XEROX

Xerox Network Systems Architecture

General Information Manual

XEROX NETWORK SYSTEMS ARCHITECTURE GENERAL INFORMATION MANUAL

XEROX

XNSG 068504
April 1985

Text and graphics prepared on the Xerox 8010 Information System, and printed on a Xerox laser printer using the Interpress electronic printing standard.

Xerox Corporation
Office Systems Division
2300 Geng Road
Palo Alto, California 94303

TABLE OF CONTENTS

0005

LIST OF FIGURES

1. INTRODUCTION

For nearly three decades Xerox Corporation has played an important role in improving the productivity of office workers. In addition to its widely-recognized line of plain paper copiers, Xerox is manufacturing and marketing word processors, facsimile transceivers, data terminals, electronic typewriters, professional workstations, electronic printers, and a host of related products. These advanced products are being used throughout the world to improve personal efficiency and productivity in handling physical and electronic documents in the office.

Productivity through systems

The size and vitality of the office systems industry is a direct result of the substantial gains in productivity users have been able to realize through modern computing and communication technology. As a result of those gains, work can flow more smoothly, data can be processed more accurately and quickly, and information can be distributed more efficiently. In turn, management can be more effective and responsive in its decision-making.

The early office machines were basically "point products," which means that their benefits were derived essentially from functions wholly performed within the product itself. For example, a point product such as a word processor made it possible for a secretary to be much more productive in many routine tasks, without requiring that the product be connected to anything else. In this fashion many of the more obvious tasks in offices were automated and significant gains in productivity were realized. Xerox was part of this initial wave of office automation with its word processing systems. Moreover, many of the most important subsequent developments in office automation have come from basic research work carried out at the Xerox Palo Alto Research Center.

Productivity through networks

One result of that research was a recognition that further major productivity gains would be made when point products could easily communicate and share work with each other. These new requirements for intercommunication and integration led Xerox to announce, with its partners, Intel Corporation and Digital Equipment Corporation, that Xerox' proprietary local area networking technology, Ethernet, would be made available to the business and academic communities. Xerox' intent was to promote widespread acceptance and use of local area network technology to provide a basic, general-purpose data transmission "highway" within a facility such as an office building, laboratory, or factory.

The importance of local area networks in such applications as office automation cannot be overemphasized. They are one of the most important means by which individual devices—workstations, word processors, electronic typewriters, filing and printing subsystems, etc.—communicate directly with each other. Being able to intercommunicate is a necessary first step toward the functional *integration* of these devices: the coordinated cross-functioning between products that will deliver the next wave of effectiveness and efficiency improvements in offices and related environments. (Note, however, that a local area network, or any other communication system, by itself does not guarantee integration; more is required as we will see.)

Productivity through integration

As an example of functional integration, when the Ethernet announcement was made, Xerox introduced a series of integrated office products called the 8000 Network System, which included the Star Information System along with a number of shared-resource "servers." Star pioneered such concepts as "bit-mapped" (or "all points addressable") displays, "windows" for simultaneously interacting with multiple processes, and a "mouse" for efficient operator control. The 8000 Network System provided not only the functions that were available with the older point products, but an entirely new set of functions such as coordinated document creation, centralized printing, and filing. These functions were possible only because of the high degree of integration among the members of the 8000 family.

More recently Xerox has broadened the integration of its product line, continuing to offer additional productivity improvements. Gradually drawn into the integrated community were such products as word processors, personal computers, very fast electronic printers, an input scanner, electronic typewriters, and special systems for electronic engineers. In each of these cases, integration means that all products work effectively with other products, exchanging data, sharing resources, and building applications, thereby leading to new levels of productivity.

Commitment to open systems

Xerox' development philosophy has emphasized taking maximum advantage of what has been developed before and of the efforts in the international standards groups that are leading to robust, durable standards. The openness of the Xerox approach works both ways: as new developments have matured at Xerox, a conscientious effort has been made to share these developments with the rest of the information systems world. Xerox is committed to an open systems approach to the development of its information systems.

Xerox Network Systems: the key to integration

Although the Ethernet local area networking technology makes it possible for intelligent devices to intercommunicate efficiently, simply having that facility does not ensure the devices will be integrated. All it guarantees is that they will be able to exchange data back and forth. Simply put, the benefits of an integrated information system can be obtained only when the individual elements work together. That, in turn, depends on what those elements do with the data they exchange.

Network architecture

The rules governing the exchange of information among networked devices, and specifying the processes through which work on that information is to be done, are collectively known as a network architecture. There are many different types of network architectures, varying with respect to their application orientation, the emphasis placed on local vs. wide area communications, etc. But they all define a structured approach to the exchange and handling of information in a network, and they all encompass greater functional scope than the comparatively straightforward matter of data transmission provided by the network itself. In fact, a general purpose network architecture is capable of employing a number of different transmission techniques as required: a local network here, a wide area network there, along with satellite links, public data networks, etc.

An analogy might be drawn between network architectures and city planning. At first glance, a "city plan" could be thought of as a map of the city's streets. But in order to have a completely integrated city plan, provisions must be made for various city services, the delivery of utilities, the extent of land use restrictions, the establishment of public activities such as education and recreation, and the integration of various transportation systems, of which the city street system is an important part, but only a part. The city streets are analogous to a local area network and the overall city plan is analogous to the network architecture.

Long-range planning

As is the case with a comprehensive city plan, a good network architecture is conceived with the understanding that not all eventual uses of the architecture can be anticipated in advance. But it accommodates known elements successfully, is faithful to long-range goals, and provides for future growth and change in ways that do not require the architects periodically to return to their drawing boards to start over.

The network architecture underlying Xerox' focus on integrated systems is called Xerox Network Systems (XNS).

XNS provides a conceptual framework for accomplishing all the functions required in a general-purpose information system.

Like the city plan that uses city streets as one of its building blocks, XNS uses Ethernet (along with other networking techniques); but XNS is not synonymous with Ethernet. Indeed, XNS imposes extensive and stringent requirements on those products designed according to its rules, requirements far more elaborate than those applicable to devices that simply connect to Ethernet.

Integrated office systems

Following the rules and specifications of XNS, it is possible to design highly integrated information systems, using hardware and software elements designed by different groups using different technologies. XNS brings to this process a broad range of facilities and functions, tested in a variety of implementations over a long development period. This makes XNS one of the most thorough and robust of the extant network architectures, particularly among those intended for commercial, office-oriented applications.

XNS is one of the major reasons why Xerox products are as capable, reliable, user-friendly, and obsolescence-resistant as they are. These qualities are specific objectives of the architecture design and reflect directly on the products that the architecture supports. Office products not designed within a framework such as XNS provides will likely be lacking in one or more of these important attributes.

Network architectures and distributed systems

A network architecture provides the conceptual framework for the design of the many functions that are necessary for network elements to work together. It also provides a series of specifications for the common functions that must be agreed upon by the network community. These functions usually take the form of protocols that provide for accomplishing specific tasks such as the transfer or storage of information. In such instances the network architecture can be thought of informally as a collection of protocols, each of which is usually identified with the service to which it corresponds.

Traditional architectures are centralized

In addition to XNS, a number of other network architectures have emerged within the computer and office equipment industries. Two examples are IBM's Systems Network Architecture, which is oriented to the interconnection of mainframe computers and terminals, and Digital Equipment Corporation's Digital Network Architecture, which supports the interconnection of that company's small- to medium-sized computers.

An important distinction exists between XNS and the typical computer-manufacturer network architecture. To one extent or another, the latter tend to be hierarchical in organization, intended for applications where one or more computers dominate the resulting information system and its users. In such systems other elements—satellite processors, terminals, workstations, etc.—are clearly subordinate to the large computers. In various ways the network architectures underlying these systems are designed to create and reinforce this relationship.

There is a place for such arrangements. Traditional mainframe computing is usually organized in such a way that information naturally flows to and from major computing centers. The operation of other system elements is intended to support the processes taking place in the centralized machines. In many respects this view of computing reflects traditional data processing implementation.

Centralized systems lack flexibility

Many networking applications, however, are not well served by this model. In particular, many of the processes and activities in modern offices are essentially autonomous, i.e., initiated and conducted by individuals at their own pace, with only occasional references to external resources. This is particularly true with the creation, editing, storage, retrieval, and printing of documents, the common currency of the office. In most cases document management requires dealing with a large series of autonomous processes, for which the centralized mainframe-oriented model of information flow and processing is often not very effective and a potential bottleneck.

XNS architecture is distributed

For this reason XNS—and network architectures similar to it—are designed to support autonomous processes, implemented by distributed, rather than centralized, processors. One of the important results of this is that XNS makes it possible for networks to grow incrementally through replication of the individual system elements and data bases. XNS is a distributed/replicative network architecture.

A more complete discussion of network architectures and the relationship of XNS to other architectures and industry standards is provided in the Appendices.

