TIMOTHY BRESNAHAN
Stanford University

SHANE GREENSTEIN
University of Illinois

Technical Progress and Co-invention in Computing and in the Uses of Computers

WHY DO SOME technologies offer opportunities for widespread economic change? Purely technical progress is rarely sufficient to make an invention economically important. Users, through their own experimentation and discovery, make technology more valuable. We call this activity co-invention to distinguish it from original invention. Co-invention is potentially complex and uncertain, and it can be a bottleneck in technical progress. Yet the complementarity between inventions by users and by technologists can benefit a wide range of economic activities. Understanding co-invention is a key to understanding the economic payoff to invention in new information technologies today.[1]

We would like to thank the National Science Foundation, the Institute for Government and Public Affairs at the University of Illinois, the National Bureau of Economic Research Project on Industrial Technology and Productivity, the Center for Economic Policy Research at Stanford University, the Stanford Computer Industry Project, and the Sloan Foundation for financial support. We also thank Ron Borzekowski, Victoria Danilchouk, Harumi Ito, and Mike Mazzeo for their research assistance. Comments at the Brookings microeconomics conference, notably from David Brownstone, Kenneth Flamm, and the editors of this volume, have been very helpful.

1. In the past two centuries general purpose technologies and associated co-invention have helped drive economic growth. These "macro-inventions" (Mokyr, 1990) are rare, important, and slow to come into use. Bresnahan and Trajtenberg (1995) suggest that co-invention by users of a general purpose technology is a key source of Romerian increasing returns. Von Hippel (1988) discusses users' role more generally. For the uncertainty and experimentation surrounding the early stages of the co-invention process, see Rosenberg (1996). The importance of co-invention in information technology has led several authors (Freeman and Soete, 1990; David, 1990) to make comparisons to historically important general purpose technologies.

To present and explain our argument, we study the transition from mainframes to client/server (C/S) systems at large establishments. Large-scale company-wide or organization-wide computing systems are well established. They are the most valuable commercial uses of computing, having been cost effective at price-performance levels in the 1950s and 1960s. Decades of invention by information technology companies and co-invention by users have improved and refined applications such as corporate accounting systems, bank automatic teller machine networks, and inventory-control or reservation systems. Yet, for all their power and all the valuable services they deliver, these systems have been the source of considerable discontent. Hard to use, hard to change, and run by unresponsive bureaucracies, these systems invited reform.

Client/server computing emerged as a viable solution to the problem by promising to combine the power of traditional mainframe systems with the ease of use of personal computers (PCs). A network could permit the functions of a business system to be divided between powerful "servers" and easy to use "clients." By the late 1980s the promise of C/S was articulated and demonstrated in prototypes, and the competitive impact was quick and powerful. Firms selling traditional large-scale computer systems saw dramatic falls in sales, profits, and market value.[2] Yet the genuine realization of the economic promise, in terms of widespread adoption of C/S systems, has been much slower to come. The explanation lies in users' co-invention processes.

We conduct our analysis at the individual establishment level. Our analysis is based on a series of surveys of computing users taken by Computer Intelligence Infocorp. (CII). The data include more than half of the large-scale computing users in the United States. We examine the variety in users' behavior as they start to use C/S technologies, move forward through a switch to C/S, and finally remove their last mainframe. To predict users' choices, we use a set of proxies for different theories of the costs and benefits of early or successful co-invention and reject those theories that are not supported by the evidence. The rejected theories are among those most influential in technologists' and economists' pop thinking about information technology: vendor

2. Ferguson and Morris (1993), Chandler (1997), Baldwin and Clark (1997), and Bresnahan and Greenstein (1992) analyze this competition.

lock-in, obstructionism by the management information systems (MIS) department, and overly skeptical as well as overly optimistic views of information technologies' potential value.

We find evidence that co-invention costs are important in understanding the switch from host-based computing to client/server computing. Change remains slow because it is difficult to reinvent organizations around a new computing platform, even a platform with large potential benefits. Complex computing applications and idiosyncratic computing applications drive up co-invention costs, which market-supplied software tools cannot easily lower. Users' activities determine the pace and the nature of gains to information technology.

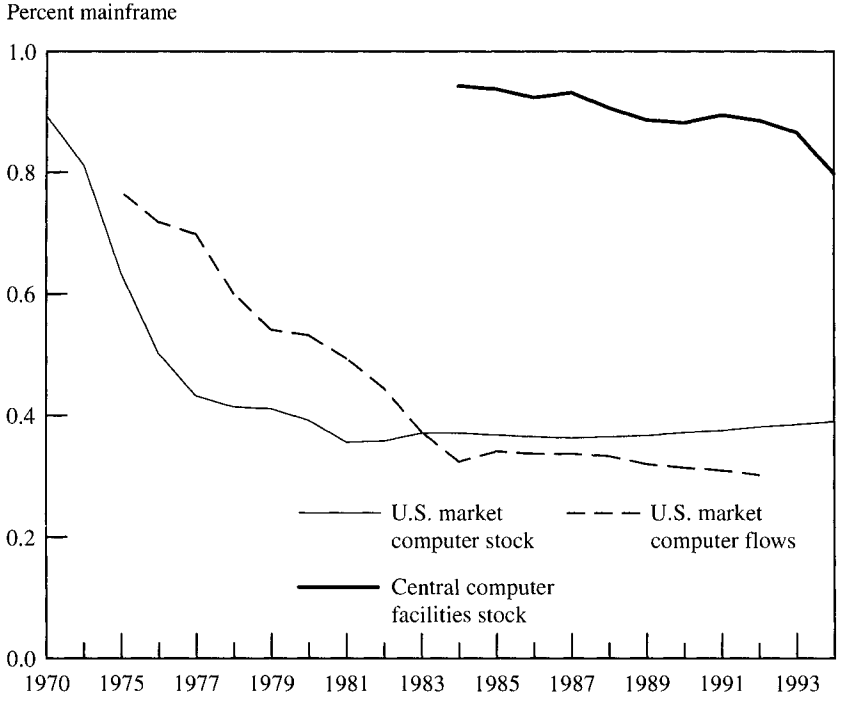
Large Organizations, Large Computers, and the Client/Server Revolution

For decades every major corporation and large organization within the United States—from AT&T to Sears, Roebuck to the U.S. Department of Defense—has employed large computer systems for critical organization-wide functions. These systems have been supported by a stable set of mainframe host vendors, such as IBM, and independent software companies, such as Computer Associates. Over time hardware and networking technology improved rapidly and in a consistent and predictable direction. Business systems and related applications changed less rapidly, but they too reached a highly productive state.

In parallel markets cheaper and smaller minicomputers, personal computers, and workstations brought computing power to smaller (and, individually, less valuable) uses. Much less orderly and more rapidly changing, this new computing was largely irrelevant to large organizations' centralized computing.

Dramatic change occurred in the late 1980s and early 1990s, when mainframe hardware and software faced real competition from networked smaller computers. Before 1989 workstations and personal computers could no more replace mainframes than could the people of Lilliput wrestle Gulliver to the ground. Yet, like the Lilliputians' ropes, networking cables created strength from numbers. Workstations, a technology originally intended to serve individuals, were deployed as servers. They did not need as much capability as hosts, since PCs, deployed

Figure 1. Use of Mainframes and Smaller Computers: Revenue-Weighted Shares in the U.S. Market and Central Computer Facilities, 1970–94



Source: The value of shipments series is drawn from a seller survey conducted by the Computer and Business Equipment Manufacturers Association. The market-wide installed base figures were calculated based on the International Data Corporation grey sheets and data books. The numerator is "General Purpose Computers" through 1983 amd "Mainframes" thereafter. The central computing facilities data come from Computer Intelligence Infocorp. data. In all three series the denominator includes workstations, minicomputers, and mainframes but excludes personal computers. In both the IDC and CII installed base series, we have used historical sales figures and depreciated at 20 percent per year.

as clients, assumed some of the computing tasks (such as effectively interfacing with people). This technical opportunity produced large market and organizational change. IBM, having dominated large-scale business computing for decades, no longer found itself a leader. The MIS departments of large organizations were excited as well as threatened by the new opportunities. Both IBM and these departments began to appear as the inert Gulliver, roped and staked to the ground.

The effects of this regime change are shown vividly in figure 1, which contrasts the use of mainframes and smaller computers by two different kinds of buyers. First, it presents the share of mainframes in the flow of computer investment and the stock of installed computers

in the United States. Mainframes' share falls sharply from 1970 until the mid-1980s as first minicomputers and then microcomputers became important in the marketplace. Second, figure 1 offers a measure of mainframes' share in large organizations' centralized computing facilities, revealing heavy reliance on mainframes throughout the 1980s. While the microcomputer and workstation revolution was going on outside the "glass house," it had not penetrated inside. The early 1990s, however, reveal a dramatic transformation: a very substantial increase in the use of smaller computers—mostly as servers—in the large data centers. With the help of networking, the revolution came to corporate data centers.

The trade press initially reported this transformation as a major improvement in computing, the replacement of an old, expensive, technology—mainframes—with a newer, cheaper, and better one—client/server computing. Users, many of whom we interviewed, took a somewhat different view.[3] While some moved forward aggressively, others waited for the C/S technology to mature, possibly as a result of early experiments with it.[4] This need for co-invention by potential users of C/S slowed "the death of the mainframe."

Figure 2 compares the capacity of mainframe and server-class computers held in the corporate data centers. The early switch to C/S did not mean a decrease in mainframe capacity. Instead, it meant a decrease in the rate of growth of mainframe capacity. On the server side, capacity growth accelerated and continued to accelerate through 1994. No simple story of the comparative technical merits of each of the two technologies is going to explain these figures. Our micro data analysis will take us many steps toward understanding what has happened so far and why.

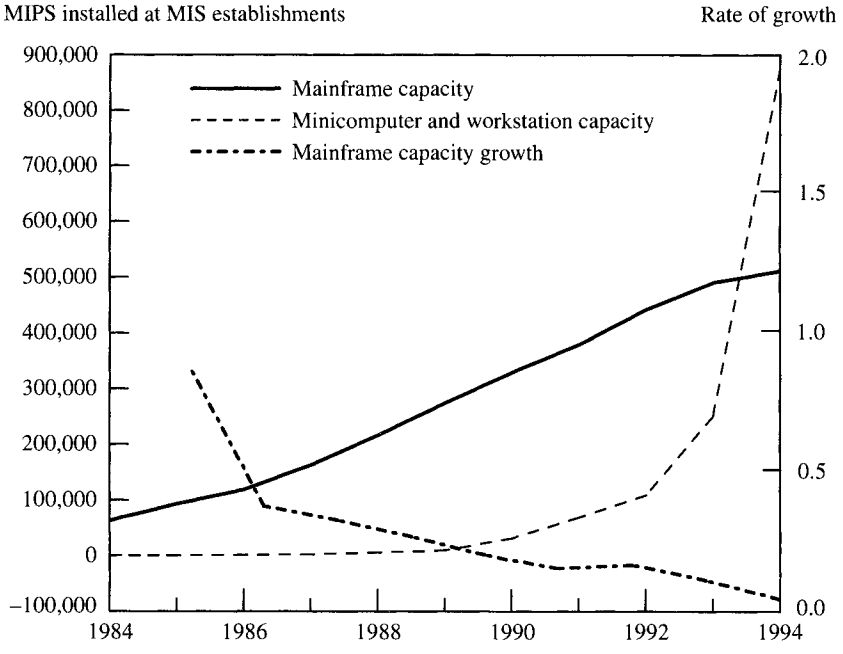
Defining Basic Terminology

Whether implemented on a traditional mainframe or in C/S, a *large-scale computing system* collects, stores, sorts, analyzes, and presents information. Such a system includes processors and controllers for directing information flows between components. It also incorporates

3. See Bresnahan and Saloner (1997) for a study based entirely on interviews from the early phases of C/S adoption. To tighten the theories and the proxy relationships in the current analysis, we conducted further interviews of large-scale application users.

4. This lag of effective use behind raw technology is systematic in large-scale computing. See Friedman and Cornford (1989) or Barras (1990).

Figure 2. Capacity of Mainframe and Server-Class Computers at Corporate Data Centers, 1984–94



Source: Computer Intelligence Infocorp. data.

Note: Capacity is measured in millions of instructions per second. Mainframe capacity and minicomputer and workstation capacity are shown on the same lefthand scale (though MIPS are not perfectly comparable across classes of computers). The percentage rate of growth of mainframe capacity is also presented. For definitions of server capacity, see text and data appendix.

components for presenting output (such as screens or printers), for receiving input (such as keyboards and tape drives), and for storing information in large databases. Still other components facilitate communication over phone lines, wide-area-network lines, or dedicated local-area-network lines. Finally, there is operating system software and application software, some of it packaged and distributed by vendors, some written and refined by users.

Commercial *large-scale computing applications* involve administrative tasks performed jointly among as many as several hundred end-users. Because of the scale of such applications, the employed hosts are primarily mainframes or superminis (supercomputers and sometimes other exotica). Workstations and PCs were not associated with large-scale applications in the early stages, but they provided decen-

tralized individual computing applications or limited interaction with the mainframe as intelligent terminals. They do, however, underlie the C/S networks in the early 1990s.

Large-scale computing is found at virtually every major organization in every industry in the United States. Some industries—such as banking, on-line services, and engineering support activity—are especially computer intensive. Tens of thousands of large computing systems are used for large-scale applications, and no two are exactly alike. In general, most enterprises view their large-scale applications as essential in the day-to-day functioning of their organizations. Computer systems control access to, and security of, enterprise information, and they embody business routines for carrying out key functions.

In practice, the typical large-scale system is expensive to set up and maintain. Any system requires frequent attention and maintenance. Efficient functioning calls for lifetime operating expenses well in excess of the capital outlays.[5] Complementary human capital investments in the MIS department and in the departments supported by the computer can be very substantial.

A computing *platform* is a reconfigurable base of compatible components on which users build applications. Platforms are most readily identified with their technical standards, that is, engineering specifications for compatible hardware and software. Broadly speaking, standards in the host-based mainframe world tend to be proprietary, controlled by the vendor of the host computer that supervises the system on which most programs are run. The extent to which nascent client/server standards are proprietary or open is a matter of considerable debate. We prefer a broad definition of platforms that includes the co-invented parts: users' knowledge about how to best configure, program, and utilize the system. What a platform can do depends not only on the technology invented by vendors and the market supply of complementary goods, but also on human capital at user establishments.

It is a simplification to group all platforms for large-scale applications into two camps: client/server computing and host-based comput-

5. Hardware capital outlays have historically ranged between a quarter and a fifth of the typical centralized facility budget. See, for example, *Datamation*, August 15, 1994, p. 48, and *EDP Industry Reports*, Executive Summaries, International Data Corporation, 1978–85.

ing. Yet this division captures most changes to usage and industry structure in the late 1980s.

In *host-centric computing*, most programs are run on a centralized computer, or host.[6] The host computer manages resource allocation among the competing uses and also performs each function. A personnel and payroll system, for example, may be simultaneously used to enter data about new employees and to implement new policies about pay. Meanwhile, the same computer may be used for operational applications, such as using data to write checks. The clusters of subapplications—data entry, data correction, data quality management, performance of the central operational task, and routine or ad hoc analysis—together define the operations of a system. Different subapplications share not only the computer's central processor but also certain key data. These subapplications can change, as users invent further routines and customize software and system configurations to their unique needs.[7]

In *client/server computing*, programs run on both clients and servers, communicating and coordinating via networks. A data entry program, for example, may run on an easy-to-use client such as a PC. Client computers have only local tasks, but they request services from servers. When a data entry task is completed, for example, the client computer requests that the personnel database server computer update the personnel records file. Each kind of computer pursues its comparative advantage. Thus, clients run the parts of large applications systems that PCs can do well, such as providing human interfaces or doing simple analysis and reporting. Servers pursue their comparative advantage, which is maintaining databases, processing transactions, and coordinating networking tasks.

Host-Centric Computing in the Late 1980s

By the time client/server systems began to threaten them, host computers together with software technologies had developed very valuable

6. The host may be assisted by other computers, such as a minicomputer used as a communications controller for a mainframe host.

7. Some users (for example, a single-branch bank) employ a single CPU and multiple peripherals, dividing up applications over the workday and workweek . Other users (for example, a large research university) employ multiple systems, specializing in different sets of applications. Yet others, such as very large reservation systems, use several systems linked together for complex, idiosyncratic, and highly effective combinations.

capabilities for large organizations. Our discussion focuses on three features of host-based computing: their high value, their complexity, and the emergence of related markets.

Host computers, using market supplied tools such as database management systems and "transaction monitors," were able to provide control of many aspects of a business to its managers. Experienced computing professionals knew how to schedule use of the host and avoid expensive peak load problems. MIS people knew, for example, just how much computer power they would need to service a specific pattern of use (for example, fifty payroll clerks entering updates to data with a peak load occurring at the end of the benefits open-enrollment period). The combination of host-centric computing with specialized management information systems made possible the automation and control of large-scale bureaucratic tasks.

It also permitted higher value tasks. For example, knowledge of a debtor's standing permits monitoring of credit relationships. Banks, in turn, have completely changed relationships with retail customers as a result of the automatic teller machine, the sorted checking account statement, and the call center that can resolve account problems over the telephone—all complex large-scale computer applications. These higher value applications tend to be co-invented by computer users only after considerable experience with automation and control applications.[8]

Alongside these extremely valuable functions, host-centric computing has some well-known and longstanding problems associated with its complexity and management. First, using the terminal sitting on the end-user's desk to interpret and pass simple commands from the host to the user and from the user to the host requires application-specific and system-specific training and experience, which is not only an undesirable complication, but also an important expense in developing new applications or modifying old ones. Further, since resource allocation and business procedural rules strongly affect applications systems design, the information systems (IS) group is solely responsible for implementing any significant change. Nontechnical end-users often cannot accomplish computing tasks without the aid of specialists, and this leads to complaints, justifiable or not, about the IS personnel.

8. See, for example, Barras (1990).

Applications may not do exactly what is needed and can be hard to use. Apparently simple queries ("Sales are down in the West region; which salespersons are responsible?") take too much effort, cost too much, and get resolved slowly. Both users and IS personnel express general dissatisfaction with results. Users can easily imagine something better, and IS personnel dislike being the object of opprobrium but often cannot improve things quickly.[9]

Despite these problems, a stable industry structure had emerged behind host-based computing by the end of the 1980s. Platforms were typically supplied by a cluster of firms, one of which was the lead vendor, responsible for development of the overall architecture of the platform. Usually, this firm, such as IBM, DEC, or Unisys, was also the seller of the host, and it offered hardware, software, and networking components. These vendors had a vested interest in improving old platforms and making their chronic problems more palatable. For every important platform there were markets in "plug compatible" hardware components, such as disk drives and terminals. There were also markets in software to run on the important platforms. Some software vendors (including all integrated sellers of hosts) sold for only a single host vendor's platforms. Other vendors (Computer Associates, for example) sold "multiplatform" software that could be used with a variety of different host platforms. Finally, a web of service and support industries developed, including systems integration, data center operations, and consultancies of all kinds. In this, as in software, there was supply from host vendors from large firms (for example, EDS, Andersen Consulting, AMS) and from small firms. All these efforts reinforced host-centric computing's strengths, but none removed its fundamental weaknesses.

Potential Payoff from Client/Server Invention

During the 1980s easy-to-use PCs spread widely through business. Individual users experienced few of the frustrations of host-centric computing. This provided a powerful incentive to integrate personal computers into large-scale applications to profit from their relatively easy use. To the extent new platforms could solve the longstanding problems

9. Friedman and Cornford (1989) document many failed organizational attempts to overcome these problems.

with large-scale computing by meeting user demand for better functionality at lower cost, their potential payoff was tremendous.

