2010

SOFTWARE BUSINESS IN INDUSTRIAL COMPANIES: IDENTIFYING CAPABILITIES FOR THREE TYPES OF SOFTWARE BUSINESS

Karin Väyrynen

University of Oulu, karin.vayrynen@oulu.fi

SOFTWARE BUSINESS IN INDUSTRIAL COMPANIES: IDENTIFYING CAPABILITIES FOR THREE TYPES OF SOFTWARE BUSINESS

Completed Research Paper

Karin Väyrynen
University of Oulu, Finland
PL 3000, 90014 Oulun yliopisto
karin.vayrynen@oulu.fi

Abstract

The development of organizational capabilities takes time and resources, and they are difficult to change once they have been created. Recently, also non-software companies – so-called industrial companies – have started to sell software and related services. So far it has been unclear which capabilities are needed in, and whether they differ for, different types of software business in industrial companies. The present research closes this gap and identifies 20 software business capabilities by studying the capabilities relevant in three types of software business in industrial companies. Of these capabilities, some proved to be relevant for all types of software business, some for software product and/or enterprise solution system business, and some for software service business. Based on our findings, industrial companies could improve their competitive position by directing their resources at the development or improvement of specifically those capabilities that proved to be relevant for a certain type of software business.

Keywords: Organization-level analysis, resource-based theory, business value of IS/value of IS, track 17, organization theory, strategy and IS, case study/studies, qualitative research

Introduction

Information technology (IT), including computer hardware and software, has become a cornerstone of companies' operations over the last four decades. Two roles that IT systems and software can have in companies can be distinguished. In the first case, IT including software is used to support and improve the company's internal operations (Bakos and Treacy 1986) and in most cases thus represents a cost centre for the company. In the second case, IT and software represent the core business of the company and are a means to generate profits. The latter type of software business is mainly practiced by software companies within the software industry and has been researched extensively (for example, see Käkölä 2003). Recently, a new development can be observed: that of software business being started and practiced by companies outside the software industry (Tyrväinen et al. 2008). Xerox, for example, operates outside the software industry, but established 35 technology spin-offs from 1979-1998. Seven of these spin-offs were pure software companies – one of them the today very successful company Adobe – and eleven offered software combined with hardware and/or networks. (Chesbrough 2003)

Primary software companies are generally defined as companies belonging to NACE (Nomenclature for economic activities) class 62, 'Computer programming, consultancy and related services' or class 63.11 'Data processing, hosting and related services'. In the study at hand, an industrial company, as opposed to a primary software company, will be defined as "a company whose core business is not classified as NACE class 62 or 63.11".

According to Tyrväinen et al. (2008: 382), "questions regarding the extent to which secondary software companies shift to primary software businesses, the industries from which they came, and why and how this happens are still valid". Their definition of secondary software companies corresponds to industrial companies doing software business. Tyrväinen et al. (2008) research this shift from secondary to primary software industry on an industry level. However, they do not answer how an industrial company's involvement with software business happens on an operational or business level. This clearly represents a research gap.

For an industrial company, software business represents a new line of business. Chakravarthy and Lorange (2007) propose two ways in which a company can arrive at a new line of business: either by leveraging existing resources and capabilities in a new market, or by building new capabilities. Capabilities are difficult to change once they have been created. Previous research has identified capabilities important for the development of software (for examples see Butler and Murphy 2003; Ethiraj et al. 2005; Sallinen 2002), capabilities important for conducting business in general (for examples see Day 1994; Grant 1996; Morgan et al. 2009; Peteraf and Bergen 2003; Teece 2007), and to some extent capabilities important in the context of software business (for examples see Ethiraj et al. 2005; Sallinen 2002). However, previous research neither studied which of these capabilities are relevant for industrial companies when starting to do software business, nor whether the capabilities needed for different types of software business in industrial companies differ from each other. For an industrial company that starts to do software business, this information would be essential, as it would allow the company to allocate its time and resources to leveraging and building exactly those capabilities that are relevant for a certain type of software business, thus improving its competitive position. The present research addresses this issue of how an industrial company should direct its efforts when starting software business by examining which capabilities are relevant for industrial companies in the three traditional types of software business: software product business, enterprise solution system business, and software service business. This research thus focuses on the capabilities needed for software business that was started by industrial companies.

The remainder of this paper is organized as follows. First, the empirical context of this research, three types of software business, is presented. Following that, previous research on the capability theory is reviewed and different capabilities important for software development and business are identified. These capabilities will then be focused on when analyzing the empirical data. Next, the research methodology is described, followed by a presentation of the results of the qualitative empirical data analysis. Finally, the main findings of this study, as well as the implications for practice and research are discussed. The paper concludes by summarizing the key findings.

Empirical Context: Software Business

Software business will be defined as in Tyrväinen et al. (2008) as 'business operations based on software being developed by the organization'. This includes business operations within enterprises classified as primary software companies, as well as similar business operations in software-oriented business units in the secondary software industry and in public administration.

According to previous research, two main types of software business are commonly distinguished: *software service business* – also referred to as software implementation or software project business – and *software product business* (Cusumano 2004: 25; Hoch et al. 1999; Tyrväinen et al. 2008). Software product business can be further classified into pure software products which are sold to the customer without making any changes to the software, and customized software which is adapted to a certain extent to the customer's specific needs. This customized software has been referred to as *enterprise solution systems* by Hoch et al. (1999), as *custom software and services* by Malerba and Torrisi (1996: 170), as *professional services* (including IT consulting, custom software, systems and network implementation, education and training, and facilities management) by Grindley (1996: 202) and as *hybrid solutions* by Cusumano (2004: 25). In **software service business**, software is developed specifically for one customer, usually in the form of a project. The software developed in software service business demands customization, user training, integration with other software systems, and maintenance services (Cusumano 2004: 26). Software products can be further classified into pure software products, and enterprise solution systems which refer to software that is adapted to a certain extent to the customer's specific needs. In **software product business**, a software program is sold to a large number of customers without customization or special installation support (Cusumano 2004: 26). **Enterprise solution systems**, in contrast to pure software products, require substantial time and effort to get the software running (Hoch et al. 1999: 36).

In the research at hand, these three main types of software business – pure software product business, enterprise solution system business, and software service business – will be distinguished and considered in the empirical data analysis.

Theory

Capability Approach

The organizational capability approach is an important stream within research on the resource-based view (RBV) of organizations (Barney 1991; Penrose 1959; Wernerfelt 1984). The RBV sees the firm as a bundle of resources – which include also capabilities – and explains the importance of resources for a company in gaining sustained competitive advantage (Barney 1991). However, later on researchers noticed that it makes sense to make a clear distinction between resources and capabilities. Amit and Shoemaker (1993) define capabilities, as opposed to resources, as firm-specific and not tradable on the market. According to them, capabilities refer to a firm's capacity to deploy resources using organizational processes to effect a desired end. Several researchers are of the opinion that the sustained competitive advantage of a company does not lie in its resources but in the company's capabilities, as these are more difficult to imitate and transfer than resources (for example, Amit and Shoemaker 1993).

The RBV has been applied extensively to the analysis of information systems and whether and how they can offer a company sustained competitive advantage (see, for example, Bharadwaj 2000; Clemons and Row 1991; Jarvenpaa and Leidner 1998; Mata et al. 1995). The capability approach, too, has been used to study and evaluate capabilities related to information systems (for example, Peppard et al. 2000; Santhanam and Hartono 2003; Weill and Vitale 2002) and software business (for example, Ethiraj et al. 2005; Sallinen 2002).

Recently, also the concept of dynamic capabilities – which according to Lopez (2005) are formed as a subgroup of firm capabilities – has been under the focus of research. Teece *et al.* (1997) define dynamic capabilities as "the firm's ability to integrate, build, and reconfigure internal and external competences to address rapidly changing environments" (p. 516). The term "dynamic" refers to the capacity to renew competences so as to achieve congruency with the changing business environment. Helfat and Peteraf (2003: 999) classify organizational capabilities, which consist of routines, into operational and dynamic capabilities, where operational capabilities have a direct effect on the firm's output, and dynamic capabilities only indirectly affect the firm's output through an impact on operational capabilities. According to them, dynamic capabilities "build, integrate or reconfigure operational capabilities".