Realization of Xerox Network Systems

This manual describes the architecture of Xerox Network Systems. It provides information on the standards and protocols that comprise the architecture. Detail specifications and specific hardware and software products are not described, but a description of the services is provided to illustrate how the protocols are used and how the network architecture integrates products to form systems. Fig. 1-1 shows the relationship between the XNS architecture, standards, and products.

The architecture is at the top of the hierarchy. It establishes the general structure and functioning of the network. The specifics

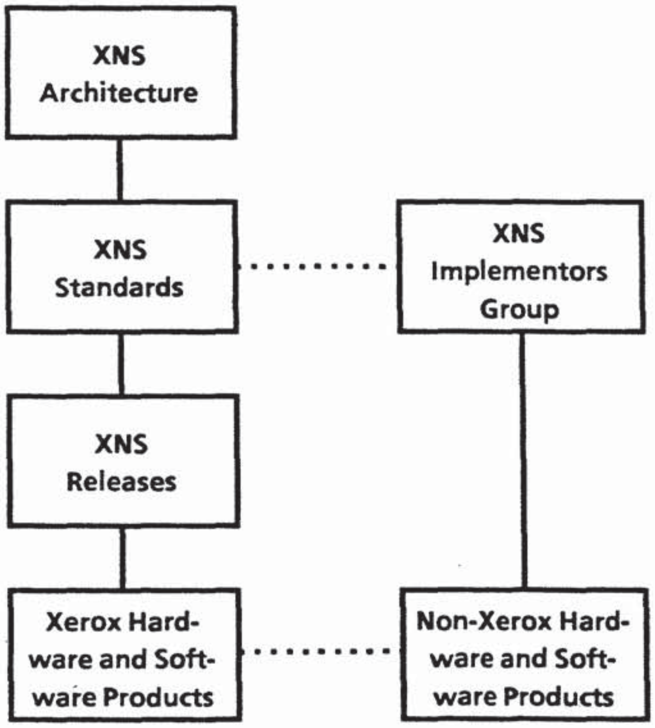

XNS Architecture
XNS Standards
XNS Implementors Group
XNS Releases
Xerox Hardware and Software Products
Non-Xerox Hardware and Software Products

Figure 1-1 **Structural relationship of architecture, standards, and products**

of the architecture are contained in the various standards documents. The Xerox Network Systems releases specify which Xerox hardware and software products conform to the standards, and their level of mutual integration.

XNS architecture is open

There is a well-defined path for other companies who choose to offer products in conformance with XNS standards (see Fig. 1-1). Xerox offers a variety of help to companies wishing to adopt any or all of the XNS standards. This help includes publication of standards and guides, assistance in implementation through the XNS Implementors Group and the XNS Institute, and a variety of joint intercompany arrangements. For further information please contact:

Xerox Corporation
Xerox Network Systems Institute
2300 Geng Road
Palo Alto, Ca 94303
(415)-496-6088

2. XEROX NETWORK SYSTEMS OVERVIEW

Xerox' goal in office automation

Xerox' goal in office automation is to provide its customers with products, systems, and services to maximize return on their information assets. Information assets come in two forms: people and hardware/software products. Increasing the return on information worker assets is increasing effectiveness; increasing the return on information product assets is increasing efficiency. Fig. 2-1 graphically displays the concept of return-on-information assets (ROIA). The boxes at the bottom of the diagram show some of the ways in which the return can be increased and the assets decreased.

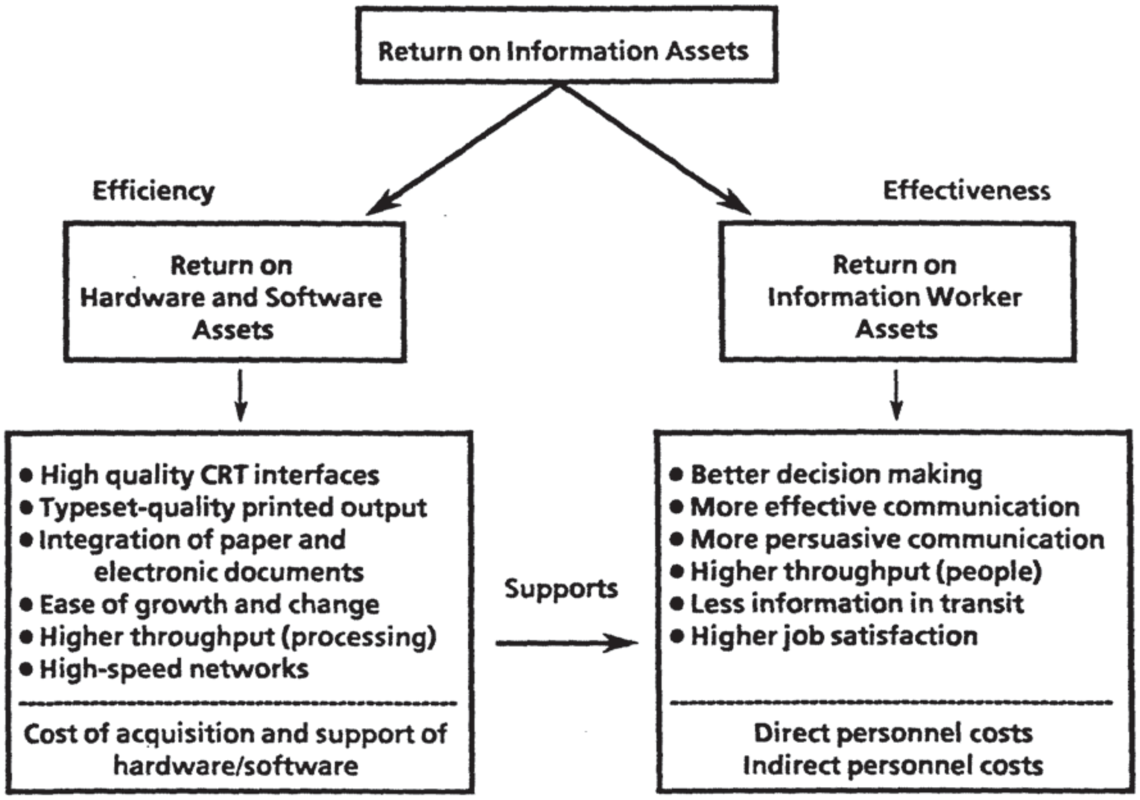

Return on Information Assets
Efficiency
Effectiveness
Return on Hardware and Software Assets
Return on Information Worker Assets
• High quality CRT interfaces
• Typeset-quality printed output
• Integration of paper and electronic documents
• Ease of growth and change
• Higher throughput (processing)
• High-speed networks
Cost of acquisition and support of hardware/software
Supports
• Better decision making
• More effective communication
• More persuasive communication
• Higher throughput (people)
• Less information in transit
• Higher job satisfaction
Direct personnel costs
Indirect personnel costs

Figure 2-1 **Return on information assets**

Typeset quality

Consider an electronic printer which can provide typeset quality directly. Not only can documents be printed on fewer pieces of paper, thereby reducing material, storage, and delivery costs, but higher-quality documents can support more effective communication and decision making.

People come together to accomplish information related tasks in offices because these tasks are generally highly interactive. The means of interaction is usually either through verbal means, including physical cues, or through the creation and reading of documents.

Documents and document management

A clear understanding of the concept of a document is essential to understanding the objectives of XNS. A document is a structured organization of information designed to communicate effectively with people. A document may be rendered by a printer, which produces a paper document from an electronic original, or it may be rendered by a workstation, which makes an electronic document visible. Alternate forms of documents include voice documents and video records. In the future it will be possible to integrate these various document forms; for example, a text document may be annotated by digitized voice.

As most documents are stored at one time or another, it is easy to confuse documents with stored information. Not all stored information is in document form. A computer data base, for example, is not a document since it is organized to provide efficient storage and access to elements of information, not to communicate with people.

The general objectives of XNS is therefore to increase the ROIA by facilitating the creation, capture, storage, communication, printing, and replicating of electronic or paper documents within the office, especially at the work group and departmental levels. This is what Xerox calls document management.

Xerox Network Systems objectives

Systems that fulfill specific short-range objectives are always possible, and may appear to have a lower initial cost. The real test of a system's quality, however, is if it can be used over a long period as applications change and new technology is introduced. In the long run, systems that lack proper architectural support are seldom the most cost effective. They become obsolete in a short time and have to be replaced, often resulting in costly disruption for users. Xerox Network Systems Architecture, with its long-range view, is for those users and suppliers who want to do it right the first time.

The XNS objectives include:

Formal definition of standards

The creators of XNS knew that this architecture would be used as a basis for the design of a wide variety of products. The architecture, therefore, would have to be powerful and open-ended. The products supported by the architecture would be designed by a number of Xerox and non-Xerox organizations.

Some of these organizations would be as distantly removed from the group responsible for the architecture as if they were outside of Xerox. For this reason, the architecture had to be formally defined and its individual elements subject to strict rules of standardization and configuration control.

