MAINTAINING A COMPETITIVE ADVANTAGE IN THE FACE OF A RADICAL TECHNOLOGICAL CHANGE: THE ROLE OF STRATEGIC LEADERSHIP AND DOMINANT LOGIC

August 23, 1996

Allan Afuah

University of Michigan Business School, 701 Tappan
Ann Arbor, Michigan 48109-1234, USA
e-mail: afuah@umich.edu, Phone: (313) 763 3740. FAX (313) 936 0282

Abstract:

The failure of industry leaders to maintain their leadership positions following a technological discontinuity has been attributed to the lack of a strategic incentive to invest and competence-destruction. Both reasons fail to explain why many industry leaders have been the first to embrace radical technological innovations. This paper argues that a firm's ability to embrace and exploit a discontinuous change is a function of the extent to which its top management's dominant logic allows it to recognize the potential of the change. Using the case of RISC technology in the computer industry, the paper shows that incumbents were first to embrace a discontinuous change when top management did not hold the industry and organizational dominant logic. The results suggest that top management logic, not organizational competence-destruction, may be what makes the difference in the face of a discontinuous technological change. They also offer a plausible explanation for why, as some economists would argue, an incumbent facing a radical technical change cannot just replicate the capabilities of a new entrant by creating an internal venture group or skunk works within itself.

1. INTRODUCTION

The record of industry leaders in maintaining their leadership positions following a technological and/or market discontinuity is not flattering. Accounts of RCA being replaced by Texas Instruments as vacuum tubes gave way to transistors; of Sony and Matsushita replacing Ampex as VCRs moved from TV studios to the home; and of Microsoft replacing IBM in software as desktop and home computing took over from mainframe computing, are now legendary. Two major streams of research have explored this phenomena. The first, grounded in

neoclassical economic theory, suggests that new entrants replace incumbents because they have greater strategic incentives to invest in radical innovation (Gilbert and Newbury, 1982; Reinganum, 1983; Henderson, 1993). Incumbents may be reluctant to invest, for example, for fear of cannibalizing existing products. This is the so-called *strategic incentive* [to invest] view (Henderson, 1993). The second, originating from organizational theory, suggests that incumbents lose their leadership positions in the face of radical innovations because such innovations are competence-destroying to them, and they are therefore less efficient in introducing and exploiting these innovations than new entrants (Cooper and Schendel, 1976; Tushman and Anderson, 1986; Foster, 1986; Henderson and Clark, 1990; Henderson, 1993). That is, even if an incumbent decides to invest in an innovation, its existing capabilities may be a handicap in its efforts to exploit the innovation.

Neither stream of research, however, explains why many industry leaders have been the first to embrace radical innovations and maintained their leadership positions. This paper argues that our understanding of why firms fail or succeed in the face of radical technological change may be incomplete without an exploration of the role of *strategic leadership*. Strategic leadership focuses on those upper level managers with overall responsibility for the firm (Hambrick, 1989). The thesis of the paper is that an incumbent is more likely to introduce and exploit a radical innovation if its top management's *dominant logic* (Prahalad and Bettis, 1986; Bettis and Prahalad, 1995), or *view of the world* (Finkelstein and Hambrick, 1990)[1] allows it to recognize the potential of the innovation. This logic, in turn, depends on top management's experiences, as well as organizational and industry logic (Nelson and Winter, 1982; Spender, 1989). It is only after management recognizes the potential of an innovation that it can effectively invest in it or fear cannibalization.

Exploring the role that top management plays in the ability of firms to innovate is not new. Bantel and Jackson (1989), for example, found that innovation was greater in banks headed

[1]Dominant logic has also been referred to by names such as *givens* (March and Simon, 1957) or *structure, mental frames* and numerous other names. Walsh (1995) provides a list of these different names.

by more educated managers who came from diverse functional backgrounds. Most of this research, however, has not distinguished between incremental and radical innovation. This paper focuses on the role of top management in exploiting radical innovation. It explores the role of top management, industry, and organizational logic in a firm's ability to recognize the potential of an innovation. Using the cases of DEC, Sun, IBM, and HP which adopted Reduced Instruction Set Computer (RISC) technology, the paper shows that incumbents were first to embrace the new technology and maintain their leadership positions when top management did not hold the prevailing industry dominant logic. The study provides strong evidence for the role of top management dominant logic in a firm's ability to exploit technological change, and suggests that it is not so much organizational competence-destruction or lack of incentive to invest in an innovation as it is the type of top management logic that deprives a firm of introducing and exploiting radical technological innovations.

2. BACKGROUND

Some explanations for why incumbents lose their leadership positions

Strategic incentive view

The debate over who is most likely to innovate dates back to Schumpeter (1934) who suggested that small entrepreneurial firms were more likely to be the source of most innovations. Later, he changed his view and suggested that large firms with some degree of monopoly power were more likely to be the source of technological innovation given their access to skilled labor, capital, and other complementary assets (Schumpeter, 1950). Later research suggested that whether incumbents or new entrants are likely to fare well is a function of the type of innovation. If an innovation is radical (drastic) in that it renders the existing technology or products non-competitive, incumbents are less likely to invest in it for fear of cannibalizing their existing products (Reinganum, 1983, 1984; Henderson, 1993). New entrants have the *incentive to invest* since they have less to lose. If, however, the innovation is incremental in that existing products stay competitive (Arrow, 1962; Tirole, 1988), incumbents are more likely to invest in the

innovation and therefore more likely to maintain their competitive positions (Gilbert and Newberry, 1984a, 1984b; Henderson, 1993).

There are other factors that influence a firm's incentive to invest in an innovation other than the degree to which the innovation destroys the firm's market power. The fear of erosion of organizational political power may also decrease the incentive to invest in an innovation (Ferguson and Moris, 1993). Emotional attachment to the old technology can be equally handicapping to a firm's ability to invest in an innovation (Burgelman, 1994).

Competence-destruction view

While the strategic incentive view explains why firms may not invest in certain innovations, it does not explain why firms that invest may still fail. This is where the *competence-destruction* view comes in. It suggests that successful introduction and development of an innovation also depends on the impact of the innovation on the organizational capabilities of the firm. If an innovation is radical in that the knowledge required to exploit it is drastically different from existing knowledge—that is, competence-destroying—the firm's existing capabilities may not only be useless, they may actually be a handicap to its introduction and development of the innovation (Cooper and Schendel, 1976; Tushman and Anderson, 1986; Leonard-Barton, 1992). This would allow the more nimble new entrants to outperform incumbents. Since the knowledge in question can be technological or market, there may be cases where the innovation destroys technological knowledge but leaves market knowledge intact, allowing incumbents to take advantage of their established marketing capabilities to exploit innovations. If the innovation requires entirely new markets, incumbents may also lose out to new entrants even when the technological knowledge is not competence-destroying (Christensen and Bower, 1996).

If the innovation is incremental in that the knowledge required to exploit it builds on existing knowledge and capabilities, incumbents are likely to continue to dominate since they already have the capabilities which new entrants have to build from scratch (Nelson and Winter, 1982; Tushman and Anderson, 1986; Banbury and Mitchell, 1995).