Amit and Shoemaker (1993) argue that firms can have sustained economic rents based on the differences in resources and capabilities between companies. They state that the challenge for managers is to *identify, develop, protect and deploy* resources and capabilities in a way that provides the firm with a sustainable competitive advantage and hence with a superior return on capital. In order to achieve this in the context of software business in industrial companies, management has to know which capabilities the industrial company has to possess to conduct different types of software business. According to Prahalad and Hamel (1990: 81), competitiveness "derives from an ability to build, at lower cost and more speedily than competitors, the core competencies that spawn unanticipated products". Therefore, in the present research we focus on the organizational capabilities an industrial company needs in order to conduct software business. This knowledge should support industrial companies when considering which of its existing capabilities it can apply in a certain type of software business, or which capabilities it still has to develop. Organizational capabilities will be defined as in Helfat and Peteraf (2003: 999): "An organizational capability refers to the ability of an organization to perform a coordinated set of tasks, utilizing organizational resources, for the purpose of achieving a particular end result."

Next, we will discuss which capabilities have already been identified in previous research in connection to software development and business.

Software Development and Business Capabilities

Examples of capabilities related to software development and business can be found in previous research. Most studies focusing on capabilities have been conducted in the context of business in general, but several focus on capabilities in software companies or in the context of software development. In this section, a review of previous research on capabilities identified in the context of software development and (software) business is conducted. The capabilities are written in italic, bold letters.

The probably most basic capability in the context of software development and software business is the ***capability of developing software applications***. This capability has repeatedly been identified in research on the primary software industry (for example, Butler and Murphy 2003; Ethiraj et al. 2005; Sallinen 2002). Hobday et al. (2005) focus on a company's ***software integration capability*** and review the development of this capability over time in a wide range of industries. They find that this capability is no longer "merely" a technical task, but a core technical, strategic and organizational capability especially for complex products and systems and other high-technology goods. The ***support and maintenance of software and IT systems*** has been found to be tightly related to software development (see Butler and Murphy 2003). The ***capability of understanding user needs*** has been identified in previous research as being essential for gaining competitive advantage, both for software companies (for example, Ethiraj et al. 2005) and non-software companies (for example, Cockburn et al. 2000). Related to understanding the user needs is the ***requirement engineering capability,*** which has been discussed in previous research on software development and software business (Butler and Murphy 2003; Ethiraj et al. 2005). The ***project management capability*** has been discussed especially in the context of software development and software business (Butler and Murphy 2003; Ethiraj et al. 2005; Sallinen 2002). Related to that the ***capability of estimating the resources needed*** within a certain project – including effort and schedule estimation and management – has been identified as being essential for successfully conducting software projects (see, for example, Butler and Murphy 2003; Ethiraj et al. 2005).

A number of capabilities have been identified to contribute to a company's competitive advantage independent of the type of business. These capabilities, thus, can be expected to being important in the context of software business in industrial companies, too. The ***marketing and sales capability*** has been emphasized both in the context of software business (Sallinen 2002) and business in general (Grant 1996; Morgan et al. 2009). Morgan et al. (2009) differentiate between capabilities concerning individual "marketing mix" processes – including selling and marketing communications – and capabilities concerned with the processes of marketing strategy development and execution. ***The ability to estimate the potential of a customer segment***, including the cost structure and profit potential, has been discussed specifically in the context of business in general (for example, Teece 2007), as well as the ***capability of having control over the costs*** involved with some business (for example, Day 1994; Teece 2007). Day (1994) researches market-driven organizations and identifies ***anticipation of market development*** ahead of the competition, the ***identification of opportunities***, as well as ***pricing*** as important capabilities of this type of organizations. Also Morgan et al. (2009) identify pricing as a capability related to "marketing mix" processes. Markides and Williamson (1994) mention ***skills in building brands*** as an indicator of customer assets and argue that these skills can be transferred into another market where brand-building is equally important as it is in the market the company originates from. Petromilli et al. (2002) discuss brand building and brand architecture.

Table 1 summarizes these capabilities (C1-C7 and C11-C17), along with six capabilities identified in the present research. These newly identified capabilities (C8-C10 and C18-C20) are displayed in Table 1 for clarity and transparency only, but will be introduced in more detail in Section "Results". The capability code (C1-C20) displayed in the left column will be used to refer to the respective capabilities when presenting our findings in Section "Results".

Table 1. Software Development and Software Business Capabilities

Code	Capability and description	Related Research
Software Development Capabilities		
C1	**Software application development capability**: The ability to develop and code software applications.	Butler and Murphy (2003), Ethiraj et al. (2005), Sallinen (2002)
C2	**Software integration capability**: The ability to integrate a software application with other software applications and IT systems.	Hobday et al. (2005)
C3	**Software support and maintenance capability**: The ability to offer the service of providing support in the event of problems with the software, and the service of maintaining and updating the software to keep it functioning.	Butler and Murphy (2003)
C4	**Capability of understanding user needs**: The ability to determine clearly which need(s) the software application should fulfil for the users/customers.	Cockburn et al. (2000), Ethiraj et al. (2005)
C5	**Requirement engineering capability**: The employment of a systematic process of discussing with the customer and finding out about the features a software application should offer.	Butler and Murphy (2003), Ethiraj et al. (2005)
C6	**Software project management capability**: The ability to create an organized process defining how to manage software projects, including the definition of the project schedule and milestones, and the definition of document types and other items produced in the project.	Butler and Murphy (2003), Ethiraj et al. (2005), Sallinen (2002)
C7	**Capability of estimating the resources needed (for software development/software business)**: The ability to estimate the quantity of (human) resources and how much time will be needed for a certain task, for example, for the development of a software application or for offering certain services.	Butler and Murphy (2003), Ethiraj et al. (2005)
C8	**Software version management capability**: The ability to distinguish different versions of a software application and to ensure that it is clear which features and software modules are related to which version.	Present research
C9	**Software training capability**: The ability to offer training and to teach people how to use a software application.	Present research
C10	**Software documentation capability**: The ability to create a manual explaining which features and functionalities a software application has and how to use the software application.	Present research
Software Business Capabilities		
C11	**Software sales and marketing capability**: The ability to identify potential customers and make them interested in the software applications and services the company offers. Also includes the development of material that can be used to promote the company's offerings.	Grant (1996), Morgan et al. (2009), Sallinen (2002)
C12	**Capability of estimating customer segment size**: The ability to assess how many customers the company expects to gain for its offerings.	Teece (2007)
C13	**Cost control capability**: The ability to create and manage an up-to-date overview of the costs that are involved with developing a software application and offering services, and the ability to take corrective measures in the event costs become higher than anticipated.	Day (1994), Teece (2007)
C14	**Software branding capability**: The ability to offer a software application and/or software services under a specific name which enables customers to identify the company's offering in a group of similar offerings (opposite: white-label software).	Markides and Williamson (1994), Petromilli et al. (2002)
C15	**Capability of anticipating market development**: The ability to anticipate how customer numbers and the needs of customers will develop and change in future.	Day (1994)
C16	**Capability of identifying business opportunities**: The ability to identify possible new sources of future rents.	Day (1994)
C17	**Capability of pricing software and related services**: The ability to define the price that is to be asked of the customer for (the right to use) a software application or for using the company's software services.	Day (1994), Morgan et al. (2009)
C18	**Software business process creation capability**: Refers to the ability of managing the process of offering software and services to the customer.	Present research
C19	**Software productization capability**: The ability to develop or change a software application so that it can be sold more easily to a larger number of customers. This includes modularization of the software, interfaces that allow easier integration of the software at the customer site, and packaging of the software for delivery to the customer.	Present research
C20	**Capability of creating software contracts**: The ability to understand the issues that have to be addressed in a contract that defines the terms of sale of (the right to use) software applications and the terms when selling software services.	Present research

In summary, previous research has already identified and discussed capabilities related to software development and (software) business. Related to the actual development of software are the software application development capability, software integration capability, software support and maintenance capability, capability of understanding user needs, requirement engineering capability, project management capability and the capability of estimating the resources needed for software development and software business. Related to (software) business are the sales and marketing capability, capability of estimating the customer segment size, cost control capability, branding capability, capability of anticipating market development, capability of identifying business opportunities, and the pricing capability.