Document compatibility

One of the key issues in the design of an information system is the preservation of compatibility for the users' work products (documents of all kinds). Once a document is created, it must be capable of being operated upon at some point in the future and at another part of the system (even in another part of the world where a different language is used). If at all possible, architectural changes that "dead-end" already-created documents must be avoided.

Product performance

The architecture must be designed so the products it supports can be engineered in accordance with it. Using proven technology, the products must still meet their performance, reliability, and cost goals. This objective separates the theoretical from the practical as each architectural provision must be implemented with real hardware and software at a competitive cost. An appreciation of this must influence architectural design work.

Product evolution

The architecture must make it possible for products and services to evolve in two directions: toward greater breadth (new areas of applications, user categories, operating environments, etc.), and toward greater depth (new functions).

Interconnection with other systems

Xerox systems must interface and interact with computer products and specialized systems from a large number of suppliers. Because Xerox' autonomous distributed/replicative system must be closely integrated with hierarchical, mainframe-oriented systems, some interfaces become very complex and create profound challenges for the architectural designer. Despite the difficulty, these challenges have been met. The end user and the industry are not well served by approaches that omit interconnection with external mainframe systems.

Industry standards

Xerox has a strong commitment to the development and use of industry standards. Xerox personnel have participated actively in standardization efforts sponsored by groups such as IEEE, ANSI, ISO, ECMA, and CCITT. XNS architecture will either adopt or be compatible with all the important relevant standards. Many aspects of XNS have yet to be the subject of external standardization since in many cases Xerox' work on XNS tends to lead official standards formulation and adoption processes by several years. Nevertheless, Xerox anticipates making XNS-derived contributions to such efforts when they are finally undertaken just as it has with past and present standards projects. A summary of the XNS relationship with key international standards appears in appendix C.

Qualities of Xerox Network Systems

The layers and functions of XNS bring together an important set of qualities on behalf of the products they support. Together these qualities ensure that a system designed with the support of XNS will be among the most powerful, cost-effective, and obsolescence-resistant systems available.

The XNS qualities include:

Maturity XNS is more than just a theoretical construct; Xerox has been working on it for over a decade. Thousands of man-years of effort have gone into its design, implementation, testing, and refinement. It is a practical system with well-established performance and functional characteristics. XNS is a key support element in Xerox' distributed information systems.

Distributive/replicative Xerox has designed XNS to support distributed processing in which autonomous devices are interconnected in a network that permits simple, low-cost, incremental expansion. Because it is easy to replicate workstations, file and print servers, and other network resources as a user's needs grow, a user will be able to benefit from the system for a long time—unlike mainframe-centered architectures.

Completeness XNS is one of the most complete architectures available. It provides two separate information exchange techniques: one for moving data and the other for invoking (and responding to) remote processes. It also provides a broad spectrum of self-contained application processes. Including these processes within the architecture rather than leaving them for user development means that an XNS-based system is more immediately useful to the user and ensures uniformity and robustness in all the important network functions.

Growth and expandability XNS is not a static architecture. One of its strengths is that new facilities can continue to be added without major disruption to the product hardware and software it supports.

Transparency to the user Despite its many functions and its potentially worldwide scope, XNS is transparent to the user. Information is exchanged, resources are added to and subtracted from the network, remote procedures are invoked, and all activities defined by the architecture take place while the user remains free to do his job without concern for what is making that job possible.

Global scope Whether the physical scope of a user's network is limited to a single building or whether it spans the world, XNS provides a single, consistent set of services that automatically adjust to the network's scope. Each user sees one logical network, even though that network may consist of many physical local area networks interconnected by a web of wide area telecommunication circuits.

An "open" system

XNS systems are multivendor systems. As new, specialized hardware and software are introduced for use in office applications, these facilities will be capable of integration within an XNS-based system. Xerox began this process in 1981 with the public disclosure of the non-application layers of the XNS architecture. The application layers are also being fully disclosed as individual applications reach technical maturity.

Xerox Network Systems concepts and facilities

The basic idea behind most network architectures, including XNS, is that of a layered structure. This means that the various functions supported by the architecture are divided into a series of layers. By convention, the most primitive tasks are located in the lowest numbered layers while the higher layers are reserved for more sophisticated tasks.

ISO Open Systems Interconnection Reference Model

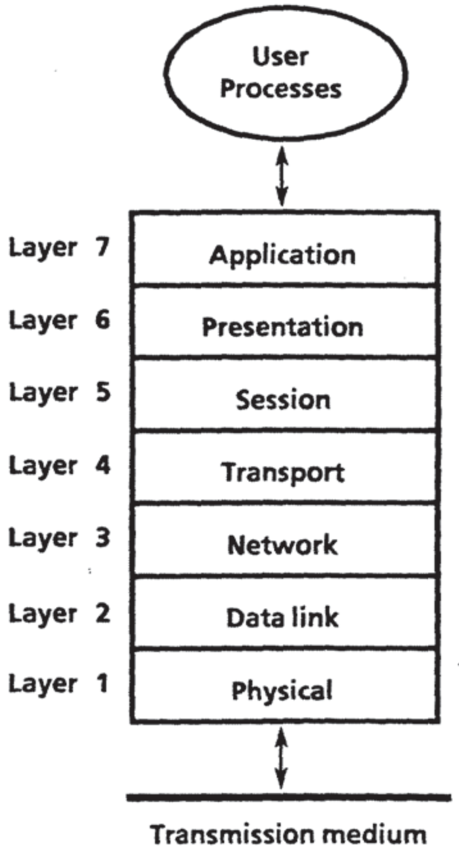


Figure 2-2 Layers in the ISO Model

A very useful model of this layered structure was developed by the International Organization for Standardization (ISO). The model is called the "Open Systems Interconnection (OSI) Reference Model," referred to as the "ISO Model" in this document. It was adopted in 1981 under the sponsorship of ISO's subcommittee 16. One measure of the great need for this clarifying set of definitions has been the widespread, adoption of the ISO Model concepts, even among people who are not professional network architects. Although lacking sufficient detail to be a standard, the ISO Model provides a common basis for discussing a complicated subject.

Figure 2-2 shows the essence of the ISO Model, which holds that all network architectures, no matter how complex or broad-guaged, can be divided into seven functional layers. Each layer is superior to the one below it, and subordinate to the one above it, in the sense of the relative primitiveness or sophistication of its function. This sense of hierarchy is an important feature of the ISO Model; a commonly-accepted perspective is that the functions of one layer use the resources of the lower layers to complete their assigned tasks. Of course, the entire hierarchy exists in the final analysis to serve a set of user processes. Because of the hierarchical nature of this model, the lower layers typically deal with ordinary data communication matters, while the upper layers deal with broader issues of information system control, management, and applications.

The individual layers in the ISO Model, and the roles they play in a network architecture, include:

Layer 1: Physical	All network architectures provide for data transmission. In this layer the fundamental tasks related to transmission take place, such as bit stream manipulations, dialing (for switched networks), modem control, etc.
Layer 2: Data Link	This layer is basically responsible for getting information reliably across a data link. Information is organized for purposes of transmission (outgoing) and reorganized for processing (incoming); transmission errors are detected (and sometimes corrected), and the rate of data flow through the link is controlled.
Layer 3: Network	Fundamentally this layer is responsible for getting units of information across the network. It provides data organization functions for information moving on and off the transmission media; it also includes higher level forms of error recovery, and most of the control of network addressing, routing and switching decisions.
Layer 4: Transport	In this layer decisions are made concerning which transmission service is appropriate, more data organization and reorganization functions are provided, and a set of network management tasks are performed.
Layer 5: Session	This layer is responsible for creating a communicating relationship between two parties for the time necessary to complete their interaction (a "session"). Related tasks such as buffering and queuing are also performed within this layer.
Layer 6: Presentation	Data conversion and similar tasks are performed here. The object is to translate data forms from those understood by user applications into those used in lower layers (or remote user processes), and vice versa. This can range from simple matters of code and format conversion, to complex matters of syntax changes (as in the extreme case of shifting between two different programming languages).
Layer 7: Application	Specific applications are performed here; for example, in a document-oriented environment, this layer might include filing, printing, mailing, etc.

The ISO Model also helps to explain interfaces and protocols. Fig. 2-3 shows the ISO Model expanded to include two separate nodes. (A node is any logically or physically distinguishable entity on the network, such as a terminal, a computer, or a program running inside a computer.) The ISO Model assumes that the two user processes have need to communicate (e.g., one user wishes to transfer a spreadsheet file to another user), or that one user process has need to use resources in the other node (e.g., the user wishes to print a document on a remote printer).

Interface between layers

A formal interface is conceived to exist between each layer of the ISO Model. That interface is the "logical view" from a higher layer to the next lower layer, and vice versa. Information related to actions wanted and results obtained, as well as actual data, is passed back and forth across these interfaces.