C/S in its early incarnations involved a mix of easy-to-use clients with powerful servers and the replacement of proprietary architectures with open systems.[10] In the early days of C/S, no one paid much attention to the problem of co-invention or worried about building the same kind of market support found behind host-based systems. In the late 1980s, technologists optimistically forecasted a reasonably quick process for inventing new C/S architectures.[11] The applications systems designer would simply have to combine the best features of existing products in an artful way. The designer would allocate each kind of computer for subtasks and distribute computing power widely across a networked set of PCs (or other clients) and workstations (or other servers), whichever was cheaper and easier to employ. Further, different kinds of people who used and maintained the system, business end-users and IS experts, would each have the right kind of computers and software. The competitive threat to host-centric systems predominantly arose from these superior technical features, not just lower costs. By this logic, it was anticipated that valuable large-scale applications would move to C/S quickly.

We argue that this "mix and match" or "best-of-both-worlds" vision is conceptually too simple. The repackaging needed to make C/S a reality involves complex adjustments. Database management systems, network operating systems, and network-connectable operating systems are not simple. Bringing together small computers with larger ones involves bridging myriad specific technical differences between clients and servers. The more complex the computing applications, the more difficult it is to cross the bridge, so complex applications should be difficult to switch. The optimistic forecasts also underestimated the importance of supply of complementary component markets. Even though every major trade publication predicted rapid growth of subcomponent markets at the beginning of the 1990s, these markets were thin

10. Technologists differ about the precise definition of C/S, and one can become deeply entangled in debates about two-tiered versus three-tiered architectures and other issues. For our analytical purposes a broad and inclusive definition is appropriate: whatever new platforms let users move applications off traditional host architectures.

11. Kador (1992), Keefe (1990), and Radding (1989) describe the near-term possibilities in very similar terms.

at first. Users could not easily obtain basic programming tools for the new platform. Thus, it was difficult to make computer operating systems on the servers and the networks as reliable and secure as those in the host-centric platforms. These difficulties could be overcome by clever programming or engineering at the user establishment, but most large computing applications establishments were staffed with programmers who understood host systems, not C/S. Programmers who had knowledge of the new platform were usually more expensive to employ. These factors raised the costs of building applications, refining them, or maintaining them on the new platform.

The switch to client/server computing grew steadily easier and more attractive during the early 1990s. The size of the potential market for C/S drew large investments in technical solutions to solve these problems. The kinds of tools users needed to build new systems improved. Human capital at MIS departments increased. Interface standards between clients and servers became more stable and predictable, though they still compare unfavorably to mainframes on this score.

Despite the speed and ambition of this technical progress, C/S did not become strictly better than mainframes. Instead, by the mid-1990s, each platform had distinct advantages and disadvantages. On the one hand, pre-existing data and programs for large applications were (necessarily) on host-based systems. If newly developed applications were simply improvements to the old, then there would be cost advantages to continuity. If new applications also needed to interact with the old programs or data, even greater advantage would arise.[12] Finally, even with many technical problems solved, C/S still called for co-invention, which was potentially costly and time consuming, especially for complex applications. Hence, some users were going to switch cautiously, if at all.

On the other hand, technologists forecasted that co-invention would be easier on the C/S platform than it had been on earlier platforms. They expected it to solve the problem of user relationships in one of two ways. First, there might be end-user programming. Business people program PCs all the time (for example, by making spreadsheets). Perhaps they could write their own applications using company data. Second, if this fails, the more rapid development cycle of C/S might let

12. See Friedman and Cornford (1989) and Ito (1996).

Table 1. Host-Centric and Client/Server Computing: Advantages and Disadvantages

Host-centric computing	*Client/server computing*
Thick component markets make co-invention easier.	Thin component markets made co-invention hard at first, easier over time.
Not costly to continue with old programs, old vendors, and established databases.	Users welcomed improvement in user-interface and functionality.
General dissatisfaction with manageable but chronic problems.	Anticipated rapid development of applications but initial shortage of human capital to make them.
Powerful and reliable for large problems because of established methods and doctrine.	Potentially revolutionary but largely untested in idiosyncratic, complex applications.

Source: Authors' comparison.

MIS personnel guess what users want, program it, get feedback, guess again, and so on. In these views C/S finds part of its value as an invention for making (co-)inventions.

The advantages and disadvantages of client/server computing versus host-centric computing are shown in table 1.

Client/Server Computing: Theories of Adoption

We investigate a number of theories that might fit the pattern of adoption of C/S; some have come from the technical literature, others from the analytical economics literature. Two emphasize the benefits of C/S, while the rest emphasize the costs of switching out of one platform and developing applications on another. These costs depend on vendor lock-in, market thickness, and on two competing hypotheses of the role of MIS departments. The theories make different assumptions about the nature of the switch to C/S, and they lead to different predictions.

The first theory is based on classic (demand-pull) diffusion theory. It was widely circulated in the trade press during the initial diffusion of client/server technology. This theory posits that complex establishments were the most unsatisfied with existing solutions to their problems and had the worst relations with their MIS departments. Host-based applications were deeply linked to workers' jobs in these establishments, and these connections were widely despised. If C/S makes computer-using jobs simpler because of easier-to-use clients, it will

rescue these establishments. Moreover, these establishments experience a very complex and unwieldy applications development process. In this theory C/S will permit shorter application development cycles and thus make it easier to learn users' needs. If cross-sectional variety in the adoption of C/S is only a function of the payoffs from C/S relative to host-based computing, then by this theory complex establishments are expected to be attracted by the large benefits of C/S and part with their existing host-based architectures before everyone else.[13] Because the technologists who invented this theory emphasize that C/S combines the best of both mainframe and personal computer technology, we call this the *best-of-both-worlds theory*.

Other technical observers saw C/S as analogous to earlier platforms, such as commercial minicomputers, which had competed against mainframes in small establishments. They thought the key factor behind diffusion would be the size of the largest available server and that diffusion would proceed up the size distribution of establishments. This *size-effect theory* is also a classic demand diffusion story, one with a second view of the product differentiation between host-centric and C/S computing.

Another view emphasized the costs to the buyer from switching platforms, which often implied a switch in vendors. The *vendor lock-in theory*, well established in the industrial organization literature, hypothesizes that establishments will have a difficult time switching because they are tied to IBM and other host vendors. In addition, past co-invention itself served to lock in establishments because of its platform-specificity. This theory comes in two closely related forms. In one form the key to understanding lock-in lies in many technical parameters of computer applications and other technical constraints on the revision of existing systems. In this view replacing old applications with C/S architectures is expensive or nearly prohibitive. In the second form lock-in depends on the degree to which organizations use proprietary software, or the degree to which vendors manage "account con-

13. In the business strategy community this idea was linked to the broader movement to "re-engineer business processes." Advocates of re-engineering saw a tight link between streamlining bureaucracy and networked computing. See, for example, Hammer and Champy (1993, chap. 5).

trol'' by selling users proprietary hardware and software.[14] In either case locked-in establishments should be the last to experiment with client/server computing and the last to finish a move once begun.

There is an alternative efficiency interpretation of platform-specific investments. Host vendors tend to serve large, general-purpose software markets, leaving smaller ones to platform-specific independent software vendors. As the C/S platform matures, software (characterized by tremendous increasing returns) will tend to be written for the larger markets first. A *market thickness theory* of software markets emphasizes this supply response rather than lock-in as the main source of the costs of switching. In the market thickness theories, experimenting with or switching to C/S will get less costly and less risky with time in an order determined by new software supply.

Two further theories are related to large organizations' division of labor between computer systems builders in the MIS department and computer systems users. In the *MIS agency theory*, systems builders may not act as the perfect agent of the overall company. They have incentives to slow platform changes. The theory suggests that their own human capital is tied to the old platform, either because of well-established cozy relationships with host suppliers or because of the wish to avoid tedious large-scale change. MIS personnel can exploit their superior knowledge of technology vis-à-vis their supervisors and thus earn an information rent. This theory predicts a slow process of change at establishments with a particularly powerful MIS department.

The alternative, the *idiosyncratic applications theory*, points out that the establishment's business operations may be idiosyncratic. The MIS department may have had to build establishment-specific applications since available market applications did not fit the idiosyncratic need. Under this theory a slow move to a new platform reflects the time needed to rewrite establishment-specific applications, not an agency problem. The MIS-behavior theories, like the vendor lock-in theory, are primarily about blockages in the path to the new platform rather than about the platform's relative attractiveness. Both the market thickness theory and the idiosyncratic applications theory emphasize the

14. Much of the popularity of the lock-in theory comes from frequent complaints by users about legacy systems and from antitrust cases against IBM.

relationship between an organization's needs and market-supplied tools.

To understand the forces holding up diffusion of client/server computing, we add one final theory. The *co-invention theory* states that the most complicated establishments have the most difficult time switching from host-based computing to C/S. In these organizations the process of integrating computers into the existing business systems has been very difficult. It has called for developing new organizational forms, new job definitions for computer-using workers, even new relationships between the establishment and its suppliers and customers, as in the case of networked computing. The new C/S platform itself is not strictly better than the old on all dimensions but instead has strengths and weaknesses. If establishments want to take advantage of the strengths and avoid the weaknesses, they must undergo tedious co-invention at the individual establishment level. Aware of the costs of co-invention from past experience, they are hesitant to start the switch, and the complexity of co-invention makes them the last to finish the journey to C/S. The theory of co-invention is relatively new.[15] Yet we believe it is important because it recognizes that the social return to new technology is driven, not by the raw technology itself, but by the organization using it.

Micro Data on Large-Systems Users and Their Choices in the 1990s

In order to model the choices of large-systems users in the 1990s, we assembled micro data on establishments previously using host-based mainframe computers and on their C/S decisions. Here we describe our data sources and provide a broad overview of the behavior found within these establishments. The next section designs an establishment-based econometric analysis of user behavior, explaining why there have been marked differences in behavioral patterns.

15. Bresnahan and Greenstein (1996), Saloner and Steinmueller (1996), and Bresnahan and Saloner (1997) report on the importance of organizational costs of switching to C/S. Brynjolfsson and Hitt (1995), Greenstein (1995), and Ito (1996) discuss co-invention with regard to information technology investments in general.

Sample

Computer Intelligence Infocorp. collects annual data on the stock of computing system hardware and software employed by business computer users across North America. We assembled its surveys into an unbalanced annual panel of users of large-scale computing systems.[16] We follow host-based computing in the period just before C/S and during the early and middle stages of C/S diffusion; that is, our analysis dataset spans December 1988 through December 1994. A few of our descriptive charts and tables, but none of our analyses, use data going back to the beginning of the CII surveys in 1984.

The unit of observation is an establishment in a year.[17] We compile information about the establishment's overall stock of large-scale computers and the software running on them. With a few exceptions, our establishments are the data processing centers of large corporations and of large divisions within very large corporations (or similar not-for-profit entities).[18]

To be in our analysis dataset, an establishment had to meet a number of requirements. First, it had to use a mainframe in 1989. To identify mainframes, we followed CII's mainframe definition, thus including large IBM (or IBM-compatible) computers, such as 3090s and 4381s, comparable machines from Unisys, DEC, and others, and vector processors.[19] Second, the establishment had to meet some complete data criteria. Once surveyed in 1989, it had to be surveyed in 1990 as well (though it did not need to have a mainframe at that time). Moreover, it had to have records on its software use in 1989.

As table 2 shows, this complete data requirement, especially the software requirement, reduces our sample size by eliminating approximately 11 percent of the surveys conducted by CII. Since CII surveys

16. The data appendix describes our panelization procedures.

17. Before 1994 CII used the concept of computing "site" as its unit of observation and switched to the standard census bureau definition of an establishment in 1994. A site corresponds to a senior MIS manager, and there can be one or more sites per establishment. Using CII unique identifiers, we aggregated sites to establishments in the earlier years. More details are provided in the data appendix.

18. Our sample does not include large commercial establishments using commercial minicomputers instead of mainframes. We also include some computer services industry establishments.

19. The data appendix shows a breakdown of large system families focused on the boundary between mainframes and smaller systems in our dataset.

Table 2. 1989 Sample Size Calculated from Computer Intelligence Infocorp. Sites

Calculation	*Number of observations*
Sites aggregated to establishments	14,082
Drop establishments whose first appearance is for only one year	−234
Drop establishments missing 1989 software data	−1,303
Our sample	12,545

Source: Authors' calculations.

70 percent of all large-systems users in any given year, including our time period, our dataset tracks close to 60 percent of all such users in the United States.

Our analysis dataset contains 12,545 observations in 1989. We follow the life history of each establishment as much as possible through 1994.[20] We know from our previous work that the founding of mainframe establishments slows in the 1990s, but our sampling frame does not allow us to say precisely how many new large-scale establishments we miss. In particular, our mainframe-based sampling frame misses new establishment in the 1990s that never had a mainframe but went directly to C/S. Thus, our analysis discusses only how existing establishments transited to client/server computing.

Adoption of Client/Server Computing: Experiment Design

We analyze variation in the adoption of C/S systems at the establishment level to test among the theories we introduced. Our regressors are drawn from CII's 1989 interviews, just before the beginning of the diffusion of C/S.[21] CII's sample is limited to MIS managers or similar officials, so our knowledge about a particular establishment is strongest on its computer operations and systems. Accordingly, most of the regressors are defined in terms of software running on the establishment's computers and in terms of similar technical data. We also know the

20. Of these establishments 2,549 exit our sample by 1994 because of closings, moving of functions elsewhere, or survey nonresponse. In our duration models we take a conservative approach to interpreting these exits and treat them as right-censored observations.

21. Most of these features—idiosyncrasies, applications, relationships with software vendors—do not change rapidly regardless of whether they apply to C/S or to host-based systems, so measuring them in 1989 provides a good indicator of the establishment's goals in the early 1990s.

industry in which the establishment is located, its address, and, for a subset of larger firms, the parent firm.

CII asks questions about the software in use at establishments. Respondents report the most important programs, not the entire inventory. They typically name about thirty programs, divided roughly evenly between applications and underlying infrastructure (systems, tools, utilities). This represents far fewer programs than the establishments are running in total. Accordingly, reports from an establishment do not directly reveal all investments in software. Instead, they reveal what the survey respondent thinks is important about the establishment. For each software program reported, we know its author, its name, and its classification category (for example, accounts receivable or communications controller).

Regressors: Establishment-Level Variables

Using the software data, we build three sets of establishment-level variables: MIS variables, software-use variables, and software-author variables. These classifications of the software packages define the exogenous variables and come close to the heart of our hypotheses about the MIS agency and vendor lock-in theories, and about the complexity, costs, and importance of the co-invention process. The pragmatic goal of these variables is to predict much of the variation in establishments' switching behavior. The classes of regressors are also linked to our theories of that variation.

Further, we introduce a rich set of size measures, all based on 1989 hardware, and a complete set of SIC dummies. This basic set of exogenous variables appears in all our models. The variables are defined, and their descriptive statistics offered, in table 3.

MIS VARIABLES. We measure the role of the MIS department by the proportion of the establishment's reported mainframe software programs that were written "in house," that is, by employees of the establishment. Our *INHOUSE* variable does not capture the myriad small subsystems MIS personnel have written (as one would find in any establishment) because CII data are based solely on what the survey respondent thinks is important. Thus, the variable measures the make or buy decision for the few, important business systems that define the character of the establishment. Extensive use of in-house software in-

Table 3. Definitions and Statistics for Regressors

Variable	*Description*	*Mean*	*Standard deviation*	*Median*	*Percent zero*
MIS variables					
INHOUSE	Percentage of software applications written by establishment employees	0.185	0.245	0.102	33.73
HOUSEDB	*INHOUSE* DB*	0.029	0.060	0.014	39.99
HOUSECM	*INHOUSE* COMM*	0.035	0.048	0.019	40.41
Software-use variables					
SCI	Percentage of scientific and number-crunching software	0.037	0.082	0.012	49.95
MANUF	Percentage of manufacturing and related software	0.009	0.050	0.000	88.00
STD	Percentage of standard business applications software	0.220	0.230	0.179	21.51
DB	Percentage of database and applications tools software	0.204	0.145	0.200	14.12
BASIC	Percentage of system software and utilities	0.246	0.179	0.250	15.91
COMM	Percentage of communication and networking software	0.284	0.176	0.276	12.67
MIPCM	*COMM* MAXMIP*	3.562	11.989	1.021	12.71
MIPDB	*DB* MAXMIP*	2.871	14.770	0.673	14.15
Software-author variables					
PROP	Percentage of software applications written by hardware vendor	0.539	0.279	0.500	7.29
PROPDB	Percentage of *PROP* and *DB*	0.154	0.147	0.135	19.12
PROPCM	Percentage of *PROP* and *COMM*	0.255	0.190	0.227	14.54
CONSULT	Percentage of software from consultants (and small software companies)	0.051	0.209	0.000	62.62
THIRDPAR	Percentage of software written by third-party authors	0.342	0.244	0.375	22.62
MPLAT	Percentage of software written to run on multiple mainframe platforms	0.068	0.133	0.000	51.30
Size and other establishment variables					
MAXMIP	MIPS of largest mainframe system	13.079	41.386	3.7	N.A.
MINMIP	MIPS of smallest mainframe system	6.046	12.204	2.0	N.A.
SYSSUM	Total number of mainframe systems	1.776	2.547	1.0	0
ONESYS	Indicator if *SYSSUM* = 1	0.617	0.486	N.A.	N.A.
MINAGE	Age of youngest mainframe system	2.146	2.216	1.0	N.A.
MAXAGE	Age of oldest mainframe system	3.298	2.662	3.0	N.A.
MBLUE	Indicator if major system architecture is IBM	0.517	0.500	N.A.	N.A.

Source: Authors' calculations.
N.A. not applicable

dicates that the establishment has a localized and idiosyncratic co-invention process. An establishment with more in-house software will focus its resources on more idiosyncratic problems or establishment-specific problems.[22] This might have a direct effect on adjustment costs. It might also be a measure of the ability of the MIS department to hold up change that threatens its rents, under the MIS agency theory.

To give us further evidence, we construct a number of variables to be interacted with "in-house" to identify establishments with especially complex and idiosyncratic computing organizations. Two such applications, defined by the kinds of software tools used to make them (*COMM* and *DB*), are presented in the next subsection. Considerably more complexity and higher co-invention costs may be in play when such an application is written on in-house software. Under either the MIS agency or idiosyncratic applications theory, the importance of in-house software should be greater at more complex establishments.[23]

Establishments differ widely in their use of in-house software. The majority have very little, one third of establishments have none, and half have fewer than 10 percent (table 3). The mean exceeds the median because there is a nontrivial minority of establishments with very large fractions of local systems.

SOFTWARE-USE VARIABLES. The best-of-both-worlds theory and the co-invention theory are related to the complexity of business computer systems and to their embeddedness in the organization. In this section we build a series of proxies for complexity and embeddedness, based on the purposes of the software in use at the establishment.

Large-scale applications can be divided into a variety of categories. Different classes of applications need different management practices for designing and implementing new computer systems, such as C/S. To isolate which establishments may be running which applications, we make use of CII's categorization of the software and its application on large-scale computing systems. We combine its detailed categories into five groupings: *DB* (database and applications tools software), *COMM* (communication and networking software), *SCI* (scientific and other number-crunching software), *MANUF* (manufacturing and related

22. Ito (1996) and Steinmueller (1996) support this view.

23. The interviews reported in Bresnahan and Saloner (1997) and our user-company interviews in connection with the preparation of this paper offer considerable support for our proxy interpretation here.

software), and *STD* (standard business applications software).[24] To create our regression variables for these variables, we calculate the fraction of software packages that fall into each grouping at each establishment in 1989. The denominator in each case is the total number of software packages at the establishment. The omitted category includes software that we find on virtually all large computers, such as operating systems. Table 3 offers descriptive statistics and other details.