Research Methodology

Research Settings

Two industrial companies were studied in this research. Both case companies want to stay anonymous, and for that reason it is not possible to give detailed background information about the companies. The names of all companies and software applications in the present research are pseudonyms. Case company 1 will be called *Conserx*, case company 2 will be called *Halcinson* in this study. Neither company sells or develops software for customers as their core business nor is listed as a software company. Thus, both fulfill the definition of being an industrial company.

Case company 1, Conserx, is a Finnish company offering, among other services, consulting and planning services to the forest industry, and was founded in the 1950s. From the 1970s onwards, software has been developed within the company for firm-internal use in order to increase efficiency and improve internal work processes. Case company 2, Halcinson, is the subsidiary of a large Central European hardware producer and employs about 2000 people in Central Europe. The company has existed since the 1970s, at first as a department of the Central European hardware producer, but at the end of the 1990s it was established as a separate company. In Conserx two distinct software business cases could be identified, and in Halcinson three distinct software business cases could be identified, resulting in a total of five software business cases researched in this study.

Research Methods

The present research is conducted as a case study. According to Schramm (1971: 6), "the essence of a case study, the central tendency among all types of case study, is that it tries to illuminate a decision or set of decisions: why they were taken, how they were implemented, and with what result. "How" and "why" questions have an explanatory character and "deal with operational links needing to be traced over time, rather than mere frequencies or incidents" (Yin 1994: 6-7). The present research is concerned with finding out how an industrial company starts different types of software business by investigating which capabilities are relevant for these different types of software business. Thus, based on Schramm (1971) and Yin (1994), the case study offers a suitable research method for the current study. The case study was carried out as a qualitative, explanatory case study. The present study represents a comparative case study which is used to develop concepts based on case comparison (see Cunningham 1997).

Validity and Reliability of This Case Study

The concepts of validity and reliability have to be considered in qualitative research (for examples see Golafshani 2003; Kirk and Miller 1986). The findings of a study may be neither reliable nor valid, or may be reliable but not valid, or may be both reliable and valid, but they can never be valid without being reliable (Gorman and Clayton 1997). The present study follows the validity and reliability tactics for case studies as discussed by Yin (1994), who argues that four tests are commonly used in judging the quality of empirical qualitative research: construct validity, internal validity, external validity, and reliability.

In the study at hand, *construct validity* has been addressed by using multiple sources of evidence (see Section "Data Collection and Analysis), by establishing a chain of evidence, and by asking the interviewees to review the description of the software business cases and to point out mistakes and misinterpretations. *Internal validity* was addressed by explanation-building and conducting a high-level time-series analysis of the software business cases and of the development of software business in the industrial companies focused upon in this research. Patternmatching has been carried out by matching the empirical data of the research with the capabilities identified in previous research. *External validity* has been aimed at by conducting the research in two different industrial companies, which pursue different types of core business and operate in different countries. Interview questions for one case were adapted based on interesting findings in the other case. Finally, *reliability* has been aimed at by keeping research diaries over the whole period of the research process, and by using the NVivo software application when analyzing and coding the empirical data.

Data Collection and Analysis

The empirical data for this research was mainly collected by conducting interviews in the two case companies, and by collecting additional material related to these two case companies. Additional material included information from the companies' websites, press releases, and newspaper articles. We had access to an assessment of Halcinson's core knowledge (conducted by Halcinson), as well as several video interviews (conducted by Conserx with current and former employees for knowledge-prevalence reasons) describing Conserx's history and development of the "Virtual Planning" concept. Data collection in Conserx lasted from March 2006 to October 2008 and thus covers a period of almost two and a half years. During this period, six individuals were interviewed in ten interviews, amounting to more than 14 hours of interview recordings. Data collection in Halcinson lasted from July 2007 to December 2008 and thus covers a period of one and a half years. During this time, five individuals were interviewed in ten interviews, amounting to more than 18 hours of interview recordings. In Conserx, we interviewed three software developers that had been involved with the development of the ProPIDesign software (two of which were transferred to Aplec, a Conserx subsidiary), one member of the department the ProPIDesign software had been developed for in the first place, Conserx's group risk manager, and Aplec's CEO. In Halcinson, we interviewed two software developers (one of them had been responsible for the development of the enterprise solution system), the manager of the IT department, Telstanet's business development manager and Telstanet's software development manager. As interviewees' positions in both companies ranged from programmers who were actually involved in the development of the software dealt with in this research, to high-level managers, we could gain insights from a range of different viewpoints. In this research, we generalize from empirical statements to theoretical statements, thus "from description to theory" according to Lee and Baskerville's (2003) classification of generalizability in information systems research.

The data was analyzed using processual analysis (Pettigrew 1997), and we followed Eisenhardt's (1989) steps of building theory from case study research. These steps include the overlap of data analysis with data collection, analyzing within-case data, searching for cross-case patterns, shaping hypotheses and enfolding literature. All interviews except two (the interviewee asked to not be recorded, so we took notes during the interview) have been audio recorded and have been transcribed after the interview. Before additional interviews were conducted in one of the industrial companies, the already conducted interviews were briefly analyzed by identifying new topics and capabilities which arose during an interview and which would need further investigation **(first analysis step: overlap of data analysis with data collection)**. In this way, topics that arose in one of the industrial companies could also be investigated in the other industrial company (which represents replication logic).

In the **second analysis step (analyzing within-case data)**, each interview transcript was coded, first on paper, and later using the NVivo software tool. The first round of codification was started before the final interviews were conducted and included more than 50 codes. In two further rounds of codification of all the interviews, the number of codes was reduced significantly. In the final, fourth round of codification, which took place after all interviews had been conducted, we concentrated on identifying the five different software business cases and on identifying the capabilities involved in the development of software business in the industrial case companies. We used the 14 capabilities identified in previous research as pre-codes (C1-C7 and C11-C17). In addition we identified six capabilities that proved important in the cases we researched, but which were not included in the list of capabilities identified in previous research (C8-C10 and C18-C20). In our analysis, we focused on identifying the capabilities the company mentioned as being directly important for being able to offer the software and/or related services to the customers, i.e. to conduct software business. Some of these capabilities the industrial company already possessed, others the industrial company had to develop in the course of starting to do business with software. In this research, we did not consider commodity and support processes necessary for the every-day business of the industrial companies.

In the **third step (verification of data description)** of the analysis, the process of development of software business was written up and described for each software business case, focusing on the capabilities identified in the relevant case. The description of each case was sent to the interviewee(s) who had spoken about the case in the interviews, and these interviewees were then asked to read the case description and point out misunderstandings and missing capabilities, or to confirm that the software business case and capabilities important for the case were described correctly. The case descriptions were then improved based on the interviewees' comments. In the **fourth step (searching for cross-case patterns)** of the analysis, cross-case analysis was carried out by comparing the capabilities needed in the different types of software business to find out whether the capabilities needed in different types of software business differ from each other. In the **fifth step (shaping hypothesis)** of the analysis, we developed our categorization of industrial companies' software business capabilities (see Section "Results"), which represents a "theory for analysing" according to Gregor's (2006) classification of information systems theories. In the **sixth step (enfolding literature)** of the analysis we compared our results to findings of previous research (see Section "Discussion"). The results of both the within- and cross-case analysis will be presented and discussed next.

Results

In this section, the five distinct cases of software business identified in the two industrial companies will be presented. In the remainder of this paper, these cases that we identified in the empirical data and analyzed will be referred to as "software business case(s)" or "SWB case(s)" when referring to the empirical cases in general, and as "SWB Case" followed by a number from one to five when talking about a specific case (for example, "SWB Case 2"). For each software business case, we identified the capabilities that played a role in the respective SWB case. Each capability was assigned a code consisting of 'C' followed by a number (see Table 1). The results are organized as follows. First we will give an overview and definition of the capabilities identified in the empirical data, followed by a description of the five SWB cases and the capabilities we identified in the empirical data to be important in these SWB cases. Then, we will summarize our findings concerning which software development and business capabilities were used in which software business case. Finally, we will propose a classification for these software business capabilities, based on the type of software business they proved to be important for.

Software Development and Software Business Capabilities in Industrial Companies

The 14 capabilities identified in previous research and presented in Section "Software Development and Business Capabilities" were used as a basis when identifying capabilities in the empirical data, and were complemented by six additional capabilities identified when analyzing the empirical data (C8: software version management capability, C9: software training capability, C10: software documentation capability, C18: software business process creation capability, C19: software productization capability, and C20: capability of creating software contracts).