Strategic leadership view

Both the *strategic incentive* and *competence-destruction* views do not explain why some incumbents are the first to embrace and exploit radical innovation. Nor do they explain why, as some economists would argue, an incumbent facing a competence-destroying technological change cannot just replicate the capabilities of a new entrant by creating an internal venture group or skunk works within itself (Henderson, 1993). The strategic leadership view (Hambrick, 1989; Bantel and Jackson, 1989) provides some answers to such questions. In this view, a firm's dominant coalition (top management or upper echelon) takes the decision to invest in an innovation.[2] It also takes the decisions to assume the roles of champion or sponsor that can be critical to the introduction and development of an innovation. Its effectiveness in taking such decisions is a function of the extent to which it recognizes the potential of the innovation. How well these managers recognize the potential of the innovation is, in turn, a function of the top management's *dominant logic* (Prahalad and Bettis, 1986; Bettis and Prahalad, 1995), or *view of the world* (Finkelstein and Hambrick, 1990). The role of strategic leadership in enabling a firm to maintain its leadership position in the face of a technological change is best seen by exploring the information collection and processing process that leads up to the decision by top management to invest in the innovation, and give it the backing that it needs.

3. THE FRAMEWORK: COLLECTION AND PROCESSING OF INFORMATION ON INNOVATION

The activities that lead to top management's decision to embrace an innovation can be viewed as a problem-solving process in which the managers collect and evaluate information in order to recognize the potential of the innovation and the rationale behind its exploitation (Galbraith, 1973; Roberts, 1988). As shown in Figure 1, top management's ability to collect and process this information is a function of its dominant logic (Hambrick and Mason, 1984; Prahalad and Bettis, 1986; Hambrick, 1989; Bettis and Prahalad, 1995), and of the type of

[2] If the decisions are taken by lower level managers, they reflect upper level manager's values (Hambrick and Mason, 1984), and structures, processes or the strategies that these upper level managers installed.

innovation it faces (Tushman and Anderson, 1986; Henderson and Clark, 1990). This top management dominant logic, in turn, is a function of its experiences, industry and organizational dominant logic.

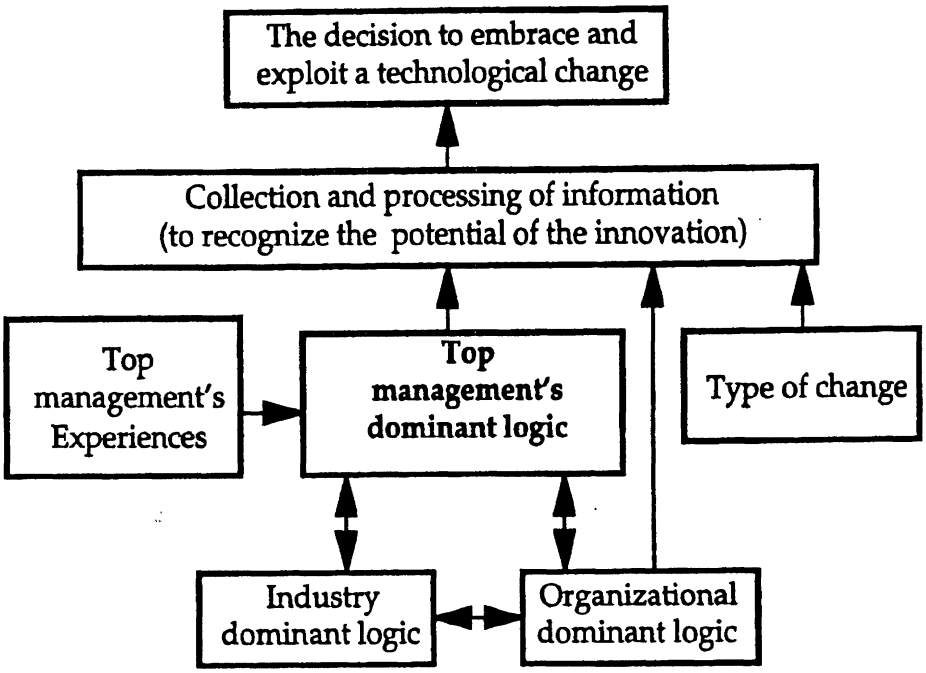

Figure 1: Top management's dominant logic and recognizing the potential of an innovation

Top management's dominant logic

Every manager brings to any innovation circumstance, a mental model that reflects his/her beliefs, knowledge, assumptions, presuppositions, and values (March and Simon, 1958; Hambrick and Mason, 1984). This model has been called different names by different researchers—for example, *givens* (Simon, 1955), *frame of reference* (March and Simon, 1958), *recipes* (Spender, 1989), and *dominant logic* (Prahalad and Bettis, 1986). (Walsh (1995) provides an excellent review of the models.) It defines the frame within which a manager is likely to scan for information and approach problem-solving. It is a function of three factors: 1) industry dominant logic, 2) organizational dominant logic, and 3) managers' experiences (see Figure 1).

Spender's (1989:69) findings that there exist a "shared knowledge-base that those socialized in an industry take as familiar professional common sense" provide evidence for industry logic, or recipes, and suggests that top management logic is likely to be influenced by such a shared knowledge-base. "The recipe suggests a pattern of appropriate resources as well as a way of looking at the world", (Spender, 1989:192). From a technological innovation perspective, such a dominant industry logic usually emerges following the emergence of a dominant design. A dominant design is one whose major components and underlying core concepts don't vary substantially from one product model to the other, ushering the beginning of a period when critical technological problems are defined, product features established, and when attention turns to routine product improvements (Anderson and Tushman, 1990; Tushman and Rosenkopf, 1992; Utterback, 1994). Prior to that, when the innovation is still in a state of flux with a high degree of market and technological uncertainty, such a recipe is not yet possible as there is still plenty of uncertainty as to what should go into the product and just what customers want (Tushman and Rosenkopf; Utterback, 1994).

Just as influential on top management logic is organization logic. Following the emergence of a dominant design, for example, when industry recipes are being established, firms develop organizational structures, systems, employees, and problem-solving strategies (rules and routines) that become the organizations memory where its view of the world (logic) is stored (Nelson and Winter, 1982; Walsh and Ungson, 1991). These organizational recipes are used to offer products based on the dominant design. The organization modifies this recipe as it learns, but ever so slightly. Organizational success reinforces not just the organization recipe, but also the managerial dominant logic (Bettis and Prahalad, 1995).

Top management's experiences and agenda also help shape its logic. The longer its tenure in an industry or organization, or the better current organizational performance, the more likely the management is to hold the industry and organizational logic, and therefore maintain the status quo (Hambrick, Geletkanycz, and Fredrickson, 1993). Functional and project tenure also matter. A manager's logic may reflect the kinds of projects that it has worked on. It is also going to be a

function of his/her personal experiences in previous organizations or functions. With this logic, top management collects and processes information in order to recognize the potential of innovations and exploit them.

Collection and processing of information—Recognizing the potential of an innovation

Of the universe of data that is available to top management, its logic would allow it to only look in certain places—*communication channels* (Galbraith, 1973; Arrow, 1974; Nelson and Winter, 1982), *field of vision* (Hambrick and Mason, 1984)—for the information it needs for the current situation. The logic also leads it to selectively gather the information that it perceives it needs; that is, top management *filters* the information to keep out what it believes it does not need (Arrow, 1974; Hambrick and Mason, 1984).

After collecting the information, top management must process it in an effort to recognize the potential of the innovation. There are two parts to recognizing the potential of an innovation. The first is to understand the rationale behind using the new knowledge from the change to offer new products. That is, to answer the question: does the new method for building the mousetrap make sense logically? Take a firm that made electromechanical cash registers in the late 1960s when electronic cash registers were about to invade the old technology. To understand the rationale behind the new technology, the firm would need to understand the concepts that underpin transistors and how they can be connected to other electronic components—instead of the usual mechanical gears, ratchets and levers of existing electromechanical machines—to perform calculations and more. The second part to recognizing the potential of an innovation is to identify the different applications for the innovation. In the cash register example, the question becomes: will the electronic cash registers perform only the activities presently performed by their electromechanical predecessors (only better, and/or at lower cost), or would they, in addition, give customers itemized printouts of purchased items and take inventory of items sold?

