business forums, no clear guidance or framework was identified prior to entering the field that specifically assists software vendors with developing strategy for their businesses. The strategic planning and vendor strategy constructs of the new overall model supplement existing strategy literature with software business contributions. The result is a simple framework that can be used to assist vendors with software business specific considerations when considering standard strategic options as part of a strategy formulation process. International growth is highlighted as one area in which while there may be fewer barriers to entry with a software product in comparison to a traditional tangible product. A number of subtleties that a vendor will need to consider in order to be successful when taking a standard product into a local market are highlighted. While not being extensive in terms of every possible strategic direction, the vendor strategy construct does now provide software businesses with a point of reference to consider the main strategies that are commonly used to achieve growth in the software industry.

One point of note on the overall model is the reference to various software business strategy tools that vendors have at their disposal if they think laterally. There is debate in literature about whether a reference industry exists for the software industry. Geisman (2006d) considers railroads for instance. However, rather than using to a single reference industry, a hybrid of multiple industries will actually better inform the running of a software business.

7.1.4 Software business performance management

The performance management component of the overall model has a backdrop of generic reference literature that underpins it. However, limited literature that discusses or applies performance management specifically to software businesses was identified prior to the research. Table 7d provides a summary assessment of the contribution this component makes to the existing body of knowledge.

Table 7d: Contribution of Performance Management

Model Component and Constructs	*Covered in literature review*	*Additional existing literature identified*	*Software business specific contribution to existing literature*	*Specific new contribution*
Performance Management	No	Yes	Yes	
Business KPIs	No	Yes	Yes	
Primary business KPIs	Partial	Yes	Yes	
Overall business KPIs	No	Yes	Yes	Yes
Total business scorecard	No	Yes	Yes	

While practitioner research and industry analyst reports touched on a number of KPIs for primary business functions such as R&D investment levels and cost of sales (Cusumano 2004a; Keller (2005b), a perceived void existed for KPIs that unified a software business as a holistic model. Probing of individual informants provided many different KPIs of interest that were used to configure and measure software businesses. Informants' suggestions of KPIs for financials and customer satisfaction point to a further

examination of literature in those areas. However, the construct of KPIs for a software business' strategic assets is more novel.

It became very clear that there was no holistic performance management perspective across a software business. This perceived gap drove the theoretical sampling to attempt to identify and assemble a set of KPIs into a single performance management canvas covering all aspects of a software business. Kaplan and Norton's strategy maps (2004) provided a useful framework for doing this. Their model provided a backdrop to arrange KPIs within, while also acting as a validation tool to ensure KPIs for different aspects of a performance framework were in place. As a result, the total business scorecard for an overall software business, while based on existing literature, is a contribution to the body of knowledge for running a software business.

7.2 Assessment of Findings

Objectively assessing the legitimacy and credibility of the findings is a key part of assessing the overall research. The empirical grounding of the new overall software business model is evaluated. Strengths and weaknesses of the new model are identified.

7.2.1 Empirical grounding of the model

Strauss and Corbin (1998) offer a set of eight criteria that can be used for assessing whether the findings of a grounded theory are empirically grounded. Table 7e compares the overall software business model against each of these criteria.

Table 7e: Assessment of Grounded Theory Findings

Criterion	Evidence
Criterion 1: Are concepts generated?	Many concepts were generated from the data. An example is that informants mentioned the pros and cons related to the amount of time that was spent interacting with customer. This led to the engagement model concept. Similarly, how the appetite for pursing growth opportunities varied depending on the profile of the CEO and major shareholders provided the basis for the ownership model concept.

Criterion 2: Are the concepts systematically related?	Yes. This is discussed below in the section 'strengths of the new model'.
Criterion 3: Are the categories well developed? Do categories have conceptual density?	Some categories [constructs] of the new model are better developed than others are. Constructs within the primary business functions and the software business strategy surfaced early in the research and considerable field data were collected to provide conceptual density. In contrast, constructs such as the business core, business foundations, business levers and [software business] commercial governance surfaced later in the fieldwork and although some theoretical sampling was undertaken, there is a limit to the amount of supporting data for these constructs. Additional time in the field could be used to develop these categories further.
Criterion 4: Is variation built into the theory?	The overall model does allow for some variation. While the model encompasses thirteen constructs that constitute a software business, a number of vendors were identified in the fieldwork that did not have either a professional services business, or a support and maintenance business, or in one case both. As the model is a reference framework as opposed to a prescriptive model with mandatory constructs, it is valid and still applicable when there are slightly different vendor profile variants that do not use all aspects of it.
Criterion 5: Are the conditions under which variation can be found built into the study and explained?	The overall model is essentially a macro-model that integrates together a diverse set of business management concepts against the backdrop of the complex software industry. The broader concepts of customer value, business, strategy and management are woven into the new model at many points.
Criterion 6: Has process been taken into account?	The findings describe a process of developing and growing a software business model. As a vendor progresses a long a maturity curve, different areas of the model will come into focus.
Criterion 7: Do the theoretical findings seem significant and to what extent?	The theoretical findings are noteworthy as they contribute to filling a gap and they provide a new and holistic of guidelines to assist in the running of a software business. When the model is practically applied in the field, the extent of its significance will become apparent.
Criterion 8: Does the theory stand the test of time and become part of the discussion and ideas exchanged among relevant and professional groups?	The section below on implications for management practice discusses opportunities for the findings to influence a wider circle of industry practitioners. At this point it is too early to judge whether this will occur and stand the test of time.

In summary, the new software business model does meet the Strauss and Corbin's (1998) criteria for the empirical grounding of grounded theory research findings.

7.2.2 Strengths of the model

The new overall software business model has a number of strengths. It provides a novel and different perspective to running a software business. While not revolutionary, the model elaborates, integrates and binds together a number of existing theories and reference frameworks into a unified model. This has the advantage that a number of constituent parts of the new model are grounded in solid concepts that are supported by extensive reference materials. The findings are of practical relevance. With the level of detail in the model constructs and the ability to apply this to a real life vendor, it is hoped that software business managers will be able to recognise and apply these ideas to their businesses.

A particular strength of the overall software business model is that the constituent constructs are systematically related and there are strong linkages across the model as a whole. A big challenge through the open and axial coding was deciding to which constructs some of the concepts should be allocated. There often appeared to be equal weight for some of the concepts to live in more than one construct. As an example, should the concept of domain experience be assigned within the business foundation construct, or the marketing and sales construct? Similarly, should customer experience be limited to the primary business functions, or is it actually an integral part of a software business' integrating business attributes? In the earlier parts of the analysis, this caused debate, but as the analysis progressed, there was an acceptance that constructs were not mutually exclusive. It was realised that this could actually be a strength rather than a weakness, as a number of the concepts actually touch the model at multiple points. Although this adds complexity, it also has the benefit of moving the model beyond a set of loosely connected mutually exclusive constructs, to a holistic framework that has multiple connections and bindings between constructs and across the model as a whole. All of the four primary business functions need to be underpinned and related to each of

the five integrating business attributes. These in turn need to performance managed and contextualised within the vendor's strategy. As an example, the capability for marketing and sales may be enabled by the capability model for distribution within the business levers construct, which is itself driven by an international growth strategy. Further exploring how each of the constructs are related and influence each other provides a useful tool for further strengthening the overall model.

7.2.3 Weaknesses of the model

Several weaknesses in the overall software business model are present. Some of the model constructs, particularly in the integrating business attributes component, are relatively intangible. It may be a challenge in practice to unequivocally define and quantify such constructs. The overall software business model is unproven and therefore lays itself open to criticism from a positivistic perspective. An intrinsic part of a grounded theory research paradigm is that newly created theories are only grounded and they are not actually proven. There are limitations to the new software business model and it is important to contextualise these within academic and professional communities. While accuracy, simplicity and generalisability are all desirable traits of a good theory, in reality the research scope requires tradeoffs amongst these three criteria (Langley 1999). In this study, the pursuit of simplicity and generality has been achieved at the expense of attaining the high a level of accuracy that was originally desired. If the intention was to achieve a more concrete theory, then it is suggested a more traditional quantitative research method should subsequently be employed. However, it is hoped that the attraction of a grounded theory generated software business model will be its originality and its ability to assist software vendor strategy formulation, where limited guidance exists today.

7.3 Implications for Theory

The implications of the findings for theory are the elaboration of the existing body of knowledge available for running a software business. Building on solid principles for strategic management and a number of other general reference disciplines, the findings contribute a foundation for a more rigorous and scientific approach to software business management and the running of a software business. Various specific implications of the research for theory are highlighted.

First, the study verifies the suggestion in the introduction chapter that a more holistic approach to running software businesses is required. The research informants unanimously supported the idea that there is not a standard set of software business specific models that they can call upon to assist them in running their software businesses. This was complemented by statements that if such a toolkit were available it could be of great benefit.

