A study of firm capabilities and performance in the Software Services industry¹

SENDIL K ETHIRAJ
University of Michigan Business School
701 Tappan Street, Room D5203
Ann Arbor, MI 48109
Ph. 734-764 1230
Email: sendil@umich.edu

PRASHANT KALE
University of Michigan Business School
701 Tappan Street, Room D4209
Ann Arbor, MI 48109
Ph. 734-764 2305
Email: kale@umich.edu

M.S. KRISHNAN
University of Michigan Business School
701 Tappan Street, Room D3251
Ann Arbor, MI 48109
Ph. 734-763 6749
Email: mskrishnan@bus.umich.edu

AND

JITENDRA V. SINGH
The Wharton School of Business
3620 Locust Walk, Suite 2000
University of Pennsylvania, Philadelphia PA 19104
Ph. 215-898 6605
Email: singhi@wharton.upenn.edu

Dated: September, 2002

¹ We thank Phanish Puranam for helpful comments on an earlier version of this paper. Errors and omissions remain our responsibility.

The names are listed in alphabetical order reflecting equal contribution by the authors.

Research funding for this project from the Mack Center for Technological Innovation at the Wharton School and The William Davidson Institute and May and Mike Hallman Fellowship at the University of Michigan is gratefully acknowledged.
An empirical study of firm capabilities and performance in the software services industry

Abstract

Recent years have witnessed a surge of interest in the notion of capabilities as an important source of competitive advantage. However, few papers in strategy provide empirical documentation of the existence of capabilities and how and why they explain performance. The present paper is an attempt to address this question. Using a large sample of detailed project level data from a leading firm in the global software services industry, we attempt to empirically study the importance of capabilities. We find that two broad classes of capabilities are significant. The first class, which we label client-specific capabilities, is a function of repeated interactions with clients over time and across different projects. This learning from repeated interactions with a given client reduces project execution costs and helps improve project contribution. The second class, termed project management capabilities, are acquired through deliberate and persistent investments in infrastructure and systems to improve the firm’s software development process. Our empirical results suggest that the marginal returns to acquiring different capabilities may be different and an understanding of such trade-offs can improve firm decisions to improve and/or acquire such capabilities. We discuss the key contributions of our paper and the implications for future research on capabilities.
1. Introduction

In recent years strategy scholars have increasingly agreed that non-imitable and non-substitutable organizational capabilities (and resources) are a key source of inter-firm performance differences (Barney, 1991; Dosi et al., 2000; Nelson, 1991; Rumelt, 1984; Wernerfelt, 1984). This recognition has, in turn, placed emphasis on the question of where and how these capabilities emerge and how they influence firm performance. There are a number of theoretical arguments about the how, where and why of capabilities (Barney, 1991; Dierickx & Cool, 1989; Peteraf, 1993; Teece et al., 1997). Rich, detailed process studies of firms that advance our understanding of the connection between capabilities and firm performance have also been done (e.g., Burgelman, 1994). However, there are relatively few large sample empirical studies that carefully document the existence of firm capabilities and how they affect performance (for exceptions see Henderson & Cockburn, 1994; Levinthal & Myatt, 1994; Makadok & Walker, 2000; McGrath et al., 1995). Though capabilities, by definition, are context-specific and difficult to generalize across firms and industries, it is nevertheless important to examine empirically the role of capabilities in explaining firm performance over time.

Using the context of the Indian software services industry, this paper studies two interrelated questions: one, where do capabilities come from, and two, how do they affect firm performance? We use detailed, project-level data from one software services firm over a six-year period to carefully test some empirical regularities posed by the capabilities based view. The rich, disaggregated project-level data about resource inputs, project characteristics, capability metrics and profitability allow us to delve deeper into the capabilities-performance link than extant research.
The Indian software industry is an attractive context for the study of firm capabilities. First, there is a widely shared view that much of the Indian software industry’s explosive growth over the last decade is accounted for by factor cost differences between India and the developed country markets (Arora et al., 2001; Nasscom, 2001). The implication is that performance differences are driven not by firm capability differences, but by country-level comparative cost advantages. While factor cost differences definitely exist, even a cursory examination of the data suggests that firm-level explanations cannot be discounted. The Indian software services industry accounted for $6.2 billion of export revenues in 2000-01 (Nasscom, 2001). A telling statistic is that 0.8 percent (25 firms) of the firms in the industry (nearly 3000 firms) accounted for 60 percent of export revenues. Indeed, a relatively small set of firms growing at a compounded average annual rate of about 45-65 percent over the last decade accounts for much of the activity in the Indian software services industry. Our premise is that a detailed examination of the economics of one or more of these 25 firms can afford a good insight into the micro-foundations of the capabilities that underlie such sustained and robust growth in a competitive industry.

Building on research on firm capabilities in general and on our detailed fieldwork and interviews with the project managers of several firms in the software services industry we argue that two sets of capabilities are important in the software services industry: client-specific capabilities and project management capabilities. Client specific capabilities develop by virtue of repeated interaction with a given client across multiple projects over time. Project management capabilities are acquired through deliberate and persistent investments in infrastructure and training to improve the firms’ software development processes. The development of these capabilities rests not only on implicit learning by doing processes but also on deliberate, proactive investments in building them. For example, based on industry experience, some
scholars have presented economic models to show that it makes economic sense for software vendors to initiate the first project with a client at low prices even if it amounts to a modest loss on the project. The first project serves as a platform for the development of client-specific and project management capabilities that help significantly reduce costs in the long-run and ultimately generate positive returns (Whang, 1995). In our empirical analyses, we estimate the marginal contribution, cross-sectionally and temporally, of the two capabilities to project profitability.

The study makes two main contributions to the extant literature on capabilities. First, we argue and empirically demonstrate that organizational capabilities are often context specific and fruitful research in this area might emanate from the enjoining of a detailed qualitative study of the capabilities specific to a context and the careful empirical estimation of their significance and value. Second, we show that not all capabilities provide the same marginal contribution to performance. This result is significant because it suggests that if different capabilities have different costs and benefits associated with their development or acquisition, managers should pay attention to understanding these trade-offs in making investments in capability development. More broadly, our study advocates a shift in the debate from whether or not capabilities matter to “what” capabilities matter and “how”.

The rest of the paper is organized as follows. The next section briefly outlines the research literature on capabilities and how capabilities influence performance differences. Section 3 describes the Indian software industry context in some detail and develops our hypotheses. Section 4 describes the data, the measures and empirical estimation procedures. Finally, we present the results and discuss their implications in the light of extant research on capabilities.
2. Theory and literature review

2.1. What are capabilities and why are they important?

The notion of capabilities can be traced back to Penrose (1959) and Andrews (1971) among others (see also Selznick, 1957). Penrose (1959: 25) suggested that resources consist of a bundle of potential services. While these resources or factor inputs are available to all firms, the “capability” to deploy them productively is not uniformly distributed. Analogously, Andrews (1971) argued that the “distinctive competence” of an organization is more than what it can do; it is what it can do particularly well. Put simply, capabilities may be seen as assets with rent generating potential. The literature largely emphasizes two kinds of rents that capabilities can generate. The first parallels the textbook notion of (Ricardian) rents from scarce resources. Ownership of a scarce resource (say land in Manhattan) enables the owner to enjoy superior rents relative to competitors who do not own the resource (they lease the land). The scarcity rents here are rooted in the inelastic supply curve for the resource. In addition, there is the implicit condition that, all else being equal, the cost of ownership is less than the lease cost. It means that at the time the scarce resource was acquired, its price was less than its (future) marginal product. Since this hinges on imperfections in the factor market, its basis in capabilities is questionable if we adopt Penrose’s (1959) definition.

