The Convergence of Telecommunications and Computing: What Are the Implications Today?

DAVID G. MESSERSCHMITT, FELLOW, IEEE

Invited Paper

As has been widely recognized for some time, the computing and telecommunications technologies are converging. This has meant different things at different times. In this review paper, we describe the current state of convergence, and speculate about what it may mean in coming years. In particular, we argue that as a result of the horizontal integration of all media (voice, audio, video, animation, data) in a common network and terminal infrastructure, telecommunications and networked-computing applications are no longer distinguishable. Considering that the old terminology is no longer meaningful, we attempt to codify networked applications in accordance with their functionality and immediacy. As application functionality is increasingly defined in software, with commensurate cost-effective programmable terminals and means for distribution of applications over the network itself, we argue that user-to-user applications will be greatly impacted, moving into the rapid-innovation regime that has characterized user-to-information-server applications in the recent past. Finally, we identify a number of areas where different technical approaches and design philosophies have characterized telecommunications and computing, and discuss how these technical approaches are merging and identify areas of needed research. We do not address complementary forms of convergence at the application or industrial level, such as convergence of the information and content-provider industries, but rather restrict attention to the infrastructure and technology.

I. Introduction

The convergence of telecommunications and computing has been noted and commented on for some time. However, there is a much richer interrelationship at present than at any time in the past. In fact, we will argue that the very terms "telecommunications" and "computing" are losing their relevance as separate identities, and also that these fields will become virtually indistinguishable in the relatively near future. This paper builds on our brief summary of technological trends in the industry in [1].

Why should we care? The convergence has, and will continue to have, a profound impact on technology, industry, and the larger society. The traditional fields of telecommunications and computing have already been irreparably changed by the other, and, as we argue below, will be even more substantially recast in the future. We argue that much more profound changes are forthcoming, changes no less weighty than the rapid disintegration of the vertically integrated industrial model (from silicon to applications). Finally, while computing in the absence of communications has led to new applications and made substantive changes to leisure and work life, computing in conjunction with communications will have a profoundly greater impact on society. This is because communications is at the heart of what makes a society and a civilization, and the convergence with computing will revolutionize the nature of that communications [2].

Manuscript received April 1, 1996; revised May 27, 1996.

The author is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA.

Publisher Item Identifier S 0018-9219(96)06187-7.

II. What Is Telecommunications and What Is Computing?

The term *telecommunications* is derived from "tele," meaning at a distance, and "communications," meaning exchanging of information. The dictionary definition of telecommunications is "communication at a distance (as by telephone)" [3], and the term is most commonly applied to the telephone, but also applications like video conferencing. At its origin, the *computer* was envisioned as a machine to perform massive numerical calculations. Indeed, this is the origin of the term "computer" as "something that can compute." Later, with the development of large peripheral storage devices, the computer became a repository of large amounts of data that could be modified, manipulated, and queried. This is reflected in the current dictionary definition of the computer, as "a programmable electronic device that can store, retrieve, and process data" [3]. These classical views of telecommunications and computing are well differentiated with respect to applications.

Recently, the infrastructure and applications for these technologies have become seriously blurred. In both the network [7] (embodied in the Internet [4] and asynchronous transfer mode (ATM) [8]) and the desktop computer [6], data has become integrated with continuous media (audio

0018–9219/96$05.00 © 1996 IEEE

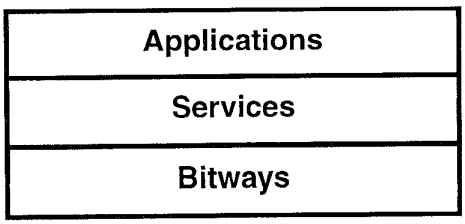

Fig. 1. A three-level model for an information network and its services and applications.

and video), enabling so-called *multimedia* applications [5], [9]. Applications are becoming blurred as well. Accessing bank records using a dual-tone multifrequency (DTMF) telephone and voice response unit, or with a networked computer over a computer network, differ as to medium but not basic functionality. Thus the classical terminology of telecommunications and computing is no longer as useful, and possibly even delusory. In light of this, it is appropriate to define a more transparent classification of networked applications that is media-blind and focuses on the functionality provided the user.

A. A Three-Level Architecture

As an aid to understanding, we adopt the three-level model of Fig. 1, similar to that proposed in [10], [11].[1] We define an *application* as a collection of functionality that provides value to a *user* (a person). In this paper, we are concerned with *networked* applications, implying that they are distributed across a distributed telecommunications and computing environment. Examples of networked applications are electronic mail (e-mail), telephony, database access, file transfer, World Wide Web (WWW) browsing, and video conferencing. A *service* is defined as functionality of a generic or supportive nature, provided as a part of a computing and telecommunications infrastructure, that is available for use in building all applications. Examples of services would be audio or video transport, file-system management, printing, electronic payment mechanisms, encryption and key distribution, and reliable data delivery. *Bitways* are network mechanisms for transporting bits from one location to another. Examples of bitways with sufficient flexibility for integrated multimedia applications are ATM [8] or internets interfaced with the Internet Protocol (IP) [14].

We can build a taxonomy of networked applications into four categories as shown in Table 1. Two categories relate to the functionality [12].

- *User-to-user applications,* in which two (or more) users each participate in some shared functionality.
- *User-to-information-server applications,* in which a user (or sometimes two or more users) interacts with a remote system to access, receive, or interact with information stored on that system.

[1] In the traditional terminology of network operators, users have accessed "services" (i.e., telephone, call waiting, voice mail), which computer users have accessed "applications" (i.e., spreadsheets and word processors). The origin of this distinction is undoubtedly the involvement of a "service provider" in telecommunications, largely absent in the modern computer industry.

Table 1 A Taxonomy of Networked Applications with Examples

	Immediate	*Deferred*
User-to-information -server	Video on demand WWW browsing	File transfer
User-to-user	Telephony Video conferencing	Electronic mail Voicemail

Each user in a networked application interacts with a local terminal, which communicates in turn with remote computers or terminals across the network.

We also separate networked applications into two classes with respect to the temporal relationship in the interaction of the user with a server or with another user.

- *Immediate,* meaning a user is interacting with a server or another user in real-time, typically with requirements on the maximum latency or delay.
- *Deferred,* meaning a user is interacting with another user or a server in a manner that implies no fixed temporal relationship and for which the delay is typically not critical.

One useful test is whether the user concentrates solely on the application (immediate) or typically moves to another task in the middle of an interaction (deferred). Immediate applications would sometimes be called *synchronous* [19] or *real-time* [12], and deferred would sometimes be termed *asynchronous* [19] or *messaging* [12].

B. Two Architectures for Networked Applications

Networked applications are physically realized by *terminal nodes* (or just terminals) interconnected by *bitways.* Functionally there are two basic architectures available for networked services, as illustrated in Fig. 2.

- *Peer-to-peer architecture* [50], in which two (or more) *peer terminals,* each associated with a local user, communicate over a bitway to provide a user-to-user networked application. The networked communications component between peers is often symmetrical (in terms of both functionality and bitway resources).
- *Client-server architecture* [21], [22], in which a *client terminal* associated with a user communicates over the bitway with a *server computer,* which is not associated directly with a user, but rather realizes an information-server function. The functionality is often asymmetric, with the server embodying the primary functionality or database access and the client terminal focusing on the user interface. (As will be described later, this partitioning is rapidly shifting.) The communications component is also often asymmetric, with the server-to-client direction typically requiring much higher bandwidth.

Often the peer or client terminal functions will be realized in software in a desktop computer, or they may be dedicated-function terminals (like a telephone or video conference set). For simplicity, we will refer to "peers," "clients," and "servers," without the associated terms "ter-

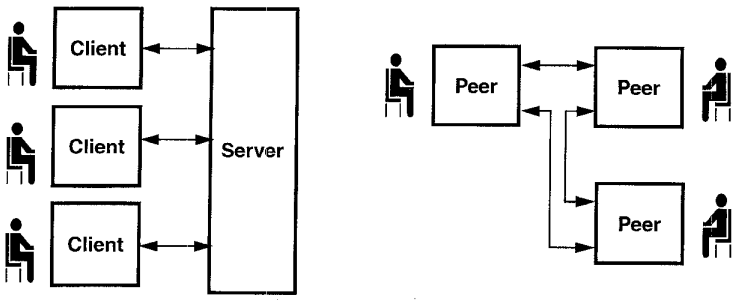

Fig. 2. A comparison of client-server and peer-to-peer architectures for networked applications.

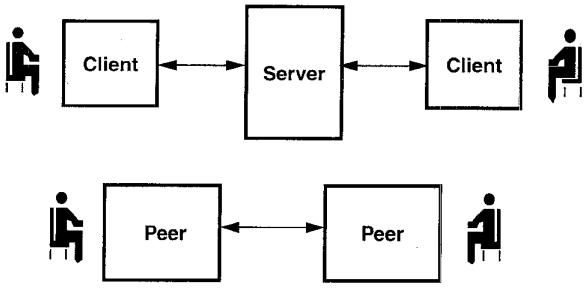

Fig. 3. A user-to-user application can be realized by either the client-server or the peer-to-peer architecture.

minal" or "computer." Note that the terminal versus bitway is (primarily) a *physical* partitioning of functionality between a terminal at the edges of the bitway, and the bitway itself. The three-level architecture of Fig. 1 is a *logical* separation of functionality, where application functionality will typically physically reside in the terminals, and services functionality may reside in the terminals or somewhere within the bitway.

A user-to-information-server application is always realized with the client-server architecture. As shown in Fig. 2, many clients will typically access a single server, which provides functionally separated but time-shared services to the clients. On the other hand, a user-to-user application can be realized in either the peer-to-peer or client-server architectures, as illustrated in Fig. 3 (for two users). In the client-server architecture, the two clients are communicating through the server, which may be realizing additional application or control functionality.

The client-server architecture is particularly appropriate for deferred user-to-user applications, since the server provides a convenient point for the necessary buffering with guaranteed availability regardless of the state of another peer. One such example in Table 1 is voice mail, where the originating user (by way of his or her client telephone) forward the voice message to a voicemail server, where it is stored to be accessed later by the destination user (by way of his or her client telephone). From the perspective of the two users, the application is user-to-user and deferred, which is how it is listed in the table. However, the interaction of each user with the voicemail server (at different times) is user-to-information-server and immediate. This example illustrates several points. For one, the server adds functionality not available in the client telephone, such as menu-based control primitives, and is available even when the destination user is unavailable. For another, a terminal can serve as either a peer or a client in the context of different services. The telephone is a client to a voice mail server, but is more commonly a peer to another telephone in a (user-to-user and immediate) telephony application.

Although clients and peers serve a similar user-interface functionality, there are some basic differences. Typically many clients will connect to a single server, whereas a peer must be prepared to connect to any other peer. In some applications, like multiway video conferencing, a peer may be connected to more than one other peer simultaneously. To establish a new instance of an application, a server must always be prepared to respond to an establishment request from a client (but does not originate requests), whereas a client may originate establishment request (but is not prepared to respond to such requests). A peer must be able to either originate or respond to establishment requests, and in this sense is a hybrid between a client and a server. A client can rely on the server for some functionality, whereas a peer must be self-contained. The biggest differences are in scalability to large numbers of users, interactive delay, and interoperability (see Section IV-B).

III. A Short History of Convergence

We are arriving at a network and terminal/computer environment that seamlessly supports user-to-user and user-to-information-server applications incorporating multimedia datatypes, one where there are no remaining vestiges of telecommunications and computing (in their classical definitions at least). It is helpful to list some of the key developments that have led us to this point, as well as ongoing developments that will have a large identifiable impact. To this end, we identify a number of distinct stages of development on the road to convergence, which we list in approximate chronological order.

A. Common Technology

The genesis of computer technology was basic technology arising from telephony; namely, the relays used in telephone switches. Subsequently, both computers and telecommunications exploited underlying advances in electronics and optoelectronics (the latter in the case of com-

munications). More to the point, functional as opposed to technological convergence occurred with the advent of stored-program control for telephony switches and the development of digital representation of telephony signals (through quantization and analog-to-digital conversion in the so-called *pulse-code modulation*) in the 1950's.