The organizationally simplest type of computing may be termed "scientific calculations." Examples of such applications include diverse activities such as large-scale simulations of weather or geologic formations, statistical forecasting, airflow simulation, and some complex financial modeling. Emphasis is put on speed of calculation and graphics rather than on data input and output. These applications began to migrate off large-scale computer systems in the 1970s and 1980s, first to (noncommercial) minicomputers, then to networked clusters of minicomputers and "diskless workstations."[25] The complex management features used for applications in large organizations were largely unneeded. These users, often resistant to centralized management of computing resources, frequently used junior scientists rather than MIS professionals. Only a few calculational applications remained on large commercial platforms by the late 1980s.

We measure these types of users with the *SCI* and *MANUF* variables. The scientific software includes simulation and large spreadsheet applications. The manufacturing software includes applications with strong engineering content, such as manufacturing support, and design tools. An establishment with higher *SCI* or *MANUF* will have more engineers and will have needs that are organizationally simpler to address. We therefore anticipate that such organizations will encounter less difficulty switching to client/server computing. From table 3 we see that *SCI* and *MANUF* are highly skewed, are often zero, and have standard deviation well over their mean. Thus, an establishment tends either to have or not to have scientific and manufacturing applications.

24. We based our groupings on a close reading of the similarities and differences between each market niche in the organizational complexity of the applications. *Data Sources, Software Census* (1993), and *Software Magazine*, Top 100 Special Edition, various years, provide related information on mainframe software markets.

25. By the 1980s the workstation had acquired a disk and thus had become a platform in its own right, primarily used for these calculation-intensive and graphics-intensive applications.

These engineering-style establishments' software variables are quite different than those of commercial establishments.[26]

We next distinguish between nonengineering applications with high adjustment costs and benefits and those without. An organizationally complex group of applications in administrative work received the industry label "on-line transactions processing."[27] This was the most adventurous commercial innovation in large-scale computing in the 1980s. It involved real-time high-speed processing, quick communication between user and processor, and fast sorting and updating of large databases. The applications were often essential or "mission-critical" for the enterprise. The provision of services to customers was tied directly to functions provided by the computer system in real time. Because the systems were closely linked to the immediate delivery of products or services, users could not tolerate long delay in the execution of tasks or unreliable service. Reservation systems, automatic teller machines, and other real-time inventory or point-of-purchase sales systems are well known examples.[28]

The organizationally simpler group of applications is often given the industry label "back-office accounting." This typically involves large databases, nightly or weekly updates to these databases, and similarly timed report writing. Examples include check-clearing, inventory management, internal control accounting, project management, personnel reimbursement, bank-account clearing, and other essential but non-urgent record keeping tasks or financial analysis. The defining features of all these applications are their scale, their regular timing in batch

26. Interviews in establishments of this type confirm that they are organizationally distinct, that the establishments tend to have a single numerical purpose, and that they are often in separate research facilities.

27. This category should be distinguished from manufacturing and similar process control, which automated the precision of finely tuned large-scale, repetitive actions in manufacturing plants and could involve real-time high-speed analysis, sorting, and updating. Yet, in spite of their scale, these applications often involved a small trained user base consisting of engineers instead of a large number of users. Moreover, these applications used primarily minicomputers and developed an industrial organization that only partially overlapped with the organization for large-scale administrative applications. See Bresnahan and Greenstein (1996).

28. Our interviews in establishments of this type show two things. First, delivering these services involves complex and distinctive software tools, so that the establishments are quite distinct in the data. Second, the computer system is deeply embedded in the business organization.

mode, and their use in improving administrative processes. The software markets for these applications tend to be very developed with many standardized products, consultants, and niche applications.

On-line transactions-processing establishments use a lot of communication and networking software (*COMM*). This software category includes numerous system programs designed either to enable mainframe-micro links or to control communications among a large user base. On-line transactions processing and a large community of users are present at establishments that heavily use this software. The costs of coordinating change to these applications should be high. The invention of any new communications software is expected to be followed by considerable co-invention by users.

CII distinguishes between "system" and "application" database programs. The former include software tools such as file management programs and powerful database programming languages. The latter include software for standard financial analysis, large-system accounting, and most back-office computing. System database programs (*DB*) are complex, powerful, difficult to use. Application database programs (*STD*) are more standardized, packaged, and generic.[29] Establishments with many database tools have more complex computing applications with numerous idiosyncratic features and significant co-invention. Establishments with standard database applications could still have complexity in their computing, but the generic features found in these applications call for less co-invention.

Finally, we interact *COMM* and *DB* variables with the *INHOUSE* variable and with two other measures of complexity, which we define below. These interactions help identify particularly complex computing establishments with the highest prospects for co-invention.

SOFTWARE-AUTHOR VARIABLES. We proxy for the closeness of relations between vendor and establishment by looking at the vendor-specificity of the software used at the establishment. To create the proxies, we divide all non-in-house software into classes according to its author: proprietary, multiplatform, third-party, or consultant, which we call

29. Our interviews in high-*STD* establishments show that such establishments are a catchall category, not clearly distinct from high-*SCI* or transactions-processing establishments.

PROP, *MULTI*, *THIRDPAR*, and *CONSULT*, respectively.[30] Proprietary software is authored by the firm that provides the host hardware. Multiplatform software is software that we found on multiple incompatible mainframe platforms in 1989. Third-party software programs are those that we encountered at more than twenty establishments within the database but not on multiple platforms. Consultant software (the omitted category) was found at fewer than twenty establishments.[31] The denominator for all of these is the total amount of non-in-house software. We further interact *PROP* with *DB* and *COMM* to measure whether an establishment is using both proprietary and complex on-line transaction applications.

Table 3 shows that half of non-in-house software is from a proprietary vendor. Third party is the next largest non-in-house software with more than a third of the non-in-house software market. The largest vendor in this category by far is Computer Associates, which grew in the 1980s through a deliberate strategy of acquisitions and mergers. Other important firms (each at about one-fifth the market share of Computer Associates in 1989) include SAS, Pansophic, MCA, Cullinet, Innovation, Syncsort, and Sterling.[32]

These software-author variables serve as proxies for two different theories: the vendor lock-in theory and the MIS agency theory. According to the vendor lock-in theory, the most locked-in users are users of proprietary software, followed by users of consultant software, in-house software, third-party software, and multiplatform software, in

30. Our previous study divided third-party software into the IBM compatible and non-IBM compatible systems, but we have not found this distinction useful for this study.

31. While twenty is a somewhat arbitrary dividing line, we thought it suited the features of these data well. Many programs show up at only a few establishments, testifying to the extent of the consultant software market for large-scale systems. We wanted to distinguish between these and the more widely diffused programs from larger, often well-known, companies.

32. Establishments do not tend to change their software suppliers very rapidly, so these relative rankings in the mainframe software market did not change much from 1985 to 1994. The interesting growing firms in the 1990s have been Legent, DBSS, and Candle, none of which was much larger by 1994 than SAS, itself representing less than 3 percent of non-in-house software. Mergers and acquisitions, notably by Computer Associates, remove many software vendors but not many successful products from the list.

that order. The market size theory sees the proprietary and multiplatform markets differently. Establishments using these types of software are in large markets. They should be more mobile as the market develops software for them.

SIZE AND CONTROL VARIABLES. Table 3 also includes variables that measure how an establishment's size affects its inclination to experiment with client/server computing. These variables are *MAXMIP*, *MINMIP*, *SYSSUM*, *ONESYS*, *MINAGE*, *MAXAGE*, and *MBLUE*. We use the MIPS (millions of instructions per second) rating of the largest and smallest general-purpose mainframe system as an indicator of the maximum and minimum demand on computing capacity (*MAXMIP* and *MINMIP*). Use of a large-capacity system indicates a large task or a cluster of smaller tasks (perhaps linked by common data). Use of a small-capacity system shows a need for mainframes instead of the next smallest alternative, a general-purpose super-mini. It may suggest anticipation of increasing capacity along well-understood mainframe growth paths as users' needs grow; super-minis have much more limited growth paths associated with them. So these variables may capture the establishment's past assessment of the pace of upgrading and replacement.

We also count the number of mainframe systems (*SYSSUM*). If the number is high, it may signal that the computing core serves a large end-user community. The coordination problems associated with a large community can slow the pace of change. Or a large establishment can be inclined to experiment because it can realize the economies of scale and scope necessary to try technical solutions with high fixed costs. The variable *ONESYS* indicates whether an establishment has only one mainframe system. This variable also captures the establishments where $MINAGE = MAXAGE$ and $MINMIP = MAXMIP$, by definition, which is an artifact of specification. It is important to keep this in mind because a typical establishment has one or two systems.

The maximum and minimum ages of the general purpose mainframe computing systems at an establishment in 1989 (*MAXAGE* and *MINAGE*, respectively) measure the distribution of time since upgrades. We include these for two reasons. First, there is a regular replacement and upgrade cycle for large systems.[33] Second, some establishments using

33. See Ito (1996) for measurement of this cycle and a theory of the lumpy investment costs that drive it.

quite old technology may be out of the business of developing new applications entirely. These variables are primarily included as controls. The variable *MBLUE* indicates whether the "major" system at a given establishment is from IBM or an IBM-plug compatible manufacturer. Because of the rarity of vendor switching, this classification will help us measure differences in the demand facing IBM relative to the other mainframe vendors.

INDUSTRY DUMMY VARIABLES. To capture industry effects, we use a set of industry dummies constructed from an IDC/Computerworld grouping of two-digit standard industrial classification (SIC) codes. We isolated any two-digit subindustry with 100 or more observations in 1989.[34] SICs for motor vehicles and motor vehicle parts were combined. We also combined various categories into a computer-producing sector. The absence of this sector is a long-recognized difficulty with standard two-digit SIC codes. To the extent that our other measures of computing applications do not capture user heterogeneity, these industry-based measures will capture some unmeasured effects—such as competitive pressures, type of user base, or the diffusion of new applications—that are correlated within industry.

Our procedures produced thirty-six industry categories (table 4). Most large organizations in the United States are represented in our dataset. Approximately 15 percent of establishments come from Fortune 500 companies, and close to 90 percent of the Fortune 500 have a representative establishment within our sample. We note the wide scope of computer-using industries throughout the U.S. economy. Yet our establishments are heavily concentrated in intensive computer-using industries, such as banking, insurance, and financial services (SIC 60-69), on-line services (SIC 73), wholesale trade (SIC 50-51), government, and a few engineering-oriented industries such as electrical, computing, and transportation manufacturing (SIC 35-38), or areas dominated by science-oriented R&D, such as oil and gas exploration (SIC 13) or education (SIC 82).

NO TIME-VARYING VARIABLES. All of the variables described here concern the state of the establishment circa 1989. None measure time-varying effects—such as the increasing attractiveness over time of

34. We did not separate any miscellaneous or catchall category as its own SIC group, nor did we remove any subindustry from its original group if the remaining subindustries would comprise fewer than fifty establishments.

Table 4. Industry Dummies

Industry	*Industry group*	*Establishment*	*Percentage of sample*	*Standard industrial classification (SIC) composition*
Mining and construction	SICGRP1	99	0.789	10–12, 14–18
Oil and gas extraction	SICGRP2	110	0.877	13
Manufacturing	SICGRP3	351	2.830	21–25, 29
Food products	SICGRP4	213	1.714	20
Paper and allied products	SICGRP5	111	0.893	26
Printing and publishing	SICGRP6	287	2.288	27
Chemicals	SICGRP7	288	2.312	28
Miscellaneous manufacturing	SICGRP8	252	2.033	30–32, 39
Primary metals	SICGRP9	145	1.188	33
Fabricated metals	SICGRP10	216	1.746	34
Computers and related	SICGRP11	764	6.226	35, 365–368
Electrical apparatus	SICGRP12	198	1.610	361–364, 369
Motor vehicles & equipment	SICGRP13	183	1.475	371
Other transportation equipment	SICGRP14	170	1.395	372–379
Instruments	SICGRP15	164	1.403	38
Transportation services	SICGRP16	236	1.897	40–47
Telephone communication	SICGRP17	134	1.068	481
Other communication services	SICGRP18	49	0.407	482–489
Electric services	SICGRP19	116	0.957	491
Gas and sanitary services	SICGRP20	165	1.339	492–495
Wholesale trade	SICGRP21	730	5.890	50–51
Miscellaneous retail	SICGRP22	271	2.192	52, 55–59
General merchandise stores	SICGRP23	136	1.084	53
Food stores	SICGRP24	146	1.164	54
Depository institutions	SICGRP25	792	6.353	60
Nondepository institutions	SICGRP26	125	0.996	61
Security & commodity brokers	SICGRP27	104	0.837	62
Insurance carriers	SICGRP28	714	5.787	63
Insurance brokers & real estate brokers	SICGRP29	171	1.363	64–69
Hotel, personal services	SICGRP30	95	0.781	70–72, 75–79
Business services	SICGRP31	1598	2.921	73
Health services	SICGRP32	553	4.480	80
Legal, social, engineering services, museums	SICGRP33	456	3.739	81, 83, 84–89
Education services	SICGRP34	726	6.146	82
Federal government	SICGRP35	425	3.420	984
State/local government	SICGRP36	997	8.402	981–983

Source: Authors' data.

client/server computing relative to host-centric computing. As part of our description of our statistical models, however, we do account for time-varying effects.

Variables Measuring the Switch to C/S Computing

Our dependent variables measure the timing of switches from host-based computing to C/S, including starting times, completion times, and intermediate milestones and lags. All the variables are based on establishments' choices of computer hardware.

Even though the CII data are remarkably complete and detailed, the very nature of our study makes measuring the dependent variables inherently difficult. Just as the boundaries between computer market segments, such as mainframe and minicomputer, have always had a degree of arbitrariness, the boundaries between client/server computing and host-centric computing are blurred. Client/server computing combines familiar computers running familiar software in unfamiliar ways. The problem is compounded by the confusion over standards and even over the definition of C/S.[35] Our solution is to limit attention to platform switches that can be reliably proxied by changes in the hardware in use at the establishment.[36]

That is the main reason we look only at establishments that once used mainframes. Switches from mainframe-class hosts to C/S inevitably involve a change from the host to the server hardware during our time period.[37] Establishments that replace host-based applications with

35. Bresnahan and Saloner (1997) report numerous anecdotes related to buyers' confusion about the definition of client/server technology, the direction of technical change, and the most efficient path for development.

36. We use hardware definitions to work around a fundamental problem of observables: the lack of an unambiguous definition of C/S computing. Since there is no standardized client/server networking platform, CII cannot ask buyers whether they are using it. As a result, we cannot build our dependent variables around a C/S platform. While there were ad hoc surveys conducted by consultants and market research firms through much of our period, they tended to be carried out on a small scale, and they usually let the respondent define C/S.

37. The second half of 1994 saw several changes in mainframe technology and marketing in response to competition from C/S, and mainframe computers became much more usable as servers. As a result, the basic assumption of this study—that an establishment using mainframes is using them for old-style, host-centric computing—might grow less valid after 1994. During our sample period, however, mainframes were overwhelmingly used as hosts not servers.

C/S applications will also replace mainframe hardware with server hardware. These establishments are typically workstations, sometimes minicomputers or PCs. Similarly, an establishment that builds new C/S applications will need new server hardware. We discard changes to PC-class servers since they are irrelevant to our study of large-scale computer facilities, which as a rule found larger servers more suitable to their needs.[38] These specification choices give us a clean observable for the dependent variable, a change in the class of in-use hardware from mainframe to workstation or minicomputer. Consequently, our treatment of the demand for client/server computing is incomplete. First, we miss switches from other kinds of hosts such as super-minicomputers to C/S. These switches are economically quite similar to the mainframe downsizing choices we study. Since they do not necessarily involve a change in hardware class, however, they are much harder to detect in the data. Second, we miss all client/server computing with PC servers, which means some "up sizing" of applications formerly run on personal computers or on local area networks. While these two events are an important part of the demand for C/S, they are economically distinct from downsizing.

SWITCH MEASURES. Our measurement of switch behavior examines the timing and size of within-platform changes in computer capacity. Our definition of mainframe capacity is based on the MIPS of installed mainframe computers. Table 5 offers the MIPS ratings of several popular mainframe computers to give an idea of their overall power. The machines listed cover a range of technical vintages but all occur frequently in our data as late as 1994. To calculate mainframe capacity at an establishment, we simply sum the MIPS of all the mainframe computers in use.

Our definitions of server capacity are also based on MIPS. These MIPS are in different units than the mainframe ones because server and mainframe instructions are not comparable.[39] Table 5 provides examples of popular server-class computers and their MIPS ratings. It is not

38. It is also difficult to detect PCs that are used as servers separately from those used as clients.

39. Attempts to benchmark client/server computing against host-centric computing in commercial environments have proved quite difficult. In scientific environments the problem is easier. See Dongarra (1996) for a systematic investigation of a wide variety of machines.

Table 5. Selected Popular Computers by MIPS and Installation Date

Maker	*Model*	*Millions of instructions per second (MIPS)*	*Month and year first installed*
IBM architecture mainframes			
IBM	3081K	14.0	April 1982
IBM	3090-200E	32.0	May 1987
IBM	9375/60	1.3	Oct. 1987
IBM	4381-92E	7.8	Nov. 1988
IBM	3090-600J	114.4	Oct. 1989
IBM	9000-210	12.0	Dec. 1990
Other architecture mainframes			
UNISYS	V380	2.0	Oct. 1985
TANDEM	CLX	9.2	May 1987
UNIX-based workstation servers			
HP	9000-720	19.0	March 1991
HP	9000-827	53.0	June 1991
SUN	SPARC-SERVER-690	28.5	Sept. 1991
SUN	SPARC-SERVER-10	48.1	May 1992
IBM	RS/6000-980	86.9	Sept. 1992
UNIX-based workstation[a]			
SUN	SUN 4	7	April 1989

Source: Authors' data.
a. Typically used as workstation not as server.

always obvious whether a server-class computer is, in fact, deployed as a server. Workstation-class hardware can be used at workstations by individual engineers rather than as servers. Therefore, we define server capacity in two ways. Our broad definition includes all minicomputers and workstations, regardless of use, under the control of management information systems. Our narrow definition of server capacity looks only at machines running particular software (see data appendix for details).

PLATFORM CAPACITY MOVEMENTS. In figure 2 we saw a decline in the growth rate of mainframe capacity in central computing facilities and an acceleration in growth, dramatic by 1994, of server capacity. We now look at the establishment level to see the micro data behind these changes. These data strongly influence our definition of dependent variables.

We first explore how the rate of increase in mainframe capacity slowed down at the individual establishment level between 1985 and

1994. A variety of behaviors might be behind the slowdown. Figure 3 breaks down the year-to-year changes in aggregate mainframe capacity (MIPS) into four categories. All categories are expressed as percentage changes in the entire stock of computing capacity relative to the entire stock of mainframe computing capacity in our sample.