The second implication for theory is that the research has contributed to the demystification of the software business landscape. The new overall business model provides a different perspective on running software businesses. The model does not introduce fundamentally new concepts, but what it does do is consolidate and bind together a range of other reference theories into one holistic integrated model. The model maps out a set of software business variables and provides a structured framework to that can be used for further investigation of the forces at work in running a software business. The overall model also introduces a number of conceptual ideas that contribute innovative and novel thinking. Capability sourcing, the software specific customer value proposition and the customer engagement model are examples of these.

The third implication for theory is potentially more controversial. There appears to be a disproportionate amount of rhetoric and commentary on what are essentially not the most important aspects of running a software business. Often vendors pursue the latest fad or direction documented in both academic and industry publications; with these being commonly cited as holistic business models or comprehensive strategies. SaaS is often cited as a business model; it is not. More accurately, it could be described as a consumption and distribution model. Service-orientated architecture (SOA) is often described as a software vendor strategy; it is not a holistic strategy. More accurately, it could be described as a product development platform or a vehicle for product segmentation. A clear implication of this research is that any proposed software business model needs to cover a whole range of different factors involved in running a business. An overall model cannot just be based around a single contemporary topic. The new overall software business model reinforces the requirement for theories in this area to be holistic and balanced.

Further research could be undertaken in a number of other areas to advance the overall software business model. First, the relationship between the application of the model for running a software business and the actual resulting commercial performance of that software business could be more closely examined. Using a sample of software vendors, a capability assessment against the thirteen constructs could be made against the actual performance of software businesses. Second, while the new overall software business model is unproven, there is now an opportunity to enhance the framework through further research. Hypothesis-based research could be undertaken to test whether the thirteen constructs can really be operationalised to provide guidance in running a software business; and identify potential areas of the model that may need to taken back to the drawing board and redeveloped. Third, further research could focus around collecting more data to further inform the integrating business attributes component of the model.

The exact composition and conceptual make up of the constructs in this area could evolve further and take on a slightly different form from what the model developed in this initial study. As some the constructs gain further density, others may lose it. This could result in further levelling of the model where some constructs become more important, while others move to the background. In addition to the refinement of constructs, further research can also result in more explicitly defined linkages between the model constructs. As this is the area in which the model makes the biggest contribution, this would benefit from further validation and clarification.

7.4 Implications for Management Practice

Development of the new overall software business model has significant implications for management practice. First, there is a significant interest and demand in the software industry for the new overall software business model that has been generated by this research. Various opportunities exist for socialising and promoting the model. Second, the model can be used to provide highly specialised strategic management consulting to assist COTS application software vendors in running their businesses.

7.4.1 Demand for the model: Opportunities for socialisation

There are numerous industry journals and newsletters where the research could be formally published as a way of reaching practitioners. However, it is suggested that the way maximum impact can be achieved from the new overall software business model for management practice is through online business networks and forums. Many possible sites discuss various concepts of software and IT, although two sites have been initially targeted as being of particular interest. 'SandHill.com' provides considerable insight and opinion on '*business strategy for software executives*' (Sandhill 2009). Conversely, the 'Business of Software' provides a network of individuals who are interested in '*building long term sustainable, profitable software businesses*' (BoS 2010).

There is an opportunity to present and socialise my findings at appropriate and relevant industry events and conferences. In Australia, the Australian Information Industry Association (AIIA) and Australian Computer Society (ACS) hold various industry events and organise special interest groups for software professionals. While in North America, the yearly Business of Software and the Sandhill Software conferences are events that are exclusively targeted at software business managers. Keynote speakers at these events include very high profile software business leaders. Attendance at these events will serve a dual purpose of providing exposure for the new overall software business model, as well as acting as a vehicle for receiving direct feedback from practitioners that can be used to refine the framework further.

More than half of the informants who were interviewed for the research were very interested in the contribution that the study could make to the existing body of knowledge for running of a software business. These informants requested a summary copy of the findings when the study was complete. Others wanted to know if the intent was to publish a version of the study in an industry or practitioner journal and they wanted to be kept informed on when this would become available. I have received an invitation to speak and present the study findings to an industry peer group of twenty or so Australian SME software business CEOs. This group, to which two of my informants belong, is loosely affiliated with the AIIA. The level of support I have received from software business managers and the amount of enthusiasm there has been for developing the thinking around running a software business reinforces the case for this research making a valuable contribution.

7.4.2 Application of the model: Strategic consulting for software vendors

The new overall software business model can be applied in a number of high value ways to assist software vendors in running their businesses. It is common in business for

management consultancies and internal management functions to undertake strategy and business improvement studies for organisations of all shapes and sizes. These pieces of work are usually undertaken using a toolkit of standard strategy and business consulting models and methodologies. The addition of the new overall software business model to this generic toolkit will enable highly specialised strategic assessment, business improvement, and strategy development activities to be undertaken for COTS application software businesses.

A systematic approach is recommended for how the overall software business model can be applied in a strategic analysis piece of work for a software vendor. Firstly, the constituent components of the model can be used as a point of reference to assess a software business' current capabilities. It is suggested that this would involve systematically working through each of the thirteen software business model constructs to analyse the importance, configuration and current strength of these as part of the overall business. The software business constructs that comprise the primary business functions component and commercial governance construct are tangible. It should relatively easy to use a scientific method to assign a form of quantitative measure. A simple numeric scale of weak to strong could be used for this.

Four of the five constructs in the integrating business attributes component of the overall model will require more subjective qualification. In some cases, these constructs of the software business model may just seem abstract, with no one within an actual vendor business has previously considering, let alone articulating, defining and measuring. It is noted that further research that more clearly defines exactly how a number of the intangible and subjective model constructs can be qualified and quantified would be beneficial. However, although it is not an exact science, practitioners should not shy away from attempting this exercise; it is critical part of defining the holistic profile of a

software business. To reinforce this point, in the case of the business core and market centricity constructs, it is suggested that if these cannot be clearly defined for a vendor business, it is unlikely that that vendor will be able to run their business with any real focus at all. With the business foundations and business lever constructs, these cannot be operated in isolation to the business as a whole; they need to have a point of reference against which they can be commercially gauged. The business core, market centricity, commercial governance and primary business functions provide such a point of reference for these to be calibrated against appropriately.

Clarity of the nine constructs of the combined primary business model functions and integrating business attributes provides a vendor with the key cornerstones of a holistic software business model. The performance management component of the overall software business model can then be used to assess the maturity and strength of each of the business KPIs for the key business capabilities. The total business scorecard subsequently arranges these KPIs in a single picture. The scorecard logically connects together a holistic set of software business attributes in a simple representation. The scorecard can then be used as a high-level tool to identify where there are weaknesses and/or gaps in a software business model where business improvements can be made. A software vendor should be able to develop a very focused and strong business model by utilising the new software business specific performance management framework.

Strategy development for a software vendor is pulled together by utilising the strategy component of the overall software business model. As well as reinforcing the imperative of strategic planning, the new model offers some solid strategy tools for developing a software business strategy that software managers may not have considered previously. The growth strategy construct provides a reference framework of commonly pursued

software business growth paths. It includes a checklist of points to consider so common pitfalls that have experienced by others in the industry can be avoided.

7.5 Conclusion

The purpose of this research was to improve the guidance available to business managers running COTS application software businesses. While existing academic literatures and industry commentary contribute many reference models for running such a business, these are disjointed and do not provide a unifying framework that links these ideas and concepts. The output of this study is now a new overall software business model that provides a single framework for managers to integrate numerous aspects of perceived best practice. In conclusion, this research has contributed to both scholarly and practitioner domains. The new overall software business model elaborates existing theories, and by building on solid principles for strategic management and various other reference disciplines, a more rigorous and scientific approach to software business management theory has been developed. The new overall software business model has real life practical application. It can be used as a tool to provide highly specialised strategic assessment, business improvement, and strategy development services for a COTS application software business.

8 REFERENCES

Abell, DF (1980) *Defining the Business. The Starting Point of Strategic Planning*. Englewood Cliffs, NJ, Prentice-Hall.

Adams, WJ & Yellen, JL (1976) Commodity Bundling and the Burden of Monopoly. *Quarterly Journal of Economics*, Vol. 90 Issue 3, pp475-498.

Ahmed, FC & Luiz F (2007) Managing the Business of Software Product Line: An Empirical Investigation of Key Business Factors. *Information and Software Technology*, Vol. 49 Issue 2, pp194-208.

Ahmed, F & Capretz, LF (2010) A Business Maturity Model of Software Product Line Engineering, *Information Systems Frontiers*, Springer, In Press.

Ahrens, T & Dent, JF (1998) Accounting and Organisations: Realizing the Richness of Field Research. *Journal of Management Accounting Research*, Vol. 10, pp1-39.

Alderman, R (1999) Only Eight Product Strategies Exist. *Electronic Design*, Vol. 47 Issue 12, p97.

Alic, JA; Miller, JR & Hart, JA (1991) Computer Software: Strategic Industry. *Technology Analysis & Strategic Management*, Vol. 3 Issue 2, pp177-190.

Amoribieta, I; Bhaumik, K; Kanakamedala, K & Parkhe, AD (2001) Programmers Abroad: A Primer on Offshore Software Development. *McKinsey Quarterly*, Issue 2, pp128-139.