A second type of rent, more closely related to capabilities, is quasi-rents. Quasi-rents are the excess of the value of a resource in its current use over its value in the next best use (Klein et al., 1978). Such rents may be a product of specialized assets or resources that are embedded within the context of the organization such that their optimal deployment is contingent on the presence of other complementary assets (e.g., managers, culture, technology, etc.) or even learning how to deploy the bundle of assets efficiently. Additionally, there is a cost involved in transferring this resource or asset along with the complementary assets to another firm thereby
reducing its productive value (Langlois, 1992). This difference in value is the quasi-rent accruing to the owner of the resource. It may be noted that quasi-rents are not a function of an inelastic supply curve (Ricardian rents) or an inelastic demand curve (monopoly rents). They are a function of the uncertainty about the production function underlying the deployment of resources. If inputs are freely available and the knowledge of exact inputs, the input proportions, and the technology underlying a firm’s production function is available to all competitors then there are no quasi-rents to be exploited since the value of the assets in alternative uses is equal to its value in current use. In this paper, our notion of capabilities reflects those assets that can generate quasi-rents.

2.2. Where do capabilities come from?

Traditionally, strategy has devoted little research attention to the issue of where capabilities come from. Nelson and Winter (1982) made one of the early theoretical attempts to understand where capabilities come from. They viewed the firm as bundles of path-dependent knowledge bases. Over time, firms’ knowledge, accumulated through “learning by doing”, is embedded in bundles of “routines” that are likened to the genetic material of the firm. Routines, a central concept in evolutionary theory, involve repetitive patterns of activity, require investment in routine-specific human and physical capital, and are easily recognized as belonging to a class (Winter, 1990). An important element of their theory is the description of the firm as a historical entity, with productive knowledge being a result of endogenous, learning by doing processes. Consequently, this perspective sees firms as entities that possess heterogeneous capabilities as a function of their routines and search processes. These capabilities are rooted in the individual skills and organizational routines that serve as organizational memory to repetitively execute the sequence of productive activities without trouble. At their core, the individual skills and organizational routines embody knowledge and competence in carrying out
the productive activities that the firm is engaged in. Building on the basic idea that history matters and that capabilities are rooted in contextually embedded knowledge underlying the production function, others have emphasized the significance of absorptive capacity (Cohen & Levinthal, 1990) and asset stocks (Dierickx & Cool, 1989) in driving capabilities-based competition.

Some researchers have also suggested that capabilities are not merely the result of tacit accumulation of experience embedded in routines and learning by doing. They are also the result of deliberate investments in organizational structure and systems to make constant improvements in those routines and practices (Zollo & Winter, 2002). Organizations strive to adapt their operating processes through proactive actions dedicated to process improvements. These may include explicit efforts to continuously learn and capture the lessons from prior experience of self or others (Collis, 1996; Zollo & Winter, 2002) and incorporate those lessons to make improvements in prevalent practices, or create organizational mechanisms to coordinate and institutionalize the improvement efforts (Kale et al., 2002). Although the notion of making deliberate investments to improve firm capabilities may be understood uniformly by most firms, there are idiosyncratic firm-level differences in the timing of this effort, the nature and amount of their investment and effort, and the internal organizational mind-set that supports this process. These differences may get reflected in significant heterogeneity across firms with respect to the capabilities that result from this effort.

In sum, it appears that in operationalizing the notion of capabilities, it is important to show that: (1) there are strong theoretical reasons undergirding how and why capabilities can generate rents; (2) capabilities tend to evolve over time to reflect the joint effects of passive learning-by-doing and deliberate firm level investments in learning and making improvements.
and, (3) capabilities are hard to imitate or easily acquire in factor markets, and this forms the basis for rent generation.

As far as empirical research is concerned, the literature on capabilities is still largely qualitative and anecdotal in nature. Several excellent papers in this tradition provide useful insight into the historical development and evolution of capabilities (Iansiti & Khanna, 1995; Raff, 2000; Rosenbloom, 2000; Tripsas & Gavetti, 2000). Unfortunately, such methods do not permit the estimation of the significance and value of the capabilities. By contrast, an examination of the large-sample empirical literature on capabilities suggests that few studies have managed fully or adequately, to capture the spirit of the idea. Most studies usually measure capabilities using aggregate indicators such as R&D intensity (Silverman, 1999) at the firm level. If capabilities critically reside at the operational level within firms, aggregate firm-level measures may tend to mask much of the variance within firms. In a notable exception, Henderson & Cockburn (1994), using survey data attempted to get at more disaggregated measures of R&D capability at the program level. They measured architectural competence (ability to integrate knowledge within the firm) and component competence (locally embedded knowledge) at the R&D program level to predict drug discovery productivity. Similarly, McGrath et al. (1995) measured firm competence at pursuing new initiatives and identified that deftness and comprehensiveness are important pre-cursors to competence acquisition and ultimately to the emergence of competitive advantage.

Our study continues in this empirical tradition. We construct disaggregated capability measures that can be measured as close to the operational level as is practically possible, and we also track the evolution of these capability measures over time.
3. Overview of the Indian Software Services Industry

In this section we provide a brief overview of the Indian software services industry. We focus on the demand- and supply-side economics of the industry and attempt to draw out the nature and type of capabilities that might have the potential to generate rents.

The Indian software services industry is relatively young, with many of its most mature companies incorporated in the late-70s and early 80s. Although information technology (IT) started becoming an important productivity tool in Indian business since the early 1980s, the domestic market for IT services was small and continues, even today, to account for only about 25-30 percent of the industry’s sales (Nasscom, 2001). The Indian software services industry received a big boost in the early 1990s when the demand for IT services in the developed world outstripped the available supply of skilled labor. India, which at that time was graduating 150,000 English speaking engineers a year with only a limited demand for their services within the Indian economy, was well placed to take advantage of this opportunity. Companies in the developed world began focusing on India to leverage its low cost, English-speaking IT manpower. Given the low levels of initial investment required to enter the software services business and the minimal regulatory intervention from the Indian Government, there were few, if any, entry barriers or constraints in these early years. Several hundred Indian firms were founded to exploit this opportunity. As the industry witnessed very rapid growth, there was a general consensus that macro factors such as the access to the large, low-cost, English-speaking technical manpower in India and improvements in IT-related infrastructure (e.g., high bandwidth communication lines) set up by private and state-owned companies positively influenced the competitive advantage and growth of Indian companies (Arora et al., 2001; Nasscom, 2001).
3.1. Demand for Indian software services

Faced with the small and undeveloped nature of the domestic software services market, Indian software firms focused primarily on serving the needs of the export market. Their early work, however, was neither technologically very sophisticated nor critical to clients’ businesses. Clients usually did the high-end work such as requirement analysis and top-level design either in-house or by using US based consultants. They outsourced the low-end, labor-intensive work such as low-level design, coding, testing, support and maintenance to Indian companies to leverage their low-cost activity base. Clients or their US-based consultants retained the high-end work in the early years either because their Indian vendors did not possess the requisite skills to undertake these activities or, even if they did, the clients did not have sufficient confidence to entrust such activities to them. Thus, the origin of the Indian software industry was firmly rooted in performing low-end, technically less demanding and labor intensive work for the global IT industry and exploiting labor cost arbitrage opportunities between India and developed country markets (Nasscom, 2001).