These two developments presaged two profound shifts in telecommunications. First, computers became common as control and signaling points in telephony networks, enabling more functionally complex telecommunications services, and second, digital representations of audio, image, and video signals allowed them to be stored and manipulated by standard computational hardware. While the first factor has resulted in a major shift toward the automation of the telephone networks, it has had relatively little influence on the computer industry. The second development has had far wider implications outside communications, such as compact digital audio, digital high-definition television (HDTV), and the extremely flexible manipulation of signals by standard or custom digital hardware (the latter called *digital signal processing*). Only today is this technology joining the computing mainstream, as enabled by the increasing performance of desktop computers.

B. Networked Computers

Two seminal developments were the desktop computer [25], as well as networks devoted specifically to the communication among them, first in the "local area network" [26] and later the "wide area network" (two early examples of which are synchronous network architecture (SNA) and ARPANET [27], the latter having evolved into the Internet). Early examples of applications enabled specifically by the networked computer include e-mail, file transfer, concurrent databases, and recently the WWW. The stand-alone desktop computer had previously enabled its own set of high-value applications, such as desktop publishing, spreadsheets, and other personal-productivity applications. The networked computer provides a ready large-scale market for new applications, thereby reducing the barriers to entry for new applications developers.

Computer networking, like the control computer before it, was widely adopted in the telecommunications industry as the basis of signaling and control. This signaling function was originally realized in-band on the same voice channel, but was replaced by a signaling computer network called common-channel interoffice signaling (CCIS) [28]. CCIS enabled the advance from simple circuit-connection functions to much more advanced features (like caller identification), and ultimately will provide terminal-to-terminal signaling capabilities (a basis for dynamic deployment, see Section III-G).

Up to this point, there remained an infrastructure for computing that emphasized data-oriented media (graphics, animation), and a relatively separate telecommunications infrastructure that focused on continuous-media signals (voice and video). These converged in a relatively superficial way, at the physical and link layers, where telephone and videoconferencing and computer networks shared a common technology base for the physical layer transport of bits across geographical distances. The telecommunications industry made extensive use of computer and software technologies in the implementation of the configuration and control of the network. The computer industry made use of the telecommunications infrastructure to network computers, which enable networked applications. However, it is fair to say that the disciplines remained intellectually separate, sharing common hardware and communications media but pursuing distinct agendas and possessing distinct cultures.

C. Programmability and Adaptability

There are a number of inventions embodied in computing, but arguably the most important is *programmability*. The expanding importance of programmability flows from extraordinary advances in the cost/performance of the underlying electronics and communications technologies. In the context of any single application (like control, voice, audio, video, etc.), the performance requirements in relationship to the capabilities of the underlying technology passes through three stages.

- Initially, the application is very expensive to implement, and cost effectiveness dictates a customized hardware design. In this stage, efficiency (in metrics like processing power, bandwidth, etc.) is critical to cost-effectiveness, and hence commercial exploitation.
- Next, programmable software-defined implementations become feasible, and eventually cost effective. At this point, efficiency remains a dominant consideration, but the lower design cost and lower time to market afforded by a software definition can often overcome the manufacturing-cost penalties of general-purpose hardware.
- Finally, technology advances far enough that software-defined implementations become the norm. At this point, the greater efficiency of a custom hardware implementation is definitively overcome by its lower volume of manufacture, greater design costs (including especially the cost of tracking advancing technology), and greater time to market.

The final stage—a software-defined solution—has an important implication; namely, the basic functionality need not be included or defined at the time of manufacture, but rather can be modified and extended later. This property—that the basic functionality can change and advance over time—is the key to the triumph, for example, of the personal computer over the stand-alone word processor.

The advances in underlying technology are such that software-defined implementations are cost effective for audio as well as virtually all data media, and as time passes will become viable as well for video at increasing temporal and spatial resolution. Thus the programmable implementation can be expected to spread to all corners of the computer and communications world (although there

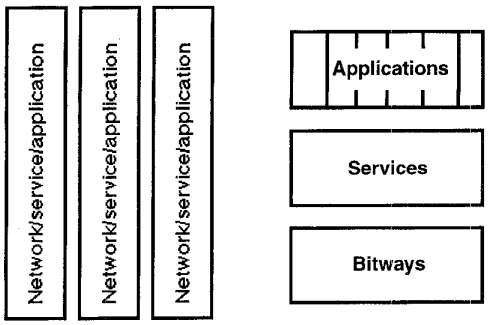

Fig. 4. Two architectural models for provisioning networked applications: vertical and horizontal integration.

will always remain high-performance functions that are implemented directly in hardware).

The modern trend is toward *adaptability,* a capability that (usually) builds on programmability and adds the capability to adjust to the environment. For example, in a heterogeneous environment it is helpful for each element to adapt to the capabilities of other system elements (bandwidth, processing, resolution, etc.).

D. Horizontal Integration

There are two architectural models for provisioning networked applications, as illustrated in Fig. 4. In the most extreme form of *vertical integration,* a dedicated infrastructure is used to realize each application. The premier example is the public telephone network, which was originally designed and deployed specifically for voice telephony. In contrast, the *horizontal integration* model is characterized by the following.[2]

- One or more integrated bitways that transport integrated data and stream media like audio and video with configurable quality-of-service (QoS) parameters (see Section IV-A).
- A set of services, such as middleware services (directory, electronic funds transfer, privacy key management, etc.) and media services (audio, video, etc.) that are made available to all applications.
- A diverse set of applications made available to the user.

A key advantage of the horizontal model is that it allows the integration of different media within each application, as well as different applications within the bitway. (For this reason, this is often called an *integrated-services* network in the telecommunications industry.)

Another useful distinction among networks is whether or not they are *content-aware,* and whether or not they are *application-aware* [32]. Vertically integrated networks are frequently application-aware, meaning they are cognizant of the applications they are carrying (e.g., videoconferencing versus file transfer), whereas horizontal bitways are often application-blind (e.g., the current Internet). Vertically integrated networks are frequently even content-aware (e.g., a video-on-demand network that is cognizant of what movie is requested). A primary source of the rapid innovation in the Internet is the clean separation it forms between bitway, service, and application, making the service and application layers application-blind and thus allowing new applications to be constructed without modification to the bitway.

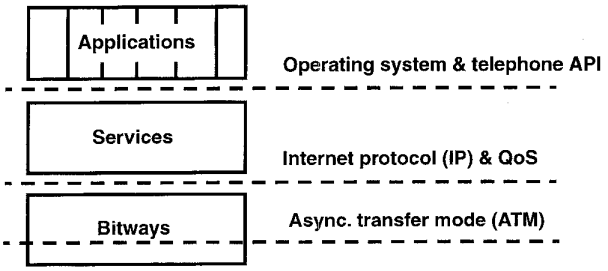

Fig. 5. Important examples of open horizontal interfaces.

An important feature of horizontal integration is the *open interface,* which has several properties: It has a freely available specification, wide acceptance, and allows a diversity of implementations that are separated from the specification. Another desirable property is the ability to add new or closed functionality.[3] Open interfaces enforce modularity and thus allow a diversity of implementations and approaches to coexist and evolve on both sides of the interface [10], [49]. Some of the most important open horizontal interfaces in the computer industry are illustrated in Fig. 5. ATM is a protocol designed specifically to accommodate a diverse mix of traffic types [8]. The Internet Protocol [4] is an open standard for interconnecting bitways below it, where those bitways may incorporate a diverse set of technologies (including ATM). IP also allows for a diverse set of media types and applications to reside above it. Another critical interface is the operating system (OS) application program interface, which allows a diverse set of applications to coexist on the same bitways and services infrastructure, while hiding as much as feasible the details from that infrastructure. Horizontal interfaces also exist for the control and signaling (e.g., control of telephony network features from a desktop computer application in the telephone application-program interface (TAPI), which supports computer-telephony integration [33]–[35]).

One critically important consideration is complexity management [59]. One purpose of open horizontal interfaces is to contribute to the modularity by the separation or independence of the definition of the application from the "execution engine" upon which it runs, which we call *platform independence.* Increasingly, applications can be simultaneously developed for multiple target platforms, by generating distinct platform representations from a common functional description, based on appropriate software toolsets. An even more powerful concept that is currently

[2] We use the terms vertical and horizontal integration as an *architectural* model. These terms are used by economists as well, but in the different context of the partitioning of organization and ownership [29]–[31]. Architecture and ownership tend to be coupled, but not completely as illustrated by the Internet where the ownership of a horizontal infrastructure is highly fragmented.

[3] We define *closed* functionality as not published or extensible by other parties. *Proprietary* functionality may be published and extensible, but is subject to intellectual property protection.

emerging is *middleware,* which is a horizontal layer residing on top of a set of networked computers, providing a set of distributed services with standard programming interfaces and communication protocols even though the underlying hosts and OS may be heterogeneous [36].

Open horizontal interfaces are not completely successful at isolating horizontal functional layers. For example, one open interface is dependent on the suite of primitive functions offered by a lower interface, a phenomenon called *protocol dependence.* D. Clark has defined a special type of open interface called a *spanning layer*, which adds the characteristic that the extent of its adoption is nearly ubiquitous [37]. A specific spanning layer called the "open data network bearer service" is proposed in [10]. Spanning layers are particularly useful because higher interfaces can presume their existence and the services they provide, thus effectively isolating the design of the horizontal layers above and below.[4]

The computer industry is well along in the evolution to horizontal integration. The networked desktop computer resulted in the division of the industry into distinct horizontal segments (hardware, network, OS, and application). Today, we are in the process of integrating nondata media such as audio and video into this same environment, supported at both the bitway level [large-area networks (LAN's) and the Internet] and on the desktop. The telecommunications industry was once vertically integrated, with a focus on provisioning a single application with a dedicated network, such as voice telephony, or video conferencing, or cable television. Today this industry is also moving toward architectural horizontal integration at the bitway level with ATM bitways that flexibly mix different media; however, it remains largely vertically integrated at the services and applications layers, as bitway providers aspire to valued-added applications such as video on demand and differentiated terminals such as "set-top boxes."

We hypothesize that powerful economic and technological forces are driving us toward horizontal integration. Advances in technology have already resulted in the integration of different media in both the bitway (such as ATM or the Internet) and in the terminals (such as desktop computers). This level of horizontal integration offers the service provider substantial administrative benefits, relative to the alternatives of separate or overlay bitways, and adds value to the user, since different media can easily be incorporated into *multimedia* applications.

The separation of the applications from bitways and services best serves the user by encouraging a diversity of applications, including many defined for specialized as well as widely popular purposes. Vertical integration discourages this diversity because a dedicated infrastructure demands a large market, and because users do not want to deal with multiple providers. Horizontal integration lowers the barriers to entry for application developers since most of the infrastructure (bitways and services and even programmable terminals) is already available. Applications can be defined in software and coexist in the same programmable terminals with other applications, reducing the distribution cost and the incremental cost of a new application. Finally, it is unlikely that a single company can accumulate the range of expertise required to provide the best solutions across such a wide range of media and technologies.

Open interfaces offer vendors a large and immediate market for new applications. The resulting diversity of applications increases the utility of the open interface to the user. This positive reinforcement leads eventually to a dominant open interface, to be displaced only by a new interface that offers significant functional or performance advantages. The same inherent value of application diversity does not apply to bitways and services. They are generic and widely applicable to different applications, difficult to differentiate except in terms of cost and performance, and are capital intensive and benefit from economies of scale.

The computer industry is far along in the evolution to horizontal integration. The desktop computer freed the user of the constraints of the computer center bureaucracy and lowered the barriers to entry of application developers, which in turn offered greater value to the user. Our speculation is that the telecommunications industry will be pushed by market forces in the same direction, even though many companies would doubtless prefer vertical integration and closed solutions.

E. Untethered, Nomadic, and Mobile Services

Here we use the term "untethered" to refer to *wireless* access to a bitway, "nomadic" to refer to *geographic flexibility* in accessing a bitway, and "mobile" to refer to bitway access while the user is *in motion.*[5] In a sense, these three concepts build upon one another, but not strictly. While mobile services are necessarily untethered, nomadic services are not. Mobile services are by definition nomadic. These three concepts lead to a different but overlapping set of challenging technological issues.

