
Beyond Chief Information Officer to Network Manager

John J. Donovan



Harvard Business Review

No. 88504

HBR

SEPTEMBER–OCTOBER 1988

Beyond Chief Information Officer to Network Manager

John J. Donovan

Decentralized computing is sweeping business like a wave rolling onto a beach. Its advance is unstoppable—and for some powerful reasons. Economics is one important factor. On a mainframe computer, the average cost per mip (millions of instructions per second), a benchmark of hardware performance, approaches \$200,000. On a personal computer the cost is only about \$4,000. A similar cost comparison applies to software. Individuals and small groups can develop computer applications faster and less expensively than large teams can. To avoid diseconomies of scale in programming, companies disperse the software development process.

Strategic and organizational factors are also driving the decentralization of computing power. By definition, information systems that link companies with distant customers or suppliers—the much-heralded use of information technology as a competitive weapon—cannot be restricted to a headquarters computer facility under tight central control. Within companies, the emergence of confident and capable computer users creates a powerful constituency fa-

voring decentralization. Employees want to operate their own systems, in their own way, and when it's most convenient for them.

The migration of computing power from corporate headquarters to divisions, plants, and desktops promises to reduce costs, enhance competitiveness, and renew organizational creativity. But it also poses profound technological and political dilemmas for senior executives responsible for computer operations. If they try to exploit the favorable economics of PCs and locally developed software, they may wind up with hundreds of isolated applications—what I call information islands—unable to share data. If senior executives focus on avoiding these information islands by resisting decentralization, they may not only forfeit attractive business opportunities but also invite mutiny among users wanting to develop systems of their own.

There is no shortage of cautionary tales about the perils for managers of decentralized computing. For example, the senior executive in charge of computer systems at a lawn-equipment company allowed its two major factories to develop their own information systems for tracking work orders and inventory. Both used stand-alone IBM System/36 minicomputers, but their application teams wrote incompatible programs customized to their factories' operations. This created problems when the CEO asked for regular

John J. Donovan is Chairman and Chief Executive Officer of The Cambridge Technology Group in Cambridge, Mass., an organization that builds major strategic applications for a variety of organizations in the public and private sectors. Professor Donovan is also a tenured faculty member at MIT's Sloan School of Management.

consolidated reports on shipments and inventory. The chief information officer's staff had to collect data from each factory separately and combine the information by hand. Responsiveness to the individual needs of each factory interfered with corporate-level data collection and analysis.

These problems are not confined to the private sector. One state government's information center allowed individual agencies like public welfare and finance to purchase their own computer hardware and specify the most urgently needed applications. To avoid connectivity problems, software engineers in the information center wrote the programs and made sure they conformed to standards. But long backlogs developed, and frustrated users began squeezing free applications from their hardware vendors or burying the acquisition of unauthorized software in equipment purchase orders. Ultimately, the government wound up with dozens of data bases and spreadsheets that could not feed into its central computer. It still has not unraveled the mess.

In the world of business computing, the 1980s has been the decade of the chief information officer. Company after company has named a senior executive, reporting to the CEO, to preside over its strategic information agenda and data-processing operations. CIOs and their staffs have been busy managing enormous hardware and software budgets, designing strategic applications, training new users, and building and running vast computer systems.

I believe the 1990s will witness the emergence of a new breed of senior information executive—the network manager—whose priorities and challenges will differ in many important respects from the CIO's. Unless CIOs successfully transform themselves into network managers, they will be ill-equipped to confront the user dissatisfaction, organizational squabbles, and technological roadblocks invariably triggered by the advance of decentralized computing.

What's the difference between a CIO and a network manager? Network managers understand that in a world of accelerating decentralization, the most effective way to oversee a company's computer resources is to relinquish control of them and instead focus on the networks that connect them. Network managers won't merely accept the inevitability of decentralized computing. They will encourage it by surrendering authority over hardware purchases and software development while seizing control of communications systems and policies. These areas will grow increasingly complex as hardware and software migrate down the organization, and the business relies more heavily on electronic links with customers and suppliers.

In practice, network management means evaluating hardware technologies with as much emphasis on telecommunications capabilities as on sheer processing performance. It means developing systems-level software tools that guarantee network security and create consistent, easy-to-use interfaces between workstations of different power built around different architectures. It means implementing connectivity standards throughout the organization so that users are free to revise their applications without jeopardizing the company's entire network. In short, it means setting technological and organizational ground rules to guide self-directed computer users.