Anderson, EE & Chen, YM (1997) Microcomputer Software Evaluation: An Econometric Model. *Decision Support Systems*, Vol. 19 Issue 2, pp75-92.

Anderson, J & Wood, R (2002) Seven Management Lessons from Microsoft. *Business Strategy Review*, Vol. 13 Issue 3, pp28-33.

Andrews, KR (1971) New Horizons in Corporate Strategy. *McKinsey Quarterly*, Vol. 7 Issue 3, pp34-43.

Ansoff, HI (1987) Strategic Management of Technology. *Journal of Business Strategy*, Vol. 7 Issue 3, pp28-39.

April, A; Hayes, JH; Abran, A & Dumke, R (2005) Software Maintenance Maturity Model (SM^{mm}): The Software Maintenance Process Model. *Journal of Software Maintenance & Evolution: Research & Practice*, Vol. 17 Issue 3, pp197-223.

Armstrong, CP (1999) Information Technology Assimilation in Firms: The Influence of Senior Leadership and IT Infrastructures. *Information Systems Research*, Vol. 10 Issue 4, pp304-327.

Arora, A; Arunachalam, VS; Asundi, J & Fernandes, R (2001) The Indian Software Services Industry. *Research Policy*, Vol. 30 Issue 8, pp1267-1286.

Arthur, WB (1989) Competing Technologies, Increasing Returns, and Lock-In By Historical Events. *Economic Journal*, Vol. 99 Issue 394, pp116-131.

Aurum, A; Daneshgara, F & Warda, J (2008) Investigating Knowledge Management Practices in Software Development Organisations – An Australian experience. *Information and Software Technology*, Vol. 50 Issue 6, pp511-533.

Babcock, C & Kontzer, T (2004) New Software Landscape. *InformationWeek*, Issue 1019, pp20-22.

Babchuk, WA (1996) Glaser or Strauss?: Grounded Theory and Adult Education. [On line] Available at: http://www.iupui.edu/~adulted/mwr2p/prior/gradpr96.htm

Bahrami, H & Evans, S (1989) Strategy Making in High-Technology Firms: The Empiricist Mode. *California Management Review*, Vol. 31 Issue 2, pp107-128.

Bailey, KD (1982) *Methods of Social Research*, 2nd Edition. The Free Press, New York.

Bakos, JY & Treacy, ME (1986) Information Technology and Corporate Strategy: A Research Perspective. *MIS Quarterly*, Vol. 10 Issue 2, pp106-119.

Bakos, Y & Brynjolfsson, E (1999) Bundling Information Goods: Pricing, Profits, and Efficiency. *Management Science*, Vol. 45 Issue 12, pp1613-1630.

Banker, RD & Kauffman, RJ (2004) The Evolution of Research on Information Systems: A Fiftieth-Year Survey of the Literature in Management Science. *Management Science*, Vol. 50 Issue 3, pp281-298.

Barney, J (1991) Firm Resources and Sustained Competitive Advantage. *Journal of Management*, Vol. 17 Issue 1, pp99-120.

Barney, J; Wright, M & Ketchen Jr, DJ (2001) The Resource-Based View of the Firm: Ten Years After 1991. *Journal of Management*, Vol. 27 Issue 6, p625-642.

Baskerville, RL & Myers, MD (2002) Information Systems as a Reference Discipline. *MIS Quarterly*, Vol. 26 Issue 1, pp1-14.

Beard, C & Easingwood, C (1992) Sources of Competitive Advantage in the Marketing of Technology-intensive Products and Processes. *European Journal of Marketing*, Vol. 26 Issue 12, pp5-18.

Benbasat, I; Goldstein, DK & Mead, M (1987) The Case Research Strategy in Studies of Information Systems. *MIS Quarterly*, Vol. 11 Issue 3, pp368-386.

Benbasat, I & Zmud, RW (2003) The Identity Crisis within the IS Discipline: Defining and Communicating the Discipline's Core values. *MIS Quarterly*, Vol. 27 Issue 2, pp183-194.

Bennett, C & Timbrell, G (2000) Application Service Providers: Will They Succeed? *Information Systems Frontiers*, Vol. 2 Issue 2, pp195-211.

Berquist, T (2006) 4 Ways to Play in Enterprise Open Source. *Sandhill.com website*, 23/3/06, [On line] Available at: http://sandhill.com/opinion/editorial.php?id=72

Bhattacharya, S & Krishnan, V (1998) Managing New Product Definition in Highly Dynamic Environments. *Management Science*, Part 2 of 2, Vol. 44 Issue 11, ppS50-S64.

Blooma, PN & Reveb, T (1990) Transmitting Signals to Consumers for Competitive Advantage. *Business Horizons*, Vol. 33 issue 4, pp58-66.

Bower, JL & Christensen, CM (1995) Disruptive Technologies: Catching the Wave. *Harvard Business Review*, Vol. 73 Issue 1, pp43-53.

Borés, C; Saurina, C & Torres, R (2003) Technological Convergence: A Strategic Perspective. *Technovation*, Vol. 23 Issue 1, pp1-13.

Boscha, J & Bosch-Sijtsemab, P (2010) From Integration to Composition: On the Impact of Software Product Lines, Global Development and Ecosystems. *Journal of Systems and Software.* Vol. 83 Issue 1, pp67-76.

Brooks, G (2005) 10 Challenges for Software CEOs. *Sandhill.com website*, [On line] Available at: http://www.sandhill.com/opinion/editorial.php?id=57

Brouthers, KD & Van't Kruis, YM (1997) Competing in Software: Strategies for Europe's Niche Businesses. *Long Range Planning*, Vol. 30 Issue 4, pp518-528.

Brown, SL & Eisenhardt, KM (1995) Product Development: Past Research, Present Findings, and Future Directions. *Academy of Management Review*, Vol. 20 Issue 2, pp343-378.

Brynjolfsson, E & Kemerer, CF (1996) Network Externalities in Microcomputer Software: An Econometric Analysis of the Spreadsheet Market. *Management Science*, Vol. 42 Issue 12, pp1627-1647.

Business of Software (2010) Business of Software Homepage. *Business of Software website,* Available [On line] at: http://www.businessofsoftware.org/

Chang, L & Birkett, B (2004) Managing Intellectual Capital in a Professional Service Firm: Exploring the Creativity-Productivity Paradox. *Management Accounting Research*, Vol. 15 Issue 1, pp7-31.

Chapin, N (2000) Software Maintenance Types – A Fresh View. *International Conference on Software Maintenance*, *San Jose, California.* Los Alamitos, CA, IEEE Computer Society, pp247-252.

Chapman, MR (2005) *The Product Marketing Handbook for Software,* 4th Edition. Aegis Resources.

Chen, PY & Hitt, M (2002) Measuring Switching Costs and the Determinants of Customer Retention in Internet-Enabled Businesses: A Study of the Online Brokerage Industry. *Information Systems Research*, Vol. 13 Issue 3, pp255-274.

Cheng, YT & van de Ven, AH (1996) Learning the Innovation Journey: Order out of Chaos? *Organization Science*, Vol. 7 Issue 6, pp593-614.

Chesbrough, HW (2003) The Era of Open Innovation. *MIT Sloan Management Review*, Vol. 44 Issue 3, pp35-41.

Choudhary, V; Tomak, K & Chaturvedi, A (1998) Economic Benefits of Renting Software. *Journal of Organizational Computing & Electronic Commerce*, Vol. 8 Issue 4, pp277-305.

Chrisman, JJ; Hofer, CW & Boulton, WR (1998) Toward a System for Classifying Business Strategies. *The Academy of Management Review*, Vol. 13 Issue 3, pp413-428.

Christensen, CM & Overdorf, M (2000) Meeting the Challenge of Disruptive Change. *Harvard Business Review*, Vol. 78 Issue 2, pp66-75.

Christensen, CM (2002) The Rules of Innovation. *Technology Review*, Vol. 105 Issue 5, pp32-38.

Church, J & Gandal, N (1992) Network Effects, Software Provision, and Standardisation. *Journal of Industrial Economics*, Vol. 40 Issue 1, pp85-103.

Chuang, JCI & Sirbu, MA (1999) Optimal Bundling Strategy for Digital Information Goods: Network Delivery of Articles and Subscriptions. *Information Economics & Policy*, Vol. 11 Issue 2, pp147-176.

Clemons, EK (1986) Information Systems for Sustainable Competitive Advantage. *Information & Management*, Vol. 11, November, pp131-136.

Collins, J & Hussey, R (2003) *Business Research*, 2nd Edition, Palgrave Macmillan.

Coyne, IT (1997) Sampling in Qualitative Research. Purposeful and Theoretical Sampling; Merging or Clear Boundaries? *Journal of Advanced Nursing*, Vol. 26 Issue 3, pp623-630.

Cooper, RG & Kleinschmidt, EJ (1987) New Products: What Separates Winners from Losers? *Journal of Product Innovation Management*, Vol. 4 Issue 3, pp169-184.