Nomadic telephony has long been available in the form of extension and pay telephones. (Perhaps because there is no computing "service provider," an analogous infrastructure is yet to appear in networked computing.) Untethered telephony has been offered for some time by the cordless phone [41], and later, mobile telephony arose in the extraordinarily successful deployment of cellular telephone systems [38]–[40], [42]–[44]. Computing has remained fixed-location for some time, although one might view networked client-server computing as nomadic in the sense of making an application executed on a server available to a nomadic user, should they be able to find a bitway access point. The laptop computer has supported the nomadic and even mobile computer user (although alas not the *networked* computer user, except to the extent such networking can be accomplished over the telephone). Recently, there is beginning to develop an infrastructure supporting nomadic and mobile networked computers [45]–[48].

[4] By this we mean a functional isolation. There will always remain performance and coordination issues (see Sections IV-A and IV-C).

[5] The use of these terms has been inconsistent in the literature.

Nomadic and mobile services and applications have been so successful because a fixed-location constraint is a mismatch to the roving nature of human activity (indeed, even *within* the office or residence). To the extent services and applications can be provisioned in a cost-effective mobile (or even untethered) fashion, experience has shown that users will choose this option. Thus it is clear that nomadic and mobile telecommunications and computing are extremely important for the future, while offering many serious technological challenges.

Nomadicity and mobility provide another point of convergence: the issues raised by mobile telecommunications and by mobile networked computing are similar. Both require the dynamic migration of resources (connections, internal state, processes, reserved memory and bandwidth, etc.), and both raise serious issues related to QoS (uninterrupted service, inability to reserve resources in advance without regard to location). Since telecommunications has addressed these difficult issues for some time, there is an excellent opportunity for cross-fertilization to nomadic and mobile computing.

F. Network Deployment

Beyond a couple of applications of universal interest—voice telephony and video conferencing—user-to-user applications are much fewer in number than user-to-information-server applications (although nevertheless very successful). These universal user-to-user applications have previously used the dedicated telephone network but are migrating to the Internet, for example with CU-SeeMe [18]. A less familiar example is groupware [9] and collaborative computing [24], where two or more users can performed shared functions on a document or database, as in a collaborative design project [55]–[57]. There are also less familiar applications, such as telepresence [51] and telemanipulation [52], which are important in military, outer space, and dangerous environments, but potentially also of importance in medicine [16], [17]. In contrast, there are a large and expanding number of user-to-information-server applications, such as the WWW [88].

Why are user-to-user applications so few in number? This could be inherent, or perhaps this class of applications has possibly been overlooked by the application software industry. Another is that the human factor aspects are not sufficiently developed. Another is the requirement for a cumbersome and time consuming standardization process[6] if two or more vendors are to achieve interoperability in a given application. In our view, none of these reasons is as important as a fundamental obstacle to the commercial exploitation of user-to-user applications that economists call *direct network externality* [60]–[62]. This property of networked applications, which distinguishes them from most other market goods, is that the value of an application

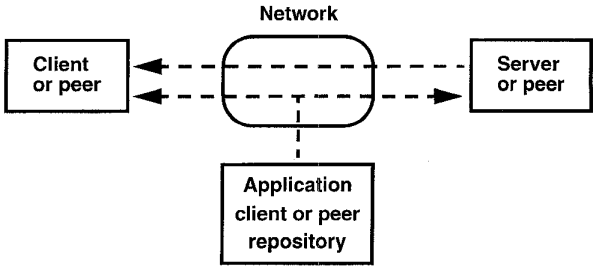

Fig. 6. Illustration of the network distribution of application client or peer software descriptions over the network.

to a particular user grows with the number of other users that have an interoperable application available;[7] that is, the community of interest willing and able to participate in that application. In contrast, early adopters derive very little value, which is an economic barrier to a vendor attempting to establish such an application. (Who is the first user to buy a video conferencing application if there are no other users with whom to conference?).

User-to-user applications display strong network externalities. In contrast, user-to-information-server applications have an weaker network externality that makes them much easier to establish in the marketplace. This is because, once an information server is made available on the network, the first user derives the same value as later users.[8]

Network externality can be partly overcome by a good mechanism for distribution of application software. If a user-to-user application can be distributed to a large number of users virtually simultaneously, interoperability and a community of available users is guaranteed, even for early adopters. For software-defined applications, this is technically feasible, since an application can be distributed over the network itself. As shown in Fig. 6, the user obtains a binary executable for a client or peer application over the network itself as a prelude to participating in the application. Developers of user-to-information-server applications like WWW browsers [89], document viewers [58], and audio and video players are distributing new versions of those applications over the network; in fact, they are bypassing many externality issues by distributing them for free (hoping to derive revenue from the interoperable server software), thus establishing a community of interest quickly. By bypassing traditional slow distribution channels, the velocity of innovation in these applications has been increased dramatically. Since user-to-user applications have a much stronger network externality, network distribution has the potential to make a much bigger impact on this class of applications.

Network distribution in the Internet remains cumbersome, however, since a user has to anticipate the need for an application and execute the relatively sophisticated and manual "network file transfer." Other problems are multiple

[6] There are some notable successes in standardization, such as the V.34 voiceband data modem [53] and MPEG [54] where the standardization process probably speeded technical advance by involving a technical community of many companies. These examples are probably more accurately described as cooperative design combined with standardization.

[7] The technical definition of a *positive consumption externality* is "the value of a unit of the good increases with the number of units sold" [60].

[8] There is some externality coming from the larger market resulting from an increasing number of clients, which in turn stimulates more activity from application developers and content providers.

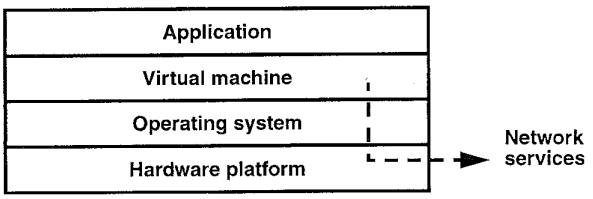

Fig. 7. The virtual machine open horizontal interface, which provides an OS-independent platform for portable applications.

microprocessor instruction sets and OS's, and security problems associated with downloading binary executables from untrusted sources. Recently, a technical advance with great promise has appeared that addresses these problems, associated with a new horizontal open interface called the *virtual machine.*

G. Dynamic Deployment and Transportable Computation

The virtual machine is illustrated in Fig. 7. A layer of software is inserted between the OS and the application that separates the application from the specifics of the OS and hardware platform. The virtual machine open interface defines a general instruction set, as well as application-program interfaces (API's) to resources like network services, all in an OS-independent way. It supports *transportable computation,* meaning that even though the program representing application functionality is stored in one node (typically peer or server), that program can be transported to and executed on another node (typically peer or client).

The virtual machine is implemented as an interpreter on various computing platforms or OS's. Thus applications can be written in a high-level language and compiled into the virtual machine instruction set, and subsequently run on any computing platform or terminal with an implementation of the virtual machine interpreter. The process is illustrated in more detail in Fig. 8, for the special case of the transport of computation from a server to a client. The client portion of an application is written in a high-level language that is compiled into a program composed of instructions for the virtual machine. This executable program is stored permanently in the repository on the server, and on demand is distributed over the network to the client. At the client, the program is run in an interpreter for the virtual machine. Thus far, this approach is embodied in several high-level application-description languages: Safe-Tcl [64], [65], Telescript [66], [67], and Java [68]–[70]. Drawing from Java terminology, in the sequel we will use the term *applet* to describe a transportable program. Built into the interpreter are various safeguards against malicious or misbehaving programs (that are inevitable when allowing programs to be loaded from untrusted sources). The program executes more slowly on an interpreter, as compared to a native applications, although "just-in-time compilers" are expected that will compile the virtual machine language into native code on the target processor as it arrives.

Transportable computation offers four important advantages:

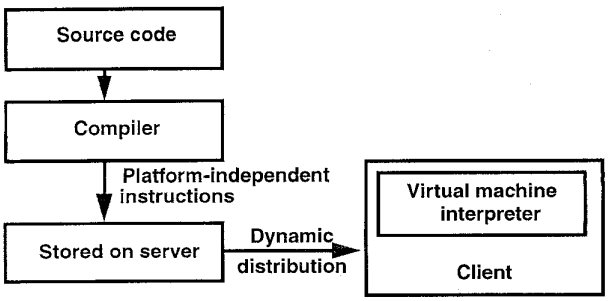

Fig. 8. The process of developing and dynamically distributing application client virtual machine code in a client-server architecture.

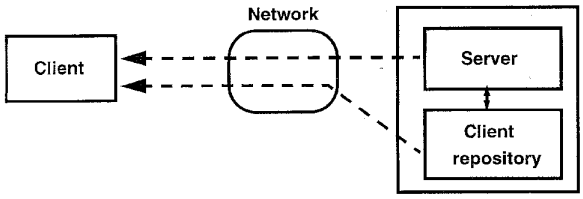

Fig. 9. A repository of client applets can reside in the server, waiting to be dynamically deployed over the network to the client on demand.

- *Scalability* (see Section IV-B). It allows both memory (transient and persistent) and computation to be located on whatever terminal or computer is most advantageous. For example, in a client-server application it allows computation to be shifted from the server to its clients, thus avoiding overload of the server.
- *Latency* (see Section IV-A2). Executing the program in local peer or client as opposed to a remote host eliminates interactive latencies due to network transport delay.
- *Interoperability* (see Section IV-F). If the programs associated with a distributed application originate from a common source, it can be assured that they are *interoperable,* meaning that they properly coordinate their operations. The conventional approach to interoperability, standardization (including *de facto* standardization), is by comparison cumbersome and time consuming. This is doubtless the most important advantage.
- *Locality of data access.* A transportable program can access and modify data stored on any computer to which it can be transported.[9]

Transportable computation facilitates the *dynamic network deployment* of applications. That is, a distributed application can be copied over the network during establishment, transparently and invisibly to the user, with guaranteed interoperability, as illustrated in Fig. 9 for a client-server architecture. The client application code is stored in a repository in the server, to be loaded dynamically across the network into the client when the user invokes

[9]This has serious security implications, and for that reason may well be prohibited by the virtual machine.

that application. In contrast to the network distribution in Fig. 6, the manual intervention of the user is not required: the deployment occurs transparently to the user whenever the corresponding application is invoked at the server. Other problems with network deployment like supporting different OS's (or versions of those OS's) are avoided, because the programs in the client repository are written to run on the "universal" virtual machine.[10]

Thus far, dynamic deployment has been applied primarily to user-to-information-server applications (adding functionality to a WWW browser in particular). As shown in Fig. 10, it has equal applicability to user-to-user applications, in this case in a peer-to-peer architecture. The peer that initiates the application contains a repository of programs for the other peer, which are dynamically loaded at establishment of an application. (Alternatively, a repository of programs for both peers could be obtained from a central server.) We have demonstrated this using Tcl as the application-description language [71] and more recently to peers consisting of Java-enabled WWW browsers [72]. Dynamic deployment will have a far greater impact on peer-to-peer applications than client-server applications, because it neatly avoids network externality obstacles. When a user purchases a user-to-user application for which one peer program is targeted at the virtual machine, he or she can readily participate in that application with any other user with the appropriate virtual machine interpreter (as opposed to the application itself). One can easily imagine a "toolbox" for defining collaborative user-to-user applications that includes standard components like "whiteboard," "shared editor," etc. It is only necessary for one user to possesses the toolbox, and temporary licenses for co-users can be created as necessary. The application can dynamically add new components, even contributed by different participants. The ability to expand the range of both types of applications will be a powerful force for the proliferation of virtual machine interpreters, and hence the economic obstacle of network externality should be greatly diminished.