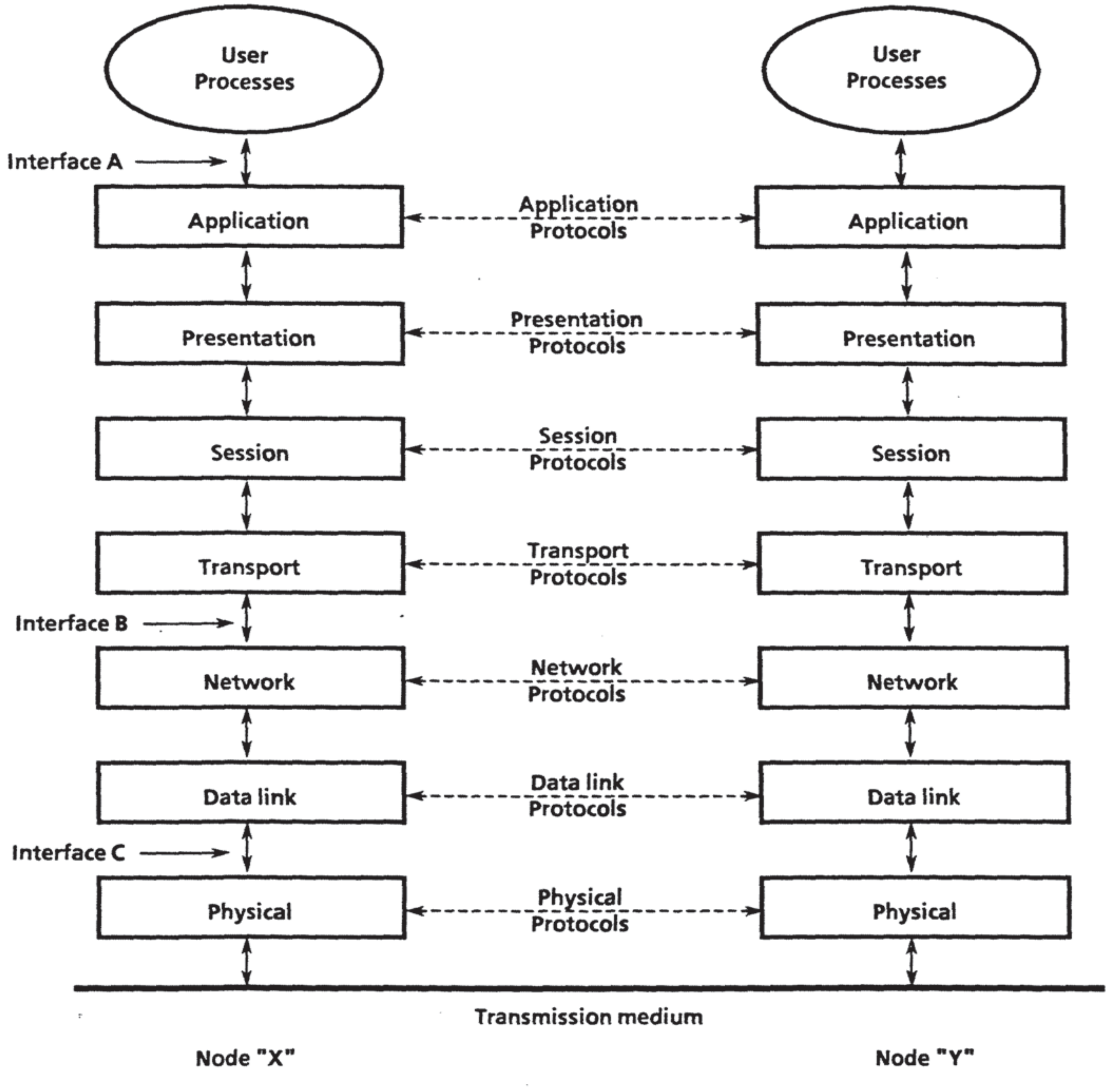


Figure 2-3 **Protocols and interfaces in the ISO Model**

A number of the ISO Model interfaces have counterparts in real networking systems. In Fig. 2-3, Interface A is the interface the user processes see to the entire network architecture. An example of Interface B is the widely discussed X.25 packet switching interface. An example of Interface C is the popular EIA RS-232-C data communication interface.

Protocols for each layer

When communication is taking place between two nodes, the work done between these nodes is conceived to be structured into a set of protocols. A protocol is an organized set of rules for getting work done. In the ISO Model each layer in one node is conceived to be working with the corresponding layer in the other node, by means of peer protocols. In other words, a given protocol represents a dialogue between two equally potent functions, each of which operates through its upper and lower interfaces to perform its specific tasks.

The ISO Model is a "guide," not a "specification." For various practical reasons, real network architectures do not necessarily formally define all the interfaces and protocols suggested by Fig. 2-3. A real architecture might have several different application protocols, no presentation protocols, and so on. Nevertheless, the basic themes of layered structure, interfaces, and protocols are important to XNS, as well as to nearly all network architecture design.

XNS structural overview

Fig. 2-4 shows the basic structure of Xerox Network Systems. As in the ISO Model, the XNS structure is organized into a series of layers, approximately corresponding to the ISO Model layers listed on the left side of the illustration.

XNS layers

The XNS architecture groups some of the ISO Model functions into fewer layers for convenience. Each XNS layer corresponds functionally to the ISO layers. Even when the functions are grouped, as in the case of Ethernet, a separation is maintained between the physical and data link layers to allow different physical media to be used with the same Ethernet data link protocol. The XNS architecture is particularly open-ended in respect to multiple transmission protocols corresponding to different types of communication services, and to multiple application protocols corresponding to different functions performed within the architectural boundaries.

As in the ISO Model, the XNS bottom layer provides for physical transmission interfaces. The functions provided by the XNS architecture rise in hierarchical order—through communication control (Internet) and remote procedures (Courier)—to the application layer. Although the lower layers are important, the full functional richness of XNS is revealed by an examination of the application layer. This layer provides a great variety of important office and computing functions to XNS users.

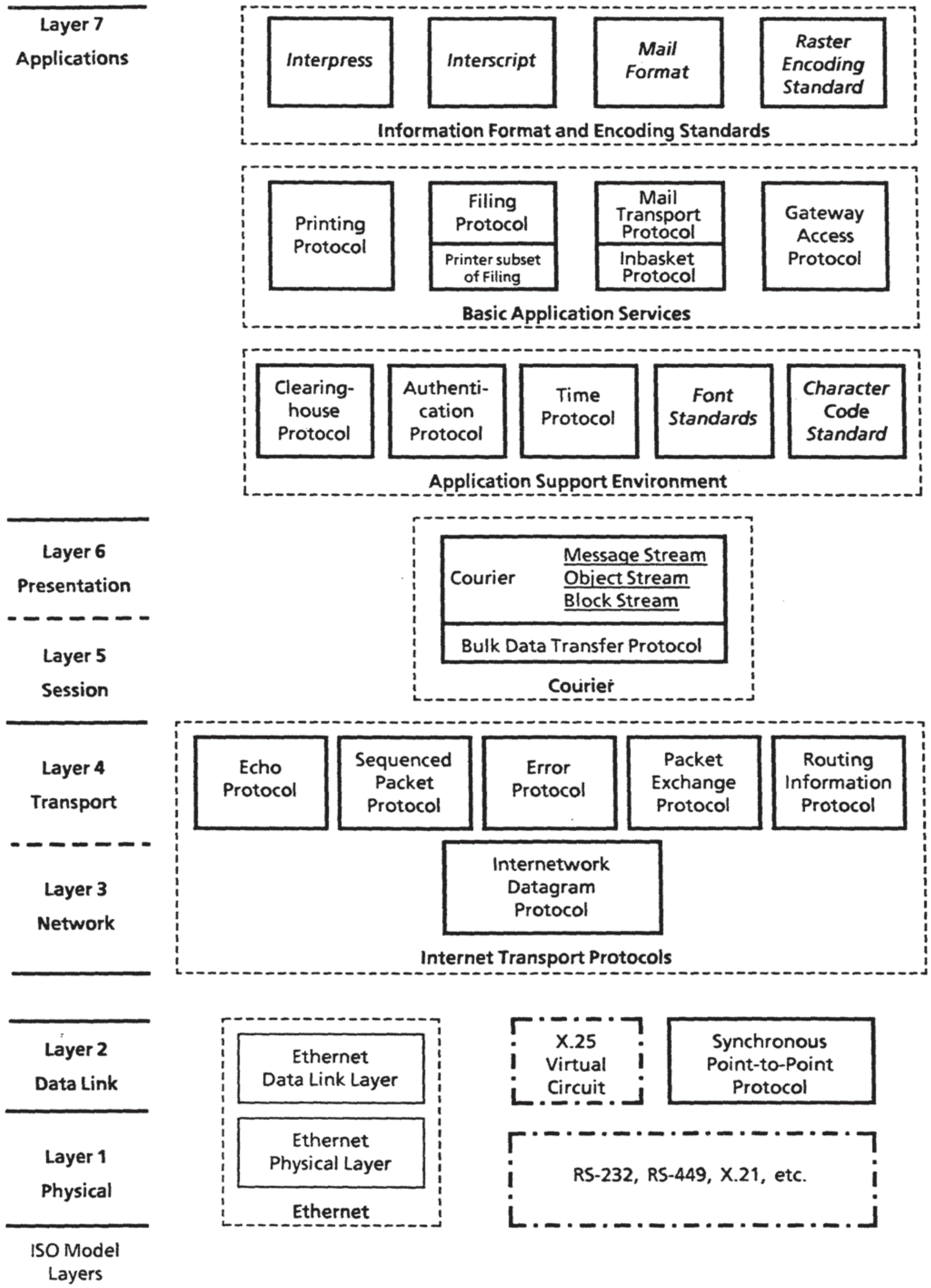


Figure 2-4 **Overview of Xerox Network Systems**

The following explanation applies to Fig. 2-4:

Physical interfaces

At the lowest layer, Ethernet provides its own unique physical interface. It is unlike traditional data communication physical interfaces, which are shown in the box to the right (RS-232-C, RS-449, X.21, etc.). These are shown in a broken outline because, strictly speaking, they are part of XNS by adoption rather than by special design.