All together, we identified 20 capabilities (see Table 1, Section "Software Development and Business Capabilities"), part of which proved to be used, and part of which turned out not to be used in software business in industrial companies. Of these 20 capabilities, we categorized ten as software development capabilities (C1 – C10), and ten as software business capabilities (C11 – C20). The capability codes presented in Table 1 will be used to identify and refer to these capabilities in the presentation of the empirical case data in the following section. Table 1 gives an overview of these capabilities and gives a definition for each capability. For capabilities C1-C7 and C11-C17, these definitions are based on the review of previous research conducted in Section "Software Development and Software Business Capabilities" and on how the interviewees described them in the context of industrial companies' software business.

The definitions of the six additional capabilities identified in the empirical data (C8-C10 and C18-C20) are based on the interviewees' description of the respective activities and processes. These six additional capabilities have not yet been discussed in previous research in the context of software business and were identified by extracting from the interviews the activities and processes mentioned by the interviewees to have been essential when offering the software and/or related services to their customers within the industrial company's software business activities. Due to space limitations, we could not include rich quotes for all of the capabilities. We thus present a number of selected rich quotes for the newly identified capabilities from the different SWB cases only, to show based on which statements we identified these additional capabilities (see Table 2). For each citation, we include information about in the context of which SWB case the statement was made.

Table 2. Selected Citations from the Empirical Data for the Newly Identified Capabilities

Code	Citations from the empirical case data
Software development capabilities	
C8	"The third challenge was versioning, because the whole system is also being developed further." (SWB Case 3)
	"Now we try to make sort of 'package-solutions', where we collect several changes. When the software was in our own use only, we made changes to the software almost daily. […] This is a good way to do things internally, because then you don't have to plan so much ahead. Now we try to not have quite every day changes, and of course we now have to document which changes we have made so that we can deliver those in a bigger badge to the customers. We have to have follow-up on which customer is using which version, so that when a change request comes, we are able to deal with it." (SWB Case 1b)
C9	"This is what those who give the software away for free don't understand. They just think 'there it is, take it'. They don't understand that the software needs a running environment, training sessions, and maintenance." (SWB Case 1a)
	„For our software product we had a separate customer support department which offered training sessions. […] The customers were offered training sessions, where we went to the customers. We had more than 140 training sessions, or 150, with 15-20 participants on average." (SWB Case 4)
C10	"If you develop software – which inevitable happens step-by-step in-house – the software documentation always lags behind. So if an opportunity to sell the software comes up, it is difficult to conjure up that documentation." (SWB Case 3)
Software business capabilities	
C18	"The first sales of ProPIDesign happened kind of by accident. We didn't have a management system for it, and we didn't have an organization for it." (SWB Case 1a)
	"Selling that kind of software system poses several big challenges. The first challenge was the project, the scope of the project. The system consists of more than a thousand separate programs. The second challenge was its packaging. The third challenge was versioning, because the whole system is also being developed further. Finally, the whole delivery, the tracking, administration and confirmation of partial deliveries, that's the fourth challenge. […] At the end is the question of how long this whole project should be running. Those things lead to a quite extensive project, as we had to solve all those challenges." (SWB Case 3)
C19	"When we used the Oracle database, everything was in the same database like ProPIDesign – all the data was together. […] We thought that when we gave the software to customers, we weren't going to give [application x] to them, so we had to define the interfaces in such a way that would enable some other document management system to connect to ProPIDesign. […] This led us to try to keep those different pieces of software sufficiently separated from each other, to make it possible to sell only one part of them." (Case 1a)
C20	"Now we are in the seller role ourselves, we know which things we have to be careful with in the contracts: not to promise to develop the product too quickly, too cheaply and so on." (SWB Case 1a)
	"If you sell the software, it depends on what the other party expects. The contract has to be written in a way that those expectations don't grow so big in the contract that in the end you are not able to fulfil them anymore." (SWB Case 3)
	"Through the external business we were, for the first time, confronted with purchasing specifications and things like that. Up until then we were always something like a subcontractor; we didn't ever have to sign contracts that stated that if something doesn't work, we have a liability of millions of Euros. This was a new aspect that came with this external business." (SWB Case 5)

Next, the two software business cases identified in Conserx, and the three software business cases identified in Halcinson will be discussed briefly. In the following description of the SWB cases, we describe the main characteristics and capabilities of these cases, followed by Table 3 summarizing which capabilities could be identified in which of the software business cases. When analyzing SWB Case 1, we noticed that we have to distinguish between the time when the software application was given to Conserx's customers for the first time (SWB Case 1a) and the time when Conserx started to actively sell the application and related concepts to its customers (SWB Case 1b). This distinction has to be made due to the differences in capabilities needed in the two cases.

Within-case Analysis of Case Company Conserx

SWB Case 1: Enterprise Solution System Business

Already since the 1970s, a number of software applications have been developed by Conserx, ranging from document management systems used only inside Conserx to applications used when offering consulting and planning services to customers. One tool in particular, ProPIDesign, has proven important for the company in offering its consulting and planning services to its customers. When customers asked to get the ProPIDesign software for their own use in the mid 1990s, Conserx agreed to give the software application to one of its customers as an additional service (**SWB Case 1a**). Before deciding to make the software application available to its customer, Conserx discussed with the customer about their needs (C4) and estimated whether it had sufficient resources to ensure the installation and maintenance of the software (C7). Conserx possessed already the software training capability (C9) because of its internal software development and operation. In addition, Conserx was able to integrate the software at the customer site (C2) because of its previous experience with integrating the system at different Conserx sites. Conserx improved its software documentation to a certain extent for the external customers (C10) and offered support and maintenance for the software (C3). During these first sales, Conserx made contracts with its customers about the use of the software (C20), but only charged the customer for integration, maintenance and support of the software application, not for the software application itself (C17).

At the end of the 1990s Conserx decided to start offering the ProPIDesign software and the additional software applications more actively to customers (**SWB Case 1b**). Conserx developed a concept branded as “VirtualPlanning” (C14) which included a number of software applications and services built around the ProPIDesign application (C1). When offering the software application more actively, Conserx made use of some of the capabilities it had already needed during first sales of the software (C3, C4, C7, C9, C20). In addition, they improved their existing software version management (C8) in order to have better control over the different versions installed at different customer sites. ProPIDesign was already productized to a certain extent to make its installation and use at different Conserx sites easier. However, when starting to sell the software actively to external customers, Conserx productized ProPIDesign further by defining interfaces more clearly (C19) to make the integration of the software at the customer site easier (C2). In addition, Conserx developed a process for conducting business with the VirtualPlanning concept (C18) and defined license fees customers had to pay when using one or more software applications of the VirtualPlanning concept (C17). Conserx developed marketing material and managed to sell one or more software applications of the “VirtualPlanning” concept to 17 customers from 2001 to 2007 (C11).

SWB Case 2: Software Service Business

In 2006, Conserx decided to transfer almost all of its in-house software developers from different departments to Aplec (a spin-off company established by Conserx already in 2000) in order to make its software development resources and capabilities available to large parts of its organization. The almost 30 developers should then offer software development services to all departments and subsidiaries of Conserx, and companies acquired by Conserx, instead of to only a few departments as was the case before the transfer of the software developers.

Aplec offered software development services to its customers (C1), as well as software support and maintenance services (C3). The software developers had in most cases already a good understanding about the customers’ needs, as the developers had been part of the customer organizations for a long time (C4). One of the big challenges, however, was the development of an organized process for offering these services (C18) and of clearly defined project management practices (C6). The development of the software project capability was very challenging and took longer than Aplec had expected. As Aplec charged the customers for staff-hours spent when developing a software application for the customer, Aplec had to be able to estimate the required time for developing this application in advance quite accurately (C7). This demanded the development of a well-working requirement engineering process (C5). Both of these capabilities, in turn, were necessary in order to have control over the costs (C13), which was one of the most essential capabilities for Aplec. Aplec used hour-based pricing (C17) when offering its development services to Conserx, and drew up the contracts with its customers (C20). Even though Aplec mostly sold its services to Conserx and related companies, it took an active role in marketing and selling its services (C11).