Over time, however, there was a change in this trend brought about, in most part, by the industry structure that subsequently emerged. The low entry barriers in the Indian software services industry triggered abnormally high levels of entry by new firms. Between 1989 and 1998, over 3000 software services firms were founded. Most of them aspired to serve export markets. With this increase in supply, domestic competition in the labor market (i.e. for trained engineers) shot up. It was not enough anymore to possess low-cost labor resources and exploit arbitrage opportunities. Firms needed to improve the productivity of labor to compete effectively in the market. Thus, they began a systematic push to build high end software capabilities, move up the value chain and improve per employee revenue figures. Consequently, the leading Indian software firms today are also the leading firms in the global market. Among the approximately
549 software and programming firms in the technology sector traded in the US capital markets, Indian firms are among the top in rankings based on return on equity (e.g., Infosys Technologies ranked 7 and Wipro Ltd. ranked 12) (Market-Guide, 2002).

Since the mid-nineties there has been a distinct shift in the nature of software projects undertaken by some of the leading Indian software firms. These firms gradually shifted their role from that of merely implementing a design provided by their overseas clients to becoming active participants in the design of the complete application product. As a consequence, some of the leading Indian software companies now undertake a range of jobs from highly labor intensive code migration work such as the integration of old mainframe-based systems into new e-commerce platforms, or developing new code for pre-designed applications and software tools, to projects that involve both conceptual design and implementation of customer relationship applications and supply-chain management systems.

The capacity to execute such a range of jobs enabled several firms to deliver end-to-end solutions and, thereby, lay claim to a larger share of the client’s IT budget and compete for it with leading firms in the US such as IBM and Accenture. Therefore, while comparative cost advantages do exist for Indian software firms, they are by no means sufficient for competitive advantage. First, the nature of services provided by these firms involves specialized work (e.g., designing supply chain management systems or setting up trading exchanges) that requires not only technical knowledge of software design but also an in depth understanding of the client’s industry and business process. Secondly, even firms such as IBM, Sapient, and Accenture have set up subsidiaries in India exploiting the same cost advantages. These subsidiaries undertake software projects involving both design and development on latest technology platforms and business applications. This development not only supports the argument that Indian software
teams are capable of executing these high value projects but also erodes much of the Indian firms’ traditional cost advantages.

3.2. Supply of Indian software services

Two aspects of the Indian software services industry are critical to understanding its supply-side economics. The first involves where the software is developed and how the development process is managed and organized. The second concerns the type of contract used to provide the services. We elaborate on each of these factors below.

Indian software firms have traditionally executed two types of projects: onsite and offshore. In onsite projects the Indian firm supplies (or exports) its overseas clients with software professionals with particular technical skills required by the client and the entire project is developed and executed at the clients’ site. In offshore projects, on the other hand, the Indian firm may send a few software professionals to the client site to understand its requirements and specifications, but thereafter the entire software is developed in India. The post development support and maintenance of the software is also carried out largely from India. Obviously, the offshore development model where the bulk of the work is done in India is more cost effective due to labor market arbitrage. In some cases, a hybrid of the two types is also observed. From a cost standpoint, the greater the proportion of work completed offshore, the lower the cost of project execution. This is primarily because onsite employees need to be paid in accordance with host country norms, which erodes a significant proportion of the cost-based advantages that Indian firms enjoy.

In the early days of the Indian software industry, Indian companies executed a majority of the projects onsite. This happened because, first, the overseas clients did not have full

---

1 Sections 3.2. and 3.3. draw heavily from an excellent article on the Indian software industry by Arora et al. (2001).
2 In industry parlance, this practice is also sometimes known as bodyshopping.
confidence in the Indian firms’ ability to execute projects in conformance to their needs. Secondly, the Indian firms also did not have sufficient understanding of clients’ needs and often required close and regular interaction with the client. But, over time, as overseas clients developed confidence in the software capabilities of their Indian vendors and the vendors, in turn, developed a better understanding of clients’ needs, it made economic sense to relocate the bulk of the project development activities to India to take full advantage of its low cost development base. The improvements in the infrastructure for long-distance communication and data-transfer facilitated this process further.

The second significant feature of software services, both in India as well as worldwide, involves the type of contract adopted in software outsourcing arrangements with software firms. Contracts can be broadly classified into two categories: Fixed-Price and Time & Material (Banerjee & Duflo, 2000). In fixed price contracts the vendor charges a fixed fee for its software development services which is usually negotiated before the start of the project. Although the vendor bears most of the risk in this case, tight and efficient project management can yield potentially higher margins. In a Time & Materials (T&M) contract, the vendor provides services at a pre-negotiated rate for every person-hour of effort expended on the project and is paid on these terms either at the end of the project or at periodic intervals when project milestones are reached. In such contracts, although a vendor is usually protected against any cost and schedule over-runs that may arise due to changes in client needs or specifications, there is also the potential for reduced incentives for the vendor to execute more efficiently.

In the context of the Indian software industry, in its early days most software firms usually preferred undertaking T&M contracts with their clients. T&M contracts, however, tend to yield lower margins since the contract is behavioral, i.e., the client needs a safeguard that the
Indian firm is not billing them more person-hours than is necessary to execute the project. So clients typically reduce their risk by negotiating hard on the price per person-hour. From the Indian vendor firm’s point of view, their reduced risk came at the cost of reduced margins. Therefore, for firms that had the capability to manage their projects efficiently and assume the risks, it made economic sense to move to a Fixed Price contract since they provided higher margins. From the client’s point of view, they were willing to concede higher prices since Fixed Price contracts reduced their need for behavioral monitoring and had strong penalties associated with delays or defects in project completion. So a combination of improved project execution and management capabilities and the Indian firms’ impetus to improve profit margins led to an increasing thrust toward taking on more Fixed Price contracts. On the supply-side, the steady transition from onsite to offshore projects and from T&M to Fixed Price contracts meant that an increasing share of project management risk had to be borne by the Indian firms. This meant that the firms had to acquire the requisite capabilities to compete effectively. The following section elaborates what these capabilities were and how they played a critical role in Indian firms’ successful evolution.

3.3. Capabilities of Indian software services firms

Against the backdrop of the demand- and supply-side economics of the Indian software services industry, two broad sets of capabilities are critical. The first are what we term client-specific capabilities. As software firms work with their clients over time, they tend to develop several client-specific patterns of interaction that become cost effective over repeated interactions. For instance, one of the firms we interviewed mentioned that clients tend to have fairly idiosyncratic ways of doing things and it takes some time to understand and appreciate this. One of this firm’s clients wanted a team of its employees to be stationed at the client site for three months after completion of the project. Initially the Indian firm resisted such a demand
since it led to increased costs. However, over a period of time, the firm was able to convince the client that it could provide the same quality of after-sales service with the support team based in India. This was possible because with repeated interaction with the same client over time, software vendors develop a better understanding of the information infrastructure at the client site and have better clarity of requirements from these software projects in the context of the client’s business environment. This knowledge is acquired through several interactions with the client over various stages of the development cycle such as requirements specification, business process design, data preparation, software installation, debugging and testing, etc. Hence long-term relationships and repeated interactions with clients resulted in client-specific learning that had a positive effect on both revenues and costs.