The more successful a firm is in solving a problem using certain channels, filters, and problem-solving strategies, the more its top management will turn to them each time it has a

problem. Success reinforces the logic. Changes to a management's logic are triggered by a problem or crisis (March and Simon, 1958; Prahalad and Bettis, 1986). Such a crisis may force them to seek information in different channels with filters that have been modified to reflect the crisis. Problem-solving also changes reflecting the extent to which the management has unlearnt the old problem-solving strategies and learnt the new ones.

Type of technological change

How top management collects and processes information on a technological change is also a function of the type of innovation. If the innovation is incremental in that the knowledge required to exploit it builds on existing knowledge and capabilities, a manager's dominant logic is an asset since it guides the manager to the right information sources and the right problem-solving strategies (Nelson and Winter, 1982; Tushman and Anderson, 1986; Banbury and Mitchell, 1995). If, however, the change is radical in that the knowledge required to exploit it is drastically different from existing knowledge, the firm's dominant logic may not only be useless, it may actually be a handicap (Cooper and Schendel, 1976; Tushman and Anderson, 1986; Foster, 1986; Utterback, 1994). This is because such knowledge is likely to require different information channels, filters and processing mechanisms from those that existing dominant logic reinforces (Nelson and Winter, 1982; Henderson and Clark, 1990).

4. RESEARCH METHOD

To explore the role of top management logic—and assorted industry and organizational logic—in a firm's ability to introduce and exploit innovation, the adoption of RISC technology by computer workstation makers was chosen as the research vehicle. Several steps were taken to assure construct validity, external validity, and reliability (Cook and Campbel, 1979; Yin, 1989).[3] First, evidence was sort from multiple sources. The first source was the field-based interviews of project managers, lead engineers and senior managers of 65 RISC projects in the

[3]Internal validity issues are covered in the analysis sections that follows the case narratives.

semiconductor and computer industries. University-based researchers who were instrumental to the commercialization of RISC were also interviewed. The second source of data was archival. It consisted of both manual and electronic searches. In the manual search, company records, consulting reports, and key electrical engineering and computer sciences practice journals—*ACM (Association of Computing Machinery) Communications, UNIX Review, Electronics, IEEE Computer, IEEE Spectrum and Computer Architecture News.*—were searched for RISC data. The electronic search consisted of Lexis/Nexis searches of all the files in the computers and communications (CMPCOM) library.[4] The archival data was used to construct a technical history of RISC which was verified by industry experts, and checked during the interviews. The final source was consulting reports from Dataquest, IDC and Workstation Laboratories Inc.

Second, multiple cases were pursued—six cases in detail, although only four are chronicled below—to allow for both the literal and theoretical replication that are essential if the propositions outlined in the framework above are to be reliably explored (Yin, 1989). For each of those propositions, several pieces of evidence were sort from each of the six cases.

Third, in all of the interviews, the same protocol was followed.

5. RISC TECHNOLOGY AND INDUSTRY LOGIC

RISC is an innovation in designing the central processing unit (CPU)[5] of a computer that considerably speeds up the processor. In order to understand the rationale behind this technology and why it was so difficult for some computer industry leaders to embrace it, it is best to start with some technical history. In the 1960s and 1970s, several factors, two of which are described here, influenced computer design. In the first place, main memory—the gut of the computer in which programs are normally stored during execution—was very slow and expensive. Consequently, a primary goal for computer designers was to minimize how much of this memory

[4]Specifically, once in the ALLNWS file of the CMPCOM library, the following command was used: SECTION(WORKSTATION) OR HEADLINE(WORKSTATION) OR BYLINE(WORKSTATION) OR DATELINE(WORKSTATION) AND DATE=19XX, where19XX is the year being searched.

[5]The CPU is sometimes referred to as the brain of the computer because it does all the calculations and controls all the electrical signals of the computer.

was needed by any program during its execution, and how many times the CPU had to access the main memory for instructions during execution of the program.

In the second place, most programmers used assembly language[6]—a somewhat awkward non-expressive computer language that uses a mnemonic for each instruction, and is difficult to program in. To improve the efficiency of these programmers, computer designers shifted some of the programming burden from programmers to hardware by designing computers that did as much per command from the programmer (line of assembly language code) as possible. An example serves illustrate the influence of these factors on computer design. To add two numbers **A** and **B**, the programmer could use the [one] complex instruction **Sum R3 A B** to perform the whole task, or the [three] simple instructions **MV R1 A** (move A to register R1), **MV R2 B** (move B to register R2) and **Add R1 R2 R3** (add the contents of Register R1 and R2, and store the sum in register R3) to achieve the same task. The outputs (end results) of the one complex instruction, or of the three simple ones are the same. But in the first case, the programmer only has to write one line of code that takes up one unit of main memory and the CPU only has to access the main memory once. The computer hardware is then charged with the rest of the job needed to give the end result.[7] In the second case, the programmer has to write three lines of code which take up three units of main memory and the CPU must access the main memory three times. Given that memory was expensive and slow, and computer architects wanted to make life easier for programmers, it is easy to see why designers would choose the one complex instruction over the three simple ones. The one complex instruction is also said to be semantically rich since with just this one command, a programmer can tell the computer to do what would ordinarily take three instructions. Since, in issuing only one software command, the programmer charges

---

[6] This is a low level language (than say, FORTRAN) that consist of mnemonics such as LDA (for Load Register A), and translates directly (on a one-to-one basis) to machine language code such as 082 which the computer's CPU recognizes.

[7] An IBM invention called microprogramming, facilitated the use of complex instructions. A microprogram was a lower level group of instructions that resided on the CPU and "interpreted" the complex instructions for execution by the processor circuitry. This microinstructions, although technically software, were implemented on hardware right on the processor and could be fetched and executed very quickly. Although part of the processor hardware, microprograms could be altered after the computer had been designed, making them a valuable tool in fixing bugs.

the computer's hardware with the minute details that allow the computer to produce the same results as the three simple instructions, it is said that the burden of closing the semantic gap between programmer and machine is being moved from software to hardware. Effectively, semantically-rich instructions, the so-called "complex instructions", were preferred in computer design. This resulted in a technological trajectory of sorts, as shown in Figure 2 where the more complex (semantically-rich) an instruction, the better. Computers with such instructions are so-called CISC[8] (Complex Instruction Set Computer) machines.

As designers pursued this trajectory, some of the factors that had perpetuated CISC technology were improving. First, main memory got cheaper and faster, relaxing the constraint on memory space and the need for semantically rich instructions. Second, systems programmers increasingly used higher level languages (as opposed to assembly language) which compilers translated into simple code, also reducing the need for semantically rich instructions.[9]

[8] Although so-called Complex Instruction Set Computers (CISC) existed a long time before RISC, the acronym CISC was coined by Professor David Patterson of UC Berkeley and his students *after* they had already popularized "RISC" in the computer architecture community.

[9] Other factors also had an impact on instruction set design. For example, optimizing compilers improved drastically, allowing higher level languages to be translated to very simple and efficient code. This includes efficiently replacing complex instructions with simpler faster ones. Innovations like cache memory also gained more acceptance, as its implementation became cheaper and more efficient. Thus, those portions of memory that the computer needed frequently during the execution of a program could be stored in cache, therefore reducing the need to access main memory and the need for semantically rich instructions.