Cooper, RG & de Brentani, U (1991) New Industrial Financial Services: What Distinguishes the Winners. *Journal of Product Innovation Management*, Vol. 8 Issue 2, pp75-90.

Creswell, JW (1998) *Qualitative Inquiry and Research Design Choosing Among Five Traditions*. Thousand Oaks, London, Sage Publications.

Curtis, S; Gesler, W; Smith, G & Washburn, S (2000) Approaches to Sampling and Case Selection in Qualitative Research: Examples in the Geography of Health. *Social Science & Medicine*, Vol. 50 Issue 7-8, pp1001-1014.

Cusicka, JJ; Prasada, A & Tepfenhart, WM (2008) Chapter 6 Global Software Development: Origins, Practices, and Directions. *Advances in Computers*, Vol. 74, pp201-269.

Cusumano, MA & Selby, RW (1998) Microsoft Secrets: How the World's Most Powerful Software Company Creates Technology, Shapes Markets and Manages People. *Free Press, 1 Touchsto Edition.*

Cusumano, MA (2004a) *The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad*. Free Press.

Cusumano, MA. (2004b) Reflections on Free and Open Software. *Communications of the ACM*, Vol. 47 Issue 10, pp25-27.

Davenport, TH; Harris, JG & Kohli, AK (2001) How Do They Know Their Customers So Well? *MIT Sloan Management Review*, Vol. 42 Issue 2, pp63-73.

Davies, H & Walters, P (2004) Emergent Patterns of Strategy, Environment and Performance in a Transition Economy. *Strategic Management Journal*, Vol. 25 Issue 4, pp347-364.

Day, GS & Schoemaker, PJH (2000) Avoiding the Pitfalls of Emerging Technologies. *California Management Review*, Vol. 42 Issue 2, pp8-33.

Deans, GK; Kroeger, F & Zeisel, S (2002) The Consolidation Curve. *Harvard Business Review*, Vol. 80 Issue 12, pp20-21.

Dekleva, S. (1992) Software Maintenance: 1990 Status. *Journal of Software Maintenance: Research and Practice*, Vol. 4 Issue 4, pp233-247.

Dembrowski, S & Hanmer-Lloyd, S (1995) Computer Applications – A New Road to Qualitative Data Analysis? *European Journal of Marketing*, Vol. 29 Issue 11, pp50-62.

Denning, PJ (2004) The Social Life of Innovation. *Communications of the ACM*, Vol. 47 Issue 4, pp15-19.

Business Insights (2008) The Top 10 Software Vendors: Positioning, Performance and SWOT Analyses. *Business Insights*, Published March 2008 – Global, [On line] Available at: http://www.globalbusinessinsights.com/content/rbtc0104m.pdf

DePalma, DA (2006) Localization Maturity Model. *Common Sense Advisory Research*, 16/8/2006, [On line] Available at: http://www.commonsenseadvisory.com/research/report_view.php?id=35#

Desmond, JP (2005) Overall Industry Thriving. *SoftwareMag.com*, October, [On line] Available at http://www.softwaremag.com/L.cfm?Doc=2005-09/2005-09software-500

Dewire, DT (2000) Application Service Providers. *Information Systems Management*, Vol. 17 Issue 4, pp14-19.

Dick, B (2005) Grounded Theory: A Thumbnail Sketch. [On line] Available at http://www.scu.edu.au/schools/gcm/ar/arp/grounded.html

Dix, J (2006) Has Open Source Changed the Game? *Network World*, Vol. 23 Issue 16, p50.

Downes, L & Mui, C (2000) Unleashing the Killer App: Digital Strategies for Market Dominance. *Harvard Business School Press*, pp35-39.

Downey, G (2001) Oracle Chief Slams System Integrators. *Computer Dealer News*, Vol. 17 Issue 6, p12.

Dun & Bradstreet (2006) Business Who's Who, Advanced Search, Dun & Bradstreet Companies Database, [SIC Code = 7372 {Prepackaged Software}] [On line] Available at http://bww.dnb.com.au.simsrad.net.ocs.mq.edu.au/AdvancedSearch.asp

Dver, A (2003) *Software Product Management Essentials*. Anclote Press.

Ein-Dor, P & Segev, E (1993) A Classification of Information Systems Analysis and Interpretation. *Information Systems Research*, Vol. 4 Issue 2, pp166-204.

Eisenhardt, K (1989) Building Theories from Case Study Research, *Academy of Management Review*, Vol. 14 Issue 4, pp532-550.

Eliashberg, J & Robertson, TS (1998) New Production Preannouncing Behavior: A Market Signaling Study. *Journal of Marketing Research*, Vol. 25 Issue 3, pp282-292.

Ellison, G & Fudenberg, D (2000) The Neo-Luddite's Lament: Excessive Upgrades in the Software Industry. *RAND Journal of Economics*, Vol. 31 Issue 2, pp253-272.

Esposito, T (2006) Professional Services – A Strategic Weapon for Product Focused Firms. Sandhill Business Strategy for Software Executives, *Sandhill.com website*, [On line] Available at: http://www.sandhill.com/opinion/daily_blog.php?id=25

Faletra, R (2004) Get Ready For Fast And Furious Consolidation In Software In 2005. *CRN*, Issue 1126, p94.

Farrell, J & Saloner, G (1985) Standardization, Compatibility, and Innovation. *RAND Journal of Economics*, Vol. 16 Issue 1, pp70-83.

Feeny, DF & Ives, B (1990) In Search of Sustainability: Reaping Long-term Advantage from Investments in Information Technology. *Journal of Management Information Systems*, Vol. 7 Issue 1, pp27-46.

Fichman, RG & Kemerer, CF (1999) The Illusory Diffusion of Innovation: An Examination of Assimilation Gaps. *Information Systems Research*, Vol. 10 Issue 3, pp255-275.

Fiegenbaum, A & Howard, T (1995) Strategic Groups as Reference Groups: Theory, Modeling and Empirical Examination of Industry and Competitive Strategy. *Strategic Management Journal*, Vol. 16 Issue 6, pp461-476.

Flick, U (2002) *An Introduction to Qualitative Research.* London, Sage Publications.

Fontana, A & Frey, JH (2000) The Interview: From Structured Questions to Negotiated Text. In NK Denzin & YS Lincoln (Eds), *Handbook of Qualitative Research*, 2nd Edition, (pp645-672). Thousand Oaks, Sage Publications.

Fontana, J (2005) Shopping for Software Maintenance. *Network World*, Vol. 22 Issue 17, p74.

Foster, E (1997) Vaporware is More Than Just Annoying: It Can Be Downright Expensive. *InfoWorld*, Vol. 19 Issue 48, p90.

Fudenberg, D & Tirole, J (1998) Upgrades, Tradeins, and Buybacks. *RAND Journal of Economics*, Vol. 29 Issue 2, pp235-258.

Fuller, A; Croll, P & Garcia, O (2001) Why Software is Riskier than Ever. *2nd Asia Pacific Conference on Quality Software* (APAQS 2001), Hong Kong, pp113-119.

Gable, GG; Taizan, C & Wui-Gee, T (2001) Large Packaged Application Software Maintenance: A Research Framework. *Journal of Software Maintenance & Evolution: Research & Practice*, Vol. 13 Issue 6, pp351-371.

Galbraith, C & Schendel, D (1983) An Empirical Analysis of Strategy Types. *Strategic Management Journal*, Vol. 4 Issue 2, pp153-173.

Gallaugher, JM & Wang, YM (2002) Understanding Network Effects in Software Markets: Evidence from Web Server Pricing. *MIS Quarterly*, Vol. 26 Issue 4, pp303-327.

Gartner (2005) Gartner Highlights Key Emerging Technologies in 2005 Hype Cycle. *Gartner website*, [On line] Available at: http://www.gartner.com/press_releases/asset_134460_11.html

Geisman, J (2005a) Software Industry Turmoil: Will Short-Term Licenses Lead to Long-Term Headaches or Opportunities? The Pricing Advisor, June 2005, *Professional Pricing Society website*, [On line] Available at: http://www.pricingsociety.com/articles/softwareindustryturmoil.htm

Geisman, J (2006a) Licensing Options – The Next Segmentation Weapon? Software 2006 Conference, Santa Clara California, USA, *Sandhill website*, [On line] Available at: http://www.sandhill.com/conferences/sw2006.php

Geisman, J (2006c) Avoiding the Perils of Pricing. *MarketShare Inc*. SoftwarePricing.com website, [On line] Available at: http://www.softwarepricing.com/readingroom/Articles/perils-of-pricing.cfm

Geisman, J (2006d) Licensing and Pricing. New Segmentation Weapons? *Marketshare Inc*, Software 2006 Conference, Software Pricing website, [On line] Available at: http://www.softwarepricing.com

Ghemawat, P & Ghadar, F (2000) The Dubious Logic of Global Megamergers. *Harvard Business Review*, Vol. 78 Issue 4, pp65-72.

Glaser, BG & Strauss, AL (1967) *The Discovery of Grounded Theory: Strategies for Qualitative Research.* Chicago, Aldine.