On the revenue side, clients were more willing to agree to higher prices on repeated projects as they developed confidence in the capability of the Indian firm to execute projects per their specifications\(^3\). We reckon that Indian firms were able to capture at least some of the switching costs faced by the client in finding a new vendor and building a new working relationship. On the cost-side, there was tangible cost reductions associated with learning the client’s requirements over time. For instance, one of our interviewees pointed to a client who never specified requirements very clearly at the outset and tended to repeatedly come back and ask for new features after the project was delivered. While this tended initially to be disruptive and create problems for the Indian firm, over time it learned to work around this. Rather than deliver very finished projects, the firm began to involve the client firm at the prototype stage itself and then started building in the client firm’s needs as the project went along. In this

\(^3\) It may be noted that this conclusion is based on our interviews with several companies. In the dataset used for empirical estimation there is no evidence that revenues from repeat clients were systematically higher than that of first-time projects after controlling for project characteristics. Revenues tended to remain constant after adjusting for inflation and exchange rate changes.
manner, they were able to avoid the post-delivery negotiations and completion delays and the associated costs. Such client-specific tailoring of projects that is believed to be quite common in the industry also enhances the software firm’s understanding of the client’s business domain. Thus, repeat projects for clients helped develop important client-specific capabilities that contribute to higher profits by reducing costs. Therefore, we hypothesize that,

**H1:** Development of client-specific capabilities based on repeated interaction with clients is positively related to project performance.

A second capability that is more fungible across clients and industry domains is the software development and project management capability. For the most part, skills related to project design, development and planning or execution are critical here (Humphrey, 1989; Jalote, 1997). The following three capabilities are particularly important: (i) **Software design and building capabilities:** Software companies must have the capability to first properly understand the requirements of the clients and design an appropriate system or architecture to address them. Secondly, they must possess the capability to efficiently and effectively build the code based on the product design as well as coordinate the entire code development process that is usually distributed across many teams and/or sites. These capabilities are usually reflected in the defects identified in the product/software during the design and development process. (ii) **Effort estimation and management capabilities:** Companies have to be skilled in not only accurately assessing the requirements of the client but also in assessing the resource inputs or effort required to build and execute the project. They need to be able to identify appropriate resources (for instance people with the necessary skill, experience, availability, etc.) and create and use prior experience/data to arrive at accurate estimates of the resource/effort requirements. Further,
they also need to have the skills to ensure the effective management and deployment of the required resources. Poor capabilities in effort estimation and management are usually reflected in increased manpower cost and/or effort overrun. (iii) **Schedule estimation and management capabilities:** Once companies have a tentative idea of the resource inputs necessary to build and implement the project, they must be able correctly to assess the duration and schedule required for completing the project for a given level and quality of resources. They also need to possess the management skills to ensure that the project resources are garnered, deployed and managed to complete the project within the planned schedule. Again, poor capabilities on this dimension are reflected in project completion delays and schedule slippages.

Given the importance of the above capabilities, in recent years there has been great emphasis on software engineering and project management and firms have been making investments in improving their processes and capabilities. The Capability Maturity Model (CMM) developed by the Software Engineering Institute (SEI) at Carnegie Mellon University is a widely adopted framework to improve software capabilities. Built on theories of quality and continuous process improvement, but tailored to software development, the CMM was first initiated to provide the Department of Defense a standard means for measuring contractor capability via its definition of process maturity (Humphrey, 1989). As the CMM became widely adopted as a standard quality process model in the defense sector, commercial organizations within these corporations began to investigate whether they, too, could benefit from the approach to software process improvement expressed in the CMM. In the last few years software process improvement based on the CMM and related frameworks has emerged as an integrated solution to the software problems in various corporations and empirical evidence in support of the same has been reported (Herbsleb *et al.*, 1997; Krishnan, 1996).
Recognizing the importance of project management capabilities, several Indian firms have been the leaders in adopting CMM guidelines to improve their software development processes. According to SEI, in 2001, of the 42 companies worldwide certified as having attained a level-5 capability, 25 are based out of India. Meeting these guidelines is not a trivial task. Firms need to make substantial investments in firm infrastructure, systems and human capital. The CMM specifies five maturity levels, each consisting of several Key Process Areas (KPAs), to assess an organization’s process capability by measuring the degree to which processes are defined and managed (see Paulk et al., 1993: Figure 1). Firms first need to do a detailed comparison of their development and project management processes with the CMM process quality framework and identify specific aspects that need to be improved. Having made these organizational and process changes, software firms need to set up a rigorous metrics program to collect data on relevant measures to assess various aspects of their development process and institute audit systems to track non-conformance of best practices, process deviations, and exceptions. Measurement based feedback to improve the process capability is achieved through monitoring and review forums to track improvements and make required changes on a regular basis. In addition, software firms need to invest in training programs to improve their process maturity under the CMM framework. Personnel Training is one of the main aspects of the CMM framework. The mandated activities under this include on-going training in both new technology and software process and project management skills for software developers and managers. Hence, to achieve higher levels of process maturity, firms need to organize rigorous training of all their employees not only with a view to improve their development/project management practices but also to institutionalize the entire capability improvement initiative (Paulk et al., 1993).
The CMM framework offers generic guidelines for institutionalizing disciplined practices across various activities in software development. Since the nature of software development is such that software teams can quickly turn to ad hoc practices, institutionalization of disciplined practices and continuous feedback based improvements can evolve as a core project management capability of the firm. Overall, firms can eventually develop their software development and project management capabilities as a result of cumulative and integrated effort across the many process areas of CMM as described above. Eventually the possession of these capabilities should lead to better project level performance for the firm. Past studies have already recorded how strong software and project management capabilities and processes are linked to substantial improvements in the quality of the products/services provided as well as the efficiency of providing them (Harter et al., 2000; Herbsleb et al., 1997; Krishnan, 1996). Eventually, we believe that these benefits should result in improved performance at the project level in terms of profitability and contributions.

Therefore, we hypothesize that,

**H2**: Higher levels of project management capabilities will lead to higher levels of project performance.

This paper sought to understand the origin, significance, and value of firm capabilities. Until now, we have argued that capabilities reflect the distinctive deployment of resources and also that they are contextually grounded. We have then elaborated the specific capabilities that are important in the software services industry, namely, client-specific capabilities that arise in the context of repeated interactions with clients over time and project management capabilities that develop through deliberate and persistent investments in the effort and infrastructure to create them. While client-specific capabilities help reduce project execution costs, project
management capabilities help maintain low-cost and high-quality all along the software development and management value chain. These capabilities, in turn, provide opportunities for rent generation for firms. We next turn our attention to the significance and value of these capabilities as a matter for empirical investigation.

4. Methods

4.1. Model specification and estimation

We assume that the firm produces a single output, software services, using skilled labor. Since the software services business is highly labor intensive, the scale of the firm’s operations and its costs depend almost exclusively on manpower. The firm’s profit function from each project is given by,

\[ \pi_{ijt} = G(P, C, \omega_{ijt}, \zeta_{ijt}, \theta_{ijt}) \]

where, \( p_{ijt} \) are the profits from project ‘i’ for client ‘j’ in year ‘t’. This recognizes that profitability can vary as a function of time, client characteristics, and project characteristics. \( P \) are the prices and \( C \) are the costs respectively of project ‘i’ for client ‘j’ in year ‘t’. Both prices and costs depend on project specific characteristics, \( d \), such as size, complexity, duration, type of contract and so on. The last term in the equation, \( ?, \), captures the capabilities measures, which vary with client and time and also depend on project specific characteristics \( d \).

This reflects our hypotheses that capabilities tend to influence both the prices and costs of software services. As argued in the previous section, capabilities allow the firm to command a price premium (shift the demand curve), and also reduce costs (shift the supply curve).

We estimated the following equation using project level data,

\[ \log(\pi_{ijt}) = \log(w_{ijt}) + \log(z_{ijt}) + \theta_{ijt} \]

where the dependent variable, \( p_{ijt} \), is project-level contribution (revenue minus cost), \( w_{ijt} \) is the vector of input characteristics that determine costs, \( z_{ijt} \) is a vector of project specific controls, and
$\gamma_{ij}$ reflect the capability measures. The coefficients on the logged variables are directly interpreted as elasticities, while the coefficients on variables measured in levels vary with the magnitude of the variables. We only logged independent variables that did not have zero values and retained variables with zero values as levels to avoid estimation difficulties.