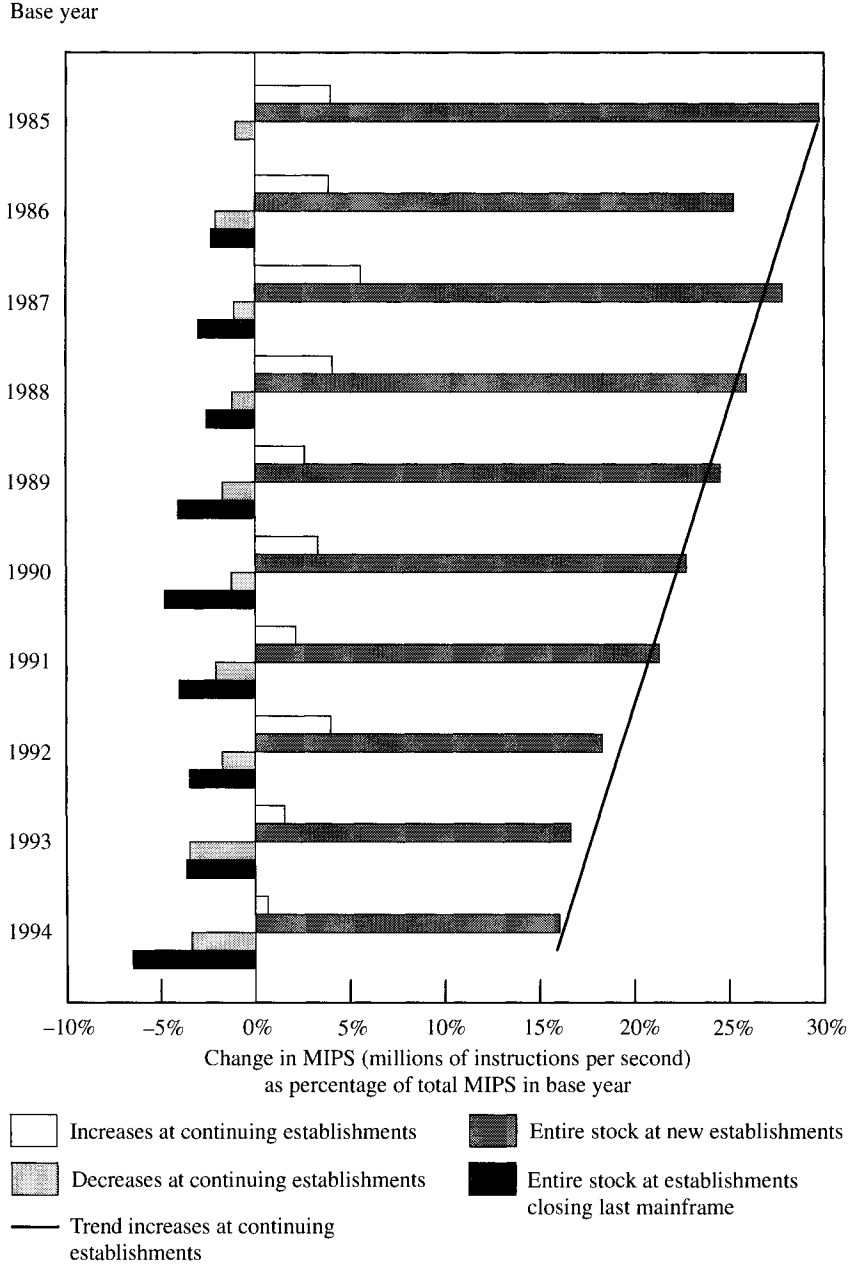
Two positive categories of capacity change are shown. The first is the increase in MIPS at establishments that already have a mainframe. These expansions were typically undertaken to make applications more responsive or to permit more applications to use existing data. The second category is total MIPS at new establishments or establishments using their first mainframe (often having migrated from a smaller platform.) Figure 3 also displays two negative categories of capacity change. The first bar on the lefthand side shows the total decrease in MIPS at continuing establishments experiencing a decline in mainframe capacity. In the late 1980s and very early 1990s, the decline could be attributable to establishments waiting for client/server computing. Later it is often a result of switching some applications to C/S. The second bar on the left represents the total MIPS at establishments closing their last mainframe. In the 1980s this usually coincided with the closing of an establishment.[40] In the 1990s switches to C/S contributed as well. Establishments that do not change their mainframe MIPS in a particular year—about 60 percent of establishments—are not shown in figure 3. To make the figure easier to read, we have included a trend line for the increases in MIPS at continuing establishments.

Overall figure 3 shows the composition of the downturn in mainframe capacity growth. The quantitatively most important change occurs in the category "increases at continuing establishments." These capacity increases represent about 30 percent of total MIPS in the mid-1980s and about 15 percent by the mid-1990s. Note the steadiness of the decline after 1987. In the category "entire stock at establishments closing last mainframe," no trend is detectable before 1994. Only at the very end of our sample period are establishments closing their last mainframe faster than historical averages.

We now examine a similar (simpler) figure showing the growth of

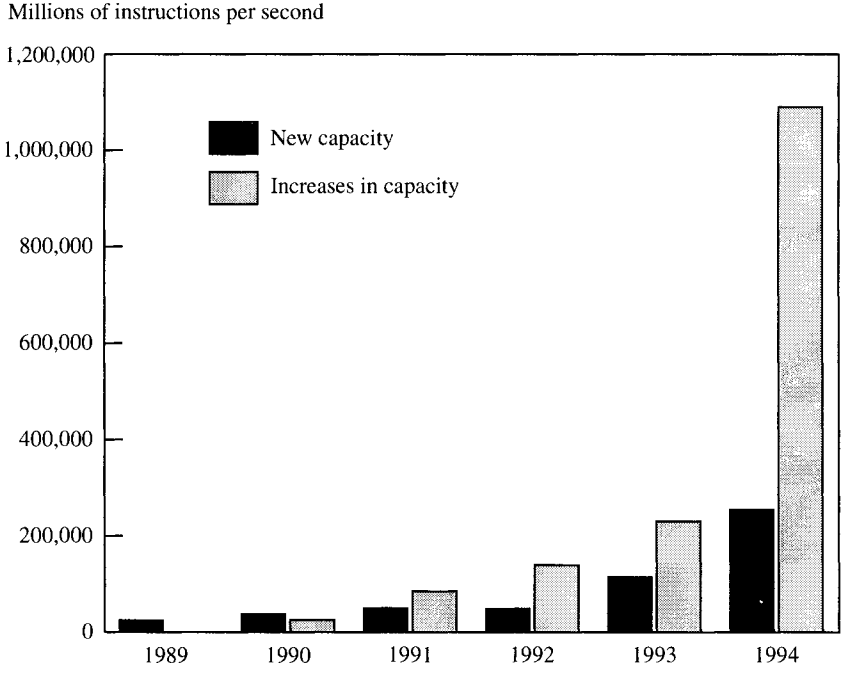
40. Closures include cessation of business at that location and relocation of computer support activities to an outsourcer's establishment or to another establishment of the same company through "data center consolidation." A few establishments simply stopped participating in the CII sample.

Figure 3. Four Elements of Mainframe Capacity Change, 1985–94



Source: Authors' calculations from CII data.

Figure 4. Growth of Client/Server Hardware as Measured by Increases in Server Capacity, 1989–94

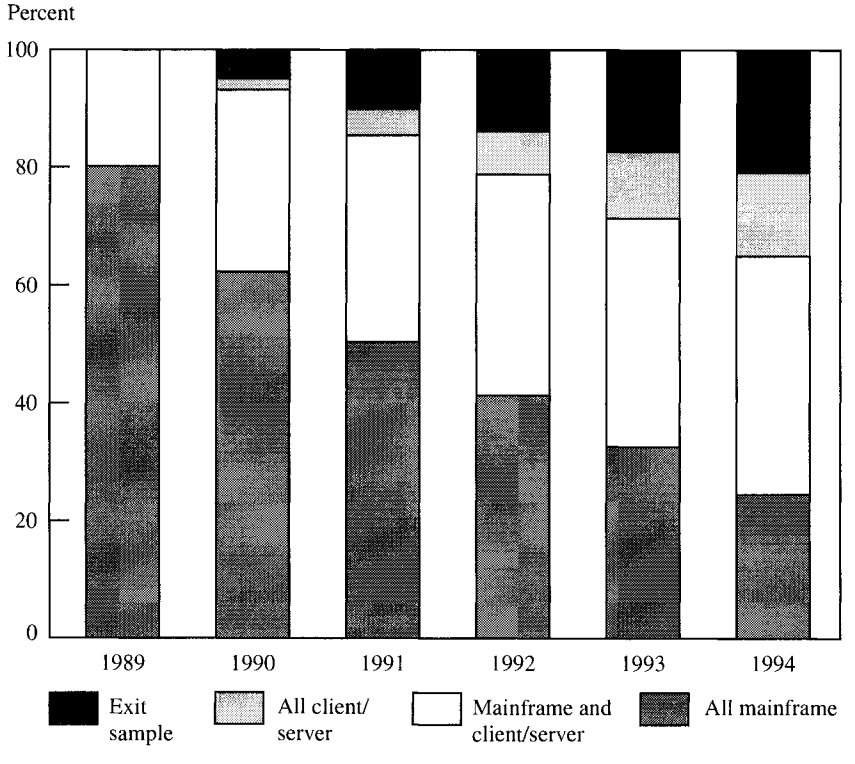


Source: Authors' calculations from CII data.

client/server hardware. Figure 4 shows that most growth in the stock of C/S hardware occurs within establishments that already have some of this hardware. Every year new establishments use C/S, but these new users do not account for much of the C/S capital stock, particularly in 1993 and 1994. Overall, the growth of the C/S capacity at experienced users is more important than the extension of C/S capacity at a greater number of establishments. This clearly reflects the long lags between initial experimentation with client/server computing and growth in use at any particular establishment.

These simple facts partly explain the switch process. The mechanism by which establishments make a *partial* switch from host-based to client/server computing involves bringing up new applications on C/S systems, avoiding an expensive expansion in host capacity. Establishments do not seem to be doing much partial decrease in their mainframe

Figure 5. Transition at Sample Establishments from Mainframe to Client/Server Computing, 1989–94



Source: Authors' calculations from CII data.

capacity. The *total* switches to C/S appear to be coming late in our period. Moreover, the overall downturn in mainframe investment was clearly not a unitary phenomenon but varied widely.

We now look at the transition from mainframe to C/S. Each establishment each year between 1989 and 1994 is classified into one of four states: all mainframe, mix of mainframe and server, all server, and exit sample. Figure 5 shows how the mixture of states is changing over time for the broad server definition. There is a clear downward trend, consistent with the overall platform shift. Notice, however, that movements from the 100 percent mainframe state to the mixed state decline over time, while movements from the mixed state to 100 percent C/S increase over time. Yet the trend is not old enough for 100 percent C/S to be very

common. At the end of our period, the mixed state is still the most common in the sample, followed by old-fashioned 100 percent mainframe.

Figures 3, 4, and 5 reveal several important trends. First, there is a dramatic shift in the 1990s in the relative use of mainframe computers versus smaller computers at corporate data centers. Second, the total quantity of mainframe computing power continues to expand, albeit more slowly, despite substantial substitution out of mainframes. Third, server capacity starts to grow before mainframe capacity falls off, and its growth accelerates with time. Fourth, some establishments expand both mainframe and C/S computing; others shut down mainframes completely, moving entirely to new platforms.

We thus can learn two simple lessons about the competitive threat to mainframes posed by client/server. First, a slowed rate of new investment in mainframes is more important than active disinvestment. Second, disinvestment is growing as a phenomenon. Such behavior is suitable for the adoption of new technological systems. A rational response by a buyer to a new technology is to adopt it slowly, using old capital to the end of its useful life.[41] As the older capital ages, it becomes economical to replace it completely. Compared with existing establishments, new establishments tend to choose C/S platforms because they find it easier to adjust their investment behavior.[42]

Figures 3, 4, and 5 justify our decision to analyze two important dependent variables: first use of C/S and removal of the last mainframe. They also justify our decision to ignore studying incremental decreases in mainframe capacity.

Establishment behavior varies widely in these figures. Our explanation of this variation comes from an underlying variety in each establishment's circumstances. Establishments have pre-existing applications that vary in effectiveness and maintainability, and they face a wide variety of costs and benefits associated with the switch to a new

41. This is a clear implication of sunk (irreversible) costs. Dixit and Pyndick (1994) talk about the implications of irreversibility for risky investments. Ito (1996) estimates a structural model of mainframe computer investment under irreversibility. Likewise, Bresnahan, Greenstein, and Ito (1996) provide evidence on the rare reversal of investments in computer capacity. These two empirical studies treat the period before competition from C/S.

42. Our sampling frame keeps us from saying anything about the technological choices of new establishments. The interview study of Bresnahan and Saloner (1997) has some related anecdotes.

platform or, indeed, with the improvement of existing applications. Moreover, they differ in the technical competence to deal with a new platform and in the role information technology (IT) plays in their firms' strategy.

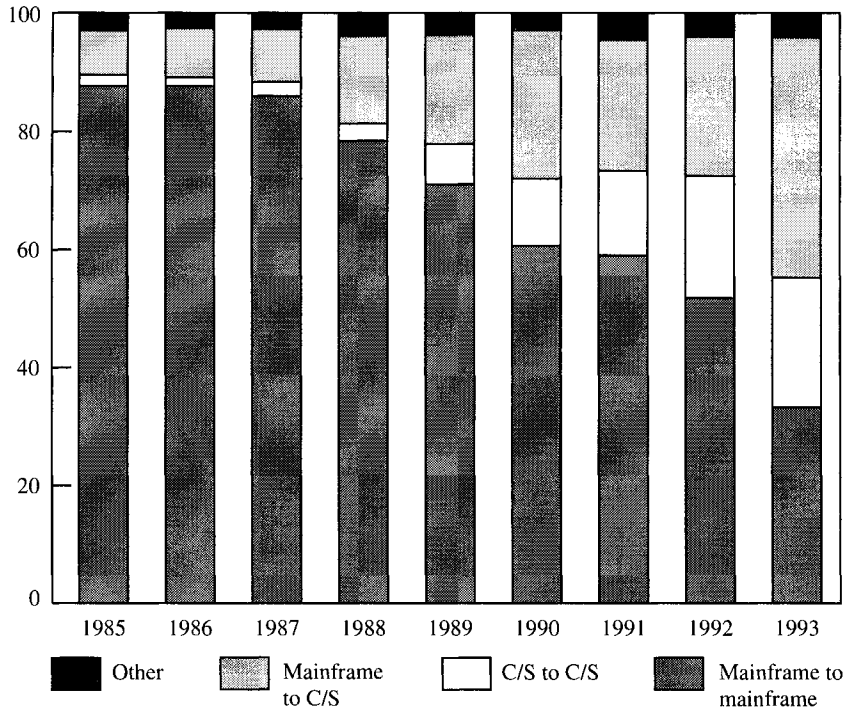
DUAL SYSTEMS OPERATION. What explains the long lag between the first use of client/server and the last use of mainframe? One cause could be within-establishment diffusion of C/S across applications. Some types of applications are very difficult to switch even now. Another possible cause concerns the risk of switching to a new platform. This possibility motivates us to look at the composition of capacity change.

All new business-systems development is risky because new computer systems can fail to do what they are intended to do. Users of large commercial computers have built up a set of risk-reduction mechanisms. One of these is dual systems operation (DSO): the old and the new systems are used simultaneously until the new system's reliability can be guaranteed. DSO is common where risk is high (for example, in mission-critical applications). It gives the new application's users an opportunity to learn and its programmers a chance to test its performance. This insurance is not free. Dual systems operation is very costly in both a monetary and organizational-specific sense; computers and software applications costing hundreds of thousands of dollars per year can be duplicated for many months. The advantage is that risks of failure can be avoided by keeping open the option of going back to the old system.

Of course, retiring applications on old platforms—unlike piece-meal expansion or incremental retrofitting of old systems with new capabilities—involves tremendous risk until the new applications are refined, running, and reliable. When the development of new applications involves platform changes, risk is at its highest.

For empirical purposes a DSO occurs when an increase in MIPS one year is followed by a decrease in the following year. DSOs can occur either within a platform or as part of a platform shift. Figure 6 sets forth the composition of DSOs from 1985 to 1993. In the mid-1980s almost all DSOs involved only mainframe capacity. Our establishments exhibit caution in installing new computer systems even in this era of stable platform use. Very near the end of our sample period, the composition of dual systems operations changes. The incidence of mainframe to server DSOs, in which servers experience an increase in ca-

Figure 6. Composition of Dual Systems Operations, 1985–93



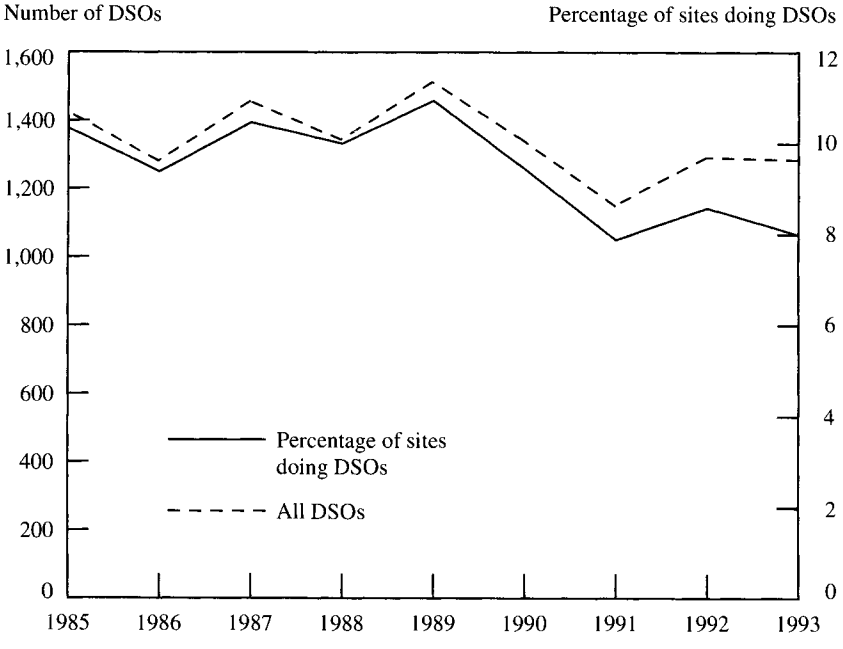
Source: Authors' calculations from CII data.
Note: It takes three years of data to define dual systems operation. We label DSOs by the middle of the three years. As a result, there are no 1984 or 1994 DSOs.

pacity one year and hosts undergo a decrease in the next year, increases considerably.[43] This means that there is real replacement of mainframes by client/server computers; it is not just that new applications are being built on C/S and old ones left on the mainframe.

Given the higher risk associated with systems development requiring platform shifts, we should expect establishments to increase their overall DSO rate in the 1990s as a risk-avoidance measure. On the other hand, it has been contended that C/S promises rapid applications de-

43. The incidence of server-to-server DSOs increases as well. This result, however, is less interesting since it simply reflects the growing importance of server capacity in the installed base.

Figure 7. Overall Trends in Dual Systems Operations, 1985–93



Source: Authors' calculations from CII data.

velopment cycles and thus operation of dual systems might be avoided. We look for preliminary evidence in figure 7, which shows the absolute rate of DSOs—of all kinds—and the percentage of establishments that undertake a DSO. Both are down in the late part of our sample, if not by a lot. Many establishments in the 1990s are switching platforms, but they undertake DSOs about 8 percent of the time, while the rate in the late mainframe-only era was 10 to 11 percent. The downturn occurs at the same time as the increase in the importance of mainframe to C/S switches, an interesting coincidence.

Two very different forces drive DSOs over time. We expect platform switches to be riskier and thus to have more DSOs. Yet the simpler program development environment of client/server may be reducing their incidence.

ENDOGENOUS VARIABLES. To learn when an establishment began using C/S systems, we construct *CS-START*, which measures the date an

Table 6. Number of Establishments Crossing Switch Thresholds over Time

	Number of establishments		
Year crossed	*CS-START*	*CS-5MIP*	*CS-ALL*
Broad definition			
1989	2,490	790	0
1990	1,702	935	236
1991	995	842	325
1992	779	813	359
1993	774	1,035	516
1994	775	994	421
Never crossed or crossed after 1994	5,030	7,136	10,688
Narrow definition			
1989	1,302	421	0
1990	949	427	236
1991	697	456	325
1992	557	463	359
1993	606	662	516
1994	1,205	1,103	421
Never crossed or crossed after 1994	7,229	9,013	10,688

Source: Authors' data.

establishment makes its first investment in client/server hardware. We construct a broad and narrow definition of this variable using our two definitions of servers. More precisely, our broad definition of *CS-START* is the year in which broad nonmainframe MIPS for an establishment exceeds its level in December 1988. The narrow definition is parallel. We define *CS-START* as an increase over 1988 to avoid false positive switches. Hardware that might be used as servers had been used earlier to assist hosts.[44] Table 6 shows that the narrow and broad definitions provide different measures of *CS-START*. As expected, the broad definition indicates more numerous and earlier starts than does the narrow definition.