Dynamic deployment benefits from (and may even require) broadband networking, since application executables will sometimes be large. This will be an important driver for broadband access to the bitway, just as low-latency downloading of executables is a primary driver for broadband local-area bitways.[11]

H. Intelligent Agents

Dynamic deployment does not exploit the full power of transportable computation, which is embodied in the more general concept of an *intelligent agent.* An intelligent agent is a transportable program that includes four attributes and capabilities [73], [74]:

- *Autonomy.* It contains all information necessary for its execution.
- *Social ability.* It can interact with other agents or its environment.
- *Reactivity.* It can exert actions based on attributes of its environment.
- *Proactive.* It can initiate actions by itself.

The capabilities of the intelligent agent open up a number of possibilities. Intelligent agent technology originated in artificial intelligence, where one can imagine sophisticated human-like qualities such as adaptation to the environment and higher-level cognitive functions. Here, we can conceptualize more mundane applications that provide useful generalizations of user-driven information retrieval [75] or even as basic as e-mail [76]. In this application domain, agents can act as "itinerant assistants" that are not restricted to particular servers, but cruise the network gathering or disseminating information. Such "itinerant agents" represent a different dimension of mobility; rather than the user being mobile, the user is represented by a mobile agent.

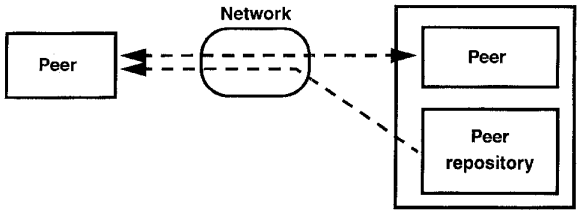

Fig. 10. Application peer functionality can also be defined by an applet, dynamically loaded from the other peer to dynamically invoke interoperable application peers.

I. Complete Convergence: The Logical Conclusion

As networked applications become more sophisticated, especially as enabled by the interoperability and scalability benefits of dynamic deployment, we expect the application types and architectural models to become increasingly mixed. Typical collaborative applications will combine user-to-user and user-to-information server functionality, as in a collaborative design involving two or more users and a common information server (storing the design being modified). The compelling performance benefits (see Section IV-A) of the peer-to-peer architecture for the user-to-user interactions suggest that the peer-to-peer messaging will enjoy increasing popularity in such applications. An example of a resulting mixed architectural model is shown in Fig. 11 for three users (with associated mixed client and peer functionality) and a single information server. All client/peer and server terminals or hosts can include repositories of applets, yielding the flexibility to locate computation wherever it results in the best responsiveness, lowest latency, and can access the data it needs while insuring interoperability.

[10]The success of this approach depends on the availability of a support library as a part of the virtual machine that abstracts machine-dependent functions like the graphical-user interface, sound and video input and output, etc. into platform-independent calls by the application. The success of platform independence in this sense is as yet inconclusive.

[11]This also has impact on the symmetry or asymmetry of the bitrate. Some bitway providers are assuming a much higher bitrate in one direction (server to client) than the other (client to server). Such assumptions do not take into account the peer-to-peer applications, nor the dynamic distribution of peer applications.

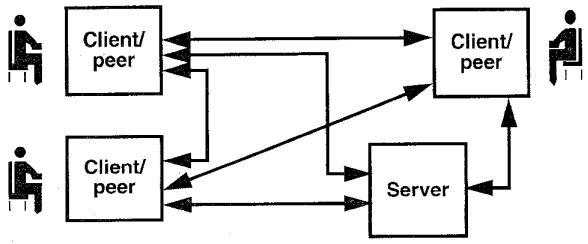

Fig. 11. Networked applications of the future can be expected to combine user-to-user and user-to-information server functionality in a mixed peer-to-peer and client-server architecture.

At the beginnings of convergence, telecommunications and computing shared many common technologies, but were distinguished in two primary ways. First, telecommunications focused on immediate user-to-user applications (principally telephony and video conferencing), while computing focused on initially stand-alone, and subsequently deferred user-to-user and immediate user-to-information-server applications, both on their dedicated and separate infrastructures. Second, telecommunications focused on continuous-media like audio and video, while computing focused on information storage, retrieval, and manipulation. These distinctions are no longer useful, for two reasons. First, all applications and media will share a common horizontally integrated infrastructure. Second, largely as a result of this common infrastructure, networked applications will no longer be neatly segmented. With dynamic deployment of networked applications, and the removal of traditional network externality obstacles to networked applications (particularly user-to-user applications), we can expect a proliferation of user-to-user collaboratory applications that freely mix user-to-information-management components.

The dynamic deployment of interwoven user-to-user and user-to-information-server multimedia applications in a horizontally-integrated terminal and network environment represents the pinnacle of convergence. Networked applications that freely mix the constituent elements traditional to telecommunications and computing will become commonplace. At this point, there no longer exists any technological or intellectual differences that distinguish telecommunications from computing. At this point, the dynamism and rate of progress in user-to-user applications becomes as great as has been recently experienced in user-to-information-server applications. As the availability of appropriate networked terminals is becoming widespread (for example Internet-connected personal computers with multimedia capabilities), this pinnacle of convergence will soon be upon us.

IV. Some Common Themes in Convergence

The preceding has given a historical perspective on convergence, from the perspective of the users, applications, and industries. Here we discuss some broad technical themes that have distinguished telecommunications and computing and have been at the foundation of the intellectual dissimilitudes. These dissimilitudes represent opportunities, since each field has its own perspectives that suggest new research directions. Understanding how these technical distinctions are disappearing is also another way to appreciate the implications of convergence.

A. Best-Effort Versus Quality-of-Service (QoS)

Arguably the greatest distinction between telecommunications and computing has been in performance metrics. A model of service provided by many computer systems and computer networks is *best-effort,* which can be described as "always strive to achieve better performance though more advanced technology or improvements in architecture, but there is no absolute performance standard; we are never satisfied." Since best-effort service does not take account of application needs, it is a "resources are cheap" model, in which applications may be provided considerably greater performance than they need, possibly at the expense of other applications that may receive less resources than they need. In fact, in the case of limited resources, most best-effort systems strive to achieve "fairness," attempting to apportion those limited resources according to some equality criteria. An early example of best-effort service (uncharacteristically in telecommunications) is *digital speech interpolation (DSI),* which statistically multiplexes speech sources, apportioning the available bit rate equally among the stochastically varying active sources [77]. Because the speech quality deteriorates gracefully with increasing traffic load, high traffic can be accommodated, at the expense of no guarantee on the quality of speech reproduction for any user. Best effort is the philosophy of design of the present Internet. For example, *fair queuing* allocates bitway bandwidth in packet networks during periods of congestion equitably among competing sources [78], [79].

With rare exceptions like DSI, telecommunications has focused on QoS guarantees, which can be described as "reliably achieve a level of performance that the user finds acceptable, but no better than that." Thus QoS is resource-conserving, assuming resources are expensive and must be conserved. Because bitways support a variety of applications, each with a different standard of what the user finds acceptable, it is usually assumed that bitways provide variable QoS (a different QoS to each application). This requires resource-allocation mechanisms that adjust resources (such as bandwidth, buffer space, etc.) to the provisioned QoS [80]. It is inherent in QoS that there have to be pricing mechanisms that distinguish different QoS; otherwise, the application will always choose the highest available QoS. Resource-allocation and pricing and billing mechanisms add a significant level of complexity to the bitway. Further, provisioning run-time variable QoS adds additional processing mechanisms that may actually slow down the bitway, since switching electronics is a significant bottleneck in today's bitways and there is often an inverse relationship between speed and complexity in electronics.

Another related difference in approach is one of trust. Perhaps because of its QoS objectives and related pricing, telecommunications has placed defenses against hostile

users, for example deploying policing policies at network access points. Networked computing has placed more trust in the users, for example, by building flow control mechanisms into protocol suites but not enforcing them within the bitway.

These different philosophies have been driven by their different applications. In particular, telecommunications has focused on continuous-media like audio and video, where improvements in performance beyond a certain level are not perceived by the user. Further, the focus has been on immediate applications like telephony or broadcast television with broad appeal, rather than high performance applications for smaller customer groups. In computing, on the other hand, there are always technology-driving applications that stress the available technology. Further, networked-computing applications have typically been deferred, and thus have not required performance guarantees. Consistent with horizontal integration, networks of the future will integrate deferred and immediate networked applications. Thus there has been considerable effort in mixing the QoS and best-effort service models [82], [84], including into today's premier horizontally integrated bitway technologies, ATM [81], and the Internet IP [85].

Three QoS performance attributes of a bitway that we can consider guaranteeing are[12]:

- The *rate* with which bits are generated, the temporal evolution of that rate, and the duplex symmetry or asymmetry of that rate.
- The *latency* between the time information enters the bitway and when it emerges at the destination. Components of the latency include the time required to accumulate a packet at the source (if the information is packetized), the speed-of-light propagation delay, queuing delays in switches, and processing and buffering delays in the computer OS's.
- The *reliability* with which the information is delivered. There are two forms of unreliability: *loss* (packet never arrives) and *corruption* (packet with a valid header arrives, but with bit errors in the payload).

There are significant requirement variations in all three of these performance attributes across different applications, and there are typical divergent assumptions made in telecommunications and network computing. These differing assumptions have resulted in different technological solutions to resolve in horizontal integration. We will now review each of these three performance attributes in turn.

1) Rate: A basic distinction can be made between two basic types of bit streams.

- *Continuous media* like digital audio, video, and animation [90], [91], are digital representations of analog signals and generate a continuous stream of information bits constrained by the underlying structure of the analog signal being represented. For example, video is composed of a sequence of image frames, which are desirably reconstructed in a reasonable facsimile of their original order and relative timing.
- *Sporadic media* like graphics and data, where information bits are allowed to flow in an unpredictable and unstructured fashion.

In terms of rate characteristics, continuous media can be represented by a continuous stream of bits with variable or constant bitrate, whereas sporadic media may have periods of very high bitrate interspersed with dormant periods.

The telecommunications infrastructure traditionally focused on the continuous-media extreme, fixed bitrate (circuit) transport with no statistical multiplexing [92], whereas computer networking has focused on sporadic media with extremes of statistical multiplexing advantage. Circuit switching avoided congestion losses, but is forced to perform admission control in the form of blocking at establishment during traffic overloads. Computer networking has not used admission control, offering service to all comers, but has utilized best-effort techniques to divide the available capacity among all services. As mentioned below, both communities appear to be evolving toward a horizontally integrated bitway infrastructure supporting both service models.

2) Latency: Quite distinct transport requirements apply to immediate and deferred applications. For immediate applications, interactive latency is often a critical element of subjective quality; thus transport latencies are often required to be both *short* (tens or hundreds of milliseconds) and *guaranteed.* The desire for low latency in such applications is a key reason for the choice of a short packet size in ATM, as this reduces the time required to accumulate a packet at the bitway access point for a low bitrate service such as voice. Guaranteed latency is particularly important for immediate applications built on continuous-media services, such as voice telephony and video conferencing. These services typically require a synchronous reconstruction with strict temporal requirements, and thus any data arriving with excess latency is not used, just as if it had been lost. This has led to attempts to insure bounded delays in packet networks [93]. Other immediate applications have less critical latency requirements; for example, video on demand may allow multiple-second delays.

A primary advantage of the peer-to-peer architecture (when compared to the client-server architecture in user-to-user applications) is low latency, which is one reason it has been widely applied to immediate user-to-user applications in telecommunications. Client server adds not only server delay, but also possibly excess propagation delay due to more circuitous routing.

Statistical multiplexing accommodates streams with aggregate peak bitrates larger than the available bandwidth, and is therefore extremely efficient for sporadic media. A side effect of statistical multiplexing is latency associated with the buffering required to accommodate high instantaneous bitrates. In addition, sporadic media often require reliable delivery, which can only be achieved over unreliable transport bitways through multiple transmissions, with the side effect that latency cannot be guaranteed.

[12]There are, of course, other performance attributes, such as establishment or setup time, outage probability for wireless links, blocking probability for admission control, etc.

Fortunately, sporadic media can tolerate the larger latencies imposed by statistical multiplexing and reliability.