Within-case Analysis of Case Company Halcinson

SWB Case 3: Enterprise Solution System Business

From the mid-1980s onwards, Halcinson developed a manufacturing execution system (MES) to automate the production of the hardware units they were producing. In the early 2000s, Halcinson was negotiating a production technology deal as the seller to one of its customers. The customer was interested in acquiring, apart from the technology, also the software utilities Halcinson had developed for its internal MES system. Halcinson started a project to find out whether they were able to sell the software, and in the course of the project defined the scope of the project, improved version management, and defined a way for packaging and delivery of the software (C6).

After assessing the resource need and resource and capability availability for selling the software to the customer and customizing the software (C7), Halcinson discussed with the customer to find out how the customer wanted to use Halcinson's software (C4). Finally, Halcinson agreed to sell the software to this customer to generate revenue, but without planning on repeating sale of this software.

Halcinson had processes for the internal support, maintenance and version management in place, but selling the software to outside customers demanded an improvement of these capabilities (C3, C8). Halcinson cooperated with an outside company who would take care of the installation and integration of the software at the customer site (C2). When writing up the software contract (C20), Halcinson defined a way of pricing the software and additional support and maintenance services (C17). Before the sale of the software to the customer, no professional organization for the sale of software existed within Halcinson, and thus a software business process had to be developed (C18). The improvement of version management and software maintenance, as well as the professional packaging of the software were part of the development of this software business process. Also the existing software documentation had to be improved to a more professional level (C10).

SWB Case 4: Software Product Business

At the end of the 1990s, Halcinson anticipated a change in its core market and wanted to have a turn-key system platform available for when this change happened (C15). Halcinson established a software spin-off, Telstanet, which would develop a software product (application layer software) to be integrated with its core businesses hardware units. However, the market did not change as expected, and Halcinson started to give the software product to the customers of its core business essentially for free as an addition to its hardware units. Telstanet represented a quasi cost center for Halcinson, as Halcinson paid Telstanet's wages and used Telstanet's software product to add value to its own core business product, the hardware units. One decade after start-up, Halcinson sold Telstanet because they felt they had not yet gotten their investments back. According to Telstanet's CEO, Halcinson had not understood the value of software.

Telstanet developed the software product (C1) based on the core needs of the customers (C4), productized (C19) and branded it (C14), marketed the software to Halcinson's customers (C11), offered support and maintenance (C3) and training (C9) to its customers, and developed software documentation for the software (C10). Project and software version management played an important role (C6, C8). Halcinson wanted Telstanet to apply the processes defined for the production of the hardware units also for software development, but that did not work. Consequently, Telstanet developed a process for software development (C18). Telstanet had to estimate the amount of time and resources it would have to invest in new customers (C7). Halcinson made the contracts with the customers on Telstanet's behalf (C20), and most of the times, Telstanet was not aware of these contracts' content.

SWB Case 5: Software Service Business

In the middle of the first decade of the 2000s, Halcinson decided to concentrate on its core business, on the production of electronics hardware and closely related business, and not to offer Telstanet's services to new customers any more. Telstanet's application layer software was not directly related to Halcinson's core business, and for that reason Halcinson decided to reduce their investments in Telstanet. As a result, Halcinson assigned Telstanet to acquire external business.

Telstanet was forced to find external customers by identifying new opportunities (C16) for selling its software development services (C1) to finance at least part of its business operations. Telstanet had to develop a process for business development, which was related to gaining new customers and to marketing and selling their services (C18). The development of a well-defined requirement engineering process (C5) and the improvement of their ability to estimate the resources (time, staff-hours) needed for developing software applications for external customers (C7), as well as the development of cost control measures (C13) were especially challenging. When working with external customers, Telstanet had to be much more exact with milestones and project schedules than during projects with Halcinson customers (C6). They also had to improve their marketing and sales capability, as contact to customers was not any more established by Halcinson, but had to be taken care of by Telstanet (C11). Additionally, they had to define prices for their services (C17) and had to learn how to make software contracts (C20), as Halcinson did not take care of these things on Telstanet's behalf any more.

Cross-case Analysis: Capability Categorization Based on the Type of Software Business

In a cross-case analysis, we studied for each capability whether it was applied in all three types of software business – software product business, enterprise solution system business, and software service business – or whether it was applied only in a certain type or types of software business. If a capability was applied in at least one of the enterprise solution system business cases or at least one of the software service business cases, we viewed the capability to be important for enterprise solution system business or software service business, respectively. We found that a number of capabilities were important for all three types of software business in the industrial companies we studied, and named this group of capabilities **core software business capabilities**. A number of capabilities were applied in both pure software product business and enterprise solution system business cases (which both represent software product business, see Section "Empirical context: Software business"), but not in software service business cases. We thus named this group of capabilities **software product business capabilities**. The group of **software service business capabilities** includes the capabilities which were applied only in software service business cases, and the group of **enterprise solution system business capabilities** includes the capability that was applied only in enterprise solution system business cases. Surprisingly, we found that three business capabilities we would have expected – based on the review of previous research – to be relevant for all types of software business in industrial companies actually were not commonly used. This category will be called **professional software business capabilities** and includes the capability of estimating the customer segment size, the capability of anticipating market development, and the capability of identifying business opportunities. Table 3 presents the capabilities we identified with help of the within-case analysis in the five software business cases, grouped according to above presented classification. For each capability, the table shows whether the respective capability was applied in a certain SWB case ("yes") or not ("-"), and whether a capability is classified as software development (SWD) or software business (SWB) capability.

Table 3. Software Business Capabilities Relevant in the Five SWB Cases

			Case 4	**Case 1a / 1b**	**Case 3**	**Case 2**	**Case 5**
Code	**Capability type**	**Software business type**	**Software product**	**Enterprise solution system**		**Software service**	
Core software business capabilities							
C1	SWD	Software application development capability	yes	- / yes	-	yes	yes
C3	SWD	Software support and maintenance capability	yes	yes / yes	yes	yes	-
C4	SWD	Capability of understanding user needs	yes	yes / yes	yes	yes	-
C6	SWD	Software project management capability	yes	- / -	yes	yes	yes
C7	SWD	Capability of estimating the resources needed	yes	yes / yes	yes	yes	yes
C11	SWB	Software sales and marketing capability	yes	- / yes	-	yes	yes
C17	SWB	Capability of pricing software and related services	-	yes / yes	yes	yes	yes
C18	SWB	Software business process creation capability	yes	- / yes	yes	yes	yes
C20	SWB	Capability of creating software contracts	yes	yes / yes	yes	yes	yes
Software product business capabilities							
C8	SWD	Software version management capability	yes	- / yes	yes	-	-
C9	SWD	Software training capability	yes	yes / yes	-	-	-
C10	SWD	Software documentation capability	yes	yes / -	yes	-	-
C14	SWB	Software branding capability	yes	- / yes	-	-	-
C19	SWB	Software productization capability	yes	- / yes	-	-	-
Software service business capabilities							
C5	SWD	Requirement engineering capability	-	- / -	-	yes	yes
C13	SWB	Cost control capability	-	- / -	-	yes	yes
Enterprise solution system business capabilities							
C2	SWD	Software integration capability	-	yes / yes	yes	-	-
Professional software business capabilities							
C12	SWB	Capability of estimating customer segment size	-	- / -	-	-	-
C15	SWB	Capability of anticipating market development	yes	- / -	-	-	-
C16	SWB	Capability of identifying business opportunities	-	- / -	-	-	yes

The capability of pricing software and related services (C17) was important in the enterprise solution system business cases and the software service business cases, but had not been applied in SWB Case 4, software product business. However, in SWB Case 4, the spin-off's CEO mentioned that Halcinson's lack of understanding for the value and cost of software was a main reason why the spin-off's operations did not pay off for Halcinson, and why the spin-off was sold to another company in the end. Therefore, we argue that the capability of pricing software and related services would also be important in software product business – thus in all types of software business – and classified it as "core software business capability".

Professional software business capabilities were, based on our review of previous research, expected to be important in all types of software business, but, based on the five software business cases we researched, turned out not to be in the context of industrial companies' software business operations. Therefore, the following capability categories proved to be important for industrial companies' software business activities: core software business capabilities, software product business capabilities, software service business capabilities and enterprise solution system business capabilities. Figure 1 summarizes our main finding, the categorization of industrial companies' software business capabilities according to the type of software business they are relevant to, based on the five software business cases we studied in the two industrial companies. From this figure, we left out the category of professional software business capabilities, as this category of capabilities did not seem to play an important role in the industrial companies' software business cases we studied. However, we want to emphasize that this category might be important in other industrial companies' software business activities and should thus be considered by industrial companies, too.