CS-5MIP attempts to measure an intermediate milestone in the building of client/server systems. *CS-5MIP* measures the first year in which an establishment's server MIPS is at least five MIPS above its 1988

44. Many host-based systems used minicomputers, such as PDP-11s, as controllers of traffic flows. Also, some larger establishments contained a host-based super-mini computer, such as an early generation VAX. The explosion in minicomputer and workstation MIPS shown in figure 2 is not an expansion of this kind of capacity.

level. Five MIPS was chosen because it cannot be reached by a trivial amount of investment and thus should represent a substantial server or a set of servers. Once again, there is a broad and a narrow definition, and the broad definition shows faster switches (see table 6).

CS-ALL is the first year in which all of an establishment's large-system MIPS are in server system hardware. This definition is synonymous with an establishment retiring all its mainframe system hardware (because all establishments within the panel had to begin with at least one mainframe system). Thus, the broad and narrow definitions of *CS-ALL* are the same. In table 6, as in figure 5, we see that comparatively few establishments reach the all C/S status. This means that *CS-ALL* is right-censored by the end of our sample somewhat more frequently than are the other variables. The measure may be less informative as a result, and therefore it may be difficult to examine the behavior of the most mainframe-preferring establishments.

To buttress the amount of information in *CS-ALL*, we created *GROW-BOTH*, a dummy variable that is one if an establishment meets two criteria. First, server capacity (broad definition) must be higher in 1994 than in 1988. Second, total mainframe computing capacity must be higher in 1994 than in 1988. *GROW-BOTH* is our proxy for establishments that had no intention of retiring mainframe computing capacity in the early 1990s even after they began experimenting in client/server systems. We think of very late *CS-ALL* (unfortunately not observed because of the censoring) and *GROW-BOTH* as similar dependent variables.

Finally, we define variables related to the use of dual systems. Both *EASY* and *DSO-HARD* are measures of the process of adopting client/server systems. An easy DSO occurs when the length of time running the new client/server system and the old host-based system is short; server capacity increases in the same year mainframe capacity decreases. (This uses our broad definition of MIPS.) A hard DSO takes place when server capacity increases one year before a decrease in mainframe computing capacity, that is, when there is a mainframe-to-server DSO. Because it requires comparing successive years of stocks, the *HARD* variable needs three consecutive years of data. Table 7 shows the incidence of DSOs over time.[45]

45. Establishments that we classify as easy in one year and as hard in another are included only as hard.

Table 7. Dual Systems Operations and Switches to Client/Server Computing over Time

DSO	*1988–90*	*1989–91*	*1990–92*	*1991–93*	*1992–94*
Easy switch	273	402	224	171	167
Hard switch	84	133	122	186	165
Total switches	367	535	346	357	332

Source: Authors' data.

SEVEN EMPIRICAL MODELS. We first estimate five duration models to study the process of change from mainframe to client/server architectures. We estimate the models based on duration analysis for three stages of C/S switch (*CS-START*, CS-*5MIP*, and *CS-ALL*) for both our narrow and broad definitions of server capacity. All five models have time-invariant covariates measured as of 1989. The dependent variables are the length of time before reaching the C/S milestone. The survival distribution is Weibull, and we estimate by maximum likelihood. Durations can be censored either by the end of our sample period in 1994 or by the exit of an establishment from the CII surveys. For example, if we never see an establishment install a server, and the establishment exits the sample after 1991, we treat its C/S start date as "greater than 1991."[46] The results are given in table 8 and include such useful descriptive statistics as the predicted mean duration to each switch.

We also estimate a probit model where the dependent variable is one at establishments where both host and server MIPS are growing. This model uses exactly the same regressors as the duration models. Its sample, however, consists only of establishments that have started to install servers (broad definition) before 1994, and its dependent variable is a dummy for growth in both server and mainframe MIPS from 1989 to 1994. We use this grow-both model to buttress our interpretation of the C/S finish duration. An establishment that adds mainframe MIPS while starting to use client/server is unlikely to remove its last mainframe anytime soon. Finally, we estimate a three-branch logit model for *EASY*, *DSO-HARD*, and other. This model also uses the same regressors as above but adds a dummy for the year in which the switch began. The coefficients of the *DSO-HARD* branch of the logit and of the *GROW-BOTH* probit are also reported in table 8.

46. Such an establishment's contribution to the likelihood is the probability of reaching the date we last see it without the switch event occurring. We treat censoring by exit and censoring by the end of the sample in 1994 identically.

These seven models should determine whether our proxies for organizational complexity, vendor lock-in, size of establishment, and high co-invention costs are related to the adoption of client/server technologies at the establishments in our dataset. One may reasonably ask how we control for the relative costs and qualities of host and C/S platforms over time. We proxy these as functions of time, rather than by explicit measures of host versus C/S indexes of price per unit performance. There are two reasons for this set of specification choices. First, only a small portion of the relevant costs can be measured. Even though hedonic and related methods have been successful in measuring computer hardware price/performance, for the kinds of systems we study, the computer hardware is only a small percentage of overall costs.[47] These developments increased the attractiveness of C/S over mainframes as time passed, while the easy-to-measure relative hardware costs moved more slowly. The 1989–94 period saw improvements in the diversity and reliability of C/S software and wide compliance with stable interface standards; the pool of talented C/S programmers also grew. These developments are daunting challenges for price index measurement. Second, our time period is short, only five years. The ability to estimate the coefficient of any economy-wide, time-varying regressor in such a context is limited.

We do, however, include time effects. In the Weibull duration model the hazard function for a switch can grow or fall. This amounts (heuristically speaking) to including time and time squared in the underlying model for studying the attractiveness of client/server computing compared with host-based computing. The very use of a duration model permits the relative attractiveness of C/S to be growing over time. One interprets the switch hazard function as the fraction of establishments for which a critical attractiveness threshold has been passed. Our Weibull model adds an acceleration term. The scale parameter, μ, allows hazards to rise or fall over time.[48] Thus, it permits acceleration in the relative attractiveness growth rate. For the easy/hard DSO model, we include an unrestricted function of time as regressors shifting the value

47. On the measurement of hardware prices, see Gordon (1990).

48. In our study all of the spells begin at the same time, so there is no distinction between duration dependence (the usual interpretation of μ) and time dependence (our interpretation).

Table 8. Parameter Estimates of Seven Empirical Models

	Duration models						
Variable	*CS-START broad definition (1)*	*CS-5MIP broad definition (2)*	*CS-ALL (3)*	*CS-START narrow definition (4)*	*CS-5MIP narrow definition (5)*	*GROW-BOTH probit (6)*	*Easy/hard switch logit (7)*
MIS variables							
INHOUSE	0.0794	0.0991	−0.0737	0.0189	0.0258	−0.413	−0.064
	(0.1015)	(0.0969)	(0.1035)	(0.1104)	(0.1063)	(0.20)	(0.53)
HOUSEDB	0.4536	0.1302	0.4555	0.8081	0.3037	0.438	0.708
	(0.2303)	(0.2100)	(0.2546)	(0.2548)	(0.2342)	(0.43)	(1.30)
HOUSECM	1.0062	0.1070	0.1341	0.3386	0.7109	0.416	1.85
	(0.3351)	(0.3188)	(0.3108)	(0.3352)	(0.3282)	(0.54)	(1.48)
Software-use variables							
SCI	−0.3949	−0.2325	−0.5148	−0.5987	−0.2998	−0.931	−0.318
	(0.1068)	(0.0905)	(0.1064)	(0.1116)	(0.1012)	(0.21)	(0.61)
MANUF	0.2007	0.1036	−0.0551	−0.0079	0.0302	−1.139	0.026
	(0.1924)	(0.1810)	(0.1484)	(0.1832)	(0.1772)	(0.49)	(0.94)
STD	0.3401	0.3439	0.1371	0.2269	0.3588	−0.273	0.406
	(0.0732)	(0.0696)	(0.0787)	(0.0801)	(0.0768)	(0.14)	(0.40)
[Omitted utilities]							
DB	0.1260	0.2137	−0.1952	−0.2000	0.1161	−0.686	0.040
	(0.1133)	(0.1030)	(0.1253)	(0.1230)	(0.1155)	(0.22)	(0.63)
COMM	0.2464	0.0670	−0.2134	0.0023	−0.1093	−0.637	0.972
	(0.1239)	(0.1121)	(0.1309)	(0.1353)	(0.1233)	(0.23)	(0.67)
MIPCM	−0.0040	−0.0041	0.0006	−0.0026	−0.0028	0.111	0.421
	(0.0011)	(0.0009)	(0.0050)	(0.0011)	(0.0009)	(0.26)	(0.70)
MIPDB	−0.0020	−0.0019	−0.0015	−0.0006	−0.0010	0.207	−0.349
	(0.0010)	(0.0008)	(0.0034)	(0.0010)	(0.0007)	(0.19)	(0.50)

Software-author variables							
PROP	0.1603	0.2325	−0.0994	0.1060	0.1663	−0.494	0.309
	(0.0989)	(0.0963)	(0.0957)	(0.1054)	(0.1041)	(0.19)	(0.50)
PROPDB	−0.0833	−0.1702	0.2266	0.1594	−0.1083	0.501	−0.550
	(0.1143)	(0.1075)	(0.1206)	(0.1247)	(0.1194)	(0.22)	(0.62)
PROPCM	−0.2137	−0.1158	0.2150	0.0724	0.1261	0.458	−1.469
	(0.1260)	(0.1169)	(0.1276)	(0.1351)	(0.1274)	(0.23)	(0.64)
[Omitted consultants]							
THIRDPAR	−0.1477	−0.1070	0.2311	0.0413	0.0205	0.544	−0.419
	(0.0919)	(0.0875)	(0.0929)	(0.0975)	(0.0946)	(0.17)	(0.47)
MPLAT	−0.4313	−0.4614	−0.1292	−0.1924	−0.3742	−0.128	−0.169
	(0.1052)	(0.0975)	(0.1050)	(0.1127)	(0.1053)	(0.21)	(0.56)
Size and other establishment variables							
MAXMIP	0.0005	0.0004	0.0016	−0.0002	0.0002	−0.158	0.133
	(0.0006)	(0.0005)	(0.0020)	(0.0006)	(0.0005)	(0.13)	(0.35)
MINMIP	−0.0052	−0.0044	0.0041	1.2315	−0.0020	0.012	−0.021
	(0.0009)	(0.0007)	(0.0018)	(0.0010)	(0.0009)	(0.16)	(0.62)
SYSSUM	0.0047	0.0007	0.0997	−0.0013	0.0022	0.066	−0.015
	(0.0044)	(0.0037)	(0.0210)	(0.0050)	(0.0039)	(0.013)	(0.04)
ONESYS	0.1354	0.2043	0.0223	−0.0378	0.0694	−0.243	−0.080
	(0.0434)	(0.0404)	(0.0538)	(0.0466)	(0.0440)	(0.08)	(0.25)
MINAGE	−0.0231	−0.0100	−0.0871	−0.0495	−0.0341	−0.065	0.014
	(0.0066)	(0.0059)	(0.0086)	(0.0070)	(0.0065)	(0.01)	(0.03)
MAXAGE	−0.0031	−0.0044	0.0137	−0.0054	−0.0035	−0.480	0.027
	(0.0052)	(0.0045)	(0.0080)	(0.0057)	(0.0051)	(0.91)	(0.03)
MBLUE	0.0660	0.0226	0.0092	0.1004	0.0671	0.107	0.442
	(0.0402)	(0.0381)	(0.0408)	(0.0428)	(0.0411)	(0.08)	(0.23)
Industry dummies							
SICGRP1	−0.0140	−0.0777	−0.1705	−0.1078	−0.1368	−0.489	−0.274
	(0.1034)	(0.0955)	(0.0977)	(0.1086)	(0.1041)	(0.30)	(0.79)

Table 8. Parameter Estimates of Seven Empirical Models *(Continued)*

	Duration models						
Variable	*CS-START broad definition (1)*	*CS-5MIP broad definition (2)*	*CS-ALL (3)*	*CS-START narrow definition (4)*	*CS-5MIP narrow definition (5)*	*GROW-BOTH probit (6)*	*Easy/hard switch logit (7)*
SICGRP2	−0.3375	−0.5143	−0.6675	−0.5304	−0.5928	−0.648	−1.562
	(0.0865)	(0.0704)	(0.0711)	(0.0875)	(0.0722)	(0.27)	(0.72)
SICGRP3	−0.0666	−0.0736	−0.2000	−0.0584	0.0364	0.058	−0.883
	(0.0544)	(0.0550)	(0.0550)	(0.0605)	(0.0626)	(0.23)	(0.64)
SICGRP4	−0.0053	0.0567	−0.1965	−0.0768	−0.0469	0.083	−0.960
	(0.0694)	(0.0663)	(0.0715)	(0.0756)	(0.0728)	(0.24)	(0.66)
SICGRP5	−0.2128	−0.3151	0.0037	−0.2990	−0.2990	0.249	−1.196
	(0.0874)	(0.0733)	(0.1114)	(0.0913)	(0.0838)	(0.26)	(0.76)
SICGRP6	−0.1121	−0.0930	0.2010	0.0345	0.0105	0.128	−0.856
	(0.0587)	(0.0527)	(0.0880)	(0.0681)	(0.0646)	(0.23)	(0.67)
SICGRP7	−0.3162	−0.2821	−0.3370	−0.4237	−0.4178	−0.014	−1.055
	(0.0538)	(0.0468)	(0.0532)	(0.0556)	(0.0486)	(0.22)	(0.63)
SICGRP8	−0.0562	0.0138	−0.1296	−0.1339	−0.0615	−0.155	−0.718
	(0.0618)	(0.0591)	(0.0645)	(0.0663)	(0.0652)	(0.24)	(0.68)
SICGRP9	−0.1162	−0.0993	−0.0940	−0.1938	−0.1125	−0.044	−1.502
	(0.0800)	(0.0735)	(0.0838)	(0.0829)	(0.0813)	(0.26)	(0.74)
SICGRP10	−0.1801	−0.1674	−0.2407	−0.2122	−0.1867	−0.227	−0.640
	(0.0644)	(0.0585)	(0.0603)	(0.0676)	(0.0641)	(0.24)	(0.68)
SICGRP11	−0.3166	−0.3052	−0.2531	−0.3360	−0.2797	−0.113	−0.761
	(0.0350)	(0.0307)	(0.0359)	(0.0372)	(0.0345)	(0.21)	(0.60)
SICGRP12	−0.2555	−0.2439	−0.2572	−0.2034	−0.2207	−0.136	−0.671
	(0.0636)	(0.0560)	(0.0630)	(0.0701)	(0.0640)	(0.24)	(0.68)

SICGRP13	−0.1869	−0.1891	−0.1047	−0.2234	−0.2134	−0.167	−1.482
	(0.0704)	(0.0619)	(0.0783)	(0.0747)	(0.0679)	(0.24)	(0.70)
SICGRP14	−0.2504	−0.2926	−0.2639	−0.3154	−0.3423	−0.177	−1.265
	(0.0677)	(0.0575)	(0.0736)	(0.0702)	(0.0603)	(0.24)	(0.68)
SICGRP15	−0.3401	−0.3436	−0.3176	−0.4401	−0.4903	−0.187	−1.073
	(0.0692)	(0.0593)	(0.0654)	(0.0708)	(0.0598)	(0.24)	(0.67)
SICGRP16	−0.0433	−0.0283	−0.2304	0.0172	−0.0136	−0.016	−0.391
	(0.0640)	(0.0590)	(0.0664)	(0.0741)	(0.0694)	(0.23)	(0.66)
SICGRP17	0.3806	0.1876	−0.1760	0.1956	0.0994	−0.383	−2.599
	(0.0966)	(0.0832)	(0.1118)	(0.1103)	(0.0978)	(0.27)	(0.95)
SICGRP18	0.1042	−0.0622	−0.1021	0.0178	−0.0453	0.249	−0.655
	(0.1499)	(0.1313)	(0.1679)	(0.1668)	(0.1494)	(0.33)	(0.91)
SICGRP19	−0.1930	−0.2915	−0.0421	−0.2172	−0.2767	0.384	−2.448
	(0.0800)	(0.0665)	(0.1114)	(0.0880)	(0.0750)	(0.25)	(0.84)
SICGRP20	0.0268	−0.1301	0.2678	0.0276	−0.0678	0.529	−1.657
	(0.0714)	(0.0604)	(0.1234)	(0.0826)	(0.0729)	(0.24)	(0.80)
SICGRP21	0.1444	0.1519	−0.0401	0.1071	0.1199	0.054	−1.399
	(0.0419)	(0.0409)	(0.0446)	(0.0465)	(0.0475)	(0.22)	(0.62)
SICGRP22	0.1293	0.2140	0.0731	0.1921	0.2248	0.212	−1.535
	(0.0642)	(0.0641)	(0.0822)	(0.0769)	(0.0786)	(0.23)	(0.71)
SICGRP23	0.1421	0.3017	−0.0388	0.3032	0.2290	−0.351	−1.318
	(0.0923)	(0.0943)	(0.1189)	(0.1199)	(0.1150)	(0.27)	(0.80)
SICGRP24	−0.0242	0.0641	0.1228	0.1481	0.2311	0.350	−1.691
	(0.0821)	(0.0811)	(0.1150)	(0.1007)	(0.1103)	(0.25)	(0.72)
SICGRP25	0.4022	0.5285	0.1439	0.3729	0.4796	−0.224	−1.675
	(0.0454)	(0.0487)	(0.0582)	(0.0535)	(0.0604)	(0.21)	(0.62)
SICGRP26	0.2502	0.1862	−0.0128	0.3354	0.2570	0.323	−2.449
	(0.0990)	(0.0909)	(0.1189)	(0.1217)	(0.1178)	(0.27)	(0.95)
SICGRP27	−0.0324	−0.1296	0.0142	0.0641	0.0085	0.481	−1.117
	(0.0953)	(0.0814)	(0.1342)	(0.1111)	(0.1007)	(0.26)	(0.76)

Table 8. Parameter Estimates of Seven Empirical Models *(Continued)*

	Duration models						
Variable	*CS-START broad definition (1)*	*CS-5MIP broad definition (2)*	*CS-ALL (3)*	*CS-START narrow definition (4)*	*CS-5MIP narrow definition (5)*	*GROW-BOTH probit (6)*	*Easy/hard switch logit (7)*
SICGRP28	0.4980	0.4719	0.3059	0.5052	0.5226	−0.064	−1.287
	(0.0456)	(0.0456)	(0.0662)	(0.0559)	(0.0606)	(0.22)	(0.63)
SICGRP29	0.0449	0.1908	0.0826	0.0589	0.1306	−0.026	−0.950
	(0.0774)	(0.0789)	(0.0975)	(0.0879)	(0.0903)	(0.25)	(0.72)
SICGRP30	−0.1521	0.0103	−0.2608	−0.2912	−0.1589	−0.120	−1.136
	(0.0951)	(0.0930)	(0.1001)	(0.0996)	(0.0972)	(0.27)	(0.75)
SICGRP31	0.2547	0.2021	0.0486	0.2541	0.1366	−0.057	−1.260
	(0.0309)	(0.0291)	(0.0367)	(0.0360)	(0.0337)	(0.21)	(0.59)
SICGRP32	−0.4090	−0.3469	−0.0193	−0.3489	−0.2810	0.566	−1.213
	(0.0381)	(0.0327)	(0.0514)	(0.0413)	(0.0377)	(0.21)	(0.61)
SICGRP33	0.0012	−0.0809	−0.1310	−0.0600	−0.1298	−0.070	−1.194
	(0.0475)	(0.0426)	(0.0485)	(0.0512)	(0.0466)	(0.22)	(0.62)

SICGRP34	−0.4238	−0.5005	0.0922	−0.4073	−0.4253	0.567	−1.052
	(0.0342)	(0.0291)	(0.0459)	(0.0356)	(0.0314)	(0.21)	(0.60)
SICGRP35	−0.0063	−0.0921	−0.0312	−0.0671	−0.1234	0.059	−0.925
	(0.0484)	(0.0423)	(0.0611)	(0.0521)	(0.0465)	(0.22)	(0.62)
SICGRP36	0.0341	−0.0099	0.2893	0.0592	0.0669	0.338	−1.069
	(0.0326)	(0.0292)	(0.0493)	(0.0366)	(0.0353)	(0.21)	(0.60)
CS-START 1989							0.972
							(0.178)
CS-START 1990							0.273
							(0.183)
CS-START 1991							0.241
							(0.202)
CS-START 1992							0.491
							(0.205)
Weibull scale parameter	0.7808	0.5998	0.4537	0.7238	0.5507	N.A.	N.A.
	(0.0076)	(0.0071)	(0.0096)	(0.0088)	(0.0084)		
Average expected duration	4.7 years	6.0 years	14.5 years	7.7 years	7.6 years	N.A	N.A
ln (likelihood)	−14,344	−9,985	−5,351	−11,484	−8,131	−4,029	−5,052
Number of observations	12,296	12,296	12,296	12,296	12,296	6,668	6,668
Right-censored observations	4,720	5,272	10,343	7,101	7,414	N.A.	N.A.