For the future, horizontal integration requires a high degree of flexibility in accommodating both continuous and sporadic media. Similar challenges occur in the computer OS, where additional latency is added though the statistical sharing of processing and memory resources, running counter to the latency requirements of continuous media. These are challenging issues, since the techniques usually associated with statistical speedups (caching, paging, queuing) and often at odds with performance guarantees.

One very attractive feature of transportable computation is the ability to finesse the latency issue by performing application functionality locally, avoiding bitway round-trip delays.

3) Reliability: Reliability in transport is adversely affected by congestion, which may cause *loss* by buffer overflow, and bit errors caused by noise or interference in transmission (which may cause loss if they occur in the packet headers or *corruption* if they occur in the packet payload). The techniques available for improving reliability, including forward error-correction coding, diversity, and acknowledgment and retransmission protocols, have the fundamental side effect of increasing latency.

As in rate and latency, there is a wide gulf in reliability guarantees between the approaches traditionally used in telecommunications and computer networking. Continuous media, since they represent an analog signal, can tolerate reasonable levels of loss and corruption with adequate subjective quality. On the other hand, these media often have critical latency requirements. Thus telecommunications has focused on transport techniques like circuit switching that guarantee latency but not reliability. Computer networking, on the other hand, has typically dealt with sporadic media and thus has focused on transport techniques such as packet switching and statistical multiplexing, appending transport protocols (like transmission control protocol (TCP)/IP [13]) that guarantee reliable delivery at the expense of indeterminate delay. Horizontal integration at the bitways level requires an interesting mix of these service models.

4) Where is QoS Important? Advocates of best-effort transport argue that mechanisms for controlling QoS will slow the bitrates supported by the bitways, since switching electronics is a bottleneck, and in addition the associated infrastructure required for signaling and billing will add significant costs. Thus, it is argued, a scalable best-effort bitway will provide adequate performance near-term for the lowest cost by simply provisioning adequate resources, possibly accompanied by admission control to insure that those resources are adequate under worst-case traffic conditions.

Whether or not this best-effort argument is valid, it is clear that given geometric advances with time in processing, storage, and bandwidth, many performance issues will rapidly disappear. Research should focus on serious fundamental limitations or bottlenecks that are not mitigated by technology advances. We can easily identify two such bottlenecks.

- The total traffic density of wireless (radio and infrared) access links is subject to fundamental limitations that are much more limiting than backbone bitways. These limits on traffic are strongly dependent on the corruption QoS requirement; in fact, corruption is more important than bit rate in determining capacity limits (which is diametrically opposite to backbone bitways). The disparity between the traffic capacity of wireless access links and backbone bitways will only widen as backbone bitways get faster. There are emerging millimeter-wave radio technologies that will significantly ease this bottleneck in the indoor environment, but it will likely remain an issue in wide-area access technologies.
- The latency is lower bounded by propagation delay. In a global information infrastructure, the propagation delay (several hundred milliseconds round-trip halfway around the world) is already a serious problem for interactive applications. Thus there is little headroom to increase delay through compression signal processing, queuing, etc. Propagation delay will also increasingly interfere with bitway control and coordination as the bandwidth increases.

At the same time, both processing power and bandwidth in backbone bitways advance geometrically. Disturbingly, the two lasting bottlenecks are largely ignored, while most attention is focused on bandwidth efficiency and other less critical issues. For example, video compression research focuses almost entirely on minimizing bit rate (a resource increasingly available in fiber bitways and storage systems) while ignoring the resulting stringent reliability requirements (a scarce resource on interference-dominated wireless access links) and the signal processing delay. Similarly, a disturbing tendency is to solve interoperability problems in heterogeneous environments by utilizing conversions or transcoding, operations that can introduce significant delay (as well as interfere with security and privacy by precluding encryption). Most research in terminal-to-network coordination is focused on congestion mechanisms in backbone bitways, while neglecting the more fundamental interference-related impairments in wireless access links.

Similarly, *information theory* focuses on fidelity, providing fundamental limitations on the throughput of physical channels with high fidelity and the maximum fidelity that can be achieved for a given bit rate in a signal's digital encoding [105]. For the most part, information theory ignores delay (in many aspects it explicitly allows delay and complexity to be unbounded as a key assumption, with notable exceptions like error exponent bounds). On the other hand, *queuing theory,* which has been applied extensively to both computer networks and computer systems, focuses on delay and loss due to congestion, but offers no insights on fidelity [106]. A key issue in convergence is uniform and unified ways of dealing with delay, loss, and corruption at the practical as well as theoretical levels. Particularly challenging, as mentioned before, is the problem of integration of different media

and applications with variable QoS (delay and reliability) requirements.

Associated with QoS are numerous other issues where the traditional signal processing and communications theory communities can make a strong contribution. Among them are the relationship of quantifiable transport impairments on subjective quality, the aggregation of impairments in concatenated transport media, and various optimization questions related to the allocation of end-to-end impairments to individual facilities. Also of great interest are negotiation strategies between network and terminals to arrive at acceptable solutions, and the mechanization of these negotiations.

B. Scalability

Powerful forces underlying both telecommunications and computing are exponential increases with time in the processing of electronics, the bandwidth provided by photonics, and the capacity of storage systems [86]. These advances have a strong tendency to overwhelm performance issues, given the passage of reasonable time. Nevertheless, at any given time, it is important to be able to accommodate whatever performance level or number of users necessary by simply *adding* resources to the system, as opposed to *replacing* the technology for higher performance. An architecture with this property is *scalable.* A desirable form of scalability is a resource cost that is at most linear in some measure of performance or usage. Scalability and technology advances together represent a powerful force: at any given time we can accommodate any number of users or achieve any performance at a cost roughly constant per user or proportional to performance, and over time the cost-performance (if we are willing to replace the hardware) improves geometrically.

Scalability has always been an overriding requirement in telecommunications, because of the desire to serve ever larger numbers of users in a common networked system. With network externality, the utility to each user increases with the number of users, and there may actually be economies of scale so that the cost per user decreases with the number of users, resulting in extremely favorable economics. (In addition, cross subsidies have also been used to achieve "universal" service in the telephone network.) Prenetworked computing, on the other hand, has focused on the single-user model, where scalability is not an issue. For networked computers, a strength of the peer-to-peer architecture is its inherent scalability. The server in the client-server architecture, however, represents an obstacle to scalability, both with respect to bitway bandwidth and processing power, unless the server is itself a parallel processor with scalability properties [87] or can be mirrored indefinitely.

An example where scalability is a dominant consideration is communicating a single source simultaneously to multiple sinks, as illustrated in Fig. 12. An example is multi-party video conferencing (where each user participant wishes to see all the other users) or remote learning (where each student wishes to see a common lecture). An obvious approach requiring no special measures in the bitways is for the source to *simulcast* to each of the sinks over separate streams. Simulcast is fine for a small number of sinks, but is not scalable to large number of sinks because, as the number of sinks increases, either the source processing power or access bitway bitrate will eventually be exceeded. A scalable approach is *multicast,* in which the source generates a single stream common to all sinks, and that stream is appropriately replicated within the bitway. Bitways supporting multicast are fundamentally different from unicast bitways, and a topic of intense research for both the Internet (the Multicast backbone [94]), ATM bitways [95], and ATM-based internets [96]. An alternative architecture is to add servers to the network which perform the splitting function (as for example the reflectors in CUSeeMe [18]), but this approach is also not scalable.

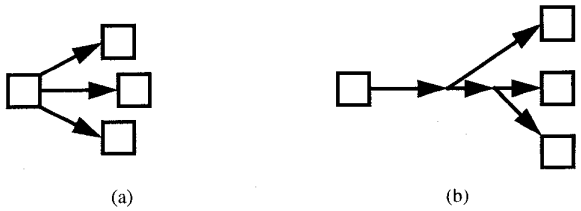

Fig. 12. Two alternatives for a service with one source and multiple sinks.

Transportable computation will play a major role in scalability. For example, the dynamic deployment of application functionality to a client reduces the transport bandwidth during the session and shifts computational cycles from the server to the client. The result can be a considerable increase in the number of clients supported by the server.

C. Terminal and Network Coordination

All networked applications require some level of coordination among the terminals (peer, clients, and servers) participating in the application, and between those terminals and the network. This coordination can occur during the setup phase, using so-called *establishment protocols* (in computer communications [99]) or *signaling* (in telecommunications [92]).[13] The distinction in terminology arises in part from the tendency to perform signaling functions over the same port and network as is used for data in computer communications (*in-band* signaling), and over a logically separate signaling network (*out-of-band* signaling) in telecommunications (as for example the modern *Signaling System Number Seven* [97]). As telecommunications moves toward horizontally integrated packet networks, there is debate as to whether to employ the in-band or out-of-band model [101]. Signaling is usually applied to the configuration of terminals and network, including the resource reservations that may be required for QoS guarantees and establishing the state in the bitway required to maintain

[13] Standards for negotiation and coordination of different system elements have been termed *etiquette standards* [98].

connections (see Section IV-D). This coordination can also occur dynamically during the execution phase of the application, called a *session* (computer communications) or *call* (telecommunications), through some form of flow control or other control mechanism.

Consider the coordination needed between a terminal originating a bit stream (called the *source*), the network carrying that stream, and the destination of that stream (called the *sink*). Both computer communications and telecommunications have used a *network-reactive signaling* model, in which the source makes a configuration request through the signaling channel and the network reacts to this request to perform the appropriate internal configuration. The network may also decline if it cannot provision the necessary resources, called *admission control* (computer communications) or *blocking* (telecommunications).

Network-reactive signaling is not the only way to perform active establishment configuration; in fact, enhanced mechanisms may be needed in the future. Consider, for example, configuration of the bitrate needed for a given service. Bitways such as ATM will be capable of provisioning a wide range of bitrates, and yet may or may not have wireless access at one or both ends. A broadband bitway with or without wireless access may have quite distinct capabilities, and the *source* may therefore have to configure in response to the network. This requires either *source-reactive signaling,* or better yet a *two-way negotiation* between source and bitway. Another important example is pricing. If bitways price their service based on resources consumed, finding the desired trade-off of resources versus price will require a negotiation, auction, or other two-way interaction.

It is also possible to coordinate a source and bitway dynamically during a session using flow control [102]. This coordination approach is common for best-effort bitways, and is especially natural for reliable delivery protocols (like TCP/IP [13]) since unacknowledged packets are an excellent estimate of traffic excesses. However, as bitway bitrates increase, propagation delay will remain constant, making flow control progressively less effective (due to the delay in receiving feedback from congestion bottlenecks coupled with more rapid variations in congestion).[14] For the future, an interesting alternative for continuous media is to use a *scalable* source coding, which presents a set of N *layers* made visible to the bitway. The convention is that if the sink has available only layers $(1, k)$, it can construct an increasingly accurate and subjectively pleasing representation as k increases. If the granularity of the layers is small and there are a large number of layers, there is no need for flow control since the bitway can simply throw away the highest layers as necessary. Scalable audio coding was used successfully two decades ago for voice transmission [77], and scalable video coders have recently been proposed [103].

Coordination issues become much more serious for bitways supporting multicast connections (Fig. 12). It is neither scalable nor reasonable to expect a source to deal with a multiplicity of downstream bitway links and sinks, including some dynamically entering or leaving the session [104]. Experimental multicast source coders for continuous media have thus of necessity been scalable. The most interesting approaches to configuration are sink (rather than source) driven [99], [100]. In a typical approach, the sink subscribes to layers $(1, k)$, makes an estimate of the resulting reliability (say by counting lost packets), and chooses to either increase or decrease k based on that estimate.

Terminal to network coordination is an area of great divergence between the traditional approaches of the computing and telecommunications communities. It is also arguably an area of great challenge for the future, with many competing approaches, as well as new requirements such as multicast. Scalable source coding and reliability measurements will be a profitable area for research.

D. Connection-Oriented Versus Connectionless

Telecommunications has traditionally focused on *connection-oriented* transport, where information is constrained to traverse the same route from source to destination. This approach enables resource allocation along that route to actively control QoS. Computer networks sometimes use *connectionless* transport, where information is routed dynamically according to congestion and availability. This approach is a natural outgrowth of the "efficiency by statistical means" orientation, and has many advantages, such as robustness to failure, ability to dynamically route around points of congestion, and the absence of state in the bitway (a tremendous simplification to the software). On the other hand, it makes QoS guarantees difficult to realize.