We used a fixed (within) effects panel data estimator to estimate the equation above. The coefficient estimates are interpreted as the amount of within-panel variation in the dependent variable (project performance) that is explained by the within-panel variation in the independent variables. Thus, the regression analysis relates changes in project contribution across different projects for a given client to changes in the independent variables after controlling for unobserved time-invariant effects, and the included control variables.

4.2. Data

We obtained detailed quantitative data at the project level from a leading, world-class software services firm headquartered in India. Over 90 percent of its revenues are export-based and more than 60 percent of it comes from North American clients. The dataset included revenues, cost, factor inputs, capability measures, various project characteristics such as size, client industry, development platform, etc., measured at the project level.

We have data on 223 projects executed by the firm over a six-year period from 1996-2001. However, after dropping projects with missing data, our sample was reduced to 138 projects in the full models and 152 projects in the partial models. We found no statistically significant differences between projects with complete data and those with missing data for those

---

4 We explain our rationale for logging the dependent variable in the section describing the measures.
5 Note that entering the variables in levels does not change the results. We report the results on the logged variables for obvious ease of interpretation.
6 Since the dataset reveals the economics of the firm and has competitive implications, we are unable to reveal the identity of the firm.
variables for which we had complete information. This increased our confidence that the analyzed data were not systematically different from the data on all the projects.

4.3. Measures

*Dependent variable*

The dependent variable is project contribution (revenues minus costs) measured in Indian Rupees (INR) recognized on the date of completion of the project. Firm performance and profitability is essentially an aggregate of project level contribution – we, however, focus on the project level because the firm capabilities that we study primarily exist and evolve at the activity or project level within firms. To be able to use this variable meaningfully we need to adjust the INR values both for inflation and INR-USD (US dollar) exchange rate depreciation over time\(^7\). As we discovered, this is a far from a simple problem, since identifying an appropriate price index specific to an export-based software services firm is quite difficult. Fortunately, we found that a simple solution is to take logs of the dependent variable and include year dummies in the regression estimation. This helps remove the effect of the adjustment factor from the parameters being estimated\(^8\). Also, note that though our dependent variable is project performance rather than firm performance, the correspondence between the two is quite large. Firm revenues are largely an aggregation of project revenues. Project performance is an imperfect indicator of firm performance only to the extent that it does not account for firm-level cost overheads.

---

\(^7\) The exchange rate depreciation over time is important since the firm incurs most of its costs (e.g., wages) in Indian Rupees whereas its revenues are in US dollars. Therefore a change in the Rupee-Dollar exchange rates will also change the coefficient estimates of the firm profit function.

\(^8\) For example, C2000 (contribution in year 2000) needs to be adjusted to the same units as C1996 (contribution in 1996). Usually, C2000 is adjusted as C2000/P1996 where the denominator (P1996) is the appropriate index for adjustment. Taking logs we obtain, \(\log(C2000) - \log(P1996)\). This means that if we take logs on the dependent variable and include year dummies, the parameters are free of the adjustment factor. The year dummies absorb the adjustment factor and since the year dummies are only controls, the main results and our interpretation remain unaffected.
Independent variables

Client-specific Capabilities. As outlined above, client-specific capabilities are a function of repeated interactions with a given client. These repeated interactions may be function of time (on a long project) or spread over several projects. We measured this in two ways. For each project in our dataset, we have information on whether the client is new (i.e., first project for that client) or a repeat client (i.e., the firm has executed projects for that client in the past). This variable, called customer type, is a dummy variable coded 0 if it is a repeat client and coded 1 if it is new. We expect the sign on this coefficient will be negative. Within our dataset, we also have information on multiple projects done for a given client. We used this information to estimate a client fixed effect. Though both measures are largely substitutes, the first measure captures some information about past projects not included in our dataset, and the second produces a within-sample estimate of the client-specific effect. Since we cannot estimate a single parameter for the client fixed effects, we only examine whether they are jointly significant after controlling for input and project characteristics and time.

Project Management Capabilities. We also measured the firm’s internal, software project management capabilities. These capabilities are not client-specific (or domain/platform specific). They are capabilities that can, by definition, be leveraged across clients, industry domains, and development platforms. We used three metric variables to measure these internal project management capabilities. The first measures the number of in-process defects identified during the project execution phase. Since in-process defects can vary by size of the project, we normalized it by a project size measure (FP) described below. In-process defects, which measure the defects detected in the product, reflect a firm’s software development and

---

9 The measure of project size, called Function Points (FP) is a composite measure of project size and complexity and described in greater detail in the sub-section below on the control variables included in the estimation.
management capabilities. Further, it has also been reported that the cost of fixing a defect detected in the early phases of software development can be substantially lower than the cost of fixing a customer reported defect or defects detected at the final stages of software development before releasing the product to customers (Jones, 1997). Hence we expect that lower in-process defects will lead to higher project contribution.

A second measure of project management capability is the extent of schedule slippage, i.e., by how much the project completion date was delayed. Such delays can adversely affect profitability since the firm can incur contractual penalties for delays and also have to bear increased labor costs. We expect the magnitude of schedule slippage to vary directly with expected project duration. To control for this, we normalized schedule slippage measured in days by project duration also measured in days. The expected sign on this coefficient is negative.

A third measure of project management capability was effort overrun, i.e., difference between actual person-months required to complete the project and person-months that were initially estimated. Effort overrun can directly affect project profitability since project costs go up. Effective project management capability involves minimizing such overruns. Since effort overrun is likely to vary directly with budgeted person-months, we normalized effort overrun by the budgeted person-months to estimate its main effect. The expected sign on this coefficient is negative. The capability measures were entered as levels (rather than logs) since zero or negative values are possible.

In summary, the capabilities measures employed here reflect our view that capabilities are distinct from either inputs or resources. While inputs or resources are accessible by all firms at prevailing factor prices, capabilities reflect the deployment of resources. Therefore, capability differences between firms are reflected in productivity differences between them, and within
firms in productivity improvement over time. Thus, in the research design employed here, changes in the capability measures reflect changes in the productivity of resources over time.

**Control Variables**

*Contract type.* There are two types of contracts commonly used in the Indian software services industry. One is the Time & Material (T&M) contract. It is, quite simply, a cost-plus contract where the project risk is borne mostly by the client. In such contracts, project profitability is a percentage mark-up over cost. The client and the software firm contract *ex ante* on the number of person-hours budgeted and the unit price per person-hour. Both parties tend to jointly decide on the number and composition of the project team. The second type of contract is the Fixed Price contract. In such contracts, the vendor software firm bids for a project at an *ex ante* agreed upon price and delivery date. Here most of the project risk is borne by the vendor firm, since there are often penalties associated with project completion delays. As such, the software service firm makes independent decisions on number of members and the composition of the project team. Consequently, project profitability depends both on accurate *ex ante* assessment of project costs and good project management. To control for the obvious effect of contract type on profitability we included a dummy variable, where a T&M contract was coded as 0 and Fixed Price was coded 1. In terms of the risk-return trade-off Fixed Price projects are expected to yield superior project profitability suggesting a positive sign of the coefficient.

*Project size and complexity.* We expect project size to affect profitability. Though we have no priors on this, we might reasonably expect that the size-profitability relationship might be increasing below the minimum efficient scale and decreasing above the maximum efficient scale. We used a measure of project size, called Function Points (FP), to capture a composite measure
of project size and complexity. We logged the FP measure in our estimation and, as a result, the coefficient can be interpreted as elasticity with respect to project profitability.