Source: Authors' data. Standard errors are in parentheses.
N.A. not applicable

of each choice. There are no time treatments in the *GROW-BOTH* model since the dependent variable is measured only at one point.

Statistical Results

Parameter estimates for the duration models can be found in table 8. All are estimated under the Weibull model of duration dependence. The Weibull scale parameters, μ, are below one, especially for the *CS-ALL* model. This means, descriptively, that the switch hazards are rising with time.[49] The obvious economic interpretation is that the attractiveness of C/S is increasing at an increasing rate. This acceleration is faster for *CS-ALL* because improvements in the reliability and security of C/S systems reduced the risk of removing the last mainframe

Parameter estimates for the grow-both probit and the easy/hard switch logit are reported in the last two columns of the table. Our discussion focuses on a selected subset of outcome derivatives with regard to a selected set of changes in regressors. This leaves open the statistical question of whether sets of variables and their interaction terms can be excluded from the models. In results shown in the appendix, we see that the answer is a resounding "no."

The seven statistical models introduced a wide variety of regressors, including many interaction terms. Much of the economic interpretation, however, can be seen in key outcome derivatives. Accordingly, we do not discuss the estimates in close detail. Instead, we base our discussion on the predictive relationship between key changes in regressors and switch outcomes.

Role of MIS Departments

We first examine the relationship between in-house software and switch outcomes, taking the opportunity to describe our calculations in detail. We seek to understand whether the in-house software in complex environments predicts either a slow adoption of C/S or a slow switch

49. Models without this acceleration are rejected against the Weibull, while the Weibull cannot be rejected against a more richly parameterized model. See appendix for details.

out of mainframes or both. This will provide evidence regarding the idiosyncratic applications and MIS agency theories.

Table 9 reports on the empirical relationship between the fraction of software written in-house and four C/S switch measures: three duration models (columns 1–3) and the grow-both probit (column 4).[50] Since the *INHOUSE* measure is interacted with other variables (*DB* and *COMM*) in the models, derivatives (of expected outcome) with respect to it are not constants. We report values at two points corresponding to simple and complex environments. In the simple environment, $DB = 0$ and $COMM = 0$. These establishments are not using complex software tools to build applications. Zeroes for these two variables are common in the data. In the environment of complex applications, $DB = 0.23$ and $COMM = 0.31$. This point is at the other, nonzero, mode of the distribution of software user types. This environment is common in our data because the two variables tend to move together; many complex database applications also serve multiple users.

In the first three columns we report derivatives of the log of the expected duration before switch events, and the estimated standard errors of these derivatives. The three columns correspond to *CS-START* under our broad and our narrow definitions, and to *CS-ALL* (which is identical under broad and narrow definitions).[51] Our models treat the survival distribution as Weibull shifted by parameters, β, so that the expected time to the event is $E[T \mid x] = \exp(\beta' x/\mu)$, where μ is a distribution parameter related to the degree of duration dependence. Thus, the derivative with respect to x_j is β_j/μ. Finally, the last column of table 9 reports the probability derivatives from the grow-in-both probit and their standard errors.[52]

The first row of the table evaluates the derivatives of the outcomes with respect to *INHOUSE* at the $DB = COMM = 0$ point. This calculation does not depend on any interaction term. In the first row and column of table 9, for example, we see that the derivative of the log of the

50. We report results from only three of the five duration models we estimated because the results of the other models are similar.

51. These are calculated from the statistical models reported in table 8, column 1, and column 3, respectively.

52. These correspond to table 8, column 6. The probability derivatives take the form $h(x)\beta\ \phi(x'\beta)$, where $\phi()$ is the normal density function. We always evaluate $\phi(x'\beta)$ at the mean of the sample. For terms without interactions, $h(x)\beta$ is simply β_j, the relevant coefficient. For terms with interactions, $h(x)$ is evaluated at the points described below.

Table 9. Outcome Derivatives for Selected Families of Regressors

Variable	$\frac{\partial \ln E[T_{CS\text{-}START}]}{\partial x}$ *broad definition* (1)	$\frac{\partial \ln E[T_{CS\text{-}START}]}{\partial x}$ *narrow definition* (2)	$\frac{\partial \ln E[T_{CS\text{-}ALL}]}{\partial x}$ (3)	$\frac{\partial Pr(GROW\text{-}BOTH \mid C/S)}{\partial x}$ (4)
MIS variables				
INHOUSE (no *COMM* or *DB*)	0.10	0.03	−0.16	−0.14
	(0.13)	(0.15)	(0.23)	(0.06)
INHOUSE (*COMM* and *DB*)***	0.59	0.39	0.13	−0.07
	(0.10)	(0.12)	(0.19)	(0.05)
Software-use variables				
SCI	−0.51	−0.83	−1.13	−0.32
	(0.14)	(0.15)	(0.23)	(0.07)
MANUF	0.26	−0.01	−0.12	−0.39
	(0.25)	(0.25)	(0.33)	(0.17)
[Omitted basic system software]				
STD	0.44	0.31	0.30	−0.09
	(0.09)	(0.11)	(0.17)	(0.05)
DB (no *INHOUSE*)*	0.05	−.06	0.06	0.06
	(0.17)	(0.19)	(0.31)	(0.06)
DB (high *INHOUSE*)**	0.34	0.50	0.56	0.01
	(0.20)	(0.24)	(0.38)	(0.08)
COMM (low *INHOUSE*)*	0.02	0.09	−0.01	−0.06
	(0.17)	(0.21)	(0.31)	(0.05)
COMM (high *INHOUSE*)**	0.67	0.32	0.15	0.03
	(0.26)	(0.29)	(0.43)	(0.09)
Vendor connections				
MPLAT	−0.55	−0.27	−0.28	−0.04
	(0.13)	(0.16)	(0.23)	(0.07)
THIRDPAR	−0.19	0.06	0.51	0.19
	(0.12)	(0.13)	(0.20)	(0.06)

[Omitted consultants]				
Host vendor: other	0.21	0.15	−0.22	−0.17
	(0.13)	(0.15)	(0.21)	(0.07)
Host vendor: *DB*	0.10	0.37	0.28	0.00
	(0.13)	(0.14)	(0.17)	(0.07)
Host vendor: *COMM*	−0.07	0.24	0.25	−0.01
	(0.14)	(0.15)	(0.19)	(0.08)
Size and other establishment variables				
SYSSUM	0.006	−0.002	0.22	0.02
	(0.006)	(0.007)	(0.05)	(0.004)
One mainframe	0.17	0.05	0.05	−0.08
	(0.06)	(0.06)	(0.12)	(0.03)
MINMIP	−0.01	0.000	0.009	0.004
	(0.001)	(0.001)	(0.004)	(0.0009)
MAXMIP (no *COMM* or *DB*)	0.001	−0.0003	0.004	−0.002
	(0.001)	(0.001)	(0.005)	(0.0009)
MAXMIP (*COMM* and *DB*)***	−0.001	−0.002	0.003	−0.003
	(0.003)	(0.0004)	(0.002)	(0.001)
Age of youngest system	−0.03	−0.07	−0.19	−0.02
	(0.008)	(0.010)	(0.019)	(0.004)
Age of oldest system	−0.004	−0.008	0.03	−0.00
	(0.007)	(0.008)	(0.018)	(0.005)
Main system IBM architecture	0.08	0.14	0.020	0.04
	(0.05)	(0.06)	(0.90)	(0.03)

Source: Authors' data. Standard errors for the derivatives have beeen calculated from the variance/covariance matrix of the parameters using an asymptotic normal approximation. Standard errors are in parentheses.

*Derivative evaluated at *INHOUSE* = 0.11, *DB* or *COMM* software from host vendor, and *MAXMIP* = 2.1.

**Derivative evaluated at *INHOUSE* = 0.46, *DB* or *COMM* software from host vendor, and *MAXMIP* = 2.1.

***Derivative evaluated at *DB* = 0.23, *COMM* = 0.31.

number of years before an establishment acquires any server under our broad definition is 0.10, with a standard error of 0.13.[53] This is economically as well as statistically insignificant. A one standard deviation change in the fraction of software written in-house (approximately 0.25) would lengthen the expected time to the first acquisition of a server by less than 3 percent. Looking across the entire first row, we see that changing the fraction of software written in-house in these simple environments does not predict much change in any of the reported duration models. Neither does it change the probability of increasing mainframe capacity (conditional on acquiring a server).[54] All coefficients are small and statistically insignificant.

A different story emerges when we evaluate the *INHOUSE* derivatives for an establishment building more complex applications (in the second row.) Now the derivative in the broad definition start model is a substantial 0.59, and it is 0.39 in the narrow definition model.[55] The corresponding standard errors are much smaller. A one standard deviation change in *INHOUSE* now raises expected durations to starting the switch to C/S by 10 to 15 percent. In environments with complex applications (high *DB*), computing complexity embedded in organizations (high *COMM*), and site-specific applications (*INHOUSE*), there is a slow start in switching to client/server computing.

Most interesting, however, these environments are not associated with slow finishes. If we look at the third and fourth column of table 9, we see that the *INHOUSE* derivatives are small and insignificant in both the simple and complex environments. This means that having more in-house software does not predict a slower removal of the last mainframe (column 3). It also does not predict acquiring more mainframe capacity in addition to some server acquisition (column 4). Thus, the percentage of in-house applications predicts delay only in complex environments, and even then it predicts delay only in the start of the switch process, not in its end.

What interpretation goes with these coefficients? The relationship of

53. This is calculated as the ratio of the coefficient of *INHOUSE* to μ. For that model (see table 8) *INHOUSE* is 0.079 and μ is 0.78.

54. We do not show derivatives for the five MIPS duration models for either the broad or narrow definition of server capacity. The table of estimates shows that each of these is very similar to the corresponding start model.

55. These are calculated as $h(x)/\mu$ = (*INHOUSE* + 0.21 *INHOUSE DB* + 0.29 *INHOUSE COMM*)/μ for the duration models and, similarly, $h(x)\phi(x\beta)$ for the probit.

INHOUSE to *CS-START* is exactly what one would expect under *any* theory of why in-house software development matters, including both the MIS agency and idiosyncratic applications theories. The changed relationship to *CS-ALL* does not support the MIS agency theory, which posits successful resistance to a new platform in order to preserve information rents. The resistance, if it was there, was swept away. Of course, the changed relationship is quite consistent with the idiosyncratic applications theory. Under that theory, establishments using specific applications are slow to start switching because they need to invent changes. Once MIS personnel have learned how to rewrite those applications, however, the move can go forward. Thus, on the basis of our data we weakly prefer the idiosyncratic applications theory.[56]

Whichever theory of the *INHOUSE* variables one prefers, the difference in the coefficients between start and stop has an important message for the future. The stock of establishments by now has largely passed through the start phase. But only a small minority of establishments have retired all their mainframe hardware. The influence of *INHOUSE* is spent.

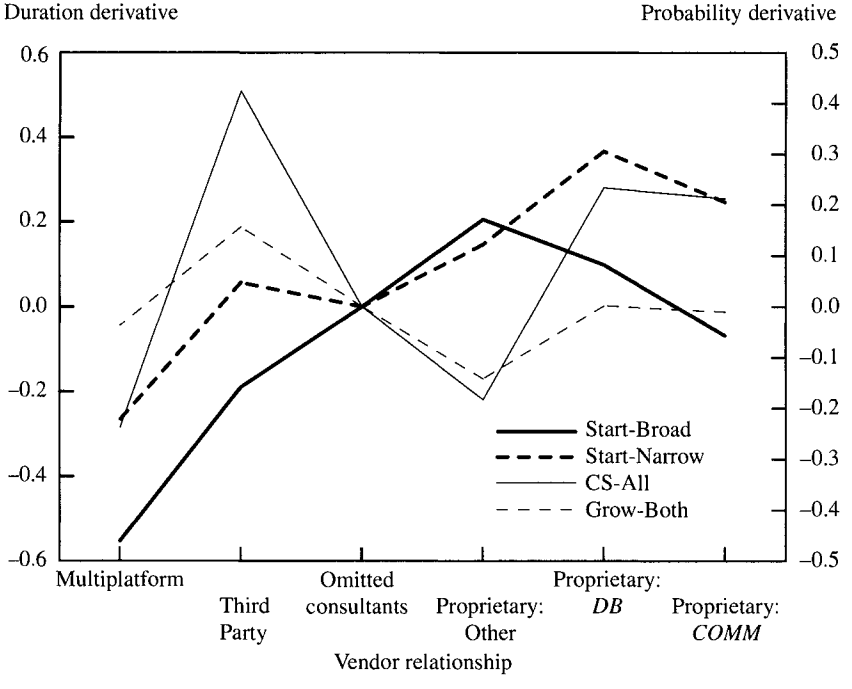
Buyer-Seller Connections, Lock-In, and Software Markets

Table 9 also presents the outcome derivatives for a series of variables related to software authorship. These variables are based on the industrial organization of mainframe computer system and software supply in 1989. We use them to capture variation in buyer-seller interactions and to test the lock-in and the market thickness theories. Even though the lock-in theory predicts slow starts, it is wrong about slow finishes. We will interpret this as evidence in favor of the market thickness theory.

Figure 8 presents the results from all four models in table 9. We have ordered the variables in such a manner that as one goes across the page, the links between the establishment and a specific host vendor grow closer. At the left is software from independent firms that in 1989 had already made their software available on a variety of platforms

56. It has the added advantage of being consistent with the prior empirical literature. Studies of power in organizations routinely reveal that MIS departments are a politically weak force. This has largely been taken up by organizational behavior scholars, such as Lucas (1984).

Figure 8. Impact of Vendor Relationship on Switch Milestones



Source: Table 9.

(*MPLAT*). At the right are complex software tools from host vendors (*PROPCM* and *PROPDB*); these categories of tools tend to be embedded in business systems running at the users' establishment and to be complementary to very specific human capital in the users' MIS department.[57] In between are software from other independent firms (*THIRDPAR*), other software from the host vendor (*PROP*), and software from consultants.

Both the broad and narrow *CS-START* duration models show that establishments with close vendor ties start more slowly. In both models the fastest-starting establishments are those whose software came from *MPLAT* vendors. The slowest-starting establishments are those that

57. For these we report the derivative with respect to an increase in the percentage of software that is both *PROP* and *DB* (*COMM*). Given the definitions of *PROP* and *PROPDB* (*PROPCM*), this is the sum of the *PROP* and the *PROPDB* (*PROPCM*) effects.

have much of one category or another from their host vendor (though the models differ on what category). Moreover, the differences between an establishment with considerable *MPLAT* software and one with considerable host-vendor software are economically and statistically significant. The rest of the coefficients are not ordered the same way in either model, however, nor does either model order them in exactly the way the theory suggests.[58] The bold lines in figure 8 corresponding to the two sets of start estimates are upward sloping but not over their entire range. These results nonetheless suggest that some establishments are more closely linked to their mainframe host vendor than are others and that these establishments start to switch more slowly.

The *CS-ALL* duration analysis and the *GROW-BOTH* probit are different. The thinner lines for these models are much less clearly upward sloping. The biggest effect, highly visible and significant in both models, is the large, positive coefficient on the (single platform) third-party software variable. The *CS-ALL* and *GROW-BOTH* models offer only very weak evidence that closer vendor ties are important. Indeed, when we add the two finish as well as the two start models to figure 8, the overall impression is not very upward-sloping.

The strongest form of the vendor lock-in theory is thus wrong in two economically important senses. First, it applies to starts much more than to finishes. The theory has been greatly concerned with persistence and permanence. Users here, even those with close vendor ties, do not appear to be "locked in" in any permanent sense. Instead, they simply appear to be more cautious in switching. The theory is almost certainly right that the users have made vendor-specific investments that must be overcome to induce them to switch. Our results emphasize that the marketplace can provide the necessary incentives. Modern information technology markets offer large switching benefits to new technologies. Moreover, they support a wide variety of sellers, many working to help users break out of locked-in positions.

Second, for finishes it is not the host-vendor-supplied software that appears to hold the users back but rather the third-party host-platform-

58. In the broad model, host-vendor software outside the key tools categories predicts slow starts. In the narrow model, difference between multiplatform software and third-party (single-platform) or consultant software is neither economically nor statistically significant. The broad and narrow models also differ on where third-party software stands relative to consultants.

specific software. Was there strategic manipulation of switching costs by vendors to keep monopoly market positions alive? If so, it did not work very well. Instead, it is the external, and uncontrolled, links between third parties and customers that have held users to the old platforms.

Indeed, the host vendors themselves have now rewritten their key software to make it work on the new platforms.[59] What happened within third-party software markets involved a hundred different stories, each with a similar outcome. Often small software firms did not find it economic to rewrite the software, since it served small, niche markets. Firms serving larger markets—whether they were multiplatform vendors or host vendors in 1989—found it much more in their interest to rewrite their products for the new, client/server platform. Both external market events and our results favor the market thickness theory over the vendor lock-in theory.

Establishment Size

The size effects in table 9 are largely absent. First, size effects with respect to computer characteristics are nonexistent. We look at the coefficients on *MINMIP* and *MAXMIP*; the latter is present at establishments with no *COMM* or *DB* software (around 1,000 establishments) and those with a good deal of both (about 10,000 establishments). These coefficients are quite small. A change in *MINMIP* from less than one to the low teens is well within our sample, as is a change in *MAXMIP* from less than one to the thirties. Yet neither of these changes alters any of the results. The story on size of the establishment is somewhat more complex. The number of systems (above one) is clearly irrelevant, just as is the size of those systems. There is, however, conflicting evidence about the presence of just one system. For starting the switch to C/S, establishments with only one mainframe are either slower to start (broad definition) or no different (narrow). For finishing, they are either no different (duration to last mainframe) or faster (not increasing mainframe capacity).

Overall, this evidence is mixed. It is wrong, however, to think of

59. For example, IBM-authored software previously available only on proprietary hardware now runs on most server platforms under the "open blueprint."