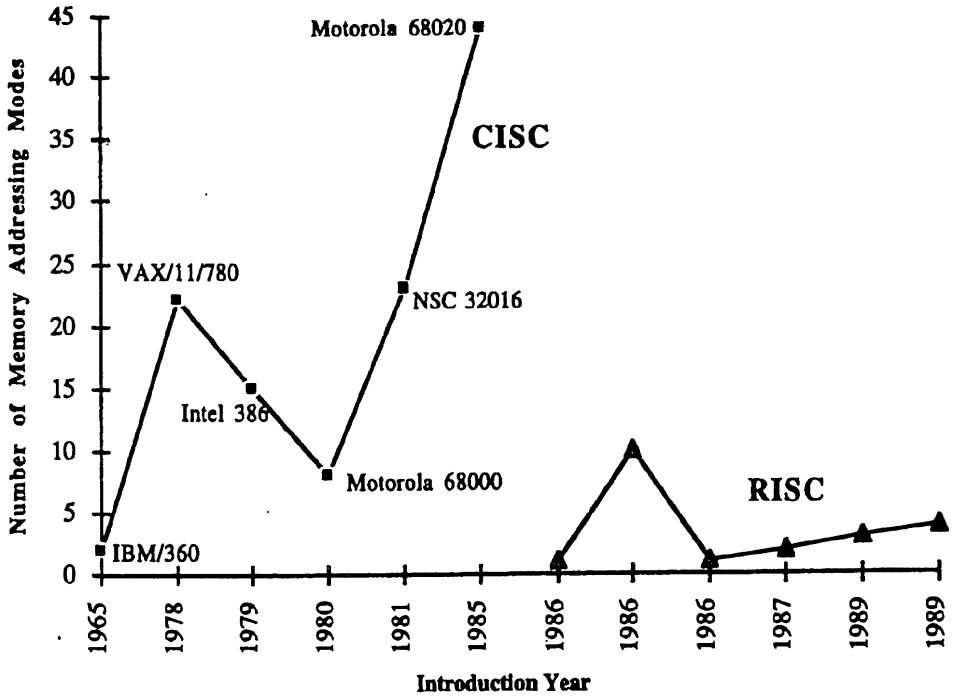

Figure 2: CISC technological trajectory and the discontinuity to RISC (Data source: Mashey, 1992)[10]

Given the changes in both factors, some computer scientists began to question the traditional approach to computer design that called for semantically-rich instructions. They suggested that computer design should use simpler instructions, not complex ones. Figure 2 illustrates this discontinuity in direction. This was a complete reversal in direction—from a core concept that had been "the more semantically-rich an instruction the better", to one where the simpler the instruction, the better. The burden of closing the semantic gap between programmer and machine was being moved back from hardware to software. This was very difficult for many computer designers to accept. In 1980, the most popular computer architecture when advocates of RISC started preaching the virtues of the new design philosophy, DEC's VAX, was—in the industry's parlance—the most CISCy (Figure 2). Computer designers as well as members of the computer science community had difficulties accepting the ideas that did not match the trend of

[10]Number of Memory Addressing Modes is just one of several proxies for complexity of an instruction. See Mashey (1992) for others.

"the more semantically rich the instructions, the better". For example, when Professor David Patterson, a RISC pioneer who would play a critical role in the evangelism of the technology, submitted a paper on RISC to "a major computer journal", it was rejected with a note suggesting: "that is not the way computers are designed".[11]

Professor John Hennessy, another RISC pioneer, experienced similar resistance with his RISC ideas. His experimental RISC design at Stanford University and suggestion that it could be turned into a viable commercial product were dismissed as an "academic fad" by CISC microprocessor makers. These rejections are part of what would inspire him to co-found MIPS Computers, to demonstrate that RISC was commercially viable.[12]

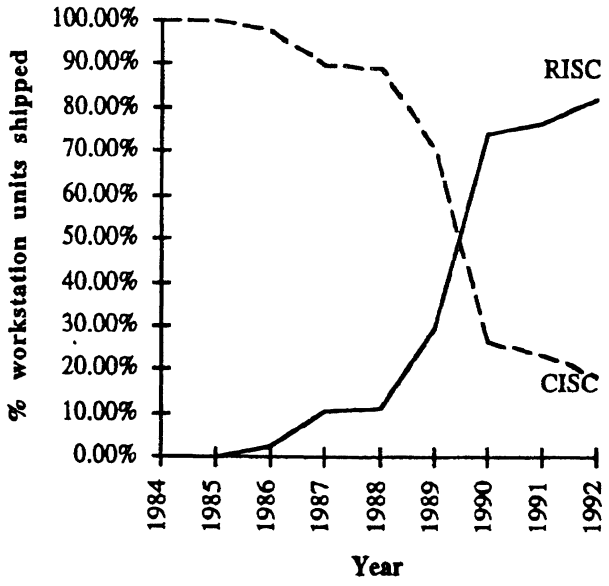

Figure 3: Diffusion of RISC in workstations.

What we now know is that with their simpler instructions, RISC microprocessors take up less chip real estate, ceteris paribus. This simplicity, coupled with the space saved, allow designers to take advantage of incremental innovations in computer design to build RISC processors that are faster than their CISC predecessors. Since simple instructions also mean that

[11]Interview with Professor Dave Patterson
[12]Interview with Professor Henneessy

the hardware has to do less per instruction, some of the burden of closing the semantic gap between programmer and machine is shifted from hardware to software. The faster RISC microprocessor and the change in relationship between software and hardware presented CISC incumbent workstation makers with architectural innovation problems (Henderson and Clark, 1990) making it difficult for them to understand the rationale behind the changes and their potential. As Figure 3 shows, the major push to convert from CISC to RISC started in 1987 and by 1992, most of the workstations being shipped were RISC-based.

The strategic *incentive to invest* and *competence-destruction* views would suggest that workstation industry leaders (see Table 1) Sun, DEC, Apollo, and HP would be the last to embrace RISC technology, given that it was competence-destroying to them and rendered their CISC workstations non-competitive. While that was the case for DEC, Apollo, and IBM, it was not for Sun and HP. We explore why in the cases that follow.

Table 1: Computer workstation market share (% units) from 1984-1993. Data source: IDC

	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	
Sun	36	26	31	27	33	34	39	39	40	38	
HP		24	16	14	12	23	20	17	17	20	Bought Apollo in 1989
DEC	5	11	8	19	20	22	17	14	12	11	
IBM			3	3	2	1	4	6	7	7	
SGI		4	2	5	2	2	3	3	5	6	Bought MIPS in 1991
NEC				1	1	1	2	3	3	2	
Intergraph				5	5	5	4	3	3	2	
Sony					3	3	3	2	1	1	
Tatung								1	1	1	
Hitachi					1	1	1	0	1	1	
MIPS							1				
NeXT						2	3				Exited in 1991
Apollo	51	25	35	19	14						Bought by HP
Other	8	11	5	8	5	6	4	13	10	11	
Total	100	100	100	100	100	100	100	100	100	100	

6. THE CASES

Digital Equipment Corporation

DEC entered the workstation market in 1985. While all the other workstation vendors used microprocessors from Motorola or Intel in their workstation designs, DEC used the same microprocessor that powered its minicomputers, the microVAX. The microVAX was DEC's single chip implementation of its very successful VAX instruction set architecture. VAX minicomputers were so popular with engineers that even IBM employees secretly bought them for use in their own plants, often covered with IBM chassis.[13] A major strategy in designing the VAX (and proud achievement of the design team) had been replacing instruction sequences with single instructions to make them more semantically rich and save memory space, while facilitating the programming task of assembly language programmers. It was the epitome of a CISC machine; it was, in the language of computer designers, an extremely CISCy machine[14]. As Figure 2 illustrates, of all the major instruction set architectures in existence in the early 1980s, the VAX deviated the most from some of the RISC ideals being advocated at the time by Patterson and his disciples. In essence, RISC advocates were telling the architects of a very successful architecture, the VAX, to do exactly the opposite of what the architects had done to be that successful.