Glaser, BG (1978) *Theoretical Sensitivity: Advances in the Methodology of Grounded Theory.* Mill Valley, CA, Sociology Press.

Glaser, BG (1992) *Basics of Grounded Theory Analysis: Emergence vs Forcing.* Mill Valley, CA, Sociology Press.

Glaser, BG (1998) *Doing Grounded Theory. Issues and Discussions.* Mill Valley, CA, Sociology Press.

Glass, RL (1998) Maintenance: Less is Not More. *IEEE Software*, Vol. 15 Issue 4, pp67-68.

Glaister, KW & Falshaw, JR (1999) Strategic Planning: Still Going Strong? *Long Range Planning*, Vol. 32 Issue 1, pp107-116.

Glesne, C (1999) Becoming Qualitative Researchers: An Introduction, 2nd Edition. New York, Longman.

Gaoa, LS & Iyer, B (2009) Value Creation using Alliances within the Software Industry. *Electronic Commerce Research and Applications*, Vol. 8 Issue 6, pp280-290.

Goodwin, NC (1987) Functionality and Usability. *Communications of the ACM*, Vol. 30 Issue 3, pp229-233.

Goold, M & Campbell, A (1998) Desperately Seeking Synergy. *Harvard Business Review*, Vol. 76 Issue 5, pp131-143.

Gosain, S; Lee, Z & Im, I (1997) Topics of Interest in IS: Comparing Academic Journals with the Practitioner Press. *International Conference on Information Systems*, Atlanta, GA.

Goulding, C (2002) *Grounded Theory: A Practical Guide for Management, Business and Market Researchers.* London, Sage Publications.

Graham, BJ (2000) Is the Price Right? *Software Magazine*, Vol. 20 Issue 4, pp14-16.

Groenewald, T (2004) A Phenomenological Research Design Illustrated, *International Journal of Qualitative Methods*, Vol. 3 Issue 1, pp1-26.

Hadley, B (2003) How to Beat the Discount Heat. *SoftwareCEO*, [On line] Available at: http://www.softwareceo.com/attachments/softwarepricing/com062403.php

Hagel III, J & Brown, JS (2001) Your Next IT Strategy. *Harvard Business Review*, Vol. 79 Issue 9, pp105-113.

Haider, A; Zahid, Q & Wasim, N (2009) The Myth of the Technical Manager. *2nd IEEE International Conference on Computer Science and Information Technology*, Beijing, China, iccsit pp255-258.

Hall, R (1992) The Strategic Analysis of Intangible Resources. *Strategic Management Journal*, Vol.13 Issue 2, pp135-144.

Hamel, G (1999) Bringing Silicon Valley Inside. *Harvard Business Review*, Vol. 77 Issue 5, pp70-84.

Harding, D & Yale, P (2002) Discipline and the Dilutive Deal. *Harvard Business Review*, Vol. 80 Issue 7, pp18-19.

Harter, DE & Slaughter, SA (2000) Process Maturity and Software Quality: A Field Study. *International Conference on Information Systems*, Brisbane, QLD, pp407-411.

Hasted, E (2005) *Software That Sells: A Practical Guide to Developing and Marketing Your Software Project*. Wiley.

Hedley, B (1976) A Fundamental Approach to Strategy Development. *Long Range Planning*, Vol. 9 Issue 6, pp2-11.

Herbert, TT & Deresky, H (1987) Generic Strategies: An Empirical Investigation of Typology Validity and Strategy Content. *Strategic Management Journal*, Vol. 8 Issue 2, pp135-147.

Herbig, P & Milewicz, J (1995) The Relationship of Reputation and Credibility to Brand Success. *Journal of Consumer Marketing*, Vol. 12 Issue 4, pp5-10.

Herbig, P & Milewicz, J (1996) Market Signalling: A Review. *Management Decision*, Vol. 34 Issue 1, pp35-45.

Herbsleb, J; Zubrow, D; Goldenson, D; Hayes, W & Paulk, M (1997) Software Quality and the Capability Maturity Model. *Communications of the ACM*, Vol. 40 Issue 6, pp30-40.

Hildebrandt, KC (1999) Market Dominance and Innovation in Computer Software Markets. *Unpublished*, University of Calgary, Proquest Digital Dissertations, Publication Number: AAT MQ47948.

Hirschheim, R & Klein, HK (2003) Crisis in the IS Field? A Critical Reflection on the State of the Discipline. *Journal of the Association for Information Systems*, Vol. 4, pp237-293.

Hogan, T (2006) Time for Software Marketing to Grow Up. *Sandhill.com website*, 29/6/06, [On line] Available at http://www.sandhill.com/opinion/editorial.php?id=79

Honjo, Y (2000) Business Failure of New Software Firms. *Applied Economics Letters*, Vol. 7 Issue 9, pp575-579.

Hopper, MD (1990) Rattling SABRE – New Ways to Compete on Information. *Harvard Business Review*, Vol. 68 Issue 3, pp118-115.

Houser, KD (1999) Protecting Software as Intellectual Property: A Checklist of Options. *Information Strategy: The Executive's Journal*, Vol. 15 Issue 4, pp44-45.

Howe, K & Eisenhardt, M (1990) Standards for Qualitative (and Quantitative) Research. A Prolegomenon. *Education Researcher*, Vol.19 Issue 4, pp2-9.

Hoxmeier, JA (2000) Software Preannouncements and Their Impact on Customers' Perceptions and Vendor Reputation. *Journal of Management Information Systems*, Vol. 17 Issue 1, pp115-139.

Hui, KL & Tam, KY (2002) Software Functionality: A Game Theoretic Analysis. *Journal of Management Information Systems*, Vol. 19 Issue 1, pp151-184.

Hunt, J (2010) Software Companies Still Fail to Tap into Top Benefits from Market and Competitive Intelligence Programs. *Julie Hunt Consulting website*, Published 01/12/2010, [On line] Available at: http://jhcblog.juliehuntconsulting.com/2010/01/software-companies-still-fail-to-tap-into-top-benefits-from-market-and-competitive-intelligence-prog.html

Hussey, DE (1968) The Corporate Appraisal. Assessing Company Strengths and Weaknesses. *Long Range Planning*, Vol. 1 Issue 2, pp19-25.

IBM (2004) IBM Annual Report 2004. *IBM website, IBM Investor Relations* [On line] Available at: http://www.ibm.com/annualreport/

IBM (2006b) PartnerWorld, the Program for IBM Business Partners. *IBM website*, [On line] Available at: www.ibm.com/partnerworld/

IDC (2006) Analysing the Future. *IDC website*, [On line] Available at: http://www.idc.com/research/reshome.jsp;jsessionid=DUNUSD34KLXHSCQJAFDCFEYKBEAVAIWD

Igel, B & Islam, N (2001) Strategies for Service and Market Development of Entrepreneurial Software Designing Firms. *Technovation*, Vol. 21 Issue 3, pp157-166.

Jensen, R & Johnson, R (1999) The Enterprise Resource Planning System as a Strategic Solution. *Information Strategy*, Vol. 15 Issue 4, pp28-33.

Jones, C (1994) Globalization of Software Supply and Demand. *IEEE Software*, Vol. 11 Issue 6, pp17-24.

Kaplan, J (2004) Redefining Maintenance. *Network World*, Vol. 21 Issue 20, p45.

Kaplan, RS & Norton, DP (2004) *Strategy Maps: Converting Intangible Assets into Tangible Outcomes*, Harvard Business Press.

Karahanna, E; Straub, DW & Chervany, NL (1999) Information Technology Adoption Across Time: A Cross-Sectional Comparison of Pre-Adoption and Post-Adoption Beliefs. *MIS Quarterly*, Vol. 23 Issue 2, pp183-213.

Katz, ML & Shapiro, C (1985) Network Externalities, Competition, and Compatibility. *American Economic Review*, Vol. 75 Issue 3, pp424-440.

Kekre, S & Krishnan, MS (1995) Drivers of Customer Satisfaction for Software Products: Implications for Design. *Management Science*, Vol. 41 Issue 9, pp1456-1470.

Keller, E (2004b) Looking for Growth in all the Wrong Places. *Managing Business Technology*, September 2004 Issue, [On line] Available at: http://www.mbtmag.com/current_issues/2004/sept/corp12.asp

Keller, E (2005b) Launching a Software Pricing Revolution. *Sandhill.com website*, 12/12/2005, [On line] Available at: http://www.sandhill.com/opinion/editorial.php?id=58

Keller, E (2005c) Enterprise Software: Still in the Stone Age. *Manufacturing Business Technology*, April 2005, [On line] Available at: http://www.mbtmag.com/current_issues/2005/apr/col2.asp

Keller, E (2005d) The New Game in Software: Applistructure, *Sandhill.com website*, 25/5/05, [On line] Available at: http://www.sandhill.com/opinion/editorial.php?id=18

Keller, E (2006a) Pushing a "Pull Model". *Sandhill.com website*, 06/04/2006, [On line] Available at: http://www.sandhill.com/opinion/editorial.php?id=76

Keller, E (2006b) SAP's SAPPHIRE '06: We Will Neuter You. *Sandhill.com website*, 19/5/06, [On line] Available at: http://www.sandhill.com/opinion/daily_blog.php?id=42&post=153

Kerstetter, J & Hamm, S (2004) The Soft Underbelly of Software Deals. *Business Week*, Issue 3914, p48.