This distinction is narrowing. The Internet, while considering various forms of QoS guarantees in the future, as well as new services like multicast, add functionality and state information to the bitway, defining what are in effect connections [107]. ATM retains the notion of "virtual circuit," or the fixed route that packets traverse, while not constraining that virtual circuit to be fixed throughout a session. This architecture enables faster routing and reduces the addressing overhead per packet (since only local addressing is required), which is important because of the small packets. There is a desire to realize connectionless IP service on public wide-area ATM networks, which requires a layer of connectionless routers interconnected by ATM virtual circuits[15] [108]–[111]. Issues of mobility raise another layer of complications for connection-oriented protocols, since connections must be destroyed and reestablished dynamically during a session [112].

The evolving approaches to connections (or lack thereof) can only be described as chaotic at present, although as

[14] A more subtle problem is lost packets due to high bit error rates on wireless access links, which do not necessarily reflect congestion.

[15] The notion of connection-oriented TCP transport riding on a connectionless IP network which in turn rides on the connection-oriented ATM network is humorous, although it does offer some robustness to congestion and equipment failure and reduced establishment overhead.

driven by QoS considerations there is a definite trend toward connection-oriented protocols.

E. Control Architecture

Telecommunications has come full circle. The earliest electromechanical relay switches relied on a self-routing strategy for telephone calls, but with the advent of stored-program control a more centralized configuration strategy was followed based on an out-of-band signaling network [28], where the control and knowledge of service semantics reside primarily in the switches. As telephone switch software has suffered from inflexibility and runaway complexity, the problems of centralized control have become evident. More recently, ATM bitways have generally adopted a "command and control" approach, utilizing for example the legacy SS7 signaling network [97] for control of ATM switches, even though ATM could be quite amenable to a distributed control approach similar to the Internet.

The computer communications community has followed a diametrically opposite approach in which the control within the network is consciously minimized. In the Internet, the network typically does not store any state of particular TCP connections, but rather distributes that state information into the stream of packets passing through the network.[16] Routing tables do reside in the network, but they are updated through a distributed adaptive algorithm. This philosophy has been successfully extended to multicast connections in the MBone [94]. A considerable burden is put on the terminal nodes to retain knowledge of connections, perform flow control, and insure reliability through automatic repeat request (ARQ) protocols, consistent with the rapidly declining cost of the required processing. Cognizance of application semantics is strictly reserved for the terminals. This approach has proven quite effective at containing complexity, and also in maintaining flexibility for ready deployment of new applications since no upgrades to the network itself are needed. Running counter to this is the client-server architecture used to realize user-to-user applications (see Fig. 4), which can be considered a centralized control at the application layer (with the important distinction that the servers are administered separately from the bitway).

Regardless of the network control, application functionality will migrate to the terminals. This is consistent with the increasing cost effectiveness of terminal intelligence, and offers compelling advantages in flexibility and rapid innovation. This trend will accelerate as network deployment becomes widespread. This raises a number of issues relative to the transitioning that may occur, especially in the traditional telecommunications infrastructure. One approach is to encapsulate the existing centrally controlled telephone network for interface to computer applications, as in the Telephony Services API and Windows Telephony API [113]. Another approach would be to migrate to ATM bitways, which accommodate more directly a distributed control model.

[16] The analogy to dynamic deployment described above is interesting.

F. Interconnection Versus Interoperability

Since telecommunications has traditionally provisioned a small set of functionally simple "universal" applications, it has focused on interconnection as a basic issue. The goal has been to attract as many customers as possible, and fully interconnect them utilizing standardized protocols. Networked computing, on the other hand, focusing on a large number of functionally complex applications, has placed more emphasis on interoperability [114]. How can the distributed pieces of a networked application interact properly in accordance with their shared functionality and communication protocols?

Looking to the future, interoperability will be an increasing issue for the converging infrastructure. Approaches to interoperability that avoid cumbersome standardization at the application layer are immature, as there are competing approaches with different strengths and weaknesses. The distributed OS attempts to make a distributed collection of processors appear as one entity, whereas distributed object-based programming models explicitly highlights the distributed environment by structuring the distributed application as a set of autonomous interacting agents or objects [115], [116]. The virtual machine (Section III-G) follows the object model, but with the twist of transportable computation. All these approaches fall within the category of middleware, although the boundary between OS and middleware is fluid [36]. It seems that the distributed OS model is an option only for coordinated "intranets" (internets under control of a single organizational entity), while the strength of the virtual machine is its applicability to the general public network. However, a great deal of research is needed to establish the best approaches, presumably merging the best features of these disparate models and defining new ones.

G. Embedded Versus General-Purpose Computing

In a software implementation, there are alternative implementation styles that also have significant impact on issues like application deployment. The highest-performance software approach uses *embedded computing,* in which a processor is dedicated to a single function or application and is embedded within a larger system [117], with a minimal OS, highly optimized special-purpose instruction set, optimized code (perhaps even written in assembly language), etc. Such tuned software implementations have been used extensively for digital signal processing functions in telecommunications.

Where lower performance is acceptable, a software implementation on a general-purpose (often desktop) computer can serve a variety of functions simultaneously. This approach is very flexible, but current desktop OS's typically do not support resource reservation for a given application to guarantee, for example, real-time performance. (There

is no fundamental reason they cannot, however.) On the other hand, as the processor speeds increase in relation to the application, a point is reached at which it no longer matters (desktop computers are completely adequate for audio applications today) [118].

Perhaps the role of embedded computing would be reduced in the future with advances in technology. However, once again the role of communications in computing looms large. When a computer is networked at sufficient speed, the need for aggregating within it a variety of functions like memory, storage, etc. becomes less compelling, because some of those functions become available on the network at sufficient levels of performance. Thus one can envision in the future embedded computers that serve the single function of running dynamically distributed applications with a minimum of local storage and peripherals. In a sense, this is a hybrid of the two models of computing, since the such a computer would be dedicated to running a single interpreter (and hence is embedded) very efficiently, and at the same time is able to serve a variety of applications (represented by the interpreted programs).

Once again, we see technology taking a full circle. At one time there were dedicated computer designs for word processing, computer-aided design, etc. Advances in technology obsoleted this approach, as users preferred a general-purpose machine. In the future, it is possible that dedicated interpreter engines for dynamically distributed network applications will reappear, partially obsoleting the general-purpose computer.

H. Heterogeneity

Historically, both computers and communications networks were relatively homogeneous entities. The modern digital telephone network, for example, at its heart provisions a single service, the 64-kb/s connection-oriented bit stream. Likewise, most terminals (telephones) perform a basic analog voiceband channel function. Before the networking of computers, the application developer only had to worry about a single homogeneous platform.

We are entering a challenging age of heterogeneity [49], [59]. Heterogeneity will occur at several levels.

- There will be *heterogeneity in customer-premise terminals,* with a number of terminal options (telephones, desktop and laptop computers, personal digital assistants, "Dick Tracy" wrist watches, etc.).
- There will be *heterogeneity in transport systems* (packet switching, circuit switching, fiber optics, wireless access, etc.). This is complicated by the many combinations of concatenated transport options that are available for any given connection.
- There will be *heterogeneity in services and applications,* as described earlier, integrated within a common bitway and terminal infrastructure.

Due to network externality, there is a strong economic push toward universal interoperability among terminals, at least for the most common services and applications, irrespective of the details like terminal type or capability, terminal manufacturer, bitway, etc. The user wants applications to operate seamlessly across this infrastructure, configuring themselves to the infrastructure. This problem is most serious for continuous-media services, where the issue is not simply functional interoperability, but also matching resources to achieve QoS guarantees and required processing performance levels.

Historically, the telecommunications industry has pursued an end-to-end application in a vertically integrated architecture, like telephony or video conferencing. Where heterogeneity has existed in telecommunications, the approach has been to partition the subsystems at the *service* level. For example, wireless cellular telephony is assembled by concatenating a wireless voiceband telephone channel with a wired voiceband channel; in other words, in the base station, a voiceband telephone channel is the assumed application. Looking ahead to horizontal integration, where there will be many different services coexisting within the same facilities, this approach will not work. It will not be possible to embed within the bitways assumptions about the services being carried, without introducing a large element of complexity and inflexibility.

The different path that is necessary for the future is to modularize bitways from the services and applications insofar as possible, with coordinated resource-allocation. The services and applications will need to adapt to a variety of heterogeneous terminal and transport configurations, as well as resource allocations, and conversely the transport and terminals will need to attempt to accommodate the differing needs of a variety of services and applications. All constituent fields will need to concentrate less on point solutions to narrowly defined problems, and more on coordination to achieve objectives like interoperability and QoS on an end-to-end system-level basis.

I. Architecture and Complexity Management

Networks of the future will need to satisfy a variety of requirements [49], which are unfortunately interrelated and interdependent. Among them, we can cite:

- point-to-point, multicast, and multisource connections;
- privacy by end-to-end encryption;
- predictability of and control over subjective quality;
- negotiation of QoS requirements on an end-to-end basis, and the allocation of impairments on a traffic-dependent fashion to concatenated transport links;
- application scalability to transport QoS parameters as well as terminal processing and display capabilities;
- low delay for critical interactive applications;
- high traffic capacity, particularly on bottleneck facilities such as wireless access; and
- interoperability across heterogeneous terminal and transport environments, and integration of heterogeneous services and applications within shared-resource environments.

All of these important objectives interact, and are sometimes at cross purposes. Finding a reasonable compromise among these objectives will require carefully crafted ar-

chitectural concepts. A key question is what horizontal interfaces should be established. Another question is how we avoid a proliferation of multiple interfaces that have not only different syntactical structure (a minor problem), but also present different semantic models of the underlying functionality. (For example, can we define parameterized QoS models that fit universally across radically different transport media like congestion-dominated backbone bitways and interference-dominated wireless access links?)

Once such architectural concepts are established, there are numerous detailed research issues that are stimulated in areas like compression, error-correction coding and modulation, and encryption. In particular, the nature of the overall network design problem forces much greater attention to architectural issues, and much greater influence of architectural issues on detailed research areas like signal processing and networking. This is a systematic way of coordinating the activities in these detail areas to meet the many interacting objectives mentioned above.

There is inadequate research that bridges the signal processing and networking worlds, and also inadequate research bridging the backbone and wireless access worlds. Today the important constraints introduced by the wireless access bottleneck are largely unrepresented in the design of backbone networks, even though they introduce important constraints.

One impact of the coming heterogeneity at the application, transport, and terminal levels is the critical importance of complexity management [59]. Complexity management has traditionally been a dominant consideration in the design of software systems, but is now also a dominant consideration in the larger context of large-scale systems including hardware, software, and physical channels. A whole host of techniques, many of them developed in the context of software system engineering, become important, such as architecture, modularity, and abstraction. More than anything, complexity management is a manner of thinking about system design. There is need for the infusion of this complexity management thinking throughout the domain of communications and computing, not just software design.

J. Economic and Business Models

In the environment of converged telecommunications and computing, the old-style design problem embodied in one organization presenting a complete end-to-end turnkey solution is gone. Rather, many vendors are participating, in effect, in the collective design of the infrastructure of the future. Such designs must take in account numerous external considerations, such as network externality, standards (or lack thereof), interoperability, adaptability and etiquette, etc. The lowered barriers to application development embodied in the migration from vertical to horizontal architectures have and will play an important role in industrial organization. Considerations such as these play a seminal role in the design of products, and should also have a larger presence in research and engineering education.

K. Applications

New developments like platform independence, network deployment, and dynamic deployment will create an environment in which the innovation in user-to-user applications will have similar characteristics to user-to-information-server applications; namely, an rapidly evolving and fragmented application space. As in client-server computing, this will be a fertile field for research.