Data link protocols

At the next lowest layer, Fig. 2-4 shows the CCITT X.25 Virtual Circuit Protocol in a dashed box to indicate that this protocol is part of XNS by adoption. It is used as part of XNS utilization of packet-switching data networks.

Internet transport protocols

Internet is shown as a set of protocols corresponding to ISO Model layers 3 and 4. The word "internet" is also used to refer to the set of all interconnected Ethernets in different locations, a relationship implemented by these protocols.

Courier

XNS implements the session and presentation layers in Courier, the XNS protocol for remote procedure calls [requests].

Application protocols

At the application layer (ISO Model layer 7) the Application Support Environment provides support resources called on by users and/or the application protocols shown immediately above. These protocols—mailing, printing, filing, and gateway access—are implemented in hardware/software to provide the XNS application services.

Document formats

Within the application layer, the standards for the format or language for the encoding of document form or content are labeled with italic type. In many respects, the utility of XNS depends as much on the innovative approach to document descriptions as it does on the actual protocols. The document encoding techniques referred to in Fig. 2-4—particularly Interscript and Interpress—make it possible for XNS documents to be edited, printed, or communicated anywhere on the system. Other encoding standards are the Character Code Standard for representing text in many languages, and the Raster Encoding Standard for representing compressed and uncompressed bitmap images.

Integration and compatibility

The internal structure of XNS enables effective integration between individual hardware and software elements within XNS-compatible products. Techniques are also provided within XNS for bi-directional protocol and format conversion, permitting other systems to achieve integration with XNS.

Note

Each architecture element shown in Fig. 2-4 is discussed in the following sections, and appendix D gives examples of protocol usage. Further information is contained in documents listed in the annotated bibliography in appendix F.

Network devices and terminology

The XNS architecture consists of a hierarchy of protocols and related formats and encodings. The different network devices connected to a network can communicate with each other if they use the same protocols. The devices that use XNS protocols and connect to the network are called system elements. They are also known as XNS hosts or XNS citizens. These system elements are generally classified as workstations or servers.

Workstations and terminals

Fig. 2-5 illustrates some key system elements. Workstations are devices with which a human operator directly interacts. The term "workstation" is loosely used in the industry to mean various things. In XNS a "workstation" refers to any system element used directly by a person. This would include the Xerox Star, the Xerox 860 word processor, and Xerox and IBM Personal Computers (when these are connected to the network via XNS protocols).

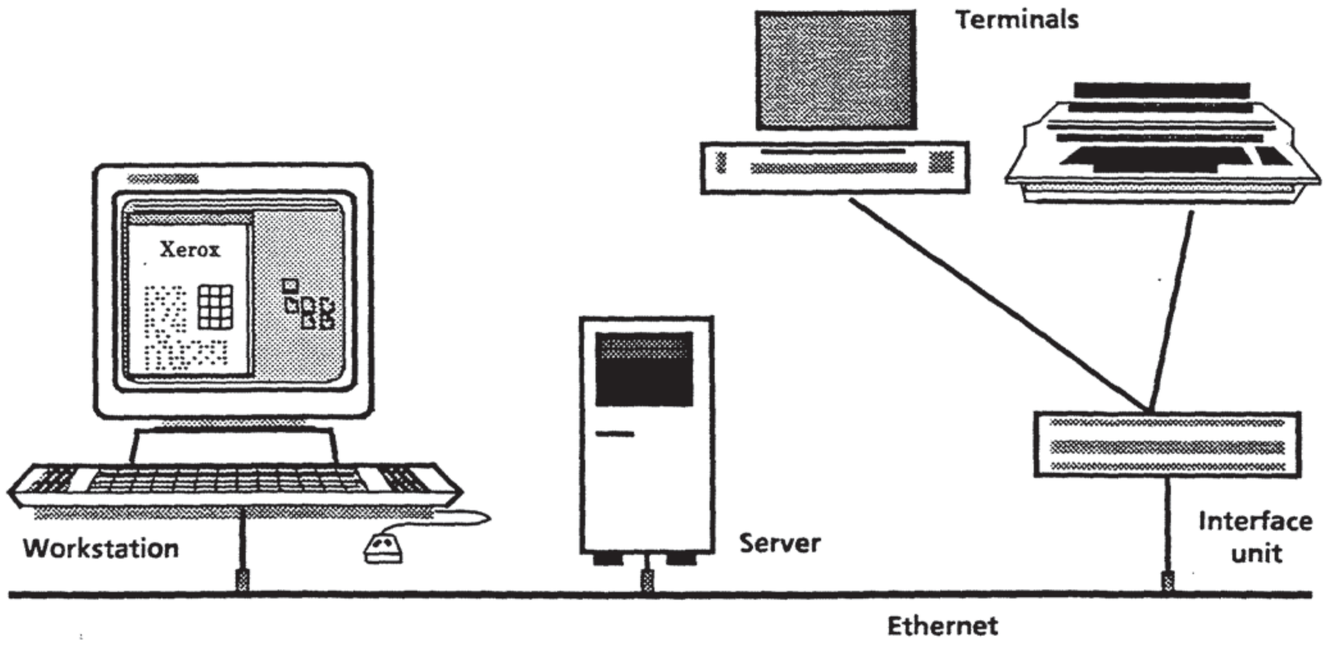


Figure 2-5 **Network devices**

Any product indirectly connected to the network using protocols other than XNS is considered a terminal. Terminals may or may not be intended for direct human interaction; a conventional ASCII keyboard/display unit would be but an ordinary minicomputer may not be. Both would be considered terminals if they were *indirectly* interfaced to the network. Other examples of terminals include the Xerox electronic typewriter or personal computers without a direct XNS connection.

The significance of direct XNS connection is that ordinarily a directly-connected device is expected to implement all the layers of XNS appropriate to its function, which would include *at least* all the layers upward through Courier (see Fig. 2-4), plus selected application protocols. These XNS system elements are assigned a unique host number identification.

This is not necessarily true for terminals which are interfaced to the XNS community by means of interface units such as the Interactive Terminal Service. Through the joint action of the interface unit and the terminal, the XNS protocol rules are obeyed, but the terminal itself is not recognized as a system element and does not have a host number identification.

Servers and services

A server is a device connected to Ethernet whose purpose is to provide a service to network users. The services are high-level functional activities such as filing, printing, mailing, and external communications. They represent high-level applications performed within the architecture. The XNS services implement various application-layer protocols, as depicted in Fig. 2-4. The services are typically collections of software acting *according to the rules of the architecture and its protocols*, to achieve the desired purpose. A server, therefore, is the physical means by which a service is performed.

The word "server" is used in several different ways, depending on the circumstances. Some servers are specifically designed to perform their assigned function, and little else. These dedicated servers tend to consist of the necessary computer peripheral related to the service—a printing subsystem, for example, or a large disk file—coupled to the network by means of an electronic subsystem.

Some servers are actually general-purpose minicomputers programmed to perform the requisite functions; file servers are often implemented this way. Such servers may be capable of accomplishing many other functions than that which makes them servers.

And finally, certain workstations themselves can also function as servers (given the appropriate peripherals and software), in addition to performing their workstation duties—sometimes alternatively, sometimes simultaneously.

One should therefore think of a "server" as any collection of the necessary electronics, software, and peripheral equipment necessary to deliver a service—regardless of what other role that ensemble may also have.

Clients

Clients are the entities which request the performance of a service. Since the XNS services implement application-layer protocols, an appropriate model is that the client uses the application-layer protocol to request the service.

An analogy for the relationship between a client and a service can be found in a human operator using a workstation and possibly other physical network resources to cause some action to take place (e.g., the printing of a document). Since the human cannot directly request the service, he or she works through the user interface of the workstation to cause the desired result. In a similar fashion, client programs use network resources, including other programs, to invoke services.