When the rationale behind RISC microprocessor design was advocated by Professor Patterson and Hennessy, DEC's top management showed some of the strongest opposition. As far as they were concerned, designing processors with semantically-rich instructions was still the optimal way to design technically superior computers and the VAX's success was proof enough.

---

[13] The VAX became the standard against which computer architectures were measured. Even today, the SPECint and SPECft measures discussed in chapter 4 are measures of performance relative to a VAX-11/780. A computer speed of 5 SPECint means that the computer in question can run SPEC's integer benchmark five times faster than a VAX 11/780.

[14] One of the reasons pioneer Patterson gave for his devotion to the RISC cause was his experience with the very CISCy VAX.

In a series of debates between 1980 and 1986, DEC officials took on Patterson and other RISC advocates, doing all they could to prove that RISC was no match to the proven CISC design principles. Commenting on Patterson and Ditzel's (1980) seminal paper that outlined the rationale behind RISC and touched off the CISC/RISC debate, Clark and Strecker (1980: 38), DEC's VP for R&D and VP for Computer Architecture and Software, respectively, wrote:

> Anecdotal accounts of **irrational** implementations are certainly interesting. Is it typical, however, that composite instructions run more slowly than equivalent sequences of simple instructions?[15] The author's emphasis in **bold.**

Both these top managers had played key roles in the design of the VAX. They did not believe that strings of simple instructions would do the job of complex dense ones, and do so more efficiently, taking shorter times. They insisted that Patterson's ideas were an academic fad. While the RISC/CISC debate was going on, Patterson and his students designed and produced their experimental *RISC* chip. They tested it and showed that, even in its most basic and unoptimized form, *RISC* was almost twice as fast as the VAX (Fitzpatrick, Patterson et al, 1982; Larus, 1982). DEC dismissed the Berkeley *RISC* as an experimental research project with no value in the real world.

Ken Olsen, DEC's CEO who, ultimately, defined the company's technical direction, did not recognize the threat that RISC posed to the VAX either. In May 1987, responding to a question on what he thought about competitors' allegations that DEC was unenthusiastic about parallel processing because the VAX was not suited to it architecturally, he said:

> "We have one very serious problem with RISC reduced instruction set computing and parallel processing: Anything we come up with has to be better than the VAX. We've made one parallel processor, and we put a lot of money and time into it. But we've downgraded it from being a product to a development project. We tried, and we're still trying, but our enthusiasm has dropped off." Lee (1987)

The "downgraded" projects included two key RISC processor projects that had been started at DEC R&D locations in the west coast of the US, three thousand miles away from the VAX and DEC's power base at "The Mill" in Maynard, Massachusetts. The first was the Titan project at

---

[15]Many industry interviewees suggested that this written statement was "extremely tame" compared to the words used during the heated RISC verbal debates that took place at conferences in those days.

the company's Western Research Center in Palo Alto near Stanford University. The project was headed by Forest Baskett, who had been a professor at Stanford University when Hennessy, a RISC pioneer, was just starting his RISC work there. The Titan, started in 1984, built a high speed RISC processor that by some accounts, ran at twice the speed of DEC's top of the line machine then, the VAX 8700. This was never really considered a commercial project by DEC. Like many ideas from another Palo Alto laboratory—Xerox's PARC—the Titan died in the laboratory.

The other RISC project was entitled PRISM, headed by Dave Cutler in Seattle, Washington. This was a late effort and in June 1988 when DEC finally realized that the threat posed by RISC was credible, PRISM was too far behind to be useful. The project was canceled, DEC chose a microprocessor from start-up MIPS Computers for its workstations, and one year later, Dave Cutler left Digital to head the Windows NT operating system project at Microsoft.

The managerial logic that prevented DEC from recognizing the rational behind RISC and its potential threat to the VAX were reinforced not only by the success of the VAX, but also by some discouraging RISC stories along the way. First, as detailed in the IBM case later, IBM had introduced the first RISC workstation in 1986 and both its technical and market performance, put mildly, were big disappointments. In 1985, HP had announced that all its computers were going to use one architecture: RISC. But HP had not introduced a RISC workstation until 1988, and when it did, its price/performance had not been what one would expect of RISC. PRISM, introduced in 1988, had not rescued a failing Apollo. These were all stories that DEC wanted to hear.

The potential threat that RISC posed to DEC's VAX eventually became real. In 1988, Sun's RISC workstations, first introduced in 1987, were selling very well. Using its RISC technology, Sun offered machines with four times the price/performance of competing DEC CISC machines.

DEC had to do something. Luckily for the company, while its upper-level management had stood behind CISC, some engineering dissenters had been secretly designing a RISC

workstation using MIPS's microprocessor. DEC's management decided to make development of this bootleg product official and promptly acquired a 5% stake in MIPS Computers, the designer of the microprocessor. In 1989, DEC introduced its first RISC workstation which it touted as a "Sun killer". The company saw RISC only as a technology to counter Sun's attack in workstation; not as a technology to give it a competitive advantage in all its computers. It saw it as a UNIX machine and therefore offered the RISC machines only with the UNIX operating system and not the VAX/VMS operating system that most of its customers had learnt to use and developed software to run on it. Its lucrative business would continue to be powered by CISC processors. But when its market share continued to erode, the company finally realized that it needed RISC for all of its computers. In 1993, it introduced its ALPHA RISC technology that would power not only its workstations but its servers and mainframes.[16]

IBM

In the twenty years since it invented RISC, IBM watched HP decide in 1983 to use RISC in all its computer products and become a market leader in minicomputers, Sun Microsystems in 1987 successfully use RISC to maintain its leadership position in workstations, Apple computer introduce the first RISC personal computers in 1993, and Intel use RISC to build supercomputers. In what follows, we attribute this inability to recognize the potential of its invention to its top management's CISC dominant logic.

John Cocke's group, at the IBM Thomas J. Watson Research Laboratory, built and tested the first RISC minicomputer called the 801 in 1979.[17] The machine's performance was a phenomenal 15.2 MIPS compared to 3.5 MIPS for many minicomputers then. But the minicomputer group did not want it. It did not see any use for a new, unproven technology especially given that its CISC mainframe and minicomputer businesses were thriving. What it saw was another "wild idea" from the "ivory tower" as the Thomas J. Watson Laboratory was sometimes referred to by IBM engineers. Around 1980, a group in the Office Products

[16]Increasingly, servers are replacing what used to be called computer mainframes and minicomputers
[17]The 801 was the name of the building in which the RISC project took place.

Division—the group that sold word processors and that would later be home to personal computers and workstations—decided to build an experimental microprocessor, in typical skunk works fashion, called the ROMP (Research/Office Products Division Microprocessor) out of the 801 processor. When the design of the microprocessor started, the state of VLSI technology was such that it was extremely difficult to implement a 32-bit microprocessor on a single chip. To make the ROMP microprocessor fit on a single chip, the design team decided to leave out the cache, a critical factor in attaining RISC's speed. Given that management had not sanctioned RISC for any application in particular, such a decision was tolerable. But the result of leaving out the cache and other features was devastating for the resulting microprocessor. The ROMP was too slow to deserve the label "RISC".