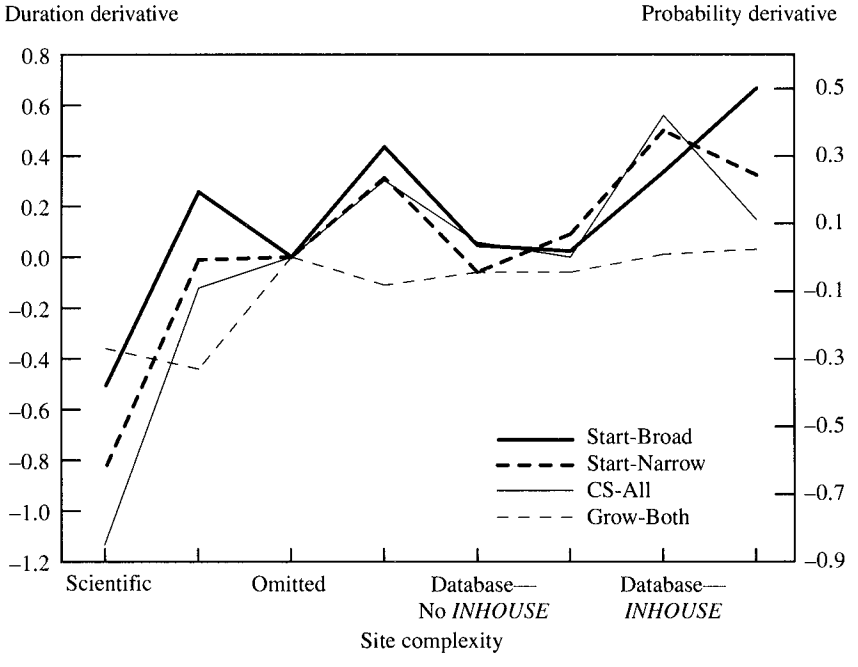
client/server as simply another new small-computer platform. We see no clear size effect, and C/S seems to be attractive to using establishments of all sizes, not overwhelmingly small ones. The entry of commercial minicomputers almost two decades ago segmented platforms by vertical product differentiation: mainframes for the largest users, commercial minicomputers for the smaller ones. The entry of C/S does not have this character; the product differentiation is along other dimensions.

This finding bodes well for networked computing in general and for client/server computing in particular. The economic and competitive impact of commercial minicomputers, of workstations, and of PCs was limited by their size and by the way they were sold and supported. They diffused to the extensive margin of new kinds of use, leaving preexisting technologies and their customers largely as they were. Client/server computing is qualitatively different. It is not limited to the smaller establishments or to the smaller applications. Networked computing can change the way applications are developed and used in all environments.

Software Use: Organizational Adjustment Costs and Co-invention

Scientific and other number-crunching uses of computing are the least tied to complex business procedures. Co-invention theory strongly predicts that these users will have the lowest co-invention costs; hence, they will move first to client/server computing and finish early. This is, indeed, what we find. Increasing the percentage of numerically intensive applications (the *SCI* variable) tends to decrease the switch times. For example, the probability derivative of -0.51 means that a unit increase in *SCI* tends to decrease the log of the mean start time for any server acquisition by 0.51 (table 9, column 1). This rate is relative to the omitted software-use category, basic system software. Since the *SCI* coefficient is smaller than all other software-use variables, establishments with more *SCI* packages and less of any other type of software tend to switch more quickly. More generally, variation in software use across establishments predicts considerable differences in timing. The use variables in table 9 are ordered according to the degree of complexity they represent in figure 9. As we move down the first column and look at the results for software uses, we notice a trend toward later

Figure 9. Impact of Site Complexity on Switch Milestones



Source: Table 9.

and later expected C/S-start dates. In particular, establishments using a greater percentage of standard business applications (*STD*) tend to move more slowly. Establishments that have written many of their own applications (high *INHOUSE*) tend to be slower the more database (*DB*) and communications (*COMM*) software they use.[60] Manufacturing and related engineering applications (*MANUF*), or *DB* and *COMM* software in establishments with no in-house applications, have intermediate effects. The pattern is hard to miss: the more computing is embedded in the using organization, the longer it takes the establishment to start using C/S. Scientific users (organizationally simple) are the fastest,

60. See the notes to table 9 regarding the point at which we evaluated the derivatives for variables with interaction effects, like *DB* and *COMM*. For these results, our definitions of high and low *INHOUSE* are at 0.5 and zero, respectively. There are quite a few zeroes in the data, and half of the reported software written in-house is well within the range of the sample.

while those using complex tools (*DB*) to build idiosyncratic (*INHOUSE*) and complex (*COMM*) applications are the slowest.

This finding is not sensitive to the particular choices we made about the interaction effects reported.[61] We also note that these differences are economically and statistically significant.[62]

How do these results change as we move to other aspects of the timing of the switch? Table 9, column 2, reports results using the narrow definition of starting the switch to C/S. Comparing the first two columns, we see that with a few exceptions the second column is equal to the first minus approximately 0.15–0.35, with the differences clustered around 0.25 for precisely estimated coefficients. Also, the estimated standard errors in both columns are quite similar. Column 3 reports the timing of finishing the switch. These coefficients are similar to those in the second column. Because of the relative rarity of finishing within our observation period, these coefficients are less precisely estimated. Thus, the statistical evidence is stronger for the start than for the finish of the switch. We summarize these patterns in figure 9, where there is a marked trend for organizational complexity to predict slow movement for both start and finish.

These results are strongly against the best-of-both-worlds theory and support instead the co-invention theory. Systematically, the organizationally complex sites adopt last.

Finally, compare the probability derivatives for the *GROW-BOTH* probit (table 9, column 4). The units are the change in the probability of increasing mainframe capacity. It is sensible, however, to examine the pattern of signs and orders of magnitude of the coefficients. We are particularly interested in whether the probit is similar to the duration model for finish. For the software-use variables the scientific and tech-

61. Interactions for *DB* and *COMM* depend on three things: *INHOUSE*, whether the software was obtained from the host vendor, and *MAXMIP*. We display how the results vary under *INHOUSE*. The coefficients of *COMM* and *DB* tend to be somewhat larger in the case we do not report, reinforcing our interpretation. *MAXMIP* simply does not influence these coefficients by much.

62. Most establishments have $SCI = 0$, and a typical value for a heavily number-crunching establishment is $SCI = 0.25$. Accordingly, the estimates in the first column of table 9 suggest that a within-sample change in *SCI* can decrease expected waiting time to start C/S by 13 percent ($(\exp(-0.58*0.25)-1)*100$) relative to the omitted category. A *SCI* establishment and an establishment with *STD* applications or an establishment writing much of its own applications using *DB* and *COMM* tools differ by more than 25 percent.

nical establishments are the fastest to remove their last mainframe and the least likely to increase both mainframe and client/server capacity. The difference between columns 3 and 4 is at the other end of the spectrum. Where the duration model says that the organizationally complex establishments are slow to finish a switch, the probit does not suggest that they are also likely to buy more mainframes. This difference is consistent with our co-invention story. If the organizationally complex establishments are holding back from switching because it is hard to move, this does not give them any particular motivation to increase mainframe capacity.[63]

An examination of the industry dummies (table 8) offers somewhat weaker but confirmatory evidence. Consider the set of industries where the establishments are both early starters and early finishers. These are oil and gas, chemicals, fabricated metals, computers, electrical apparatus, transportation equipment, instruments, transportation services, electric services, and hotel and other personal services. Users of the computing systems in all of these industries (with the exception of the last one, a very small group) are largely engineers or the science and engineering culture dominates the industry. These establishments are characterized by comparatively simple organizations and low adjustment costs. We interpret these industry differences in exactly the same way we interpret the coefficient of *SCI*.

More interesting is the contrast with the establishments that are late starters and late finishers: gas and sanitary services, miscellaneous retail, depository institutions, nondepository institutions, insurance carriers, insurance brokers and real estate brokers, business services (on-line services and system integrators), and state and local government. With the exception of the first group, which is very small relative to the others, these are unambiguously white-collar users of computers for administrative purposes and on-line transactions processing. They vary from the typically idiosyncratic users (insurance carriers, brokers, business services) to the technically conservative users of large databases (depository and nondepository institutions, state and local government). One would never mistake these organizations for engineering-oriented establishments; their main function is to provide information. Many of

63. Ito (1996) shows that these establishments have the longest gaps between new mainframe applications before the arrival of C/S, and they have substantial organizational adjustment costs associated with any kind of new computer system.

them cannot operate without their computing system. For these computing applications the costs of adjusting to new computing platforms are high, idiosyncratic, and often prohibitive. Thus, establishments with the most benefits to gain, in the long run, from the new platform are the slowest to switch. This is exactly what the co-invention theory predicts.

Looking at the industry results overall, we can identify three groups: the vanguard, middle guard, and rearguard. The first are primarily science and engineering-based industries (roughly 20 percent of all establishments). They are poised, in terms of their organization, to move very quickly to client/server computing. The middle guard represents about 35 percent of establishments. Then follows the rearguard of slow industries where organizational adjustment costs predominate. This group corresponds to perhaps 45 percent of establishments.

In general, one would expect industries to vary in both their costs and their benefits of switching to a new platform. We were surprised when most (though clearly not all) of the industries with extreme behavior seemed to be driven by extremes in adjustment costs rather than by extremes in benefits. This was further (unexpected) confirmation of the co-invention theory.

Dual Systems Operation

To examine the importance of DSO in switching to client/server computing, we estimate a model for whether establishments undertook a hard DSO (taking more than a year to switch), made an easy switch, or took some other action.[64] In the last column of table 8, we report some of the coefficients of a fully interacted three-branch multinomial logit.[65] The interesting margin is between easy switches and difficult ones. To highlight it, we have normalized the coefficients of the easy switch to zero. The regressors include all regressors in the models we discussed above and the dummies for the year in which the establishment began the switch to C/S. In the table we report the coefficients

64. The third catchall action contains establishments that increased C/S capacity and took either more than a year or more than the period of our sample to decrease mainframe capacity.

65. We adopt the multinomial logit not because we wish to have a random-choice interpretation, but because we want to achieve convenience in describing the three outcomes.

from the DSO-hard branch. These coefficients show, given our normalization, the likelihood that an establishment will have a hard versus an easy switch.

Most of our variables do not predict any difference between hard and easy switches. Indeed, it hardly seems worthwhile to examine the coefficients of the regular regressors we have been including in all these analyses. DSO is a coarse filter and an incomplete measure of the process of switching. There is, however, one very large effect in table 8—time. Over time the incidence of difficult DSOs falls relative to easy DSOs. Since the omitted category is the last time dummy, it falls steadily (even though there are some imprecisely estimated year coefficients that depart slightly from this trend). The time it takes an establishment to remove some mainframe capacity after adding some server capacity falls within our sample. This strongly suggests that making the switch is getting easier with time, exactly as the trade press reported. It suggests that the recent downturn in DSOs in aggregate may have been caused by the arrival of client/server, permitting easier new program development. If true, this interpretation underscores the possibility that idiosyncratic applications will become less of an issue for future installations of client/server networks. While this certainly does not mean that co-invention costs will disappear, it may portend a decline in co-invention costs associated with idiosyncratic applications and thin markets.

Conclusion

Information technology—broadly defined to include computers, software, and communications—is the most important technology today. It is a general purpose technology, widely useful in many sectors and, within them, for many functions. Yet its application is not automatic. Invention of any technology enables but does not direct its use. It is only the beginning of the innovation process. User co-invention completes it.

Because use of information technology does not keep pace with the technical progress in IT, three problems arise. First, white-collar work in bureaucracies, the place where most information technology enters the economy, changes slowly. Since the late 1940s it has evolved less

than the contrast between a tab card machine and a modern computer system would suggest. Second, there is a tension at the organizational boundary between IT users and MIS departments. Whether it arises from a clash of cultures or from asymmetries in information, it does not seem to go away. Third, information technology companies, however technically capable, find it difficult to commercialize their inventions. Correspondingly, the rate of technical progress in IT use is slower than the rate of progress in the underlying technology.

All three problems have arisen in wave after wave of IT invention and user co-invention. We examined a single wave, the transition from host-based computing to client/server computing. Yet the whys and hows of slow co-invention in C/S illuminate the broader question of how important technologies enter complex and diverse modern economies.

Two Demons

Observers have blamed the slow progress in IT use on one of two demons: sellers' monopoly or obstructionism in MIS departments. Our results undercut these explanations, suggesting more positive roles for both sellers and MIS departments.

There are tight bilateral links between buyers and sellers in many IT markets. Sellers' monopoly is thought to be a particularly bad problem in these markets because of co-invention. Users' investment leaves them "locked in" to sellers' proprietary platforms. Buyers are seen as passive and without real alternatives. Sellers are seen as sufficiently savvy about the technical environment—including buyers' co-invention—to manipulate it strategically. Our statistical results focus on the most famous of the alleged lockers-in, IBM, and on other mainframe vendors. These results paint a somewhat different picture. Users appear able to overcome switching costs to move to better or cheaper technology, and sellers do not appear to have maintained tight "account control" by use of their own software. We therefore reject the vendor lock-in theory.

Instead, we emphasize a completely different theory of the close relationships between IT buyers and sellers, an efficiency theory. The importance of co-invention offers a substantial market opportunity for sellers who fill the gaps between new technology and its uses. That is

how IBM built its strong market position in mainframes: it invented in the lab while it supported co-invention by its customers. Idiosyncracies and thin markets are significant economic barriers to lowering the costs of customizing IT and other co-inventive activity. Filling this gap promises considerable returns to sellers and thus encourages entry.

With regard to client/server computing, already a wide variety of firms are attempting to fill the gap. Software firms, systems integrators, consulting houses, and outsourcers all compete with technology inventors to support user co-invention. A nascent C/S monopolist will need to triumph over formidable competitors such as EDS, IBM, Oracle, Microsoft, Andersen Consulting, Sun Microsystems, and SAP. The vast differences among these firms illustrate the scale of the experiment in this information technology and demonstrate the low likelihood of monopoly.

Communication between buyers and sellers is a social benefit. Exchange of ideas regarding what the sellers have and what the users want helps the former invent the new technologies that co-inventors need, and it helps the latter adapt to changing technological realities. Communication also spreads and expands knowledge about technology and co-invention in a complex and dynamic co-invention equilibrium. Tight vertical relationships have always been a difficult area for antitrust policy; we think the current interventionist stance is unjustified.

Placing the onus on MIS departments is also unjustified. They are viewed as obstructionist and ignorant of real business needs. Many lament: if only the MIS people could be properly trained and given the right incentives, information systems would be easy to use and powerful. We demur once again and reject the MIS agency theory. Our results do not suggest that MIS departments are a powerful force against change.

Belief in the MIS agency theory has led companies to tinker with the boundary between MIS departments and business users for decades. Systems analyst positions were created. A fad developed for the internal use of the price system to give MIS departments highly levered incentives. More recently, the position of chief information officer became the rage, and there has been a wave of "outsourcing" MIS departments. None of these reorganizations has made co-invention easy. Business peoples' traditional suspicion of MIS is like blaming the messenger for the bad news.

The seller and the MIS demons arise from the same incorrect conception of technical progress. Thinking that the use of information technology has been unproductive leads to the need to find someone to blame. But investments in IT have been quite productive. There is no blame to apportion. The design and commercialization of IT products—sellers' economic function—and the conceptualization and construction of IT applications—the function of MIS departments—are difficult inventive activities carried out in uncertain environments. It is the inherent difficulty of these activities, not any failure of the firms and people doing them, that makes progress slow. Neither public nor business policy is well served by the demonization.

Invention and Co-invention

We focus instead on the sources of the actual bottleneck. There are close complementarities between invention of information technology and users' co-invention. Yet co-invention is slower. Why? Invention and co-invention draw upon different kinds of knowledge, and they exploit economies of scale at different rates.

Both IT invention and user co-invention involve the solution to very difficult analytical problems. On the invention side, these are familiar. Modern computer and telecommunications hardware are marvels of miniaturization and of complexity. Inventing this hardware calls for deep advances in the science of materials, in production processes to make integrated circuits, and in the design of complex machine logic. Some modern software, such as operating systems or database management systems, is among the most complex human artifacts ever. These systems are engineering marvels, and we are used to admiring the technical change that brings them to us.

Co-invention involves equally daunting analytical problems, though they are drawn from a very different domain. The information technology in contemporary business is so deeply embedded in policies and procedures that computer business systems define jobs, support reporting and monitoring in hierarchies, and permit the formation of teams. Co-invention is not merely the installation of a computer; it is the invention of a purpose for the system's output. That involves changing jobs, hierarchies, and other organizational structures by changing the information that flows through them. This is no trivial matter, as any

student taking a course in the economics of information knows. Theory suggests these inventions will be difficult, and our results bear this out. We find co-invention to be most difficult precisely where computer systems are most deeply embedded into the using organizations.

Technical progress, therefore, is not just bits and bytes. Its economic definition includes anything that permits more or better outputs to be made with the same inputs. The tight complementarity between IT invention and co-invention means that the slower one, in this case co-invention of organizational change, is the bottleneck.

If both invention and co-invention are so difficult, why have inventors been able to advance more rapidly than co-inventors? Part of the answer lies in the knowledge base on which invention and co-invention can draw. Technical progress in IT is organized technically: it uses science and engineering knowledge that is cumulative, hierarchical, and general. The knowledge base supporting organizational co-invention is not yet as well organized. Systematic attention to business organization is a new area of knowledge. The Harvard Business School is less than a century old, and efforts to put a deep scientific basis under organizational analysis, like those reported in the work by Paul Milgrom and John Roberts, are much more recent.[66] That co-invention is the bottleneck gives every encouragement to further development of the theory of organizational adjustment using new information technology.

This difference between the underlying scientific disciplines is apparent to anyone who knows both and provides part of the explanation of the distinct rates of progress. The rest of the explanation lies in the distinct economic organization of invention and co-invention. IT invention is subject to very large economies of scale. Once invented, a hardware design or a piece of software can be cheaply reused many times. The industrial organization of IT markets means that they are indeed reused. Some integrated circuit designs, and some pieces of package software, are sold tens of millions of times. Elements of a successful design will be reused in later designs, so the relevant scale economies are dynamic. Strong legal intellectual property protection (at least in the United States since the founding of the Court of Appeals of the Federal Circuit) and strategic mechanisms favorable to the cre-

66. See Milgrom and Roberts (1992).

ation of dominant firms mean that sellers of IT frequently appropriate much of the social return to their inventions.

Market realities have limited the effective exploitation of co-invention scale economies. Co-inventive activity is spread out over many different using functions, firms, and industries. To be useful, new information technology platforms, such as C/S, require significant customization within each idiosyncratic organization. This does not, by itself, prevent reuse of co-inventions. After all, ideas for effective use of IT could be shared across organizations formally through users' groups or informally by observation and imitation of successes.

In the case of client/server computing, this sharing was blocked. Information sharing is most effective when adopters are similar and the highest value adopters move first. They create knowledge that spills over to help later, lower value adopters. For C/S computing, it was not the highest value adopters who moved first. Our empirical findings showed that the co-invention theory, not the best-of-both-worlds theory, explains the time sequence of C/S adoption. The co-invention costs of moving to the new technology, not the benefits, determined the sequence. Thus, co-invention costs may determine the order of experience, limit the importance of spillovers, and waste accumulated experience.

There is also a broader lesson for technology policy. New technology and new engineering will not be sufficient to achieve large social gains if co-invention is the bottleneck. Certainly, there is an infrastructure to be built after a technology is invented, but the idea that the full force of the federal government should create "national information infrastructures" is at best naive. While we have no shortage of generic raw technology, there is a shortage of deep insight into co-invention. That is the bottleneck for social returns and likely the highest value locus for noncommercially motivated invention.

A more fruitful research path lies in understanding how invention and co-invention interplay to lead to economic gains and different market behavior. Do the forces pushing toward technological convergence resist generic organizational solutions because organizations are inherently idiosyncratic and complex, or are these idiosyncracies and complexities an artifact of particular technical regimes (in the delivery of services or within administrative functions) or out-moded management

arrangements? If idiosyncracy lies at the core of this resistance, what boundaries limit similarities between organizations—the markets for which they produce or the way their information technology interacts with each set of managers? If the bottlenecks lie in co-invention, what factors speed the entry of new tools to lower buyers' co-invention costs? Since spillovers accelerate when co-invention costs decline, do markets more efficiently organize communication between buyer and seller within a vertically disintegrated or integrated structure? These are difficult questions, but their answers are critical for the future rate of technical progress.