Kessler, A (2003) Monty Python M&A. *American Spectator*, Vol. 36 Issue 4, pp28-29.

Khoshgoftaar, TM; Allen, EB; Naik, A; Jones, WD & Hudepohl, JP (1999) Using Classification Trees for Software Quality Models: Lessons Learned. *International*

Journal of Software Engineering & Knowledge Engineering, Vol. 9 Issue 2, pp217-231.

Kim, KK (1989) User Satisfaction: A Synthesis of Three Different Perspectives. *Journal of Information Systems*, Vol. 4 Issue 1, pp1-12.

Kim, N; Chang, DR; Shocker & Allan, D (2000) Modeling Intercategory and Generational Dynamics for A Growing Information Technology Industry. *Management Science*, Vol. 46 Issue 4, pp496-522.

Kim, WC & Mauborgne, R (2000) Knowing a Winning Business Idea When You See One. *Harvard Business Review*, Vol. 78 Issue 5, pp129-138.

King, JL; Gurbaxani, V; Kraemer, KL; McFarlan, FW; Raman, KS & Yap, CS (1994) Institutional Factors in Information Technology Innovation. *Information Systems Research*, Vol. 5 Issue 2, pp139-169.

King, JL & Lyytinen, K (2004) Reach and Grasp. *MIS Quarterly*, Vol. 28 Issue 4, pp539-551.

Kirk, J (2006) Microsoft to Bolster Software Assurance Program. IDG News Service, *IDG News Service website*, 20/1/06, [On line] Available at http://smallbusiness.itworld.com/4394/060120softasur/page_1.html

Klemperer, P (1987) Markets with Consumer Switching Costs. *Quarterly Journal of Economics*, Vol. 102 Issue 2, pp375-394.

Krill, P (2006) Apache Chairman: Days Numbered for Commercial Software. *Computerworld*, 23/3/06, [On line] Available at: http://www.computerworld.com.au/index.php?id=840765857&eid=-255

Krishnan, MS; Mukhopadhyay, T & Kriebel, CH (2004) A Decision Model for Software Maintenance. *Information Systems Research*, Vol. 15 Issue 4, pp396-412.

Krishnan, V & Ulrich, KT (2001) Product Development Decisions: A Review of the Literature. *Management Science*, Vol. 47 Issue 1, pp1-21.

Langley, A (1999) Strategies for Theorizing From Process Data. *The Academy of Management Review*, Vol. 24 Issue 4, pp691-710.

Lay, P (2005) Reversing the Dismal Track Record of M&A in High-Tech. *TCG Advisers website*, [On line] Available at: http://www.tcg-advisors.com/Library/utb/ub_vol6_no5.pdf

Lee, BT & Barua, A (1997) Discovery and Representation of Casual Relationships in MIS Research: A Methodological Framework. *MIS Quarterly*, Vol. 21.

Lientz BP & Swanson EB (1980) *Software Maintenance Management: A Study of the Maintenance of Computer Application Software in 487 Data Processing Organizations*. Reading, MA, Addison Wesley Longman.

Lincoln, YS (1995) Emerging Criteria for quality in Qualitative and Interpretive Research. *Qualitative Inquiry*, Vol. 1 Issue 3, pp275-289.

Linderholm, O & Sanborn, S (2001) Joining forces. *InfoWorld*, Vol. 23 Issue 22, pp56-58.

Locke, K (2001) Grounded Theory's Research Practices. In *Grounded Theory in Management Research* (pp44-62). London, Sage Publications.

Llewelyn, S (2003) What Counts as "Theory" in Qualitative Management and Accounting Research? Introducing Five Levels of Theorizing. *Accounting, Auditing & Accountability Journal*, Vol. 16 Issue 4, pp662-708.

Lyytinen, K (1987) A Taxonomic Perspective of Information Systems Development: Theoretical Constructs and Recommendations. In R Boland & R Hirschheim (Eds), *Critical Issues in Information Systems Research* (pp3-41). Chichester, Wiley.

Lyytinen, K & Rose, GM (2003) The Disruptive Nature of Information Technology Innovations: The Case of Internet Computing in Systems Development Organisations. *MIS Quarterly*, Vol. 27 Issue 4, pp557-595.

Mabert, VA; Soni, A & Venkataramanan, MA (2001) Enterprise Resource Planning: Common Myths Versus Evolving Reality. *Business Horizons*, Vol. 44 Issue 3, pp69-76.

Malone, TW; Yates, J & Benjamin, RI (1989) The Logic of Electronic Markets. *Harvard Business Review*, Vol. 67 Issue 3, pp166-169.

MarketShare (1998) Excerpt from "The Vendor's Guide to Software Licensing and Pricing", *MarketShare publication*, [On line] Available at: http://www.softwarepricing.com/onlinestore/Publications/vendor%27s-guide-intro.cfm

Mason, RO & Mitroff, II (1973) A Program for Research on Management Information Systems. *Management Science*, Vol. 19 Issue 5, pp475-487.

Mason, RO; Mckenney, JL & Copeland, DG (1997) Developing an historical tradition in MIS research. *MIS Quarterly*, Vol. 21 Issue 3, pp257-278.

Mata, FJ; Fuerst, WL & Barney, JB (1995) Information Technology and Sustained Competitive Advantage: A Resource-based Analysis. *MIS Quarterly*, Vol. 19 Issue 4, pp487-505.

Matheson, LR & Tarjan, RE (1998) Culturally Induced Information Impactedness: A Prescription for Failure in Software Ventures. *Journal of Management Information Systems*, Vol. 15 Issue 2, pp23-39.

McCallin, A (2003) Grappling with the Literature in a Grounded Theory Study. *Contemporary Nurse*, Vol.15 Issue 1-2, pp61-69.

McCubbrey, DJ (2003) The IS Core-IV: IS Research: A Third Way. *Communications of AIS*, Vol. 2003 Issue 12, pp553-556.

McKinnon, J (1998) Reliability and Validity in Field Research: Some Strategies and Tactics. *Accounting Auditing and Accountability Journal*, Vol. 1 Issue 1, pp34-54.

Meyers, CB & Oberndorf, P (2001) Managing Software Acquisition: Open Systems and COTS Products. *Addison Wesley Professional*. 1st Edition.

Microsoft (2005) Microsoft Corporation Annual Report 2005, *Microsoft Corporation*, [On line] Available at http://www.microsoft.com/msft/ar.mspx

Miles, M & Huberman, A (1994) *Qualitative Data Analysis*. London, Sage Publications.

Miller, E (1999) Systems Integrators Target PDM. *Computer-Aided Engineering*, Vol. 18 Issue 6, p58.

Mills, S (2005) In the Shadow of Software's Titans. *Business Week Online*, 3/17/2005.

Mirel, B & Olsen, LA (1998) Social and Cognitive Effects of Professional Communication on Software Usability. *Technical Communication Quarterly*, Vol. 7 Issue 2, pp197-221.

Montalbano, E (2005) Salesforce.com Pushes AppExchange for Partners. *Computerworld*, 16/12/05, [On line] Available at: http://www.computerworld.com.au/index.php?id=506033649&eid=-180

Morris, CR & Ferguson, CH (1993) How Architecture Wins Technology Wars. *Harvard Business Review*, Vol. 71 Issue 2, pp86-96.

Mullin, R (2001) Software Vendors Target the Front Office. *Chemical Week*, Vol. 163 Issue 31, pp23-24.

Munson, JC (1998) Software Lives too Long. *IEEE Software*, Vol. 15 Issue 4, pp18-19.

Murphy, V (2004) Buyer's Remorse? *Forbes*, Vol. 174 Issue 8, p58.

Nault, BR & Vandenbosch, MB (2000) Research Report: Disruptive Technologies – Explaining Entry in Next Generation Information Technology Markets. *Information Systems Research*, Vol. 11 Issue 3, pp304-319.

Negroponte, N (2004) Hack out the Useless Extras. *New Scientist*, Vol. 182 Issue 4520, p26.

Nelson, RR (1991) Why Do Firms Differ, and How Does it Matter? *Strategic Management Journal*, Vol. 12 Special Issue: Fundamental Research Issues in Strategy and Economics, pp61-74.

Nidumolu, SR & Knotts, GW (1998) The Effects of Customizability and Reusability on Perceived Process and Competitive Performance. *MIS Quarterly*, Vol. 22 Issue 2, pp105-137.

Nickerson, J (1981) Why Interactive Computer Systems are Sometimes not Used by the People Who Might Benefit from Them. *International Journal of Man-Machine Studies*, Vol. 15 Issue 4, pp469-483.