Up to now, CIOs have governed computer use in decentralized environments using several distinct policies. (See the illustration, "The Four Stages of Decentralized Computing.") The logic of this framework is quite simple. CIOs make centralization/decentralization choices on three levels. The three dimensions of the framework correspond to these choices. The x-axis traces the degree to which companies distribute hardware to factories and offices. The y-axis reflects the decentralization of development functions like writing new applications and updating software. The z-axis tracks the location of decision-making authority over information systems—for example, who approves hardware purchases, or who determines what applications to develop.

The point on the framework where the three axes meet represents the set of centralized policies with which virtually every company entered the computer age three decades ago. Here, the corporate information staff wields absolute control: large mainframes available only to data-processing professionals run programs designed and written by centralized software teams. Today this set of policies is an organizational and technological dinosaur. To be sure, some companies still maintain administrative systems, like payroll and accounts payable, under rigid central controls. But the proliferation of minicomputers, technical workstations, and PCs has rendered extinct policies based on highly centralized hardware.

Point B, on the other hand, represents the "helping hand" method of managing computers in the current era of distributed hardware. Mainframes, minicomputers, and PCs are located in and are operated by the factories or branch offices that use them. But these departments do not write, update, or service the application software that powers their equipment. A central technical staff performs all development work in response to priorities set by the users. The state government information center described earlier operated under such policies.

None of these points represents a natural home base for an organization. The competitive demands of the industry, the organizational structure of the company, and the technological competence of the work force all influence the pace and method of decentralization. But virtually every company is moving (although at different rates) toward Point D, the “network” model. Let’s look at each of these policies in turn.

Big Brothers and Powerless Users

Big brother policies usually govern transaction-oriented systems where users have limited technical expertise—applications like point-of-sale scanners in supermarkets or hand-held terminals that help truck drivers track billings and deliveries. One distinguishing feature of big brother systems is their simplicity. Applications in this environment are specialized, with users choosing from no more than ten possible transactions. Moreover, since users of big brother applications tend to be line personnel, they usually lack clout within the organization to voice or act on any dissatisfaction they feel about the system.

The simplicity of the applications and the lack of expertise and organizational standing among end users allow the information systems (IS) department to maintain a tight grip on computing systems. It trains and supports users and distributes single, centrally developed versions of all software. Rather than give users source code that might permit them to tinker with an application, the IS staff distributes only binary machine code—a string of ones and zeroes that even experts might find difficult to alter and that unsophisticated users certainly couldn’t revise. In deciding how to meet users’ needs, the CIO often keeps an eye on controlling costs instead of trying to minimize response time.

This environment suffers from several built-in tensions. For example, though the users may spend the bulk of their day working with the computers, they have little control over how they operate. Companies that base service on minimizing costs often create long backlogs for users who seek to have programs updated or modified. Even if cost is not an issue, the IS staff in big brother environments often resists upgrading its responsiveness to users for fear that too many versions of an application may undermine software consistency. Backlogs and lack of responsiveness breed resentment.

That’s what happened at the department of motor vehicles in a large southern state. The department processed automobile registrations and title applications on dedicated IBM Series 1 computers located

in 187 tax offices throughout the state. To facilitate data feedback from these offices to a powerful Amdahl computer in the state capital, the IS department insisted on absolute control over the details of every application.

But its 12 programmers couldn’t make even minor changes fast enough to satisfy users. Some local offices wanted to enter names in an order different from standard. Others wanted touch screens to speed data retrieval. Still others wanted more efficient search routines for updating vehicle information on existing files. These disenfranchised users began tinkering with the programs and acquiring unauthorized software. State legislators, worried about applications overload of the IBM equipment, pressured central IS to clamp down.