Figure 1 has to be read top-down. Core software business capabilities are relevant for all three types of software business. For software product business, both core software business capabilities and software product business capabilities are relevant. For enterprise solution system business, core software business capabilities, as well as software product business capabilities and enterprise solution system business capabilities are relevant. For software service business, both core software business capabilities and software service business capabilities are relevant. Within each category of capabilities, we distinguished between software development capabilities and business capabilities.

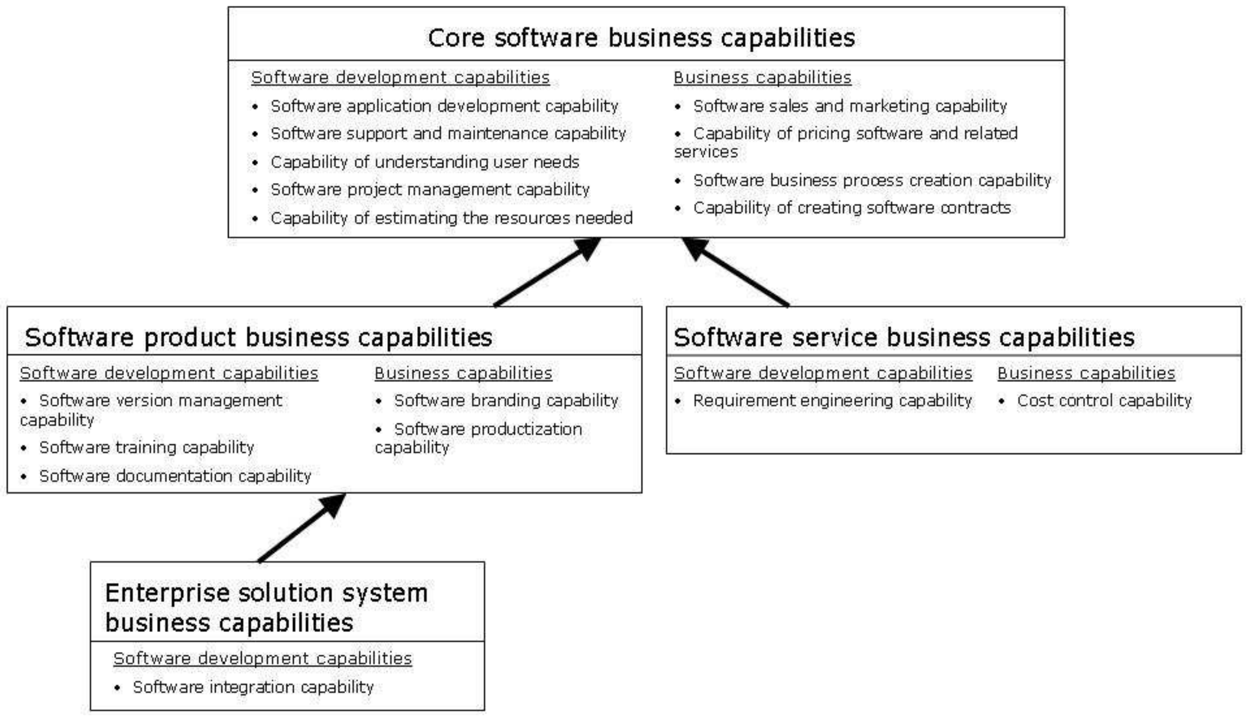
Core software business capabilities
Software development capabilities
• Software application development capability
• Software support and maintenance capability
• Capability of understanding user needs
• Software project management capability
• Capability of estimating the resources needed
Business capabilities
• Software sales and marketing capability
• Capability of pricing software and related services
• Software business process creation capability
• Capability of creating software contracts
Software product business capabilities
Software development capabilities
• Software version management capability
• Software training capability
• Software documentation capability
Business capabilities
• Software branding capability
• Software productization capability
Software service business capabilities
Software development capabilities
• Requirement engineering capability
Business capabilities
• Cost control capability
Enterprise solution system business capabilities
Software development capabilities
• Software integration capability

Figure 1. Industrial Companies' Software Business Capabilities, Categorized by Software Business Type

Our identification and classification of these capabilities can help managers in industrial companies to assess the company's current capability base and decide which capabilities the company should use, improve and develop when starting to conduct a certain type of software business. This, in turn, can help the industrial company to improve its competitive position. The above classification might also be valid for software business of primary software companies, but this would have to be verified by future research. We would expect that the capabilities classified as "professional software business capabilities", which were not regularly applied in industrial companies' software business operations, play a larger role in primary software businesses, whose core business is software. Next, we will discuss these categories of capabilities in more detail and compare our findings to previous research.

Discussion

Based on the empirical analysis, two findings are highlighted. First, based on a review of previous literature, we identified 14 capabilities related to software development and (software) business, part of which proved to be used also in software business conducted by industrial companies. When conducting a within-case analysis of the software business cases representing the empirical data, we identified six additional capabilities not yet discussed in previous research and used by industrial companies when doing software business. Second, based on the cross-case analysis of the software business cases according to the type of software business they represent, we found that some capabilities were important for all types of software business, while others were especially important for a certain type or types of software business. We also identified several capabilities that were rarely, if at all, applied by the two industrial companies in the context of software business. Based on these findings, we classified the 20 capabilities into five categories. The capabilities and categorization of capabilities are based on our study of five software business cases in two industrial companies. Therefore, it is possible that in other industrial companies than the ones we studied, additional capabilities might prove to be important in a certain type of software business. The five categories and capabilities they include, thus, should not be seen as being absolute.

Capability Categorization by Software Business Type

We classified the capabilities analyzed in this research into five categories. Next, we will discuss this categorization of capabilities, and discuss how our findings fit with and build on previous research. The capabilities we identified might prove to be generic for primary software companies, but as we did not research primary software companies, it is left to future research to confirm or dismiss this. However, the comparison of our findings to capabilities identified in previous research on software business gives some indication that our classification of software business capabilities into five capability classes most likely is valid for primary software companies, too. For the industrial companies, many of the capabilities we identified in the present research – especially the business-related capabilities – have not been applied in connection to software and software-related services and thus have not been generic before starting to do software business. Many of these capabilities became relevant only after the industrial company started to do business with software. Therefore, our findings can help industrial companies to be aware of the capabilities important in different types of software business and in this way to be better prepared for the challenges that come with software business.

Core Software Business Capabilities

The category of core software business capabilities includes for the most part capabilities we identified already in previous research to be important for software development (Butler and Murphy 2003; Cockburn et al. 2000; Ethiraj et al. 2005; Grant 1996; Sallinen 2002) and business (Day 1994; Grant 1996; Morgan et al. 2009; Sallinen 2002), which thus supports our findings. The software business process creation capability was challenging to develop especially in the enterprise solution system business cases, where the industrial company started to sell a software system developed originally for company-internal use only. In these cases, the industrial company was not prepared in advance to sell the software. In the case of the software spin-offs, the intention was from the beginning to sell software products or services, and thus the company was better prepared. However, where the industrial company imposed its own process framework on the spin-off, the development of a process for doing software business was challenging, too. The capability of creating software contracts was important in all types of software business. The industrial company has to be aware that selling internally developed software to external customers binds the industrial company in a certain way to the customer, and for that reason the contracts about use of the software, support and maintenance have to be given special attention. An interesting observation was that in one case, the industrial company took also care of drawing up contracts on the spin-off's behalf. In one case the software spin-off was not even aware of the content of the contract between the industrial company and the customer. Only after the industrial company decided to decrease investments into its spin-off, the spin-off had to learn how to make software contracts. In addition, we were able to identify the software business process creation capability and the capability of creating software contracts to be of importance for all types of software business. These two capabilities seem to be especially challenging in the context of industrial companies starting to do software business from within the company.