Services themselves may be clients of other services (e.g., when a filing service asks a Clearinghouse service for information on behalf of a client).

The discussion in this manual focuses on services, and not on the workstations and terminals or their user interface. This is appropriate because workstations are the clients of the described services, and because the user interface to the services varies among the many types of workstations supported in XNS.

User interface

The user interface on Xerox professional workstations provides an easy way for anyone to create, file, mail, or print information. Documents may contain text in many different type styles and sizes, plus charts and drawings integrated with the text. Documents appear as they will be printed ("what you see is what you get," or "WYSIWYG").

Most system options appear on "pop-up" menus and can be activated by merely pointing at an option and clicking a button on a device called the "mouse". Available network resources are shown as pictorial images on the screen (called icons). The icons in Fig. 2-6 show an "in" and "out" basket for mail, different printers, a [remote] file drawer, several documents, and folders containing any number of documents. A user interface like this makes services on the network readily accessible. User actions are intuitive, always consistent, and extremely friendly.

Multilingual capability

The Xerox professional workstations provide many different versions for the user interface, including icons, keyboards, and data formats to accomodate the needs of a worldwide user community.

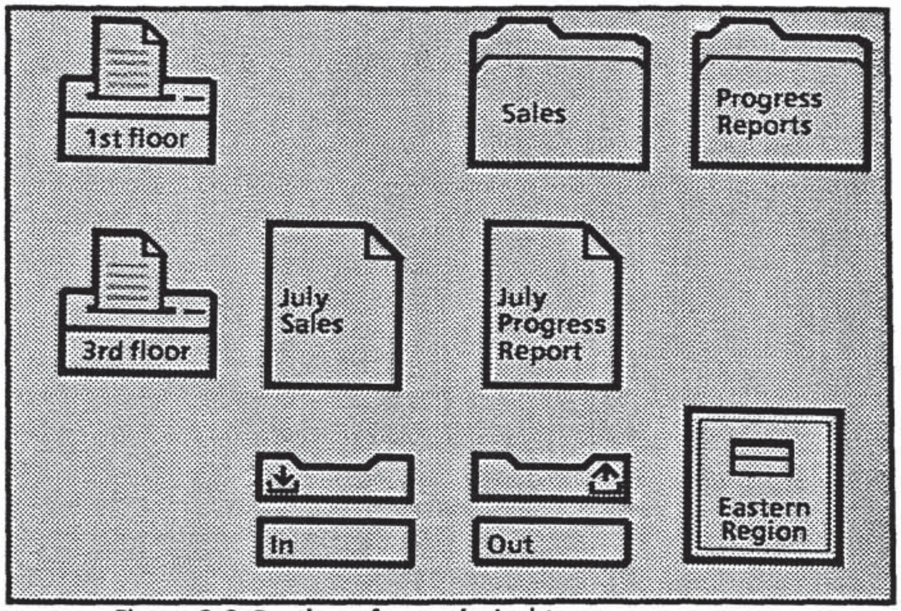


Figure 2-6 **Portion of a user's desktop**

Names

As is the case with most modern network architectures, Xerox Network Systems employs a technique in which certain network resources are "named," that is, identified with labels that relate directly to their characteristics or purpose. This may be thought of as a technique for indirect addressing. The resource is known by its name rather than by its literal or physical address or other location identification. That makes gaining access to the resource substantially easier. The access to system resources will be further explained in the discussion of Clearinghouse. XNS also provides methodologies for naming key system resources (e.g., the fonts used in printing documents).

3. COMMUNICATIONS

The XNS communication facilities are responsible for moving information between network system elements. Owing to the nature of offices (relatively short distance between elements), much of the "traffic" in an XNS network is routinely transmitted across the Ethernet "baseband" local area network. There are other forms of local area networks, however, and XNS is capable of operation over most of them, including multi-channel "broadband" networks. Beyond the relatively short range of such networks, alternative wide-area transmission techniques must be used like conventional telephone circuits (switched or leased), wideband circuits, and public data networks.

Each of these types of service requires specialized data link protocols, such as the Ethernet data link and the Synchronous Point-to-Point protocols. It is one of the strengths of Xerox Network Systems that in most cases new classes of communication services can be accommodated through the addition of new transmission protocol modules ***without disturbing the protocols in the higher layers.***

In addition to basic data link protocols, XNS provides higher level communication control functions embodied in the internet. An implementation of those control functions for purposes of interconnecting remote Ethernets is made by the Internetwork Routing Service which is an integral part of XNS communication facilities.

Ethernet

Ethernet is the local area communication network (or LAN) developed by Xerox, Digital Equipment Corporation, and Intel Corporation. Its specifications determine the kind of transmission medium (coaxial cable), electrical signaling levels, and the data link or transmission protocols.

The Ethernet transmission protocols are independent of the medium. Products supporting these protocols may communicate with each other using fiber optics, twisted pairs of wires, or even radio broadcasting into the "ether." With passage of the IEEE 802.3 standard, Ethernet is now an internationally accepted communication standard. There are some minor differences between the version of Ethernet that Xerox has been using for several years and that adopted recently bv the

IEEE 802 Committee and ISO. Now that this standard has been accepted, Xerox is migrating its products to the official 802.3 version of Ethernet, a process that started in 1985 with the incorporation of standard 802.3 transceivers and controllers into the product line.

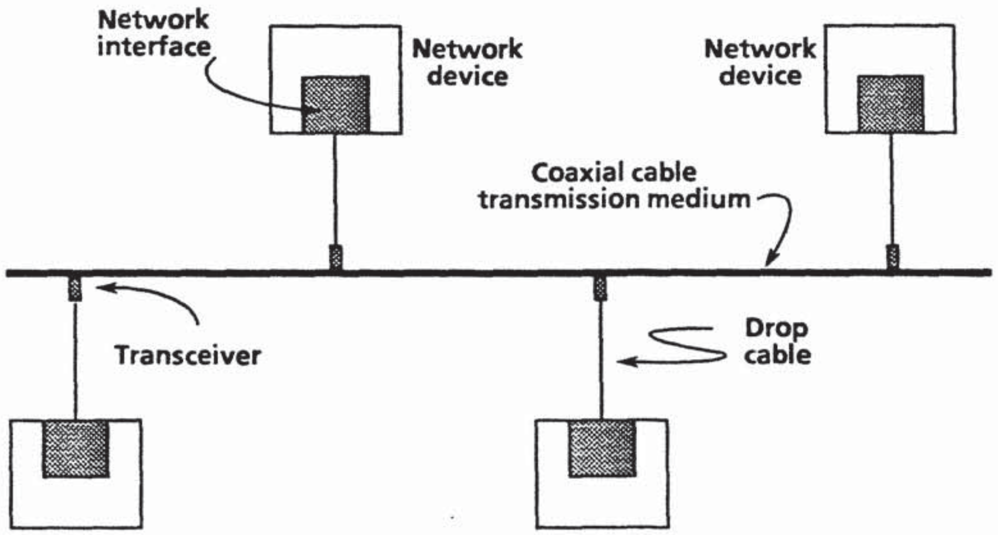


Figure 3-1 **A local area network**

Ethernet components

Fig. 3-1 shows the hardware components required for a baseband Ethernet using coaxial cable. The cable itself is passive; no central resource (power, electronics, computer) is required to allow individual system elements to communicate with each other. System elements attach to the cable via a tap, control electronics (transceiver) located near the cable, an extension drop-cable, and further electronics (network interface controller) located in the system element (network device). The Ethernet cable may be several thousand feet long and, with the use of repeating devices (which strengthen signals on the cable), is normally capable of extending through most facilities.

Network devices may be nearly anything: workstations, terminals, computers, or specialized subsystems of various kinds. The only requirement is that they possess the electrical interface with the cable and the internal logic (usually in the form of software) to interact appropriately with other devices on the cable—in other words, to implement the requirements of the network architecture associated with the network. Other devices (those not possessing the special hardware and software interfaces) can also attach to the networkby means of intermediate adapters.

Ethernet employs coaxial cable as its transmission medium. Data are transferred on the cable at 10,000,000 bits per second (10Mbps), which makes Ethernet one of the highest performance local area networks available. It is the leading example of baseband local area network technology in which computer-type (digital) electrical signaling is applied directly to the medium.

Ethernet benefits

Here are some of the respects in which local area networks differ from conventional data communications:

High-speed communication Ethernet permits the exchange of data at far greater rates than ordinary telephone-based systems (more than a thousandfold faster!). This enables the development of more sophisticated applications and also permits far greater numbers of devices to be supported over a single cable. (As document complexity grows the need for higher data rates becomes more important. For example, the digitally-encoded form of a 10-page report, containing text, graphics, and pictorial elements, might easily require 2.5 million bits; when a system is in full operation by a number of users, the efficient exchange of bodies of information of this size requires multi-megabit transfer rates.)