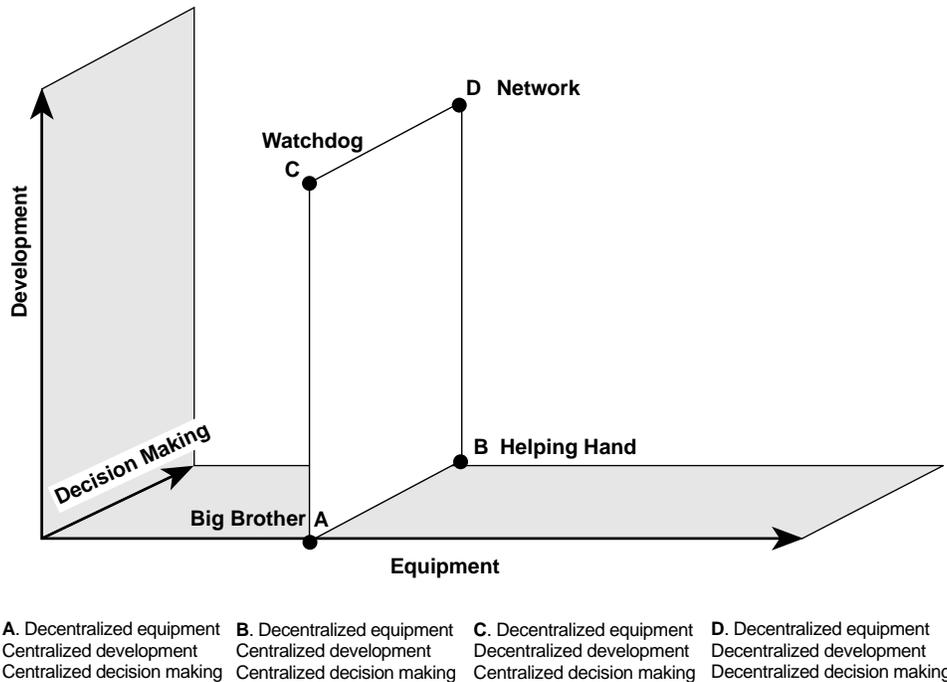
The motor vehicles department has not yet devised a solution. Ultimately, however, it will have to face up to a classic information systems dilemma. Should the department expand its programming staff to make it more responsive to user needs? Or should it accept the users’ growing confidence by loosening control over applications while requiring conformity to data communications standards? Whatever the department decides, the central lesson is clear. Once users in the big brother environment reach a critical mass of restlessness, the status quo is almost impossible to maintain regardless of the users’ lack of power to act on their grievances.

A Helping Hand for Indirect Control

Helping hand policies govern complex applications used by executives who have a deep understanding of their operation’s competitive strengths and weaknesses. The manager of an insurance company’s regional office, for example, will have unsurpassed knowledge of that territory’s risk profiles, regulatory structures, and customer preferences. It’s unlikely that computer professionals at company headquarters could generate programs that meet the needs of this regional office without its close involvement. In this case, unlike the big brother model, the central IS staff writes and maintains programs to meet specifications set by branch offices, divisions, or subsidiaries. And computer users often have direct access to senior management—a political reality that can complicate the CIO’s control over computing resources.

But maintaining control is still the CIO’s basic objective. The IS staff develops software that is based on user specifications, but that also conforms to standards set by headquarters. The IS department selects programming languages, development procedures,

The Four Stages of Decentralized Computing



and operating systems that maximize network connectivity. It uses its authority over departmental purchases of hardware, software, and peripherals to maintain consistent hardware architectures. Finally, the IS office trains and supports the technically unsophisticated end users.

The control tactics in the helping hand model are more indirect than in big brother policies. The CIO and the IS staff play supportive instead of authoritarian roles. Developers work closely with users when writing programs and discourage features that might interfere with the company's broad computing goals. Central IS often acts as a software librarian, maintaining tapes of programs and swapping applications between offices and departments. A financial analysis program developed for one branch office might be offered as a bonus to other branch offices. In this way, software becomes a bartering chip with which the CIO maintains user harmony.

Central IS must also play a mediating role as offices and divisions vie for limited resources. Indeed, scarce programming resources—particularly for maintaining existing applications—are the greatest source of instability in the helping hand environment. Analysts estimate that every dollar a CIO spends developing a program requires an additional \$4 in maintenance costs over its life. So as customized applications proliferate, IS budgets skyrocket.

Consider how changes in federal tax laws affect computer systems at a real estate development firm. The IS group may have to alter dozens of customized financial, marketing, and accounting programs that contain now-obsolete assumptions about tax rates and depreciation schedules. Then it must distribute the changes to possibly thousands of PCs simultaneously—an expensive and time-consuming operation.

While central IS defines development procedures in principle, in practice frustrated users can threaten program consistency in several ways. Consultants hired without authorization can develop chunks of programs or entire applications that don't follow IS guidelines. The availability of low-cost, off-the-shelf software can also give rise to elaborate spreadsheets and data bases outside IS control. If they're incompatible with existing programs or network software, these pockets of information can introduce inaccuracies into corporatewide data and jeopardize the smooth functioning of strategic applications.