While the design of the ROMP was going on, IBM was thinking of entering the personal computer (PC) market. But when, in 1981, it entered the market, it decided to use an Intel CISC microprocessor and not its RISC microprocessor. The Intel architecture quickly became the industry standard for personal computers. While IBM was shipping personal computers with Intel microprocessors in them and helping get the Intel standard entrenched, its ROMP RISC microprocessor sat on the shelf looking for an application. In 1983, it decided to do something about it. It decided to use it to enter the then fledgling workstation market pioneered in 1980 by Apollo Computers; not personal computers where it continued to ship Intel-based PCs, and not in its mainframe or minicomputers where its could have used the speed of RISC. Its Advanced Technology Group in Austin, Tx, was charged with this task of building a workstation using the very slow ROMP.[18] It built a workstation called the PC/RT (Personal Computer/ RISC Technology) which it introduced in 1986. The workstation was such a performance laggard that even IBM could not make a dent in the workstation market. The company's market share went up to 4% but quickly fizzled to 1.4% as customers realized that IBM was not committed to RISC.

In 1990, however, IBM finally redesigned the RISC technology adding the 801 features that had been stripped in the ROMP, and more. It used it to introduce its RS/6000 or POWER

[18]The groups very first task had been to design a machine to replace its Display Writer wordprocessor.

(Performance Optimized With Enhanced RISC) workstations that temporary gave it the honor of having the fastest workstations. Management was a little more committed to workstation technology. Still, though, the company did not see RISC as the technology for its personal computers, minicomputers or mainframes—just for workstations.

Several factors eventually made IBM realize the potential (and threats) of the RISC technology it had invented. In the first place, personal computers and/or workstations (in a stand-alone mode or networked) were being chosen by customers over its mainframe computers largely as a result of the former's price/performance advantage and independence that it offered the different functions of most firms. IBM was shipping personal computers all right, but in every one of them, there was an Intel microprocessor. What was more was that Intel was making almost five times the profits per personal computer that IBM shipped than IBM was, and commanding a near monopoly position. In the second place, CISC technology was beginning to run out of steam given the growing need for speed in all computer markets—from personal computers to mainframes. For example, even in personal computers where speed had not been a problem, the growing complexity of software, and integration of such technologies as multimedia that require faster processors, provided an opportunity for the faster RISC technology. Finally, after suffering more losses in profits in 1992 than it had in all of its previous history it decided, as part of its restructuring, to have one processor architecture for all of its computers, from personal computers to supercomputers. RISC would be that architecture. IBM formed the PowerPC (Performance Optimized With Enhnaced RISC PC) consortium with Apple and Motorola to develop and manufacture POWER architecture chips for use in everything from personal computers to supercomputers.

Sun Microsystems

Sun's quickness to embrace RISC contrasts sharply with DEC's reluctance. Below, I argue that this difference stems from the way top management at each firm perceived RISC. In particular, Bill Joy, Sun's Vice President for R&D was the difference.

A workstation has two primary components: hardware and software. At its founding, Sun had in Andy Bechtolsheim—a co-founder and designer of the company's first workstation—a hardware guru. The company still needed a software guru and turned to Bill Joy. At UC Berkeley, Joy had refined and popularized a version of AT&T's UNIX operating system and was very well respected by engineers and scientists—the primary customers for workstations at the time. Joy was hired as Sun's Vice President for R&D and quickly emerged, as described by Sun's CEO Scott McNealy in a 1988 corporate video, as Sun's "thinker and guru".

Joy was in a position to give Sun technical direction the way Ken Olsen did for DEC but, for several reasons, he did not have the type of frame that had prevented DEC's CEO from recognizing the potential of RISC early. First, while at Berkeley where RISC pioneer and evangelist, Dave Paterson, had espoused the virtues of RISC, Bill Joy had been exposed to the RISC concept. Second, his background was largely software, and he therefore did not have the ingrained frame of computer designers who had developed "the more semantically rich, the better" mentality. Finally, as "thinker and guru" he had, mimicking another famous Silicon Valley thinker and guru, postulated what had come to be known as Joy's law:[19] That the power of the fastest single-chip microprocessor in any year in the future, measured in millions of instructions per second (MIPS), would be two to the power of the difference between the year in question minus 1984. That is, effectively, computing power would double every year.[20] With such postulations, all else equal, Joy would be more open to options for speeding processors, and RISC was one of those.

With Joy's leadership, Sun started exploring the feasibility of RISC as early as 1985. Its CISC microprocessors had come from Motorola which had difficulties accepting the concept of RISC too. So Sun decided to design its own microprocessors. But rather than develop design competence from scratch, it adopted the *RISCS* and *SOAR* architectures developed by RISC

---

[19]Joy's Law is actually a corollary to Moore's Law. Back in 1964, Gordon Moore, a co-founder of Intel, had predicted that in any year, the number of transistors that could be integrated onto microchip would be $2^{(t-1984)}$. So far, he has been right.

[20]In year t, the power would be, in MIPS, $2^{(t-1984)}$

pioneer Professor Patterson and his students at UC Berkeley (Garner, Agrawal, Patterson et al, 1988). Sun engineers added extensions[21] to the Berkeley design and named it SPARC (Scalable Processor Architecture).

In contrast to DEC and IBM whose embrace of RISC could only be described as reluctant, Sun was very enthusiastic about RISC and quickly converted all its product lines to RISC and has, as shown in Table 1, maintained its workstation industry leadership position.

Hewlett-Packard

In 1983, HP decided to adopt RISC and, two years later, it announced that all its computer products would use RISC. The question one might ask is why, unlike DEC, did HP embrace RISC so quickly? The answer may lie in two words: Joel Birnbaum. The story of RISC at HP starts with the arrival of Joel Birnbaum in 1981 from IBM where he had been John Cocke's supervisor[22]. He brought along William Worley and other IBM veterans who had worked on RISC at the Watson Laboratory. When they arrived at HP, several CISC projects were underway in the firm's efforts to replace its ailing 16-bit minicomputer architecture. As the Vice President for R&D whose inputs to corporate decisions were valued, and as a believer in RISC, he sought and got approval from HP's board to adopt RISC for all of HP's computer products. The CISC programs that had been under development were killed.

Spectrum, as the RISC program was called, was headed by William Worley. The goal of the team was to design the chips for the computers, and then build a "box" which would be used by all three HP computer product divisions—workstations, minicomputers and manufacturing floor (real-time)—with very little modification, to address their different markets. The workstation group would add a graphics subsystem unit to the "box" to make it a workstation.

---

[21]Specifically, the extensions were to accommodate multiprocessors, floating point and tightly coupled coprocessors.

[22]Before Birnbaum was brought in, HP had no plans to adopt RISC. He had been brought in because he was "available" at the time.

7. DISCUSSION

The cases just chronicled provide very strong evidence that it was neither the lack of a *strategic incentive* to invest nor *organizational competence-destruction* that prevented DEC and IBM from embracing and exploiting RISC technology early; rather, it was the dominant logic of top management that made the difference. The incentive to invest argument falls short for two reasons. First, Sun Microsystems, as the industry market share leader (see Table 1), stood to suffer the most from cannibalization if it embraced the new technology; but it did embrace it. Second, the fear of cannibalization or loss of political power is preceded by the recognition of the potential of the new technology. That is, in order to fear that a new technology will obsolete its existing products, a firm must understand the rationale behind the technology and its potential for cannibalization. As we saw in the case of DEC, its top management just did not get it; it did not understand the rationale behind using simple instructions to obtain faster processors.