**Team size.** We also expect team size to have an effect on project profitability. As above, we expect project profitability will increase with team size when the team is too small and overworked and decrease when the team size is large enough to create coordination problems. Team size is logged in our model and can be interpreted as elasticity with respect to profitability.

**Person-months.** The team size measure is imperfect since there may be attrition during the project or some members may work only part-time. This will cause the team size measure to be overstated. A more precise measure of project size is person-months of labor. This accounts for both team member attrition and part-time manpower usage. Person-months are entered as logs and can be interpreted as elasticity with respect to profitability.

**Project Duration.** Duration is an important control variable since longer projects are more prone to cost overruns, either due to forecasting difficulties or employee attrition. We measure project duration as the actual months taken for project completion. This measure is entered in logs and can also be interpreted as elasticity with respect to project profitability.

**Industry domains.** Over the six years that our dataset spans, the software firm we studied has executed projects for clients in multiple industries. Since competition and appropriability conditions can vary by industry and the firm’s capabilities may not be entirely fungible across industry domains, we include industry domain dummies to control for industry specific

---

10 Earlier measures of size and complexity tended to just count the number of lines of code in software and use it as a proxy. The problem with this measure is that the number of lines of code for a given application varies directly by software platform. For example, for a given application, the number of lines of code in Cobol is several times greater than the number of lines of code in Java. The FP measure is designed to be independent of programming language (see Albrecht & Gaffney, 1983).

11 Employee attrition is historically very high in this industry ranging from 10-30 percent per year.
differences. We included three dummy variables for financial services, manufacturing, and marketing industries. The omitted category was “other.”

**Development platforms.** Profitability can also depend on the software platform on which the project is executed. The firm is likely to have better experience with platforms on which it has executed a number of prior projects and is likely to incur substantial learning and start-up costs on newer platforms. To account for this, we included four dummy variables to distinguish between Windows NT, Mainframe, Unix, and Web-based platforms. The omitted category was “other.”

**Time.** We control for time in all our models using year dummies. This control is necessary to capture the variance in the dependent variable due to exchange rate fluctuations and inflation. The time dummies are also important to account for the proportion of onsite and offshore projects that the firm has done. We found that no projects in our dataset were exclusively onsite or offshore. All projects had a component of both. Unfortunately, the firm did not have data at the project level on the proportion of work done onsite and offshore. According to the firm there were no dramatic changes from one project to another on mix of onsite and offshore work. However, there has been some change in this mix over the years. Thus including the year dummies also helps control for the change in the proportion of onsite and offshore work over time.

**5. Results**

Table 1 presents the descriptive statistics of key variables employed in the estimation. Table 2 presents the correlation matrix of variables. From the correlation matrix it is clear that all the control variables (i.e., project/team size, and project duration) are significantly positively related to project contribution. We also note that the relationship between the capability
measures and project contribution is negative as expected, though only the process defects variable is statistically significant.

Table 3 presents the coefficient estimates from the fixed-effects panel regression analysis. The second column lists the predicted sign on the key independent variables in the model. The next column labeled Model 1 presents results with just the control variables in the model. The R-squared for this model is highly significant and the control variables account for about 59 percent of the variance in the data. The model also includes client fixed effects, wherein we included a dummy variable for each client within the dataset. As expected the client fixed effect is strongly significant, suggesting that some client-specific learning and or switching cost may be associated with the higher project contribution from repeat projects for a given client.

Model 2 adds customer type (client-specific capability) to model 1. The model is again highly significant and the model R-square increases to about 0.63. It may be noted that the table reports adjusted R-squares for all the models. In displaying nested regression models, reporting R-squares tells us little about whether the added variable contributes significantly to model fit, since R-squares always increase when an independent variable is added. The adjustment of R-square to reflect the residual degrees of freedom solves this problem. In other words, an increase in adjusted R-square between successive nested models in table 3 signifies improvement in model fit. The customer type measure reflects whether the project in the dataset involved a new client or repeat client. We sought to capture aspects of client specific capabilities, such as

---

12 We examined the possibility of decreasing returns to project and team size by introducing quadratic powers of the variables. The coefficients were non-significant suggesting that the constant returns to scale assumption might be reasonable for this dataset.

13 To check for possible multi-collinearity in estimation, we computed the variance inflation factor (VIF) indices and found that the person-months variables had the highest VIF of 4.42, followed by team size (2.97), Size FP (2.46), and duration (2.36). The mean VIF for all independent variables was 2.51, suggesting marginal concern about multi-collinearity in model estimation (Chatterjee et al., 2000).

14 The adjusted R-square is computed as \((1- (1-R^2)*(N-1 / N-k-1))\) where N is the number of observations and k the number of independent variables.
specialized investments and learning from repeated interactions with the client. As expected, projects for repeat clients, after controlling for input and project characteristics, generate higher contribution than new projects. This lends some prima facie support for our expectation that client-specific capabilities are positively related to project contribution. Our results appear to also validate the widely used investment analyst metric of using percentage of repeat clients to assess competitive advantage for a firm in the software services industry. Note that the client fixed effect continues to be highly significant even in model 2. This provides greater confidence that client-specific capabilities are robustly related to project contribution.

Model 3 adds the three project management capabilities variables to the regression. The overall model continues to be significant and accounts for about 68 percent of variance in the data. We find that schedule slippage and effort overrun respectively are significantly negatively related to project contribution, suggesting that improvement in project management capabilities are rent generating at the project level. The coefficient for in-process defects is not statistically significant though its sign is in the expected direction. We reckon that this result is largely due to the lack of significant variance on this measure (Mean = 0.605; S.D. = 1.781). We also explored this result in discussions with project managers. They reasoned that the identification of defects during project execution is indeed a capability since they can be fixed at lower cost than if they were to be fixed after the project was completed. In other words, higher values on the in-process defects measure might indicate better project management skills. In addition, they pointed out that an increase in the number of in-process defects tends to disrupt delivery schedules and negatively impact project profitability. In sum, it seems that while in-process defects capture some detection capability, it also exposes some weakness in software design and execution capabilities; namely why did these defects arise in the first place. In the long-run, we expect that
this defect detection capability might be positively related to project contribution, especially with repeat projects. However, the imperative to fix defects during project execution can disrupt project schedules as well as increase project costs leading to the observed result.

Interestingly, we find that the customer type variable turns non-significant when the project management capabilities measures are included. It seems that the project management capabilities variables share common variance with the client-specific capabilities measure. This runs counter to our hypothesis that they are each distinct sets of capabilities. We discussed this result with industry experts who suggested that by working with a given client over time (i.e., repeat projects) the software firm gains a better understanding of the client’s needs and expectations. This, in turn, can help reduce effort overrun and schedule slippage accounting for the observed result. The managers in the software firm we studied tended to agree with this possible interpretation. Although it appears that the client-specific capability and the project management capability measures are capturing some shared variance, we note that the client fixed effects continues to be robustly significant even in Model 3.

The results in model 3 suggest that decreases in schedule slippage and effort overrun respectively contribute to increases in project contribution after controlling for time and other input/project characteristics. However, we can conduct further analysis to examine whether and how project management capabilities have evolved over time and how that affects contribution. In order to examine the evolution of project management capabilities, we included an interaction term of schedule slippage with 1996 (the first year of data) and effort overrun with 2001 (the last

---

15 Ideally we would have liked to examine the evolution over time of client-specific capability also. We were unable to do this since client-specific capability is measured as a dummy variable thus precluding the inclusion of an interaction term with time.
year of data). These results are reported in Model 4. The overall model continues to be highly significant accounting for about 71 percent of the variance in the data. As expected, we find that the extent of effort overrun on projects has been declining during the period 1996-2001 and this decline is positively related to project performance. Contrary to expectation, we find that schedule slippage has marginally increased during the period 1996-2001 and as a result adversely affected project contribution in more recent years.