A real estate company owned by a major entertainment conglomerate illustrates these risks. The company's local offices relied on customized sales and marketing support systems tied into a central mainframe to track their primary resources: homes, condominiums, and raw land. The CIO wanted to maintain an accurate inventory of all the company's holdings by ensuring smooth data flow between local computers and the company mainframe. To this end,

the IS department required local offices to run the core applications on a brand of personal computer chosen for its ease of connection with the mainframe. Within these hardware and software constraints, IS programmers modified and enhanced the sales and marketing applications based on local specifications.

The backlog of unmet modifications kept growing. Central IS tried to buy time by offering users remote access to other popular programs on the corporate mainframe. But local managers appealed to the company's senior executives and got permission to hire consultants to make the much-delayed software changes. Before long, information began to migrate from the mainframe corporate data base to these new PC programs—not all of which fed smoothly back to the mainframe. The company could no longer maintain an accurate inventory of its properties; and, because users often failed to back up data on their new programs, information on some properties disappeared altogether during hardware failures.

The CIO called for audits to root out troublesome programs and withheld mainframe services from the renegade offices. The political struggle became corporatewide and tempers ran high. There still is no long-term resolution to this situation. A carrot-and-stick approach based on rewards for good behavior and software audits may provide incentives to maintain the integrity of the data base. Otherwise, the CIO may have to allow local offices to take charge of their own software modifications as long as the changes meet strict mainframe-to-micro communication standards.

Watchdogs Can't Watch Everything

The factors that give rise to watchdog policies are often the reverse of those that lead to helping hand principles. Watchdog policies most often govern large, inflexible bureaucracies with clear lines of authority and hierarchy. Military installations almost always operate this way, with teams of engineers writing programs specified by higher ranking officers. In general, end users in the watchdog environment are technically proficient, but lack the business insights to specify the applications they need. As a result, central IS controls all major decisions, from hardware and software purchases to application development priorities. The IS staff standardizes operating systems, programming languages, sometimes even programming style, and the CIO enforces directives through frequent and rigorous audits. It's not

unusual for auditors to visit departments on a weekly or even daily basis to search through computer files for unauthorized programs.

Within these tight constraints, however, independent development teams essentially run their own technical shops. The local staff handles day-to-day operations like setting passwords and correcting hardware and software failures. These local teams develop and distribute software to users. Still, they must requisition the hardware and development tools from central IS and obtain permission to develop applications.

The watchdog environment has the most severe built-in tensions of any of the models I've studied and is therefore the least stable. Despite the elimination of programming backlogs (local developers write and update their own software), purchase orders and requests for permission to develop applications can pile up and slow the programming process. Bright, capable engineers resent development delays even more than their business-minded counterparts in the helping hand framework or the line personnel in the big brother environment. Moreover, strict policing by central IS can be interpreted as interference or harassment—another potential source of frustration. So it's not unusual for talented local programmers to spend a good deal of their time developing and hiding unauthorized software or otherwise skirting IS controls.

The government of a large northern California city recently experienced this problem. Each of the city's 24 departments had two or three programmers responsible for loading, integrating, and debugging software on the department's private branch telephone exchange (PBX). One important application tracked outgoing telephone calls and billed the offices that made them. In turn, each of the PBXs was connected to a citywide network through a communications program controlled by a central computer services group. But the communications software had weaknesses that the central group was slow to correct.

Long accustomed to the hierarchy of city government, local programmers wasted no time in going over the CIO's head to voice their frustration. They simply developed their own communications software, which eventually created inconsistencies in the network. Moreover, the time local programmers devoted to communications software was time they should have spent on other projects. This is a clear example of a CIO losing control of a vital resource—in this case, programmers' time—by neglecting the technical concerns of subordinates who are fully capable of addressing those concerns themselves.

The Challenges of Network Management

There is a structural elegance, based on simplicity and self-direction, that distinguishes the network model from its counterparts. End users in the other frameworks are under the control—direct or indirect, respected or circumvented—of the corporate information staff. As we've seen, such control can lead to struggles over scarce resources. This turmoil not only undermines the CIO's authority but also jeopardizes the computing goals the policies were designed to serve.

Control over day-to-day computing is not an issue for network managers. Technical staffs under the direction of the company's departments or divisions handle every aspect of their information systems. They purchase hardware and peripherals, develop and write applications, load operating systems, and respond to equipment failures. Development backlogs and approval delays are a thing of the past.

Beneath this decentralized simplicity, however, lurk imposing technological challenges. Completing the transition from CIO to network manager requires implementation of the three levels of connectivity—physical, systems, and applications—that link computers and enable them to share information. The computer industry's failure to adopt multivendor standards in so many areas—operating systems, data storage and exchange, mainframe-to-micro communications—immeasurably complicates the network manager's job. But the development of effective, customized solutions to all three levels of network connectivity is a prerequisite for evolution to the network model.