A superficial look at DEC may suggest that, as an organization, it suffered from competence destruction which prevented it from recognizing the potential of the new technology. A more detailed look, however, suggests that different coalitions at the firm held different logics of what technology was best for it. For example, Forrest Baskett's group understood the rationale behind RISC and actually built a RISC machine that worked. So was the skunk works group that built the MIPS-based RISC workstation. But the dominant coalition of the CEO and his vice presidents for engineering and computer architecture genuinely did not understand the rationale behind RISC as evidenced by the quotes—e.g., "Anecdotal accounts of irrational implementations are certainly interesting. . . ."—cited earlier in the case of DEC. Many groups at IBM did not get it either. At Sun, the dominant coalition had a logic that was more in tune with RISC. So was that at HP.

Table 2 sums up the findings. In the table, we have used an L to indicate the presence of literal replication for each of the cases (Yin, 1989). That is, evidence for one of the propositions in Figure 1 is replicated in each the case in question. A T is used to indicate a theoretical replication. That is, some of the cases produce contrary results but for the predictable reasons

outlined in the propositions. Take evidence of industry logic (column 2 of Table 2). There was very strong evidence of it in all the cases. Computer design had evolved to a point where the dominant design mentality in industry and among academics had become, "the more semantically rich an instruction set architecture, the better". One of Professor Patterson's RISC article was rejected by a reputable journal with a rebuke that "that's not how computers are designed". There was evidence of this logic at all firms. As several of the interviewees recounted, each time the idea of RISC was suggested to an engineer, "you would get the Bill Cosby "r-i-g-h-t!"".

Table 2: Evidence from the cases.

1	2	3	4	5	6	7	8
Firm	Evidence of industry logic	Coalitions with different logics	Top Management logic same as industry logic?	Top Management organizational tenure	Top Management industry tenure	Organizational performance at the time in question	Top Management personal experiences
Apollo	L	L	L	L	L	T	L
DEC	L	L	L	L	L	L	L
HP	L	L	T	T	L	L	T
IBM	L	L	L	L	L	L	L
Sun	L	L	T	L	L	L	T

Note: An L means a literal replication while an H means a theoretical replication (Yin, 1989). Apollo was part of the detailed study, although not chronicled in the case reports above.

However, in each firm, there were some managerial coalitions that did not quite go with the industry recipes and were willing to give RISC a chance (column 3). At DEC, IBM and Apollo, these coalitions were not part of top management and so lost out. At Sun and HP, however, the coalitions were top management and therefore were able to embrace and exploit RISC early (column 4).

The question now becomes, why the differences in top management logics? As indicated in columns 5 and 6, it does not appear to be due to industry or organizational tenure, or current organizational performance as observed elsewhere (Hambrick, Geletkanycz, and Fredrickson,

1993). Founded in 1982, Sun was the second workstation company, right after Apollo and its top management was the company's founders. But it embraced RISC early. Joel Birnbaum and his coalition came to HP from IBM, but had been in the computer industry for decades. Turning to column 7, only Apollo was loosing money at the time when Sun and HP adopted RISC. Sun, HP and DEC were all doing very well in the workstation industry. Yet HP and Sun entertained and embraced RISC. The adversity had little impact on Apollo suggesting that the firm's top management may not have recognized the potential of RISC. On the other hand, Sun had the largest market share and was very profitable. Yet it embraced RISC.

The study suggests that the difference in top managerial logic was in the personal experiences of the top managers. At Sun, Bill Joy, the company's technical guru, was a software engineer who had worked at UC Berkeley, home of RISC evangelist Dave Patterson. With his background in software he had not been engrained with the doctrine of "the more semantically rich an instruction set architecture, the better". Moreover, he was in a better position to understand why shifting the burden of closing the semantic gap between software and hardware could be shifted back to software and yet result in faster machines. His co-founders believed in him, styling him as the "guru". At HP, top management was also greatly influenced by Joel Birnbaum and his team from the laboratory at IBM where RISC had been invented.

CONCLUSIONS

The goal of this study was to explore the role of a firm's strategic leadership in its ability to embrace and exploit technological innovation. The study suggests that top management's logic determines its ability to recognize the potential of an innovation. The extent to which this top management recognizes the potential of the innovation then determines whether management invests in the innovation or not. The study also suggests that referring to an innovation as being competence-destroying to the whole firm may be a little superficial. A deeper look, as in this study, suggests that individuals and coalitions within the organization have different logics and therefore understand the rationale behind an innovation and its applications differently. In all the

six cases studied, different groups within each firm perceived RISC differently, according to their logic. In the end, top management's logic prevailed. This logic is, itself, a function of industry logic, and personal experiences and agenda.

This strategic leadership view offers plausible explanations for several questions, or springboards for future research. First, while the strategic incentive and competence-destruction views have offered explanations for why incumbents fail in the face of radical technological change, they have not offered explanations for why incumbents succeed in the face of competence-destroying and product-cannibalizing technological change. The strategic leadership view does. Second, it offers a plausible answer to a question often raised by some economist in arguments against the competence-destruction view: why an incumbent cannot just replicate the capabilities of a new entrant by creating an internal venture group or skunk works within itself (Henderson, 1993)? Top management does create such groups when it recognizes the potential of the innovation. In some cases, renegade coalitions with differing logics have created such skunk works. It was such a skunk works that allowed DEC to quickly enter the RISC workstation market when it finally recognized the potential of RISC and decided to enter. Third, the strategic leadership view explains why, even in firms which experience competence-destruction, there are usually entrepreneurs who exit to pursue successful ventures using the new technology: They had different logics from the dominant coalition of the firm.

The study also suggests that industry tenure, organizational tenure or current organizational performance tell just part of the story about the logic of top management. Their personal experiences and agenda may be a lot more important than time spent in the industry or organization. *Who* a manager is, and *what* functions he/she has performed may be critical. Finally, as a byproduct of its main findings, the paper provides more strong empirical evidence for the existence and impact of industry, organizational, and managerial logic.

These findings suggest an interesting question in exploring post-technological discontinuity competition: whether top management is better off concentrating on the activities up to and including the decision to adopt and turning to other strategic discontinuity activities, or

continuing on after the decision to adopt and shepherding the innovation during the rest of its life cycle.

REFERENCES

Abernathy, W. and K. B. Clark (1985). 'Mapping the winds of creative destruction', *Research Policy*, 14, 1985. pp. 3-22.

Afuah, A.N. (1996): *Innovation Management: Strategies, Implementation and Profits.* Forthcoming, Oxford University Press.

Anderson, P. and M. Tushman (1990). 'Technological discontinuities and dominant designs: a cyclical model of technological change', *Administrative Sciences Quarterly*, 35, 604-633.

Arrow, K. J. (1962). 'Economic welfare and the allocation of resources for invention.' In Richard Nelson (ed.), *The Rate and Direction of Inventive Activity*, 609-26, Princeton University Press, Princeton.

Arrow, K. J. (1974). *The limits of organizations*, Norton, New York.

Banbury C. M., and W. Mitchell (1995). 'The effects of introducing important incremental innovation on market share and business survival', *Strategic Management Journal*, 16, pp. 161-182.

Bantel, K. A. and Jackson, S. E. (1989). To management and innovations in banking: Does the composition of the top team make a difference? *Strategic Management Journal*, 10, pp. 107-124.

Bell, G. (1986). 'RISC: Back to the future', *Datamation* June 1, 1986.

Bettis Richard A. and C. K. Prahalad (1995). 'The dominant logic: Retrospective and extension', *Strategic Management Journal*, 16, pp. 5-14.

Burgelman, R. A. (1994). 'Fading memories: The process theory of strategic business exit in dynamic environments', *Administrative Sciences Quarterly*, 39, pp. 24-56.