V. Conclusions

An exciting future is at hand. The relentless march of technology has resulted in a de-emphasis of traditional performance metrics and much more focus on functionally complex and heterogeneous systems, as well as on applications. While this will result in a much richer set of applications to the end user, it also burdens technologists with the strain of relentless change. The rapidly advancing performance of electronics and photonics enables less efficient but functionally more complex software implementations, and hence greater emphasis on functionally complex services realized in a heterogeneous transport and terminal infrastructure. This implies that the traditional challenges of efficiency and performance are being partially displaced by considerations of architecture, complexity management, and greater focus on the end user and their applications and requirements. The greatest shift can be expected in user-to-user applications, a traditional focus of telecommunications, where the ability to flexibly deploy new applications is limited only by the imagination of entrepreneurs and designers, as opposed to the constraints of interoperability and standardization.

Acknowledgment

The author would like to thank the following colleagues who provided valuable comments on drafts of this paper: G. D. Forney, L. Gun, D. Leeper, and J. Major of Motorola, S. Personick of Bell Communications Research, B. Rosin of ESPI, E. Lazowska of the University of Washington, J. Godfrey of the National Research Council Computer Science and Telecommunications Board, C. Strathmeyer of Dialogic, K. Krechmer of *Communications Standards Review,* and W.-T. Chang, H. Varian, R. Wilensky, and W. Li of the University of California at Berkeley.

References

[1] D. G. Messerschmitt, "The future of computer-telecommunications integration," *IEEE Commun. Mag.*, special issue on Computer-Telephony Integration, Apr. 1996.
[2] ——, "Convergence of telecommunications with computing," invited paper in the special issue on "Impact of Information Technology," *Sci. and Soc.,* to be published.
[3] *Merriam–Webster's Collegiate Dictionary,* 10th ed. Springfield, MA: Merriam-Webster, 1995.
[4] J. A. Adam, "Upgrading the Internet," *IEEE Spectrum,* vol. 32, no. 9, pp. 24–29, Sept. 1995.
[5] ——, "Multimedia-applications, implications," *IEEE Spectrum,* vol. 30, no. 3, pp. 24–31, Mar. 1993.

[6] B. Cole, "Multimedia—the technology framework," *IEEE Spectrum,* vol. 30, no. 3, pp. 32–39, Mar. 1993.
[7] J. L. Flanagan, "Technologies for multimedia communications," *Proc. IEEE,* vol. 82, pp. 590–603, Apr. 1994.
[8] A. Iwata *et al.,* "ATM connection and traffic management schemes for multimedia internetworking," *Commun. ACM,* vol. 38, no. 2, pp. 72–89, Feb. 1995.
[9] E. Francik, S. E. Rudman, D. Cooper, and S. Levine, "Putting innovation to work: Adoption strategies for multimedia communication systems," *Commun. ACM,* vol. 34, no. 12, pp. 52–63, Dec. 1991.
[10] *Realizing the Information Future; The Internet and Beyond.* Washington, DC: Natl. Acad. Press, 1994.
[11] *The Unpredictable Certainty: Information Infrastructure Through 2000.* Washington, DC: Natl. Acad. Press, 1966.
[12] F. Fluckiger, *Understanding Networked Multimedia: Applications and Technology.* Englewood Cliffs, NJ: Prentice-Hall, 1995.
[13] D. E. Comer, *Internetworking with TCP/IP: Volume I, Principles, Protocols, and Architecture.* Englewood Cliffs, NJ: Prentice-Hall, 1995.
[14] S. Carl–Mitchell, "The new Internet protocol," *Unix Rev.,* vol. 13, no. 7, pp. 31–34, 36, and 38, June 1995.
[15] B. B. Woodward, "Will OSI be eclipsed by TCP?," in *Proc. SHARE Europe Anniversary Mtg.: Client/Server—The Promise and the Reality,* The Hague, the Netherlands, Oct. 1993.
[16] P. T. Breen Jr., "Real virtual environment applications—Now," in *Proc. Visualization '92,* Boston, MA, Oct. 19–23, 1992.
[17] R. M. Satava, "Emerging medical applications of virtual reality: A surgeon's perspective," *Artificial Intell. Medicine,* vol. 6, no. 4, pp. 281–288, Aug. 1994.
[18] R. E. Leon, "The world of Internet: A client-server architecture and the new generation of information servers," *Online CD ROM Rev.,* vol. 18, no. 5, pp. 279–284, Oct. 1994.
[19] W. Mitchell, *City of Bits: Space, Place and the Infobahn.* Boston: MIT Press, 1995.
[20] S. Simon, "Peer-to-peer network management in an IBM SNA network," *IEEE Network,* vol. 5, no. 2, pp. 30–34, Mar. 1991.
[21] A. Sinha, "Client-server computing: Current technology review," *Commun. ACM,* vol. 35, no. 7, pp. 77–98, July 1992.
[22] S. Broadhead, "Client-server: The past, present and future," *Network Computing,* vol. 4, no. 12, pp. 38, 40, 42, and 43, Dec. 1995.
[23] G. Orphanos *et al.,* "Client-server computing requirements of networked multimedia services," in *Int. Sem. Client/Server Computing,* La Hulpe, Belgium, Oct. 30–31, 1995.
[24] M. C. Hao, A. H. Karp, and D. Garfinkel, "Collaborative computing: A multi-client multi-server environment," in *Proc. Conf. Organizational Computing Syst. COOCS '95,* Milpitas, CA, Aug. 13–16, 1995.
[25] F. Ryan and A. B. Sperry, "The desktop computer emerges as a powerful engineering tool," *Electron. Design,* vol. 27, no. 14, pp. 62–67, July 5, 1979.
[26] R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks," *Commun. ACM,* vol. 19, no. 7, pp. 395–404, July 1976.
[27] G. Falk, "A comparison of network architectures—The ARPANET and SNA," in *AFIPS Conf. Proc. Natl. Computer Conf.,* Anaheim, CA, June 5–8, 1978, vol. 47.
[28] C. A. Dahlbom and J. S. Ryan, "Common channel interoffice signaling: History and description of a new signaling system," *Bell Syst. Tech. J.,* vol. 57, no. 2, pp. 225–250, Feb. 1978.
[29] T. Larsson, "Prescription for public service: Plenty of monopoly and horizontal integration," *Telephony,* vol. 203, no. 9, pp. 132–140, Aug. 22, 1983.
[30] F. Ostroff and D. Smith, "The horizontal organization," *McKinsey Quarterly,* vol. 1992, no. 1, pp. 148–168, 1992.
[31] R. E. White and T. A. Poynter, "Achieving worldwide advantage with the horizontal organization," *Business Quarterly,* vol. 54, no. 2, pp. 55–60, Autumn 1989.
[32] J. MacKie–Mason, S. Shenker, and H. R. Varian, "Network architecture and content provision: An economic analysis," in *Proc. Public Policy Corp. Strategy Info. Economy,* Evanston, IL, May 10–11, 1966.
[33] G. F. Borton, "Seeds of change in CTI," *Bus. Commun. Rev.,* vol. 24, no. 3, pp. 35–40, Mar. 1994.
[34] R. Walters, *Computer Telephone Integration.* London, U.K.: Artech, 1993.
[35] P. Strauss, "Welcome to client-server PBX computing," *Datamation,* vol. 40, no. 11, pp. 49, 50, and 52, June 1, 1994.
[36] P. A. Bernstein, "Middleware: A model for distributed system services," *Commun. ACM,* vol. 39, no. 2, pp. 86–98, Feb. 1996.
[37] D. Clark, "Interoperation, open interfaces, and protocol architecture," draft white paper at NII 2000 Forum, Washington, DC, May 23, 1995.
[38] R. Pandya, "Emerging mobile and personal communication systems," *IEEE Commun. Mag.,* vol. 33, no. 6, pp. 44–52, June 1995.
[39] J. E. Padgett, C. G. Gunther, and T. Hattori, "Overview of wireless personal communications,"*IEEE Commun. Mag.,* vol. 33, no. 1, pp. 28–41, Jan. 1995.
[40] W. W. Erdman, "Wireless communications: A decade of progress," *IEEE Commun. Mag.,* vol. 31, no. 12, pp. 48–51, Dec. 1993.
[41] W. H. W. Tuttlebee, "Cordless personal communications," *IEEE Commun. Mag.,* vol. 30, no. 12, pp. 42–53, Dec. 1992.
[42] D. C. Cox, "Wireless network access for personal communications," *IEEE Commun. Mag.,* vol. 30, no. 12, pp. 96–115, Dec. 1992.
[43] A. D. Kucar, "Mobile radio: An overview," *IEEE Commun. Mag.,* vol. 29, no. 11, pp. 72–85, Nov. 1991.
[44] D. J. Goodman, "Trends in cellular and cordless communications," *IEEE Commun. Mag.,* vol. 29, no. 6, pp. 31–40, June 1991.
[45] R. Bagrodia, W. W. Chu, L. Kleinrock, and C. Popek, "Vision, issues, and architecture for nomadic computing and communications," *IEEE Personal Commun.,* vol. 2, no. 6, pp. 14–27, Dec. 1995.
[46] S. Acharya and R. Alonso, "The computational requirements of mobile machines," in *Proc. 1st IEEE Int. Conf. Eng. Complex Computer Syst.,* Ft. Lauderdale, FL, Nov. 6–10, 1995.
[47] S. T. Vuong, O. Lau, Y. Q. Yu, H. Shi *et al.,* "Issues in internetworking wireless data networks for mobile computing," in *Proc. IEEE Pacific Rim Conf. Commun., Comput., Signal Process.,* Victoria, BC, Canada, May 17–19, 1995.
[48] D. Duchamp, "Issues in wireless mobile computing," in *Proc. 3rd Workshop Workstation Operating Syst.,* Key Biscayne, FL, Apr. 23–24, 1992.
[49] P. Haskell and D. Messerschmitt, "In favor of an enhanced network interface for multimedia services," submitted to *IEEE Multimedia Mag.*
[50] K. Young, "Look no server (peer-to-peer networks)," *Network,* pp. 21, 22, and 26, Mar. 1993.
[51] G. M. Karan, "From current to future telepresence technologies (Did the interstate system kill Route 66?)," *Canadian Artificial Intell.,* no. 37, pp. 8–17, Summer 1995.
[52] *Telemanipulator and Telepresence Technologies.* Boston, MA, Oct. 31–Nov. 1, 1994; *Proc. SPIE—The Int. Soc. Opt. Eng.,* 1994, vol. 2351.
[53] H. Krechmer, "Catching up with V. 34 modems," *Bus. Commun. Rev.,* vol. 25, no. 3, pp. 62–65, Mar. 1995.
[54] D. Le Gall, "MPEG: A video compression standard for multimedia applications," *Commun. ACM,* vol. 34, no. 4, pp. 46–58, Apr. 1991.
[55] P. Wilson, "Computer supported cooperative work (CSCW): origins, concepts and research initiatives," in *2nd Joint European Networking Conf.,* Blois, France, May 13–16, 1991.
[56] K. Ruhleder and J. L. King, "Computer support for work across space, time, and social worlds," *J. Org. Computing,* vol. 1, no. 4, pp. 341–355, 1991.
[57] E. M. Schooler, S. L. Casner, and J. Postel, "Multimedia conferencing: Has it come of age?," in *Proc. 24th Annu. Hawaii Int. Conf. Syst. Sci.,* Kauai, HI, Jan. 8–11, 1991.
[58] R. Skinner, "Cross-platform formatting programs," *Library Software Rev.,* vol. 13, no. 2, pp. 152–156, Summer 1994.
[59] D. G. Messerschmitt, "Complexity management: An important issue in communications," in *Proc. Int. Conf. Commun., Computing, Contr., Signal Processing,* Stanford Univ., Palo Alto, CA, June 22–26, 1995.
[60] N. Economides, "The economics of networks," *Int. J. Indust. Org.,* Mar. 1996.
[61] A. Christiano, "The economic theory of information networks," *The Economics of Information Networks,* C. Antonelli, Ed. Amsterdam: North Holland, 1992.
[62] ——, "Externalities and complementarities in telecommunications dynamics," *Int. J. Ind. Org.,* vol. 11, no. 3, pp. 299–450.