Let's consider each level separately. A single noisy telephone line or a faulty chip at a component installation can cripple a system that depends on exchanging information between factories, or between a mainframe computer at headquarters and microcomputers in branch offices, or between a company and its customers. So a network manager's most basic responsibility is to build and maintain a robust data communications infrastructure—the physical network—that minimizes interruptions and downtime. This means paying closer attention to the selection and maintenance of telecommunications equipment and services like long-distance data transmission, modems, multiplexers, and controllers, and evaluating these technologies on the basis of reliability and accuracy, as well as cost.

Today data transmission at many companies is the responsibility of a director of telecommunications who focuses mostly on selecting telephone compa-

nies, purchasing equipment, and cutting long-distance phone bills. These managers almost never give data networks the attention they deserve. Some forward-looking companies have assigned special responsibility for data communications to a manager who reports directly to the CIO. This is an improvement, but it doesn't go far enough. The CIO must become personally and deeply involved in communications decisions—perhaps to the point of spending several hours a day on the design, maintenance, and expansion of the physical network. Otherwise, it's unlikely that the company will make the investments or devote the staff resources necessary to build a high-performance communications infrastructure.

Systems-level connectivity is the second major challenge. A smoothly functioning network should allow the corporate information staff to monitor and address the performance of every computer on a particular network. For example, the network manager's staff should be able to assign or modify passwords quickly to block unauthorized access to new applications on a network that contains sensitive data. Likewise, the IS staff should be able to distribute new applications to whatever computers in the network need it, regardless of their operating systems or locations.

Several major vendors, including IBM, Digital Equipment, and AT&T, as well as a few independent software companies, offer systems-level tools that perform some connectivity functions. But these third-party solutions have grave shortcomings. Most of them provide security and manage remote terminals, but they generally don't integrate these capabilities under a consistent, easy-to-use interface. And as with the physical network, the absence of industry standards means that no single product can address more than a fraction of the hardware and communications protocols scattered throughout the organization. Effective systems-level connectivity may require a large investment of staff time to develop customized solutions to vexing technological problems. These investments won't be approved unless the senior executive in charge of information systems makes them a priority.

Building a reliable communications infrastructure whose performance can be monitored by the central IS staff is valuable only if the hundreds of applications running on the network can freely exchange information. True applications connectivity is a formidable challenge—even for a network composed of nearly identical computers and software—because applications are constantly changing. And modifications can interfere with an application's ability to feed data into other software.

The technical obstacles to relationship banking, a highly prized application in the competitive financial services industry, are a case in point. A single marketing representative analyzes, evaluates, and recommends changes in a customer's entire portfolio. To do this, the representative must be able to summon on command a comprehensive profile of the customer's activities in all the bank's services. That requires pulling information from the separate computer systems that manage, for example, home mortgages, certificates of deposit, checking accounts, and car loans. But each system's evolution can affect the entire network. Something as simple as reversing the order in which checking-account operators record first and last names threatens the integrity of a network unable to adjust to that change.

At this level, maintaining applications connectivity requires strict procedures to guarantee that departments notify the network manager of all software modifications. Under this system, the role of the central IS staff evolves from approving or developing applications to adjusting the interface between modified applications so they can continue to exchange data.

Applications connectivity becomes even harder when the network is supporting incompatible computer systems. For example, a bank that acquires a competitor and wants to offer relationship banking may have to gather data from two vastly different computing environments. Likewise, a network that

links a company with its suppliers almost invariably must support several operating systems. Perhaps a decade from now the computer industry will adopt connectivity standards for easier exchange of data between computers built around different architectures. Until then, however, it's up to network managers to devise customized solutions.

These solutions require the development of what I call "application filters" that translate between different systems on the network, hide each computer's particular characteristics, and adjust for application changes. Several organizations, including Covenant Insurance, the American Red Cross, and Maryland National Bank, have implemented such filters in large strategic applications. Their experiences to date have been quite encouraging. In the near future, expert systems may be able to detect and adjust automatically for application changes affecting the rest of the network. Prototypes of such expert systems have been developed, although the technology required for full-scale implementation is still several years away.

The transition from CIO to network manager will be neither immediate nor easy. But it must be made if senior executives want lower costs and enhanced strategic advantage from their computer systems. The CIO who faces the challenges of network management head-on can overcome the perils of decentralized computing and tap its vast potential.