Distributed control Ethernet is designed so that little, if any, equipment is actually associated with the network itself. This is in contrast to a conventional system in which PABXs, central office switches, "front end," and other communication control systems, concentrators, etc., are all employed simply to control communications. In Ethernet, communication control is the responsibility of the attached devices.

Expandability Ethernet facilitates graceful growth from a network of a few devices to a major network encompassing an entire facility.

Compatibility Ethernet, together with XNS comunication protocols, enforces compatibility among the attached devices.

Maintainability Ethernet can be installed and managed by a user rather than an outside vendor (such as the telephone company). This ensures maximum flexibility in the installation configuration and few obstacles when it becomes necessary to modify the network.

Reliability Ethernet is completely independent of cable amplifiers, frequency converters, and other devices, the failure of which can disrupt a major part of the network.

Fairness Ethernet accommodates the exchange of information by system elements built by different manufacturers. Any system element conforming to Ethernet rules may tap the cable. The conventions for gaining access to the Ethernet provide fairness among all system elements.

The major division in the Ethernet architecture is between the physical layer and the data link layer, corresponding to the lowest two layers in the ISO Model. The interface between the higher network client layer and the data link layer includes facilities for transmitting and receiving frames, and provides status information. The interface between the data link and the physical layer includes signals for framing (carrier sense, transmit initiation) and contention resolution (collision detect), facility for receiving and transmitting serial bitstreams, and a wait function for timing.

Physical layer

This layer performs all the functions needed to transmit and receive data at the physical level while supporting the data link layer interface. The Ethernet specification describes the 10Mbps baseband coaxial system. Other physical layers have also been specified for broadband channels, fiber optic channels, and twisted pair wires. All these diverse physical media work with the same Ethernet data link protocol. It is Xerox' intention to incorporate other physical media into its product line as they are adopted by the standards organizations and are accepted by users. It is important that there be agreement among vendors and users about the physical media that will be installed for local area networks so equipment from different vendors can operate together. This will avoid costly redundant wiring installations. Xerox has been actively supporting such standardization in the standards organizations.

Frame format

Figure 3-2 shows the five fields in an Ethernet data link layer frame; the source and destination address, the type field (called length in IEEE 802.3 standard), a data field containing the transmitted data, and the frame check sequence field, containing a cyclic redundancy check (CRC) value to detect transmission errors. The total length of the Ethernet frames can be 64 to 1518 bytes long. The Ethernet source and destination addresses are 48 bits long, thus uniquely identifying over 281 trillion network devices! Since a typical single Ethernet is limited to 1024 devices, why assign 48 bits when only 16 would suffice? The answer is that the Ethernet is only one component in a large internetwork system. The use of absolute (rather than relative) host numbers in an internetwork provides for reliable and manageable operation as the system grows, as machines move, and as the overall topology changes, if a local network can directly support these large host numbers. The 48-bit host number space is large enough to ensure uniqueness and provides adequate room for growth at little extra cost.

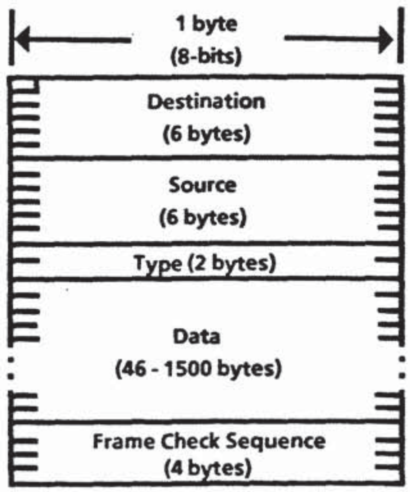


Figure 3-2 Ethernet frame

Data link layer

The data link layer itself is divided into two sublayers. The data encapsulation sublayer performs the framing, addressing, and error detection functions. The link management layer performs the channel allocation and contention resolution functions.

CSMA/CD

The general Ethernet approach uses a shared communications channel managed with a distributed control policy known as *carrier sense multiple access with collision detection*, or CSMA/CD. With this approach, there is no central controller managing access to the channel, and there is no preallocation of time slots or frequency bands. A network device wishing to transmit is said to "contend" for use of the common shared communications channel (sometimes called the Ether) until it "acquires" the channel; once the channel is acquired the device uses it to transmit a packet.

To acquire the channel, devices check whether the network is busy (that is, use carrier sense) and defer transmission of their packet until the Ether is quiet (no other transmissions occurring). When quiet is detected, the deferring device immediately begins to transmit. During transmission, the transmitting device listens for a collision (other transmitters attempting to use the channel simultaneously). In a correctly functioning system, collisions occur only within a short time interval following the start of transmission, since after this interval all devices will detect carrier and defer transmission.

To minimize repeated collisions, each device involved in a collision tries to retransmit at a different time by scheduling the retransmission to take place after a random delay period. In order to achieve channel stability under overload conditions, a controlled retransmission strategy is used whereby the mean of the random retransmission delay is increased as a function of the channel load.

Devices accept packets addressed to them and discard any that are found to be in error. It is impossible, however, to guarantee that all packets transmitted will be delivered successfully. For example, if a receiver is not enabled, an error-free packet addressed to it will not be delivered; higher levels of protocol must detect these situations and retransmit.

Taken together, the physical and data link portions of Ethernet provide the foundation for a comprehensive, sophisticated local area network capability useful in a wide variety of applications and environments.

Although Ethernet plays a central role in XNS, the architecture itself provides for a variety of ways to communicate, some of which are described below. One of the great advantages of a network architecture such as XNS is that it can be adapted to whatever communication facilities are appropriate to the application, usually without requiring major reformatting, alternative high-level protocols, and inefficient procedures.

Synchronous Point-to-Point Protocol

In addition to incorporating a flexible local area networking capability, it is also necessary for a general-purpose information system to be able to communicate over comparatively long distances.

Communication beyond local area

Fig. 3-3 illustrates this point. Consider a user who has several facilities located significant distances from each other: across a city, across a continent, or around the world. Each facility has an Ethernet local area network for communication within the facility, but the actual work done in each facility requires exchanges with the network resources and people in all other facilities. For such purposes conventional data communication techniques must be used: modems, leased or switched telephone lines, and a data link protocol specifically intended for this type of transmission.

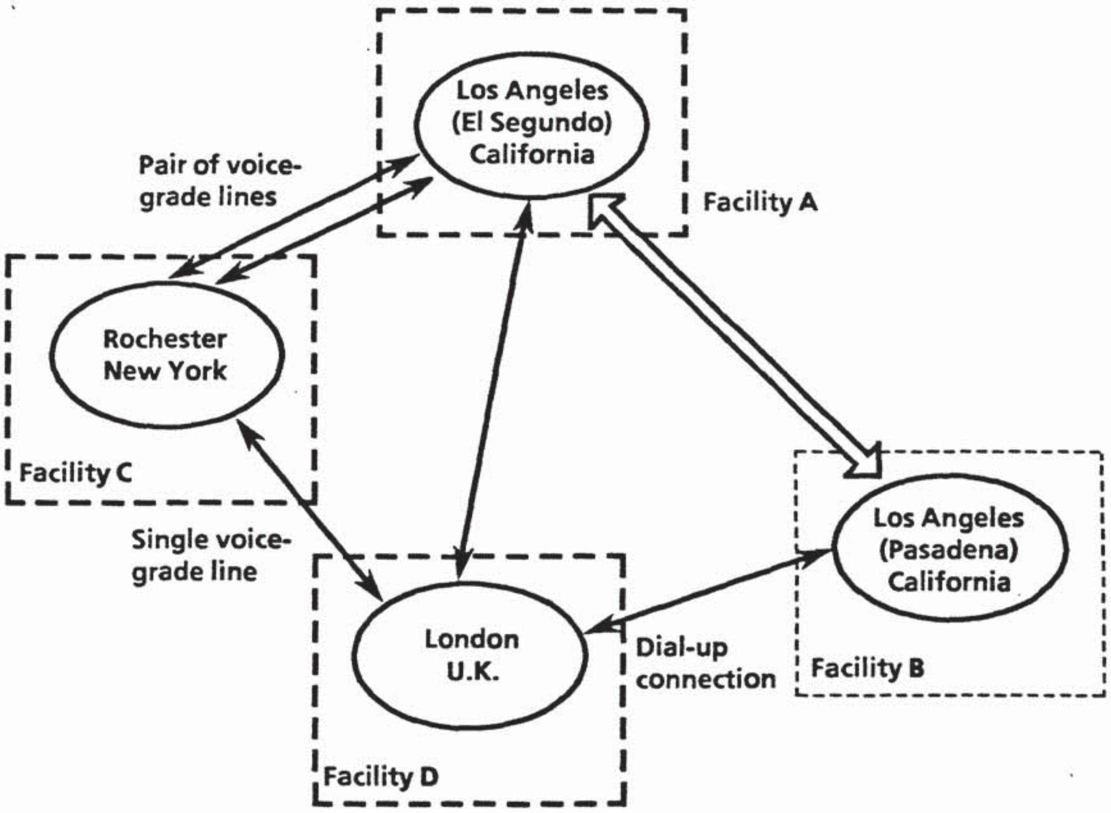


Figure 3-3 Interfacility communication using the Synchronous Point-to-Point Protocol

Fig. 3-3 shows a number of connection possibilities including: a single voice-grade leased line capable of supporting transmission at up to 9600 bps; a pair of such lines, where the data exchange requirement slightly exceeds the capacity of a single line; a switched (dial-up) connection, where only occasional exchanges are required or where leased lines are unavailable; and a wideband leased line, capable of operation at 56K or 230.4K bps, for comparatively high-volume exchanges.