Christensen, C. M, and J. L Bower (1996). 'Customer power, strategic investment and failure of leading firms', *Strategic Management Journal*, 17, pp. 197-218.

Clark, D. and W. D. Strecker (1980). 'Comments on the 'the case for the reduced instruction set computer', *Computer Architecture News*, 8:6 (October), pp. 34-38.

Cook T. D. and Campbell, D. T. (1979). *Quasi-experimentation: Design and analysis issues for for field settings*, Rand McNally, Chicago.

Cooper A. C., and D. Schendel (1976). 'Strategic response to technological threats', *Business Horizons*, 19, pp. 61-69.

Ditzel, D. R., and D. A. Patterson (1980). 'Retrospective on high-level language computer architecture'. In *Proceedings Seventh Annual Symposium on Computer Architecture*, La Baule, France (June) 97-104.

Ferguson, C. H., and C. R. Morris (1993). *Computer wars: how the west can win in the post-IBM world*, Time Books, New York.

Finkelstein S. and D. C. Hambrick (1990). 'Top-management-team tenure and organizational outcomes: The moderating role of management descretion', *Administrative Sciences Quarterly*, 35, pp. 484-503.

Fitzpatrick, D. T., D. A. Patterson et al. (1982). 'A RISCy approach to VLSI', *Computer Architecture News*, 10.

Foster, R. (1986). *Innovation: The attacker's advantage*, Summit Books, New York.

Freeman, C. (1986). *The Economics of industrial innovation*, MIT Press, Cambridge, Mass.

Galbraith, J. R. (1973). *Designing complex organizations*, Addison-Wesley, Reading, MA.

Garner, Agrawal, Patterson et al, (1988). 'The SPARC architecture, *'IEEE Proceedings. COMPCOM.*

Gilbert, R. J. and Newberry, D. M.(1982). 'Preemptive patenting and the persistence of monopoly', *American Economic Review*, June 1982, pp. 514-95.

Gilbert, R. J. and Newberry, D. M.(1984a). 'Uncertain innovation and the persistence of monopoly: comment', *American Economc Review*, Vol. 74, pp. 238-242.

Gilbert, R. J. and Newberry, D. M.(1984b). 'Preemptive patenting and the persistence of monopoly. Reply', *American Economic Review*,Vol. 74, pp. 251-253.

Hambrick, D. C. (1989). 'Guess editor's introduction: Putting top managers back in the strategy picture', *Strategic Management Journal*, 10, pp. 5-15.

Hambrick, D. C., and P. Mason, (1984). ' Uppler echelons: The organization as a reflection of its top managers', *Academy of Mangement Review*, 9, pp. 193-206.

Hambrick, D. C., and S. Finkelstein, (1987). 'Management discretion: A bridge between polar views o organizations'. In B. Staw and L. L. Cummings (eds.), *Research in Organizational Behaviour*, Vol. 9. JAI Press, Greenwich, CT, pp. 369-406.

Hambrick, D. C., M. A. Geletkanycz and J. W. Fredrickson, (1993). 'Top executive commitment to the status quo: Some tests of its determinants', *Strategic Management Journal*, 14, pp. 401-418.

Hamel Gary M. and C. K. Prahalad (1994). *Competing for the future*, Harvard Business School Press, Boston, MA.

Henderson, R. (1993). 'Underinvestment and incompetence as responses to radical innovation: evidence from the photolithographic alignment industry', *The Rand Journal of Economics*, Vol. 24, No. 2, Summer 1993.

Henderson, R., and Kim B. CLark (1990). 'Architectural innovation: The reconfiguration of existing product technologies and the failure of established firms', Administrative Sciences Quarterly, 35, pp. 9-30.

Hennessy, J. L., N. Jouppi, F. Baskett, T. R Gross, and J. Gill (1982). 'Hardware/software tradeoffs for increased performance', *Proceedings* of the Symposium on architectural support for Programming Languages and Operating Systems (March) 2-11.

Hopkins M. E. (1987). 'A perspective on the 801 Reduced Instruction Set Computer', *IBM Systems Journal*, volume 26, No. 1, 1987.

IDC (International Data Corporation), 1988 Report on Workstations.

Larus, James R. (1982). 'A Comparism of Microcode, Assembly Code, and High-Level Languages on the VAX-11 and RISC I', *Computer Architecture News*, Vol. 10, No. 5 Sept. 1982.

Lee, K. (1987). 'Olsen: DEC's success lies in software; Digital Equipment Corp. president Kenneth H. Olsen; interview', *Computer and communications decisions*, May, 1987.

Leonard-Barton, D. (1992). 'Core capabilities and core rigidities: A paradox in managing new product development', *Strategic Management Journal* 13: 111-26.

Levy, H. M. and D. W. Clark (1982). 'On the use of benchmarks for measuring performance', Computer Architecture News, Vol. 10, No. 6, December, 1982.

March, J. G., and Simon, H. (1958). *Organizations*, John Wiley, New York.

Mashey, John R. (1992). 'CISCs are not RISCs, and not converging either', *Microprocessor Report*, March 25, 1992.

Nelson, Richard R., and S. G. Winter (1982). *An evolutionary theory of economic change*, Harvard University Press, Cambridge, Mass.

Patterson D. A., and J. L Hennessy (1990). *Computer architecture: a quantitative approach*, Morgan and Kaufman Publishers, San Mateo, CA.

Patterson, D. A., and D. Ditzell (1980). 'The case of the reduced instruction set computer', *Computer Architecture News*, vol. 8. No. 6., October 15, 1980, pp. 25-33.

Prahalad C. K. and R. Bettis (1986). 'The dominant logic: A new linkage between diversity and performance', *Strategic Management Journal*, 7, pp. 486-501.

Reinganum, Jennifer F. (1983). 'Technology adoption under imperfect information', Bell Journal of Economics, 14, pp. 57-69.

Reinganum, Jennifer F. (1984). 'Uncertain innovation and the persistence of monopoly. Reply', *American Economic Review*, Vol. 74, pp. 243-245.

Roberts, E.B. (1988). 'What we've learned: Managing invention and innovation', *Research Technology Management*, 31, 11-29.

Schumpeter, J. A. (1934). *The Theory of Economic Development*, Harvard University Press. (A translation of the German version of 1912: Theorie der wirtschaftlichen Entwicklung. Leipzig, Duncker & Humboldt. 1912).

Schumpeter, J. A., (1950). *Capitalism, Socialism and Democracy*,. Harper, New York.

Simon, H. A. (1955). 'A behavioral model of rational choic', *Quarterly Journal of Economics*, 69, pp. 99-118.

Spender, J. -C. (1989), *Industry recipes: An inquiry into the nature and sources of management judgement*, Basil Blackwood, Cambridge, MA.

Tushman, M.L. and L. Rosenkopf (1992. 'Organizational determinants of technological change: towards a sociology of technological evolution', *Research in Organizational Behavior*, 14, pp. 311-347.

Tushman, Michael L., and Anderson, Philip (1986). 'Technological discontinuities and organizational environments', *Administrative Science Quarterly*, 31, pp. 439-465.

Utterback, J.M., 1994, *Mastering the Dynamics of Innovation* (Harvard Business School Press, Cambridge, MA).

Walsh, J. P. (1995). 'Managerial and organizational cognition: Notes from a trip down memory lane', *Organization Science*, 6, pp. 280-321.

Walsh, J. P., and G. R. Ungson (1991). 'Organization memory', *Academy of Mangement Review*, 16, pp. 57-91.

Yin, R. K. (1989). *Case study research: Design and methods*, Sage, Newbury Park, CA.