Data Appendix

For each year from 1984 through 1994, we received data from Computer Intelligence Infocorp. based on its annual surveys of business computer users in those years. These surveys contained detailed information about each system and software package in use at the surveyed establishment in a particular year. Individual systems were also given identifiers, but it was very difficult, if not impossible, for us to track use of particular systems at an establishment from one year to the next. Dynamic changes to components of systems each year make it hard to compare systems undergoing partial change, and CII had the practical difficulty of maintaining consistent system identifiers for tens of thousands of systems across time for such a large and comprehensive survey.

Therefore, we aggregate systems in each year up to the establishment level. In particular, we count the total number of systems and aggregate the capacity of systems, measured in millions of instructions per second (MIPS), at every establishment. We use CII's 1994 definition for the MIPS of each system and the closest indicator for a model where a precise model number was not available. If CII's label was imprecise, we made our best guess in matching the system with CII's guidebooks; if it was very imprecise, we did not count this system in our totals. We are confident that we correctly recorded the MIPS ratings for all but a trivial minority of systems.

Defining Mainframes

The dividing line between a large system and a supermini is sometimes an arbitrary one, determined partly by industry convention. We follow CII's definitions as to what is a mainframe and what is not. Appendix table A-1 shows the main system families from the main vendors. We display all the system families amounting to more than 5 percent (in system counts) of a vendor's mainframe or nonmainframe installed base. As you can see, the largest class of mainframes are IBM machines. The IBM nonmainframe families listed are all marketed as midrange, workstation, or small business systems. IBM-compatible machines from Amdahl, Fujitsu, and Hitachi are all mainframes. Within the DEC product line, the dividing line falls just below the VAX 9000. Our mainframe class also contains all "vector processor" systems, such as those made by Cray and Tandem, but they are a very small percentage of the total number of systems in the dataset. General purpose mainframe systems from Unisys, Bull, and others are also counted as mainframes, although their midrange systems are not.

Unit of Observation

Our unit of observation is the establishment over time, which means we need to aggregate the data to the establishment level. Prior to 1994, CII maintained a universe of "sites." A site was closely aligned with a data processing manager, and there could be more than one site in the same establishment. The 1994 survey put CII on standard economic definitions: it aggregated organizations with multiple sites at the same address to establishments, and CII now reports only establishment data. This definitional change affected approximately 600 establishments. We applied the same aggregations to our data prior to 1994; thus, we use CII's 1994 establishment definition applied retroactively.

We received data from CII in two shipments. One covers the period 1984–1992 and consists of sites that have at least one mainframe. In 1993 we began to receive data on small systems for establishments that were *once* mainframe users but had (before 1993) retired their last mainframe. For these sites, we can have a gap in the panel. A site that retired its last mainframe in 1991 will not have a 1992 sample entry, for example. We "backcast" 1992 server systems for such a site based

Table A-1. Important Processor Families and the Mainframe-Server Boundary

IBM	*UNISYS*	*Amdahl*	*DEC*	*Tandem*	*AT&T*	*BULL*	*HP*	*SUN*
1989 share in mainframe systems								
73.4%	10.2%	3.4%	0.8%	2.4%	2.7%	2.4%	0	0
1994 share								
68.3%	10.3%	5.8%	0.8%	4.4%	2.1%	1.8%	0	0
Mainframe families		[all]			[all]		[none]	[none]
308X	A		System 10	8500		DPS7		
3090	1100/X		System 20	8600		DPS8		
4331	2200		ACP 10000	8800		DPS9		
4381	V		VAX 9000	800				
9000	B2000							
9370								
Nonmainframe families		[none]			[none]		[all]	[all]
RS/6000	5000		all other VAX	3B		DPS6		
AS/400	6000		VS	8200		level 6		
System/36B	B800		Microvax	9000 Tower				
System/38	B1000							
Series/1	MICRO-A							

Source: Authors' calculations.

on its 1993 entry, using the then-reported acquisition-date field for the servers.

We removed establishments that appear in the dataset for only one year. We excluded other establishments, for which data might have started to appear some year after 1984 or stopped appearing before 1994. Entry of establishments is common as new sites become large-system users during this time period. Exit occurs for establishments that close, cease their computer use (most often this function is moved elsewhere in the company), or stop replying to CII's surveys.

Server Definitions: Broad versus Narrow

Our broad definition of servers includes any minicomputer or workstation class system at the central MIS establishments. For comparison purposes, we selected and aggregated a subset of the small systems—those that could be more narrowly defined as servers. Our narrow server definition is based on the software running on its small system. Using CII's application definition for each software package (there are approximately fifty such definitions in 1989), we include systems running software encompassing five or more categories (other than spreadsheet, CAD/CAM, manufacturing, or graphics). The fact that an establishment has fewer than five applications indicates that this hardware is probably being employed for a single dedicated use such as a controller function to run a manufacturing process, or as a non-networked workstation.

We exclude systems on which no software is reported to be run. If an establishment does not report any software for a piece of hardware, we presume that it does not perform an important (server-like) function. Notice that unlike our examination of mainframe software, here we assign particular software programs to the system on which they are used. To the extent that CII under-reports nonmainframe software, we will throw out some potential servers in the narrow definition.

We make a couple of exceptions to the above. First, when an establishment is running more than 10 percent of its mainframe software in the *SCI* and *MANUF* application categories defined in the text, we call it a server for engineering and scientific applications. In these cases we do allow spreadsheet, CAD/CAM, manufacturing, or graphics software, as all are common on servers in engineering-oriented shops. Finally, we assume that small systems at establishments that cease using

mainframes take on the server role for that establishment. Therefore, systems that would have been removed under the above criteria are retained as servers if their establishments have zero large systems by 1994.

Model Specification

The duration models we present assume the Weibull distribution. We investigated the importance and statistical reliability of this assumption in the model with *CS-START* as the dependent variable under the broad definition.

We re-estimated the model under several different distributions for the baseline hazard, including exponential, log-normal, log-logistic, and gamma, in addition to Weibull. Some of the baseline hazard distributions are restricted forms of others. For the Weibull distribution, the exponential is a special case (constant hazard, $\mu = 1$). A simple likelihood ratio test can be set up to test this restriction; in our case, it is quite resoundingly rejected. The Weibull model is in turn nested in the gamma model. This model, which attempts to discern a third derivative beyond the Weibull's acceleration, is difficult to estimate on the available data, and we cannot reject the Weibull against it.

The log-normal and log-logistic models are not nested in the Weibull nor vice versa. These models are single-parameter acceleration models as well.

It is possible to examine the validity of each distributional assumption individually by looking at the "generalized" residuals produced by the estimation:

$$e_i = -\log S(t_i \mid x_i)$$

Unfortunately, where the t_i is censored, this e_i will also be censored, and the usual test statistic is no longer appropriate. Instead, we plotted the residual against time, incorporating the censoring. The resulting graph should be a straight line with slope of 1 and an origin of 0, if the model is correctly specified. We plotted this graph for the Weibull, log-normal, log-logistic, and exponential models. As it turns out, only the last does not follow the straight-line pattern. The sign and significance of the covariate parameters do not change much among the nonrejected baseline hazard specifications.

Table A-2. Effect of Removing Sets of Regressors: Test Statistics and an Unconventional Significance Level

Regressor	*Degrees of freedom*	*Likelihood ratio statistic*	*Critical value: χ^2(d.f., .00001)*
Role of MIS	3	38.5	25.9
Software author	5	57.1	30.9
Software use	7	71.0	35.3
Size and other	7	119.6	35.3
SIC	36	1,119.0	86.1

Source: Authors' calculations.

The partial likelihood approach may be beneficial in cases where the baseline hazard is not known. In fact, the part of the model that depends on the values of the covariates alone is estimated separately. Two variations of the model were estimated with different methods for handling "tied" values of the dependent variable. In both cases the derivatives in table 9 with respect to covariates were very similar.

We conclude that the Weibull assumption, though used only for computational convenience, is empirically adequate. More complex models (gamma or partial likelihood) or alternative models of the same complexity (log-logistic and log-normal) lead to the same substantive story.

Excluding Sets of Regressors

In appendix table A-2 we show test statistics for excluding each set of the regressors from the broad definition *CS-START* model. The null hypothesis associated with excluding any of the analytical variables sets is easily rejected.[67] Accordingly, we do not report the comparable test statistics for other models.

References

Baldwin, Carliss, and Kim Clark. 1997. "Sun Wars: Competion within a Modular Cluster." In *Competing in the Age of Digital Convergence,* edited by David B. Yoffie. Harvard Business School Press.

67. We can, however, remove most of the sets of regressors without altering the remaining coefficients very much. The exception is the SIC variables, whose removal alters other coefficients.

Barras, Richard. 1990. "Interactive Innovation in Financial and Business Services: The Vanguard of the Service Revolution." *Research Policy* 19 (June): 215–37.

Bresnahan, Timothy F., and Garth Saloner. 1997. "Large Firms' Demand for Computer Products and Services: Competing Market Models, Inertia, and Enabling Strategic Change." In *Competing in the Age of Digital Convergence,* edited by David B. Yoffie. Harvard Business School Press.

Bresnahan, Timothy F., and Manuel Trajtenberg. 1995. "General Purpose Technologies: 'Engines of Growth'?" *Journal of Econometrics*, Special Issue 65 (January): 83–108.

Bresnahan, Timothy F., and Shane Greenstein. 1992. "Technological Competition and the Structure of the Computer Industry." Stanford University, Center for Economic Policy Research. Working Paper.

———. 1996. "The Competitive Crash in Large-Scale Commercial Computing." In *The Mosaic of Economic Growth*, edited by Ralph Landau, Timothy Taylor, and Gavin Wright, pp. 357–97. Stanford University Press.

Bresnahan, Timothy F., Shane Greenstein, and Harumi Ito. 1996. "The Sources and Effects of Investment Irreversibility: Large-Scale Computing." Mimeo. Stanford University.

Brynjolfsson, Erik, and Lorin Hitt. 1995. "Information Technology as a Factor of Production: The Role of Differences among Firms." *Economics of Innovation and New Technology* 3 (3–4): 183–99.

Chandler, Alfred P. 1997. "The First Fifty Years of the Computer Industry." In *Competing in the Age of Digital Convergence*, edited by David B. Yoffie. Harvard Business School Press.

David, Paul A. 1990. "The Dynamo and the Computer: An Historical Perspective on the Modern Productivity Paradox." *American Economic Review* 80 (May): 355–61.

Dixit, Avinash, and Robert Pyndick. 1994. *Investment under Uncertainty*. Princeton University Press.

Dongarra, Jack J. 1996. "Performance of Various Computers Using Standard Linear Equations Software." Working Paper CS-98-85. University of Tennessee Computer Science Department.

Ferguson, Charles H., and Charles R. Morris. 1993. *Computer Wars: How the West Can Win in a Post-IBM War*. Random House.

Freeman, Christopher, and Luc Soete. 1990. "Fast Structural Change and Slow Productivity Change: Some Paradoxes in the Economics of Information Technology." *Structural Change and Economic Dynamics* 1 (December): 225–42.

Friedman, Andrew L., with Dominic S. Cornford. 1989. *Computer Systems Development: History, Organization, and Implementation*. Wiley.

Gordon, Robert J. 1990. *The Measurement of Durable Goods Prices*. University of Chicago Press.

Greenstein, Shane. 1995. "Lock-in and the Costs of Switching Mainframe Computer Vendors in the U.S. Federal Government in the 1970s." *IEEE Annals of the History of Computing* 17 (3).

Hammer, Michael, and James Champy. 1993. *Reengineering the Corporation: A Manifesto for Business Revolution*. Harper Business.

Ito, Harumi. 1996. "Adjustment Costs in Mainframe Computer Investment." Stanford University.

Kador, John. 1992. "Downsizing Is Ready for Prime Time Midrange Systems." *Computerworld*, May 12, 1992.

Keefe, Patricia. 1990. "The Aroma Is Appetizing . . . But the Client/Server Main Course Is Still Simmering." *Computerworld*, January 1, 1990, pp. 35–37.

Lucas, Henry C., Jr. 1984. "Organizational Power and the Information Services Department." *Communications of the ACM* 27 (January): 58–65.

Milgrom, Paul, and John Roberts. 1992. *Economics, Organization, and Management*. Prentice Hall.

Mokyr, Joel. 1990. "Punctuated Equilibria and Technological Progress." *American Economic Review* 80 (May): 350–54.

Radding, Alan. 1989. "Size Is Beside the Point: All that Really Matters Is Fit." *Computerworld*, June 12, 1989, pp. 69–71.

Rosenberg, Nathan. 1996. "Uncertainty and Technological Change." In *The Mosaic of Economic Growth,* edited by Ralph Landau, Timothy Taylor, and Gavin Wright, pp. 334–53. Stanford University Press.

Saloner, Garth, and W. Edward Steinmueller. 1996. *Demand for Computer Products and Services in Large European Organizations*. Graduate School of Business, Research Paper 1370. January.

Steinmueller, W. Edward. 1996. "The U.S. Software Industry: An Analysis and Interpretive History." In *The International Computer Software Industry: A Comparative Study of Industry Evolution and Structure*, edited by David C. Mowery, 15–52. Oxford University Press.

Von Hippel, Eric. 1988. *The Sources of Innovation*. Oxford University Press.

Comments

Comment by David Brownstone: This is a thoughtful and interesting paper that examines the diffusion of a new intermediate-good technology. The authors have painstakingly constructed a dataset appropriate for modeling the adoption of client/server computing in place of mainframes for large enterprise-wide business systems such as accounting, payroll, inventory, and reservation systems.

These data allow the authors to define a number of milestones along the conversion from mainframes to client/server for each establishment in their data. The time to reach each milestone is explained by a Weibull duration model. The appendix shows that the authors are careful to check the sensitivity of their specification against different baseline hazard functions and exogenous variables. The Weibull model is an accelerated failure time model, which means that any change in a regressor resulting in an increase in the level of the hazard function also causes the slope of the hazard function to increase. Since all the regressors are fixed at their 1989 values, the resulting model is not particularly flexible. It would be interesting to check whether the Weibull shape parameter (which controls the slope of the hazard function) also varies with the key regressors in table 9. I doubt that this would change the author's qualitative conclusions since the increasing hazard functions imply no large unexplained heterogeneity.

The authors' conclusion that "co-invention" is a key factor in explaining the variation in adoption times rests with the statistical and economic significance of the variables indicating a large proportion of communication and database software written internally by the enterprise. They claim that these sorts of applications are deeply embedded in the organization and thus require considerable organizational change

to accommodate the switch from host-centered mainframes to client/server systems. While their arguments and evidence are compelling, the evidence does not warrant the implication that there is any market failure here. The co-invention process represents a real transaction cost, so we should not expect firms to ignore these costs and try to switch at the first possible moment. The authors also present evidence that the switching process became easier at the end of the data period (1994). They attribute this to more specialized consulting and third-party software being made available for client/server systems. If firms contemplating switching to client/server anticipated these developments, then they would have additional incentive to delay switching.

Although the authors have collected an impressive amount of data, they do not have any data on the prices of the hardware or software purchased or available to firms. I suspect that the relative prices of both client/server hardware and software were dropping (relative to mainframe software and hardware) during the period covered by their data. This implies that even in the absence of co-invention costs, firms would accelerate their conversion to client/server systems over time. Nevertheless, the co-invention costs identified in this paper are important factors explaining the diffusion of this particular new technology.

This paper makes significant contributions to our understanding of the micro details of technology diffusion in an important industry. It is hoped that this understanding will lead to better analysis of policies designed to remove alleged bottlenecks to technical progress.

Comment by Ken Flamm: To begin, the authors should be congratulated on charting a new course through an unmapped sea of difficult empirical questions. I will focus my comments on three issues: the stylized story told in the paper, the data that are being used and their limitations, and the interpretation of some of the results of their model.

The story told is that in the late 1980s a qualitatively new set of computer systems was associated with a qualitatively new way of organizing computer use—client-server (C/S) computing. Bresnahan and Greenstein focus on explaining differences in the way different types of users adopted the new computing technology.

One fundamental issue that needs to be raised is whether there really was some well-defined new technology known as client/server computing that suddenly appeared on the scene in 1989. ("Before 1989 work-

stations and personal computers could no more replace mainframes than could the people of Lilliput wrestle Gulliver to the ground.'') In fact, at least some observers (including this one) would argue that personal computers and workstations had been slowly but surely replacing mainframes (and minicomputers) in a variety of applications throughout the 1980s and that what the authors are really studying in this paper is the determinants of the slow but sure collapse of the last bastion of organized resistance to this trend, the centralized computing facility. ''Client/server'' computing, it could be argued, was merely the label finally slapped on this trend by the managers of the central computer facilities, and the trade press serving them, to rationalize their inevitable capitulation and to articulate a new philosophy designed to stabilize their continuing role in business organizations.

In fact, a trend toward more distributed computing had been under way for more than twenty years. It was driven by continuing declines in the costs of producing usable chunks of computer power. During the 1960s and 1970s, computing power was delegated to the end-user in the form of increasingly powerful and sophisticated terminals used to enter and display data, and in inexpensive, distributed small systems running stand-alone applications but linked to larger systems through time-sharing operating systems and the relatively primitive data communications networks. The spread of departmental minicomputers—often linked to larger mainframes—was another dimension of the increasing distribution of computing power among the end-users and away from the high priests of computing who were controlling the big iron in the central computer facility.

This is not to deny that smaller systems finally began to penetrate into central computer centers in the mid- to late 1980s and that the authors' study of this process provides valuable economic insights. But it does raise the question of whether the dichotomy of the small and networked computer versus the large computer is parsed in the most useful way. For example, IBM's large S-390 systems (classified by the authors as mainframes) have in fact been marketed as the servers in client/server systems during the period the authors study; similarly IBM's AS/400 series (classified as a client/server system) was selling well before 1989 and the official dawn of C/S computing. And personal computers (omitted entirely from this study) frequently were employed as servers in organizations during the period covered by this paper.

A second interpretative issue is raised by the paper's focus on central computing centers. The entire period studied was a period in which central computing centers saw their role as the main repository for computer hardware in most businesses shrink significantly, while departmental computing expanded. Much of the movement toward smaller, distributed, client/server systems surely took place at the periphery long before it penetrated into the resistant center.

The picture is further confused by another significant feature of computing in the 1990s—a trend toward outsourcing of central computer services. Replacement of company-owned mainframes with purchased services provided by third-party mainframes is going to be interpreted by the structure imposed on this data as a shift toward C/S computing even in instances when arguably it is not.

I will not attempt to do justice to what is a complex and ambitious modeling effort, but I would like to make a couple of observations. The first is that the authors seem to be estimating a demand function for client/server computing systems. One problem that leaps out is the absence of any data on price for these systems. Some clever points are advanced in arguing that this omission is not fatal, notably that the Weibull switching model implicitly includes a time trend effect and that computer hardware costs—which might arguably be measured with some modest effort—were a relatively small share of the overall cost of C/S computing. (Some of the factors mentioned by the authors as affecting the latter, like the available pool of C/S programmer talent, might be showing up in the industry dummies, complicating their interpretation.) But the absence of any price variable—when coupled with the report in an earlier version of this paper that a simple regression of MIPS (as opposed to a duration model of the switch to C/S) on these same variables and time dummy variables yielded nothing with any real explanatory power other than the time dummies—does make one wonder whether a more flexible or accurate proxy for price effects would make a difference.

A second issue in the modeling effort is the interpretation of some of the dummy variables. Is it correct to argue that a higher fraction of software in communications applications should be interpreted as signaling a more complex software environment? Fast switchers are scientific and engineering industries, while slow switchers are white-collar industries (ignoring a few pesky exceptions in both groups). But is it

correct to argue that one group switches faster than the other because of lower adjustment costs? Absent actual data on software complexity, or adjustment costs across industries, how is one to judge? The authors may be correct in these interpretations of their results, and they even make interesting hypotheses, but their findings certainly do not emerge organically as conclusions derived from their empirical analysis.

Having quibbled and carped, as is my duty, I must say I enjoyed the tale told in this paper. I look forward to the authors' next voyage into these waters.