Software Product Business Capabilities

A number of capabilities proved to be important for both software product business and enterprise solution system business, but not for software service business. In general, this suggests that software product business and enterprise solution system business are closely related on a capability-level. Hoch et al. (1999) argue that the class of software products includes both stand-alone software products for mass markets, and enterprise solution systems which have to be customized to a certain extent and integrated with existing software systems. Thus, the finding that enterprise solution system business and software product business require very similar capabilities, fits with Hoch et al.'s (1999) classification of software business. Of the five capabilities we identified in this category, the branding capability has already been identified in previous research, but we identified four additional capabilities based on the analysis of the empirical data in the present research. The branding capability has been mentioned in previous research in relation to business in general (Markides and Williamson 1994; Petromilli et al. 2002), but not specifically in relation to software business. Our finding that this capability is rather important in software product business and enterprise solution system business than in software service business is understandable. In software service business, software is usually developed for one specific customer and not sold to other customers (Hoch et al. 1999; Sallinen 2002). Software products and enterprise solution systems, on the other hand, are sold to a larger number of customers, and the development of a brand for the software product can strengthen the sales possibilities for this software. The case of ProPIDesign and the Technology Package offer good examples. At the time when the ProPIDesign enterprise solution system was only given to single customers without having any intention to sell it to additional customers, the development of a brand was not considered. Conserx focused on developing a brand for the enterprise solution system only *after* it had decided to sell the system actively to its customers. In contrast, the Technology Package deal was planned as a one-time sale from the beginning, thus the development of a brand would not have added value to the offer.

In addition to the software branding capability, we identified the software version management capability, the software training capability, the software documentation capability, and the software productization capability to be important specifically for software product and enterprise solution system business. Software products and enterprise solution systems often demand the development of upgrades, and version management is an important tool to ensure that these upgrades are directed at the right target group. We want to emphasize that in the cases of enterprise solution systems, the industrial companies had already productized the systems to a certain extent because they were used at different sites within the industrial company. Also, certain version management practices were in place already before selling these systems. These capabilities, however, had to be improved in both industrial companies when selling the systems to external customers. We want to highlight this finding, because when an industrial company starts to sell software that has been developed merely for company-internal use, it is important to acknowledge that there might be a need to improve these capabilities even though they already exist in some form.

Enterprise Solution System Business Capabilities

In our research, we identified one capability to be specifically important for enterprise solution system business – the software integration capability. Other than the capabilities we identified as software product capabilities which are relevant in both software product and enterprise solution system business, the software integration capability was not relevant in the software product business case. As Cusumano (2004: 24) defined, software products are sold to a large number of customers "without customization or special installation support". Hoch et al. (1999) argue that enterprise solution systems need professional services around the core software, and Cusumano (2004) says that they need customization or special integration work. This would explain why, according to our findings, the software integration capability is important in enterprise solution system business, but not in software product business.

Software Service Business Capabilities

Both the requirement engineering capability and the cost control capability proved to be especially important in software service business, as opposed to software product and enterprise solution system business. Butler and Murphy (2003) identified the requirement engineering capability in a company that offers software products as well as software services, but do not show in connection to what part of the business (product or service) the capability was identified. Ethiraj et al. (2005) identified the requirement engineering capability to be of importance in the context of software service business. Our findings are thus supported by Ethiraj et al. (2005), and add to Butler and Murphy's (2003) findings by specifying that the requirement engineering capability is specifically relevant in software service business.

Surprisingly, also the cost control capability proved to be of special importance in software service business, but not in software product and enterprise solution system business. This capability has been identified in previous research on business in general (Day 1994; Teece 2007), but not in relation to software business. The most obvious difference between software product and software service business is in their cost structure and their earnings structure. In software service business there are almost constant, and significant, marginal costs, which are the costs the company has for producing each unit of output. For each new customer, a software application has to be developed specifically for this customer's needs. In software product business, first a software product is developed, which can then be reproduced with almost zero marginal costs over and over again. This means that in software product business, profit margins are much higher than in software service business (for examples see Cusumano 2004; Hoch et al. 1999). Because marginal costs are high, software service companies are very sensitive to cost pressures in their projects (Carmel and Sawyer 1998). The differences in the cost structure of software service and software product companies, thus, justify our finding.

Professional Software Business Capabilities

Professional software business capabilities are the capabilities that were, based on the review of previous research, expected to be important for software business, but were not consistently applied by the industrial companies when doing software business. The capability of anticipating market development and the capability of identifying business opportunities were applied only by Telstanet, but the application of these capabilities does not seem to be related to the type of software business practised, but rather to the circumstances of Halcinson's intentions for Telstanet. The capability of estimating the customer segment size was, to our surprise, not relevant in any of the software business cases. To identify the target segment and evaluate its profit potential is a core element of every business model (Teece 2007), and thus should also be relevant when an industrial company starts to do business with software. This finding lets us assume that the industrial companies might not recognize software business as an important form of business. In addition, these capabilities might be seen as important only for a company's core business. Software business, as discussed, did not represent the industrial companies' core business. This could explain why the industrial companies studied in this research did not apply these capabilities in their software business operations.

Implications for Practice

The findings of our research can help industrial companies to direct their resources to the development of the most important capabilities when they start to do software business. Each industrial company most likely possess a different set of capabilities that exists before the company's involvement with software business, depending on the company's core business activities and internal software development practices. Industrial companies that develop software for in-house use, for example, probably already possess some of the software development capabilities that were identified to be also necessary for software business. Therefore, each industrial company would have to assess its current capability base to find out which capabilities it would have to develop for a certain type of software business in addition to already existing capabilities. Core software business capabilities should be considered by the industrial company in every type of software business. For software product business, the industrial company should, in addition, focus on software version management, training, documentation, branding and productization capabilities. For enterprise solution system business, software integration, too, plays an essential role. For software service business, the industrial company should place special attention to the development of well-working requirement engineering and cost control practices. As found in our research, the so-called "professional software business capabilities" were hardly applied in the cases we studied in this research. It might be that these capabilities are considered mainly within a company's core business, and are forgotten in the case of business activities outside the company's core business. Therefore, we advise industrial companies to put special attention to these professional software business capabilities to improve the likelihood of gaining competitive advantage and economic value with their software business.

Suggestions for Future Research

We suggest that future research would study whether our classification of software business capabilities is also valid for companies that conduct software business in the primary software industry, without having originated from industrial companies. Special focus could be placed on whether the capabilities that we classified as "professional software business capabilities" play a more important role in professional software business than in software business initiated and conducted by industrial companies.

Our identification of capabilities relevant for software business in industrial companies offers the possibility to conduct a quantitative study in a larger number of industrial companies to find out whether our findings can be generalized for software business in industrial companies.

In addition, based on the path-dependency principle of the resource-based view (Barney 1991), it would be important to study which of the capabilities an industrial company already possesses before the involvement with software business – for example, based on firm-internal software development – can be applied in software business, whether and how they have to be improved, and whether the applicability of these capabilities is dependent on the type of software business conducted.

Limitations of the Research

The present research identified which capabilities are relevant for software product business, enterprise solution system business, and software service business. As this study has been conducted in industrial companies, these findings are limited to software business in industrial companies, and would have to be verified for software business conducted in the primary software industry.

This research represents qualitative research which was mainly based on qualitative interviews. Thus, it is possible that additional capabilities which have not been mentioned in the interviews play an important role in the development of software business in industrial companies. As only a limited number of software business cases have been studied, our findings have to be verified by researching a larger number of cases.

Conclusions

In an attempt to answer the question which capabilities industrial companies need when they start to conduct software business, this research added to previous research in three ways. First, we identified 14 capabilities that are, according to previous research, relevant for software development and business in general. Second, when studying five cases of software business started by industrial companies, we identified six more capabilities that proved to be relevant for industrial companies' software business operations. Third, we found that the three types of software business – software product business, enterprise solution system business, and software service business – require to a certain extent the same capabilities, but also require certain software business type-specific capabilities. Based on this finding, we suggest five categories for industrial company's software business capabilities. Our findings have practical relevance for industrial companies that start to do software business by giving guidelines of where to direct their resources to, i.e. the development of which capabilities to focus on for a certain type of software business.

Acknowledgements

I would like to thank the Oulu University Scholarship Foundation, the Tauno Tönning Foundation, the Jenny and Antti Wihuri Foundation, and the ROOSTER project for the financial support of my research. In addition, I want to thank Dr. Juhani Warsta and Prof. Veikko Seppänen for their support during this research, as well as the anonymous reviewers and Prof. Juhani Iivari for their invaluable suggestions on how to improve this paper.

References

Amit, R., and Shoemaker, P.J.H. 1993. “Strategic Assets and Organizational Rent”, *Strategic Management Journal* (14:1), pp. 33-46.