[63] K. R. Conner and R. P. Rumelt, "Software piracy: An analysis of protection strategies," *Manage. Sci.,* vol. 37, no. 2, pp. 125–139, Feb. 1991.
[64] J. K. Ousterhout, "Tcl: An embeddable command language," in *Proc. Winter '90 USENIX Conf.,* Washington, DC, Jan. 22–26, 1990.
[65] N. S. Borenstein, "E-mail with a mind of its own: The Safe-Tcl language for enabled mail," in *ULPAA '94 Conf.,* Barcelona, Spain, June 1–3, 1994.
[66] J. Tardo and L. Valente, "Mobile agent security and telescript," *IEEE CompCon,* 1996.
[67] D. Woelk, M. Huhns, and C. Tomlinson, "Uncovering the next generation of active objects," *Object Mag.,* vol. 5, no. 4, pp. 32–39, July/Aug. 1995.
[68] K. Arnold and J. Gosling, *The Java™ Programming Language.* Reading, MA: Addison-Wesley, 1996.
[69] P. Wayner, "Net programming for the masses," *BYTE,* vol. 21, no. 2, pp. 101, 102, and 104, Feb. 1996.
[70] A. van Hoff, "Java and Internet programming," *Dr. Dobb's J.,* vol. 20, no. 8, pp. 56, 58, 60, 61, and 101–102, Aug. 1995.
[71] W.-T. Chang, W. Li, D. G. Messerschmitt, and N. Zhang, "Rapid deployment of CPE-based telecommunications services," in *Proc. Global Commun. Conf.,* San Francisco, Dec. 1994.
[72] W.-T. Chang and D. G. Messerschmitt, "Dynamic deployment of peer-to-peer networked applications to existing WWW browsers," in *Proc. Telecommun. Info. Network Architect. (TINA) '96 Conf.,* Heidelberg, Germany, Sept. 3–5, 1996.
[73] M. J. Wooldridge and N. R. Jennings, Eds., in *Intelligent Agents. Proc. ECAI-94 Workshop Agent Theories, Archit., Lang.,* Amsterdam: Springer-Verlag, Aug. 1994.
[74] M. Wooldridge and N. R. Jennings, "Agent theories, architectures, and languages: A survey," in *Proc. ECAI '94 Workshop Agent Theories, Architect. Lang.,* Amsterdam, The Netherlands, Aug. 8–9, 1994.
[75] O. Etzioni and D. S. Weld, "Intelligent agents on the Internet: Fact, fiction, and forecast," *IEEE Expert,* vol. 10, no. 4, pp. 44–49, Aug. 1995.
[76] V. Vittore, "Intelligent agents may jump start PDA's," *America's Network,* vol. 98, no. 24, pp. 28–29, Dec. 15, 1994.
[77] E. Lyghounis, I. Poretti, and G. Monti, "Speech interpolation in digital transmission systems," *IEEE Trans. Commun.,* vol. COM-22, no. 9, pp. 1179–1189, Sept. 1974.
[78] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queuing algorithm," *Internetworking: Research Experience,* vol. 1, no. 1, pp. 3–26, Sept. 1990.
[79] A. G. Greenberg and N. Madras, "How fair is fair queuing?," *J. Assoc. Computing Machinery,* vol. 39, no. 3, pp. 568–598, July 1992.
[80] A. A. Lazar and G. Pacifici, "Control of resources in broadband networks with quality of service guarantees," *IEEE Commun. Mag.,* vol. 29, no. 10, pp. 66–73, Oct. 1991
[81] G. E. Konstantoulakis and G. I. Stassinopoulos, "Transfer of data over ATM networks using available bit rate (ABR)," in *Proc. IEEE Symp. Comput. Commun.,* Alexandria, Egypt, June 27–29, 1995.
[82] N. C. Audsley, A. Burns, R. I. Davis, and A. J. Wellings, "Integrating best-effort and fixed priority scheduling," in *Proc. IFAC Workshop Real Time Programming '94,* Lake Constance, Germany, June 22–24, 1994.
[83] T. Nishida and K. Taniguchi, "QoS controls and service models in the Internet," *IEICE Trans. Commun.,* vol. E78-B, no. 4, pp. 447–457, Apr. 1995.
[84] H. Schulzrinne, J. Kurose, and D. Towsley, "An evaluation of scheduling mechanisms for providing best-effort real-time communications in wide-area networks," in *Proc. IEEE INFOCOM '94,* Toronto, Ont., Canada, June 12–16, 1994.
[85] S. Shenker, D. D. Clark, and L. Zhang, "Services or infrastructure: Why we need a network service model," in *Proc. 1st IEEE Int. Workshop Community Networking,* San Francisco, CA, July 13–14, 1994; New York: IEEE, 1994, pp. 145–149.
[86] C. Mead, "VLSI and the foundations of computation," in *Information Processing '83. Proc. IFIP 9th World Computer Congress,* Paris, France, Sept. 19–23, 1983, R. E. A. Mason, Ed. Amsterdam: North-Holland, 1983, pp. 271–274.
[87] K. Hwang, *Advanced Computer Architecture: Parallelism, Scalability, Programmability.* New York: McGraw-Hill, 1993.
[88] T. Berners–Lee and R. Cailliau, "World-Wide Web," in *Proc. Int. Conf. Computing High Energy Phys. '92,* Annecy, France, Sept. 21–25, 1992.
[89] K. Reichard and E. F. Johnson, "GUI Web browsers," *Unix Rev.,* vol. 13, no. 8, pp. 69–74, July 1995.
[90] A. Campbell, G. Coulson, F. Garcia, and D. Hutchison, "A continuous media transport and orchestration service," in *Proc. ACM SIGCOMM '92 Conf. Commun. Applicat., Architectures Protocols,* Baltimore, MD, Aug. 17–20, 1992.
[91] B. Wolfinger and M. Moran, "A continuous media data transport service and protocol for real-time communication in high speed networks," in *Network Operating System Support Digital Audio Video. 2nd Int. Workshop Proc.,* Heidelberg, Germany, Nov. 18–19, 1991; R. G. Herrtwich, Ed. Berlin, Germany: Springer-Verlag, 1992, pp. 171–182.
[92] J. C. McDonald, Ed., *Fundamentals of Digital Switching,* 2nd ed. New York: Plenum, 1990.
[93] D. C. Verma, H. Zhang, and D. Ferrari, "Guaranteeing delay jitter bounds in packet-switching networks," in *1st Int. Workshop Network Operating Syst. Support Audio Video,* Berkeley, CA, Nov. 8–9, 1990.
[94] M. R. Macedonia and D. P. Brutzman, "MBone provides audio and video across the Internet," *Computer,* vol. 27, no. 4, pp. 30–36, Apr. 1994.
[95] K. Chong–Kwon, "Blocking probability of heterogeneous traffic in a multirate multicast switch," *IEEE J. Selected Areas Commun.,* vol. 14, no. 2, pp. 374–385, Feb. 1996.
[96] G. J. Armitage, "Multicast and multiprotocol support for ATM based internets," *Computer Comm. Rev.,* vol. 25, no. 2, pp. 34–46, Apr. 1995.
[97] G. G. Schlanger, "An overview of signaling system no. 7," *IEEE J. Selected Areas Commun.,* vol. SAC-4, May 1986.
[98] K. Krechmer, "Technical standards: Foundations of the future," *Standards View,* Mar. 1996.
[99] L. Zhang *et al.,* "RSVP: A new resource ReSerVation Protocol," *IEEE Network,* vol. 7, no. 5, pp. 8–18, Sept. 1993.
[100] D. J. Mitzel and S. Shenker, "Asymptotic resource consumption in multicast reservation styles," in *ACM SIGCOMM '94 Conf. Commun. Architect., Protocols Applicat.,* London, UK, Aug. 31–Sept. 2, 1994.
[101] T.-H. Wu, N. Yoshikai, and H. Fujii, "ATM signaling transport network architectures and analysis," *IEEE Commun. Mag.,* vol. 33, no. 12, pp. 90–99, Dec. 1995.
[102] I. W. Habib and T. N. Saadawi, "Access flow control algorithms in broadband networks," *Computer Commun.,* vol. 15, no. 5, pp. 326–332, June 1992.
[103] D. Taubman and A. Zakhor, "Rate and resolution scalable subband coding of video," in *'94 IEEE Int. Conf. Acoust., Speech, Signal Process.,* Adelaide, Australia, Apr. 19–22, 1994.
[104] J.-C. Bolot, T. Turletti, and I. Wakeman, "Scalable feedback control for multicast video distribution in the Internet," in *ACM SIGCOMM '94 Conf. Commun. Architect., Protocols Applications,* London, UK, Aug. 31–Sept. 2, 1994.
[105] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* New York: Wiley, 1991.
[106] L. Kleinrock, *Queuing Systems.* New York: Wiley, 1975.
[107] C. D. Cranor and G. M. Parulkar, "An implementation model for connection-oriented internet protocols," *Internetworking: Research Experience,* vol. 4, no. 3, pp. 133–157, Sept. 1993.
[108] B. J. Vickers and T. Suda, "Connectionless service for public ATM networks," *IEEE Commun. Mag.,* vol. 32, no. 8, pp. 34–43, Aug. 1994.
[109] N. Kavak, K. Laraqui, A. Nazari, and P. Emberg, "Experience and analysis of a connectionless server for provision of broadband data communication service," in *Proc. Interworking '94—2nd Int. Symp. Interworking,* Sophia Antipolis, France, May 4–6, 1994.
[110] D. G. Harris, A. C. Perry, and M. D. Batts, "The provision and evolution of connectionless data services in the public network," in *2nd Int. Conf. Broadband Services, Systems Networks,* Brighton, U.K., Nov. 3–4, 1993 (Conf. Publ. No. 383).
[111] I. S. Venieris, E. N. Protonotarios, G. I. Stassinopoulos, and R. Carli, "Bridging remote connectionless LAN/MAN's through connection oriented ATM networks," *Computer Commun.,* vol. 15, no. 7, pp. 418–428, Sept. 1992.

[112] K. Keeton *et al.,* "Providing connection-oriented network services to mobile hosts," in *Proc. USENIX Mobile Location-Independent Computing Symp.,* Cambridge, MA, Aug. 2–3, 1993.
[113] P. Strauss, "Welcome to client-server PBX computing," *Datamation,* vol. 40, no. 11, pp. 49–52, June 1, 1994.
[114] F. Bar, M. Borrus, and R. Steinberg, "Islands in the bit-stream: Mapping the NII interoperability debate," BRIE Working Paper #79, Univ. Calif. Berkeley, 1995.
[115] F. Manola, "Interoperability issues in large-scale distributed object systems," *ACM Computing Surveys,* vol. 27, no. 2, pp. 268–273, June 1995.
[116] R. S. Chin and S. T. Chanson, "Distributed object-based programming systems," *ACM Computing Surveys,* vol. 23, no. 1, pp. 5–48, Mar. 1991.
[117] W. Wolf and T. Yen, "Embedded computing and hardware-software co-design," in *Proc. WESCON '95,* San Francisco, CA, Nov. 7–9, 1995.
[118] R. S. Rao, "Embedded computing with a comprehensive reduced instruction set processor," in *Proc. 4th Int. Conf. Signal Process. Appl. Tech., ICSPAT '93,* Santa Clara, CA, Sept. 28–Oct. 1, 1993.

David G. Messerschmitt (Fellow, IEEE) received the B.S. degree from the University of Colorado, Boulder, and the M.S. and Ph.D. degrees from the University of Michigan, Ann Arbor.

He is a Professor in the Department of Electrical Engineering and Computer Sciences at the University of California at Berkeley. From 1993 to 1996 he was Department Chair. Prior to 1977, he was at AT&T Bell Laboratories in Holmdel, NJ. He has served as a consultant to a number of companies, and is a co-founder and member of the Board of Directors of TCSI Corporation (NASDAC). His current research interests include issues overlapping signal processing (especially video and graphics coding) and transport in broadband networks with wireless access, network services, and protocols for multimedia, wireless multimedia computing, and the economics of networks.

Dr. Messerschmitt is a member of the National Academy of Engineering and the Computer Sciences and Telecommunications Board of the National Research Council.