To understand this result, we examined the data more closely and found two possible reasons that account for this result. First, the firm has steadily increased the number of fixed price projects over the years and has not managed this transition well. As observed in our data, Fixed Price projects tend to suffer from a greater schedule slippage than T&M projects. This result is also corroborated in the regression results where we find that T&M projects, contrary to expectations, tend to be more profitable than fixed price projects. Second, for the fixed price projects we find a greater difference between the team size and person-months measures. While team size reflects the maximum number of persons who worked on a given project, the person-months measure more accurately captures the labor input in the project. At one extreme, if all members of the team worked full time on a given project we will find that team size and person months are perfectly correlated. The difference in the two measures appears when team members work part-time and/or when there is high turnover in the project teams. We find that the difference between the two measures has increased over the years suggesting that an increase in project team turnover might account for the increase in schedule slippage. During the period

\footnote{We did not include an interaction term with capabilities and time as a continuous measure since this implies the assumption that capabilities change linearly over time. We found no theoretical or empirical basis for such an assumption. Note that our results are robust to including year as a continuous variable. However, we were unable to include both interactions in the same model. The high correlations between the two interaction terms created estimation difficulties. Similarly, interacting both capability measures with the same year (i.e., 1996 or 2001) created estimation problems since they were highly correlated. We switched the years to avoid this problem. Just in the interest of reporting all results, an interaction of schedule slippage with 2001 is negative and significant which is consistent with our interpretation.}
1996-2001, the attrition due to employee resignation fluctuated between 9-16 percent. Increase in turnover directly contributes to schedule slippages since there is a setup cost when new employees enter a project mid-way. Some time may be lost in getting the newer employees on board the project that may lead to greater schedule slippage.

6. Discussion

Our results are broadly supportive of our hypotheses that capabilities contribute positively to project level performance. We found limited evidence that client-specific capabilities were positively related to project performance. In the partial model, new projects, on average, result in approximately 19 percent lower contribution than repeat projects after controlling for input and project characteristics. This effect disappears, however, when the project management capabilities are introduced into the equation. It appears that the dummy variable capturing repeat or new clients is too coarse a measure to adequately reflect client-specific capabilities. Ideally, we would have liked to use a count of all the projects undertaken for a client during the entire course of interaction with that client (and not just a count of projects within the sample period) to measure client-specific capabilities. Unfortunately, the firm was unable to provide this data. Nevertheless, the significance of the client-specific capabilities measure in the partial model is indicative and might be useful to examine carefully in future research. In addition, we have reason to believe that the client fixed effect might also be picking up some variance in the learning associated with repeated interaction with clients. This interpretation is consistent with past studies that attributed fixed effects to persistent capability differences (Henderson & Cockburn, 1994).

Project management capabilities were generally predictive of higher project contribution. Two of the three dimensions of project management capabilities were statistically significant. A
one percent increase in schedule slippage resulted in a 0.08 percent drop in project contribution. A one percent increase in effort overrun causes a 0.02 percent drop in project contribution. It seems that the effect of effort overruns just increases labor cost and causes less damage to project contribution. Our results suggest that although both types of capabilities, namely schedule estimation and management and effort estimation are significantly related to performance, the former makes a higher marginal contribution to performance. These results enable firms to assess the tradeoffs and decide whether to meet the project milestone by adding more resources or deliver the project with a delayed schedule.

Finally, we found some evidence for the evolution of capabilities over time and its impact on project contribution. We found that a tighter control of effort overrun in projects in 2001 resulted in better project contribution as compared with projects in 1996-2000. On the other hand, we found that an increase in schedule slippage from 1996-2001 has had a significant negative effect on project contribution. In sum, it seems that the firm needs to focus significant efforts on improving its capability to tightly manage schedule slippages.

We believe our study of firm capabilities and their effects on performance in the context of the software services industry has raised several important issues. First, our study highlights that identifying the capabilities that are the sources of performance differences need to be contextually grounded. Each industry is driven by its own demand- and supply-side economics. It is important to take this into account in identifying and measuring capabilities. It seems that there are indeed significant differences in the way the same firm deploys its resources, both

---

17 The semi-elasticities for the coefficients on the capabilities measures were computed as $m_i = \frac{\partial F(X, i)}{\partial X_{ij}}$ with all the variables fixed at their means.

18 We avoid calculating semi-elasticities for the interaction terms. The typically high correlations between the main effect and the interaction effect result in imprecisely estimated coefficients. As a result, computing accurate economic significance of these coefficients becomes difficult.
across projects and over time. There is reason to take more seriously Penrose’s (1959) key insight that resource productivity differences lie at the heart of firm capabilities.

Second, it would appear that measuring capabilities at more micro levels within the firm is quite promising. It not only helps better estimate their economic significance but also provides clear guidelines to the firm on where and how it needs to improve its capabilities. Measures of capabilities at the aggregate firm-level, while useful in identifying between-firm differences, provide little understanding of the micro-foundations of such inter-firm differences. Our analyses also show that the marginal returns to different capabilities are not uniform. Given scarce managerial resources, it is useful for firms to first identify the capabilities that provide the highest marginal returns to performance and then direct the bulk of its resources to acquiring them. For example, in our dataset, if we assume the marginal cost of acquiring different project management capabilities is the same, then the firm would be well advised to expend resources to tightly manage schedule slippages. Improvements in this capability promise to provide higher marginal improvement in project contribution performance.

Third, this paper adds to the important earlier work on capability measurement and performance done by Henderson & Cockburn (1994), McGrath et al. (1995), and Makadok & Walker (2000). While Henderson & Cockburn (1994) measured firm capabilities and studied their influence on its research productivity, McGrath et al. (1995) examined the significance of the antecedents to firm capabilities on perceptual measures of performance such as meeting staffing objectives, level of client satisfaction, and so on. More recently, Makadok & Walker (2000), in line with our work identified and measured the critical capabilities in the savings and loan industry and found that forecasting ability is a significant driver of firm performance. Along the same lines, we establish a clear link between the development of firm capabilities and
financial performance at the project level in the software services industry. We believe that the benefit of firm capabilities should not just get reflected in operational or perceptual performance but also in terms of its ultimate financial performance as evidenced here.

7. Limitations and directions for Future Research

In this paper we have attempted to empirically study capabilities and how they affect performance in the context of a software services firm. However, like any study, our study also has some limitations. First, it is based on a single service industry with its own peculiar characteristics. It is not clear to what extent the substantive results of this paper are generalizable across industries. At the same time, as we have asserted above, capabilities are usually context-specific, i.e., industry specific, and capabilities that are generalizable across industries are likely to be overly abstract. A second limitation is that our study is based on data from a single firm. Ideally, we would have liked to include data from a few more firms. However, getting access to such detailed data of great competitive significance is a difficult challenge. In our case, this involved several years of data collection, ongoing negotiations with the firm concerned, and the signing of non-disclosure agreements. Third, since our analysis is based on data from only one firm we could not make any explicit inter-firm comparisons of competitive advantage. However, the firm is among the top five firms in the industry on several dimensions such as growth, total revenues, and so on. This gives us some confidence for drawing the capabilities-performance link.