Organisation for Economic Co-operation and Development (2002) OECD Information Technology Outlook 2002.

Organisation for Economic Co-operation and Development (2008) OECD Information Technology Outlook 2008.

Oracle (2005) Oracle Annual Report 2005, Fiscal Year 2005, Form 10-K, *Oracle Corporation website*, [On line] Available at http://www.oracle.com/corporate/investor_relations/index.html

Oren, SS & Smith, SA (1981) Critical Mass and Tariff Structure in Electronic Communications Markets. *Bell Journal of Economics*, Vol. 12 Issue 2, pp467-487.

Orton, JD (1998) From Inductive to Iterative Grounded Theory: Zipping the Gap between Process Theory and Process Data. *Scandinavian Journal of Management*, Vol. 13 Issue 4, pp419-438.

Orlikowski, WJ & Iacono, SC (2001) Research Commentary: Desperately Seeking the 'IT' in IT Research – A Call to Theorizing the IT Artifact. *Information Systems Research*, Vol. 12 Issue 2, pp121-134.

Parker, LD & Roffey, BH (1997) Back to the Drawing Board: Revisiting Grounded Theory and the Everyday Accountant's and Manager's Reality, *Accounting Auditing and Accountability Journal*, Vol.10 Issue 2, pp212-247.

Patin, A (2002) Software Maintenance Agreements. *Network World*, Vol. 19 Issue 7, p44.

Patton, MQ (1990) *Qualitative Evaluation and Research Methods*, 2nd Edition. Newbury Park, California, Sage Publications.

Paulk, MC; Curtis, B; Chrissis, MB & Weber, CV (1993) Capability Maturity Model, Version 1.1. *IEEE Software*, Vol. 10 Issue 4, pp18-27.

Phan, DD (2001) Software Quality and Management. *Information Systems Management*, Vol. 18 Issue 1, pp56-67.

Pollack, J (1999) Durable Goods and Secondary Markets. *Jordan Pollack website*, [On line] Available at: http://www.jordanpollack.com/softwaremarket/secmark.html

Porter, ME (1985) *Competitive Advantage: Creating and Sustaining Superior Performance*. The Free Press.

Porter, ME (1980) *Competitive Strategy: Techniques for Analyzing Industries and Competitors*. The Free Press.

Porter, ME (1991) Towards a Dynamic Theory of Strategy. *Strategic Management Journal*, Special Issue: Fundamental Research Issues in Strategy and Economics, Vol. 12 Issue 8, pp95-117.

Porter, ME (1996) What is Strategy? *Harvard Business Review*, Vol. 74 Issue 6, pp61-78.

Postmusa, D; Wijngaardb, J & Wortmann, H (2009) An Economic Model to Compare the Profitability of Pay-Per-Use and Fixed-Fee Licensing. *Information and Software Technology*, Vol. 51 Issue 3, pp581-588.

Prahalad, CK (2006) Innovation through Co-Creation. *SandHill.com*, Published 21/07/06, Available at: http://www.sandhill.com/opinion/editorial.php?id=92

Pulkkinen, M & Forsell, M (2007) Managing a Software Development Organization with a TQM Approach for Balance in a Period of Rapid Growth, *Advances in Information Systems Development.* Springer, US.

Prasad, RM & Prasad, SB (2002) Is the Enterprise Software Sector Still in Transition? A Research Note. By: *Technovation*, Vol. 22 Issue 12, pp769-674.

Rathnam, RG; Johnsen, J & Wen, HJ (2004) Alignment of Business Strategy and IT Strategy: A Case Study of a Fortune 50 Financial Services Company. *Journal of Computer Information Systems*, Vol. 45 Issue 2, pp1-8.

Raval, V (1999) Seven Secrets of Successful Offshore Software Development. *Information Strategy: The Executive's Journal*, Vol. 15 Issue 4, pp34-39.

Raynor, ME (1998) That Vision Thing: Do We Need It? *Long Range Planning*, Vol. 31 Issue 3, pp368-376.

Roeding, CR; Purkert, G; Kindner, SK; Muller, R & Hoch, DJ (1999) *Secrets of Software Success: Management Insights from 100 Software Firms Around the World*. Harvard Business School Press.

Rohlfs, J (1974) A Theory of Interdependent Demand for a Communications Service. *Bell Journal of Economics & Management Science*, Vol. 5 Issue 1, pp16-37.

Rosenbush, S (2004) Software Makers: Soon They'll Be Fewer. *Business Week Online*, 7/9/2004, ppN.PAG.

Rowlands, BH (2005) Grounded in Practice: Using Interpretive Research to Build Theory. *Electronic Journal of Business Research Methods*, Vol. 3 Issue 1, pp81-92.

Rubenstein, B (2001) Microsoft's Strategy Stresses Intellectual Property Protection. *Corporate Legal Times*, Vol. 11 Issue 119-136.

Rubin, HJ & Rubin, IS (1995) *Qualitative Interviewing: The Art of Hearing Data*. Thousand Oaks, Sage Publications.

Ruokonen, M (2008) Market Orientation and Product Strategies in Small Internationalising Software Companies. *The Journal of High Technology Management Research*, Vol. 18 Issue 2, pp143-156.

SAP (2006a) SAP Business Solutions and Applications. *SAP website*, [On line] Available at: http://www.sap.com/solutions/index.epx

SAP (2006d) SAP Partners. *SAP website*, [On line] Available at: http://www.sap.com/partners/index.epx

SAP (2008) SAP Annual Report 2008, *SAP*, Waldorf, Germany, pp1-252.

Sadagopan, S. (2006) Enterprise Software & The Future Of Smaller Players. *Sandhill.com website*, 24/7/06, [On line] Available at: http://www.sandhill.com/opinion/daily_blog.php?id=44&post=181

Sandhill (2009) About SandHill.com. *Sandhill.com website*, [On line] Available at: http://www.sandhill.com/sandhillcom/index.php

Saran, C (2004) Badly-managed Offshore Software Development Costs Firms Millions. *Computer Weekly*, 15/6/2004, p5.

Schelp, M (2006) Software Licensing: Problems, Opportunities and a Winning Formula. Ventana Consulting, *SoftwarePricing website* [On line] Available at: http://www.softwarepricing.com/readingroom/Articles/Ventana.cfm

Schilling, MA & Hill, CWL (1998) Managing the New Product Development Process: Strategic Imperatives. *Academy of Management Executive*, Vol. 12 Issue 3, pp67-81.

Schofield, JW (2000) Increasing the Generalizability of Qualitative Research. In R. Gomm, M. Hammersley & P. Foster (Eds), *Case Study Method: Key Issues, Key Texts* (pp69-97). London, Sage Publications.

See Pui Ng, C; Gable, GG & Chan, T (2002) An ERP-client Benefit-oriented Maintenance Taxonomy. *Journal of Systems & Software*, Vol. 64 Issue 2, pp87-109.

Sainio, LM & Marjakoskia, E (2009) The Logic of Revenue Logic? Strategic and Operational Levels of Pricing in the Context of Software Business. *Technovation*, Vol. 29 Issue 5, pp368-378.

Shay, JP & Rothaermel, FT (1999) Dynamic Competitive Strategy: Towards a Multi-Perspective Conceptual Framework. *Long Range Planning*, Vol. 32 Issue 6, pp559-572.

Shapiro, C & Varian, HR (1998) Versioning: The Smart Way to Sell Information. *Harvard Business Review*, Vol. 76 Issue 6, pp106-114.

Shapiro, C & Varian, HR (1999) *Information Rules*. Boston, Harvard Business Press.

Sharif, N (1994) Integrating Business and Technology Strategies in Developing Countries. *Technological Forecasting and Social Change*, Vol. 45 Issue 2, pp152-165.

Sheth, JN & Sisodia, RS (2002) Competitive Markets and The Rule of Three. *Ivey Business Journal*, Vol. 67 Issue 1, pp1-5.

Silvers, GA (1992) Software Industry Merger Mania. *Information Systems Management*, Vol. 9 Issue 1, pp86-88.

Snow, CC & Hrebiniak, LG (1980) Strategy, Distinctive Competence, and Organizational Performance. *Administrative Science Quarterly,* Vol. 25 Issue 2, pp317-336.

SoftwarePricing.com (2006) A Software Pricing Primer. *SoftwarePricing.com website*, [On line] Available at: http://www.softwarepricing.com/readingroom/Articles/Prcg-Primer.cfm

Spiggle, S (1994) Analysis and Interpretation of Qualitative Data in Consumer Research. *The Journal of Consumer Research*, Vol. 21, Issue 3, pp491-503.

Susarla, A; Barua, A & Whinston, AB (2003) Understanding the Service Component of Application Service Provision: An Empirical Analysis of Satisfaction with ASP Services. *MIS Quarterly*, Vol. 27 Issue 1, pp91-123.

SIIA (2006) Key Trends in Software Pricing and Licensing. SoftSummit 2006 Conference, Santa Clara, CA. *SoftSummit website,* [On line] Available at: www.softsummit.com/surveyresult

Sink, E (2006) *Eric Sink on the Business of Software*. Apress.