Bakos, J.Y., and Treacy, M.E. 1986. “Information Technology and Corporate Strategy: A Research Perspective”, *MIS Quarterly* (10:2), pp. 107-119.

Barney, J.B. 1991. “Firm Resources and Sustained Competitive Advantage”, *Journal of Management* (17:1), pp. 99-120.

Bharadwaj, A.S. 2000. “A Resource-based Perspective on Information Technology Capability and Firm Performance: An Empirical Investigation”, *MIS Quarterly* (24:1), pp. 169-196.

Butler, T., and Murphy, C. 2003. “Unpacking Dynamic Capabilities in the Small-to-Medium Software Enterprise: Process, Assets and History”, in *Proceedings of the Eleventh European Conference on Information Systems,* Naples, Italy, pp. 327-337.

Carmel, E., and Sawyer, S. 1998. “Packaged Software Development Teams: What Makes Them Different?”, *Information Technology and People* (11:1), pp. 7-19.

Chakravarthy, B., and Lorange, P. 2007. “Continuous Renewal, and How Best Buy Did It”, *Strategy & Leadership* (35:6), pp. 4-11.

Chesbrough, H. 2003. “The Governance and Performance of Xerox’s Technology Spin-off Companies”, *Research Policy* (32:3), pp. 403-421.

Clemons, E.K, and Row, M.C. 1991. “Sustaining IT Advantage: The Role of Structural Differences”, *MIS Quarterly* (15:3), pp. 275-292.

Cockburn, I.M., Henderson, R.M., and Stern, S. 2000. “Untangling the Origins of Competitive Advantage”, *Strategic Management Journal* (21:10/11), pp. 1123-1145.

Cunningham, B. 1997. “Case Study Principles for Different Types of Cases”, *Quality & Quantity* (31:4), pp. 401-423.

Cusumano, M.A. 2004. *The Business of Software*, Free Press, New York.

Day, G.S. 1994. “Capabilities of Market-Driven Organizations”, *Journal of Marketing* (58:4), pp. 37-52.

Del Canto, J.G., and González, I.S. 1999. “A Resource-Based Analysis of the Factors Determining a Firm's R&D Activities”, *Research Policy* (28:8), pp. 891-905.

Eisenhardt, K.M. 1989. “Building Theories from Case Study Research”, *The Academy of Management Review* (14:4), pp. 532-550.

Ethiraj, S.K., Kale, P., Krishnan, M.S., and Singh, J.V. 2005. “Where Do Capabilities Come From and How Do They Matter? A Study in the Software Services Industry”, *Strategic Management Journal* (26:1), pp. 25-45.

Golafshani, N. 2003. “Understanding Reliability and Validity in Qualitative Research”, *The Qualitative Report* (8:3), pp. 597-606.

Gorman, G.E., and Clayton, P. 1997. *Qualitative Research for the Information Professional - a Practical Handbook*, Library Association Publishing, London.

Grant, R.M. 1996. “Prospering in Dynamically-Competitive Environments: Organizational Capability as Knowledge Integration”, *Organization Science* (7:4), pp. 375-387.

Gregor, S. 2006. “The Nature of Theory in Information Systems”, *MIS Quarterly* (30:3), pp. 611-642.

Grindley, P. 1996. “The Future of the Software Industry in the United Kingdom: The Limitations of Independent Production”, in *The International Computer Software Industry*, Mowery, D. C. (Ed.), New York, Oxford University Press, pp. 197-239.

Helfat, C.E., and Peteraf, M.A. 2003. “The Dynamic Resource-based View: Capability Lifecycles”, *Strategic Management Journal* (24:10), pp. 997-1010.

Hobday, M., Davies, A., and Prencipe, A. 2005. “Systems Integration: a Core Capability of the Modern Corporation”, *Industrial and Corporate Change* (14:6), pp. 1109-1143.

Hoch, D.J., Roeding, C.R., Purkert, G., Kindner, S.K., and Muller, R. (Eds.) 1999. *Secrets of Software Success: Management Insights from 100 Software Firms Around the World*, Harvard Business School Press.

Jarvenpaa, S.L., and Leidner, D.E. 1998. “An Information Company in Mexico: Extending the Resource-based View of the Firm to a Developing Country Context”, *Information Systems Research* (9:4), pp. 342-361.

Kirk, J., and Miller, M.L. 1986. *Reliability and Validity in Qualitative Research*, Sage Publications, Beverly Hills.

Käkölä, T. 2003. “Software Business Models and Contexts for Software Innovation: Key Areas for Software Business Research”, in *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS)*, Big Island, Hawaii, USA.

Lee, A.S., and Baskerville, R.L. 2003. "Generalizing Generalizability in Information Systems Research", *Information Systems Research* (14:3), pp. 221-243.

Lopez, S.V. 2005. "Competitive Advantage and Strategy Formulation – the Key Role of Dynamic Capabilities", *Management Decision* (43:5/6), pp. 661-669

Malerba, F., and Torrisi, S. 1996. "The Dynamics of Market Structure and Innovation in the Western European Software Industry", in *The International Computer Software Industry*, Mowery, D. C. (Ed.), New York, Oxford University Press, pp. 165-196.

Markides, C.C., and Williamson, P.J. 1994. "Related Diversification, Core Competences and Corporate Performance", *Strategic Management Journal* (15:Special Issue), pp. 149-165.

Mata, F.J., Fuerst, W.L., and Barney, J.B. 1995. "Information Technology and Sustained Competitive Advantage: A Resource-based Analysis", *MIS Quarterly* (19:4), pp. 487-505.

Morgan, N.A., Vorhies, D.W., and Mason, C.H. 2009. "Market Orientation, Marketing Capabilities, and Firm Performance", *Strategic Management Journal* (30:8), pp. 909-920.

Penrose, E. 1959. *The Theory of the Growth of the Firm*, New York, John Wiley& Sons.

Peppard, J., Lambert, R-, and Eswards, C. 2000. "Whose Job is it Anyway? Organizational Information Competencies for Value Creation", *Information Systems Journal* (10:4), pp. 291-322.

Peteraf, M.A., and Bergen, M.E. 2003. "Scanning Dynamic Competitive Landscapes: A Market-Based and Resource-Based Framework", *Strategic Management Journal* (24:10), pp. 1027-1041.

Petromilli, M., Morrison, D., and Million, M. 2002. "Brand Architecture: Building Brand Portfolio Value", *Strategy & Leadership* (30:5), pp. 22-28.

Pettigrew, A. 1997. "What is a Processual Analysis?", *Scandinavian Journal of Management* (13:4), pp. 337-348.

Prahalad, C.K., and Hamel, G. 1990. "The Core Competences of the Corporation", *Harvard Business Review* (68:3), pp. 79-91.

Sallinen, S. 2002. *Development of Industrial Software Supplier Firms in the ICT Cluster*, Doctoral thesis, University of Oulu, Department of Marketing.

Santhanam, R., and Hartono, E. 2003. "Issues in Linking Information Technology Capability to Firm Performance", *MIS Quarterly* (27:1), pp. 125-153.

Schramm, W. 1971. *Notes on Case Studies of Instructional Media Projects*, A. f. E. Development. Stanford University, Washington, DC.

Teece, D.J. 2007. "Explicating Dynamic Capabilities: The Nature and Microfoundations of (Sustainable) Enterprise Performance", *Strategic Management Journal* (28:13), pp. 1319-1350.

Teece, D.J., Pisano, G., and Shuen, A. 1997. "Dynamic Capabilities and Strategic Management", *Strategic Management Journal* (18:7), pp. 509-533.

Tyrväinen, P., Warsta, J., and Seppänen, V. 2008. "Evolution of Secondary Software Business: Understanding Industry Dynamics", in *Open IT-Based Innovation: Moving Towards Cooperative IT Transfer and Knowledge Diffusion*, León, G., Bernardos, A., Casar, J., Kautz, K., and DeGross, J. (Eds.), IFIP International Federation for Information Processing, Vol. 287, Springer, Boston, pp. 381-401.

Weill, P., and Vitale, M. 2002. "What IT Infrastructure Capabilities are Needed to Implement E-Business Models?", *MIS Quarterly Executive* (1:1), pp. 17-34.

Wernerfelt, B. 1984. "A Resource-Based View of the Firm", *Strategic Management Journal* (5:2), pp. 171-180.

Yin, R.K. 1994. *Case Study Research*, SAGE Publications Ltd, Thousand Oaks.