In spite of some of the data limitations outlined above, some of the unique aspects of our dataset outweigh some of these disadvantages. First, the data on resource inputs and capabilities allow us to empirically measure Penrose’s (1959) key distinction between available resources (factor inputs) and how they are deployed (i.e., productivity of resources). Second, the data on
project-level contribution performance suffers from relatively fewer problems associated with aggregate accounting data. Finally, our design allows us to combine the depth and richness of longitudinal, single firm case studies with the rigor of large-sample empirical estimation.

Lastly, our identification of capabilities in the software services industry is by no means comprehensive. For instance, people related capabilities are conspicuously absent from our study. Unfortunately, we were unable to obtain data on skills and qualifications of project team members. In future research, we hope to deal with some of these limitations.

In conclusion, the paper attempted to take an initial step in teasing out the importance of capabilities and estimating their impact on performance. We hope that the spirit of this paper in advocating the importance of contextually grounded empirical studies of firm capabilities will spur further research along these lines in other industries. The shift in research focus from whether or not capabilities matter to “what” capabilities matter and “how” they matter, promises to enrich research on capabilities.
References


<table>
<thead>
<tr>
<th>Variable</th>
<th>Mean</th>
<th>S.D.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log(Contribution)</td>
<td>4.809</td>
<td>1.274</td>
<td>0.993</td>
<td>8.327</td>
</tr>
<tr>
<td>Log(Size)</td>
<td>6.411</td>
<td>1.521</td>
<td>1.611</td>
<td>10.234</td>
</tr>
<tr>
<td>Log(Team size)</td>
<td>2.078</td>
<td>0.670</td>
<td>0.693</td>
<td>4.700</td>
</tr>
<tr>
<td>Log(Person months)</td>
<td>3.049</td>
<td>1.056</td>
<td>0.000</td>
<td>6.327</td>
</tr>
<tr>
<td>Log(Duration)</td>
<td>4.847</td>
<td>0.612</td>
<td>2.890</td>
<td>6.256</td>
</tr>
<tr>
<td>Customer type</td>
<td>0.238</td>
<td>0.427</td>
<td>0.000</td>
<td>1.000</td>
</tr>
<tr>
<td>Process defects</td>
<td>0.605</td>
<td>1.781</td>
<td>0.000</td>
<td>19.000</td>
</tr>
<tr>
<td>Schedule slippage</td>
<td>0.038</td>
<td>0.150</td>
<td>-0.342</td>
<td>1.369</td>
</tr>
<tr>
<td>Effort overrun</td>
<td>-0.022</td>
<td>1.364</td>
<td>-12.298</td>
<td>6.990</td>
</tr>
<tr>
<td>Contract type</td>
<td>0.619</td>
<td>0.487</td>
<td>0.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>
Table 2  Pearson's correlation matrix of variables

<table>
<thead>
<tr>
<th>Variables</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log(Contribution)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Log(getSize)</td>
<td>0.489*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Log(Team size)</td>
<td>0.609*</td>
<td>0.679*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Log (Person months)</td>
<td>0.675*</td>
<td>0.670*</td>
<td>0.741*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Log(Duration)</td>
<td>0.459*</td>
<td>0.536*</td>
<td>0.470*</td>
<td>0.639*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer type</td>
<td>0.100</td>
<td>0.169*</td>
<td>0.177*</td>
<td>0.235*</td>
<td>0.195*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process defects</td>
<td>-0.319*</td>
<td>-0.638*</td>
<td>-0.302*</td>
<td>-0.411*</td>
<td>-0.276*</td>
<td>-0.117</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schedule slippage</td>
<td>-0.138</td>
<td>0.026</td>
<td>0.034</td>
<td>-0.052</td>
<td>-0.045</td>
<td>0.087</td>
<td>-0.028</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Effort overrun</td>
<td>-0.077</td>
<td>-0.037</td>
<td>0.033</td>
<td>0.043</td>
<td>-0.002</td>
<td>0.148*</td>
<td>0.006</td>
<td>-0.142</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contract type</td>
<td>-0.254*</td>
<td>-0.080</td>
<td>-0.117</td>
<td>-0.059</td>
<td>-0.038</td>
<td>0.023</td>
<td>0.115</td>
<td>-0.102</td>
<td>-0.138</td>
<td></td>
</tr>
<tr>
<td>Year</td>
<td>0.120</td>
<td>-0.081</td>
<td>0.013</td>
<td>-0.156*</td>
<td>-0.249*</td>
<td>-0.375*</td>
<td>0.113</td>
<td>-0.140*</td>
<td>0.043</td>
<td>-0.085</td>
</tr>
<tr>
<td>Effort overrun * Y2001</td>
<td>0.089</td>
<td>0.009</td>
<td>0.056</td>
<td>0.081</td>
<td>0.031</td>
<td>0.028</td>
<td>0.007</td>
<td>-0.195*</td>
<td>0.544*</td>
<td>-0.083</td>
</tr>
<tr>
<td>Schedule slippage * Y1996</td>
<td>0.027</td>
<td>-0.044</td>
<td>-0.006</td>
<td>0.077</td>
<td>-0.031</td>
<td>0.136*</td>
<td>0.034</td>
<td>0.353*</td>
<td>0.052</td>
<td>-0.080</td>
</tr>
</tbody>
</table>

Note:- Stars indicate coefficients significant at 0.05 level or lower.
Table 3 Coefficient estimates of fixed effects panel regression analyses

<table>
<thead>
<tr>
<th>Independent variables</th>
<th>Pred. Sign</th>
<th>Dependent variable: Log(Contribution)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Model 1</td>
</tr>
<tr>
<td>Customer type</td>
<td>-</td>
<td>-1.027 ***</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.316)</td>
</tr>
<tr>
<td>Process defects</td>
<td>-</td>
<td>-0.038</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.044)</td>
</tr>
<tr>
<td>Schedule slippage</td>
<td>-</td>
<td>-2.850 *</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(1.431)</td>
</tr>
<tr>
<td>Effort overrun</td>
<td>-</td>
<td>-0.172 *</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.080)</td>
</tr>
<tr>
<td>Schedule slippage * Y1996</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Effort overrun * Y2001</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Controls</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contract type</td>
<td>-0.321 †</td>
<td>-0.245</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.190)</td>
</tr>
<tr>
<td>Project size (FP)</td>
<td>0.050</td>
<td>0.069</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.082)</td>
</tr>
<tr>
<td>Team size</td>
<td>0.349 †</td>
<td>0.373 *</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.204)</td>
</tr>
<tr>
<td>Person months</td>
<td>0.454 ***</td>
<td>0.399 ***</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.154)</td>
</tr>
<tr>
<td>Duration</td>
<td>0.387 *</td>
<td>0.453 *</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.200)</td>
</tr>
<tr>
<td>Domain controls</td>
<td>Sig.</td>
<td>Sig.</td>
</tr>
<tr>
<td>Platform controls</td>
<td>Sig.</td>
<td>Sig.</td>
</tr>
<tr>
<td>Year effects</td>
<td>Sig.</td>
<td>Sig.</td>
</tr>
<tr>
<td>Constant</td>
<td>0.695</td>
<td>0.513</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.794)</td>
</tr>
<tr>
<td>Adjusted R-square</td>
<td>0.592</td>
<td>0.636</td>
</tr>
<tr>
<td>N</td>
<td>152</td>
<td>152</td>
</tr>
<tr>
<td>F-Value</td>
<td>25.22 ***</td>
<td>25.07 ***</td>
</tr>
<tr>
<td>F-Test for client fixed effects</td>
<td>2.09 ***</td>
<td>2.31 ***</td>
</tr>
</tbody>
</table>

* Variables are logged

*** p<=0.001; ** p <=0.01; * p<=0.05; † p<=0.10

Standard errors are reported in parentheses