Skaates, MA & Seppanen, V (2002) Managing Relationship-driven Competence Dynamics in Professional Service Organisations. *European Management Journal*, Vol. 20 Issue 4, pp430-437.

Smail, J (2004) The Business Case for Application Service Providers. Unpublished, [On line] Available at: http://faculty.ed.umuc.edu/~meinkej/inss690/smail/ASP.htm

SMS (2004) Five Things You Need to Do to Become a Software Product Company, *Software Market Solution website*, [On line] Available at: http://www.softwaremarketsolution.com/Becoming_a__software__product_/becoming_a__software__product_.htm

Software Equity Group LLC (2006) Software Industry Equity Report 2006 Q1. Software Equity Group LLC, *Software Equity Group's website,* [On line] Available at: http://www.SEGReports.com/reports/2006-Q1.pdf

Software Engineering Institute (2006) Capability Maturity Model® Integration (CMMI). *CMMI© website*, [On line] Available at http://www.sei.cmu.edu/cmmi/cmmi.html

Sommer, B (2006) 5 Paths to Software Growth. *Sandhill.com website*, [On line] Available at http://sandhill.com/opinion/editorial.php?id=82

SourceForge (2006) SourceForge.net: What is SourceForge.net? *SourceForge.net website*, [On line] Available at: http://sourceforge.net/docs/about

Sternberg, RJ (2000) Images of Mindfulness. *Journal of Social Issues*, Vol. 56 Issue 1, pp11-26.

Stewart, H; Hope, C & Muhlemann, A (1998) *Journal of Retailing and Consumer Services*, Vol. 5 Issue 4, pp209-222.

Stolpe, M (2000) Protection Against Software Piracy – A Study of Technology Adoption for the Enforcement of Intellectual Property Rights. *Economics of Innovation & New Technology*, Vol. 9 Issue 1, pp25-52.

Stone, M (1985) Strategies for Marketing New Computer Products. *Long Range Planning*, Vol. 18 Issue 3, pp41-54.

Strauss, A & Corbin, J (1990) Grounded Theory Research: Procedures, Canons, and Evaluative Criteria. *Qualitative Sociology*, Vol. 13 Issue 1, pp3-21.

Strauss, A & Corbin, J (1998) *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 2nd Edition. Thousand Oaks, CA, Sage Publications.

Suddaby, R (2006) From the Editors: What Grounded Theory is Not. *Academy of Management Journal*, Vol. 49 Issue 4, pp633-642.

Sundararajan, A (2004) Nonlinear Pricing of Information Goods. *Management Science*, Vol. 50 Issue 12, pp1660-1673.

Swanson, EB (1976) The Dimensions of Maintenance. *International Conference on Software Engineering, San Francisco* (pp492-495). Long Beach, CA, IEEE Computer Society.

Swanson, EB (1994) Information Systems Innovation Among Organizations. *Management Science*, Vol. 40 Issue 9, pp1069-1092.

Szajna, B (1994) Software Evaluation and Choice: Predictive Validation of the Technology Acceptance Instrument. *MIS Quarterly*, Vol. 18 Issue 3, pp319-324.

Tabrizi, B & Walleigh, R (1997) Defining Next-Generation Products: An Inside Look. *Harvard Business Review*, Vol. 75 Issue 6, pp116-124.

Tam, KY & Hui, KL (2001) A Choice Model for the Selection of Computer Vendors and Its Empirical Estimation. *Journal of Management Information Systems*, Vol. 17 Issue 4, pp97-124.

Taylor, S & Todd, P (1995) Assessing IT Usage: The Role of Prior Experience. *MIS Quarterly*, Vol. 19 Issue 4, pp561-570.

Teece, DJ; Pisano, G & Shuen, A (1997) Dynamic Capabilities and Strategic Management. *Strategic Management Journal*, Vol. 18 Issue 7, pp509-533.

Tellis, GJ & Merle CC (1981) An Evolutionary Approach to Product Growth Theory. *Journal of Marketing*, Vol. 45 Issue 4, pp125-132.

The Economist (2004) Software Takeovers. To Buy or Be Bought. *The Economist*, 29/12/04.

The Economist (2005a) Business Software. Softwar or Hard? *The Economist*, 23/03/05.

The Economist (2005b) To buy or be bought. *The Economist*, Vol. 374 Issue 8407, pp47-48.

Tian, J (1999) Measurement and Continuous Improvement of Software Reliability Throughout Software Life-Cycle. *Journal of Systems & Software*, Vol. 47 Issue 2-3, pp189-195.

Treacy, M & Wiersema, F (1993) Customer Intimacy and Other Value Disciplines. *Journal of Health Care Marketing*, Vol. 71 Issue 1. pp84-93.

Tuckett, A (2004) Qualitative Research Sampling: The Very Real Complexities. *Nurse Researcher*, Issue 1, pp47-61.

Turner, BA (1981) Some Practical Aspects of Qualitative Data Analysis: One Way of Organising the Cognitive Process Associated with the Generation of Grounded Theory. *Quality and Quantity*, Vol. 15 Issue 3, pp225-247.

Van Der Zee, Han (2002) Measuring the Value of Information Technology. PA, Idea Group Publishing.

VanDoren, V (2000) System Integrators Wear Many Hats. *Control Engineering*, Vol. 47 Issue 14, pp8-11.

Venkatraman, N & Prescott, JE (1990) Environment-Strategy Coalignment: An Empirical Test of Its Performance Implications. *Strategic Management Journal*, Vol. 11 Issue 1, pp1-23.

Verlage, M & Kiesgen, T (2005). Five Years of Product Line Engineering in a Small Company. *Proceedings of the 27th International Conference on Software Engineering*, pp534-543.

Vincent, PB (2006) Avoiding Software's Perfect Storm. *Sandhill.com website*, 14/7/06, [On line] Available at: http://www.sandhill.com/opinion/editorial.php?id=91

Voas, J (1998) COTS Software: The Economical Choice? *IEEE Software*, Vol. 15 Issue 2, pp16-19.

Vollmar, KR (2001) Toward Increased Public Confidence in Software Reliability. *Journal of Public Affairs*, Vol. 5, pp7-78.

Walsh, B (2006) *Micro-ISV: From Vision to Reality*. Apress.

Wernerfelt, B (1984) A Resource-based View of the Firm. *Strategic Management Journal*, Vol. 5 Issue 2, pp171-180.

Weber, R (1999) The Information Systems Discipline: The Need for and Nature of a Foundational Core. *Proceedings of the Information Systems Foundations Workshop. Ontology, Semiotics and Practice*. [On line] Available at: http://www.comp.mq.edu.au/isf99/Weber.htm

Weber, R (2003) Still Desperately Seeking the IT Artefact. *MIS Quarterly*, Vol. 27 Issue 2, ppiii-xi.

Weick, KE (1976) Educational Organizations as Loosely Coupled Systems. *Administrative Science Quarterly*, Vol. 21 Issue 1, pp1-19.

Weick, KE (1989) Theory Construction as Disciplined Imagination. *Academy of Management Review*, Vol. 14 Issue 4, pp516-531.

Wilson, I (1992) Realizing the Power of Strategic Vision. *Long Range Planning*, Vol. 25 Issue 5, pp18-28.

WenShin, C & Hirschheim, R (2001) A Paradigmatic and Methodological Examination of Information Systems Research From 1991 to 2001. *Information Systems Journal*, Vol. 14 Issue 3, pp197-235.

Wohl, AD (2004) Changing Software Pricing. *Amy D. Wohl's Opinions website*, 11/10/04, [On line] Available at: http://www.wohl.com/wa04-86.htm

Wuebker, G & Simon, H (2005) Price Bundling – A Powerful Strategy To Increase Profit. *Quarterly Journal of Professional Pricing*, Vol. 14 Issue 1, [On line] Available at: http://www.pricingsociety.com/articles/pricebundlinga.htm

Xiaotong, L (2004) Informational Cascades in IT Adoption. *Communications of the ACM*, Vol. 47 Issue 4, pp93-97.

Yin, RK (1984) *Case Study Research: Design & Methods*. Beverley Hills, CA, Sage Publications.

Yoffie, DB (1996) Competing in the Age of Digital Convergence. *California Management Review*, Vol. 38 Issue 4, pp31-53.

Yoffie, DB & Cusumano, MA (1999) Building a Company on Internet Time: Lessons from Netscape. *California Management Review*, Vol. 41 Issue 3, pp8-28.

Zatolyuk, S & Allgood, B (2004) Evaluating a Country for Offshore Outsourcing: Software Development Providers in the Ukraine. *Information Systems Management*, Vol. 21 Issue 3, pp28-33.

Zajac, EJ; Kraatz, MS & Bresser, RK F (2000) Modeling the Dynamics of Strategic Fit: A Normative Approach to Strategic Change. *Strategic Management Journal*, Vol. 21 Issue 4, pp429-453.

Zvegintzov, N (1998) Software Should Live Longer. *IEEE Software*, Vol. 15 Issue 4, pp19-20.