0036

In each case the use of synchronous modems at the end of each communication circuit is implied (when common-carrier digital services are used, special service unit devices are substituted for conventional modems). The modems or service units are interfaced with specific devices connected to each facility's local area network. The interface between these devices and the modem/service units corresponds to the interface between the data link and physical layers in the ISO Model. It is usually implemented according to the familiar provisions of EIA specifications RS-232-C or RS-449, or CCITT Recommendations such as V.4 or V.35.

Bit-synchronous transmisssion

The protocol used in XNS between remote local area networks is the Xerox Synchronous Point-to-Point Protocol, which corresponds to the data link layer of the ISO Model. Its name derives from the fact that it expects to use synchronous (rather than the lower speed asynchronous) transmission, and from the fact that it is intended for operation between two specific points rather than among a number of points (which is usually referred to as multipoint, or multidrop operation).

The Synchronous Point-to-Point Protocol is one of a class of data communication link control protocols that have come to be called bit-oriented. Essentially, this means they are capable of efficiently transmitting digital information without regard to its content, much as the public postal system functions without regard to the content of the billions of envelopes within the system. In this respect, the XNS Synchronous Point-to-Point Protocol resembles IBM's Synchronous Data Link Control (SDLC) protocol, the High-Level Data Link Control (HDLC) international standard, and others.

This XNS protocol is designed to support operations over the variety of transmission schemes illustrated in Fig. 3-3. Among its special features are:

Simplicity

The protocol defines a simple connection between communicating entities, and is easily implemented.

Transparency

In keeping with the concept of a layered architecture, the protocol makes no assumption about data that is transmitted, nor does it constrain the data in any way.

Compatibility and maintainability

A "version negotiation" provision which makes it possible for two communicating stations to agree on which version of the protocol is to be used. This is a particularly useful system maintenance feature.

Error detection

Since telephone circuits are "noisy," the protocol provides an error detection mechanism analogous to that provided with Ethernet (as in the case of Ethernet, however, error *correction* is a responsibility of a higher-level XNS protocol).

Operational flexibility

Both half- and full-duplex operation are supported over both leased and switched circuits; in full-duplex (simultaneous two-way) operation, a balanced style of transmission is provided in which neither station must be relegated to a "master" or "slave" role.

The Synchronous Point-to-Point Protocol is designed to transmit a series of frames, or packets. The organization of these packets is generally consistent with the HDLC international standard. It provides for:

—transmitting the unique XNS-standard 48-bit host address

—encapsulating and decapsulating packets up to maximum of 65,535 bytes in length (although in practice the packet length for most systems would be far less)

—several different formats for control packets, data packets, and private packets

—special sequences marking the beginning and end of a packet

—a frame check sequence, for error detection.

Relationship to HDLC

Like the Ethernet, the Synchronous Point-to-Point Protocol strives for peak performance from a normally errorless medium. When transmission errors do occur, the higher level XNS protocols provide error correction to achieve reliability. In contrast, if full HDLC were used as the point-to-point data link protocol, both HDLC and the higher-level XNS protocols would be incurring the overhead necessary to ensure reliability. The Synchronous Point-to-Point Protocol adopts the framing and error detection scheme of HDLC, but not the error correction. However, within XNS, full HDLC is also supported, as are other external protocols.

Protocol application

As a Xerox standard, the Synchronous Point-to-Point Protocol is applicable to Xerox forwarding system elements (such as internetwork routers), terminal system elements (or remote system elements), cluster system elements, and interfacing elements, as shown in Fig. 3-4. The protocol also defines a two-way alternation mode to manage half-duplex medium contention, so it is applicable to half-duplex as well as full-duplex circuits.

The Internet

To serve large and geographically dispersed office environments, Ethernet's basic communication capability must be augmented in various ways. Interconnecting multiple Ethernets will circumvent the maximum end-to-end cable length restriction, but requires mechanisms for internetwork communication. An internetwork, or internet as it is often

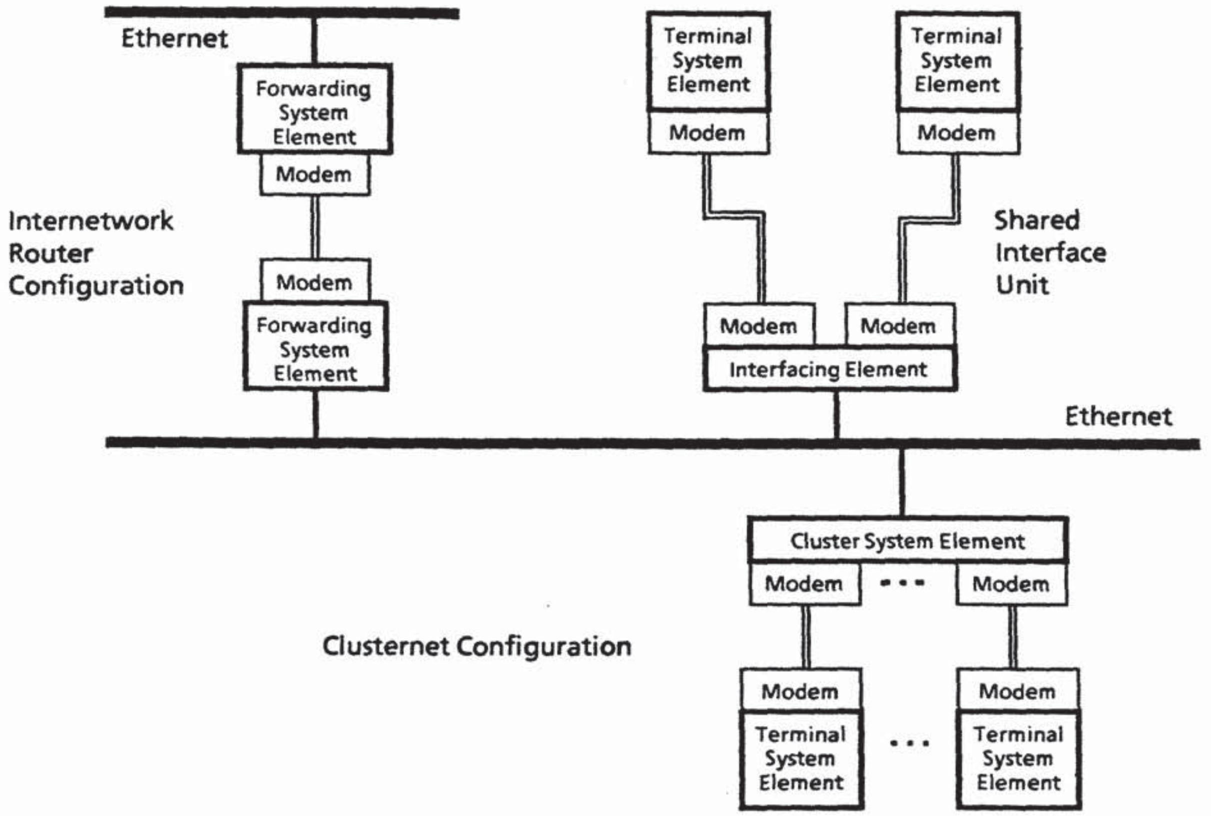


Figure 3-4 **Examples of internetworking configurations**

called, is simply an interconnection of networks. The internet architecture, consisting of the Internet Transport Protocols, provides the network and transport layer functions of the ISO Model.

Internet architecture

The Xerox internet architecture offers a richer addressing scheme and a more sophisticated routing algorithm than available in local area networks. It enables Ethernets to be interconnected by telephone lines, public data networks, or other long-distance transmission media. It also allows the communication system to be reconfigured to satisfy the immediate and future requirements of the user. For example, the Network System may have only one Ethernet initially and then be expanded (without software modification) to contain two or more Ethernets, which are interconnected directly or via other communication media.

Internet Transport Protocols The internet layer consists of a family of Internet Transport Protocols shown in Fig. 3-5. These protocols cause information to move between sources and destinations of information in an organized and reliable manner. The several protocols are:

Internet Datagram Protocol This defines the fundamental unit of information flow within the internet—the internet datagram packet.

0039

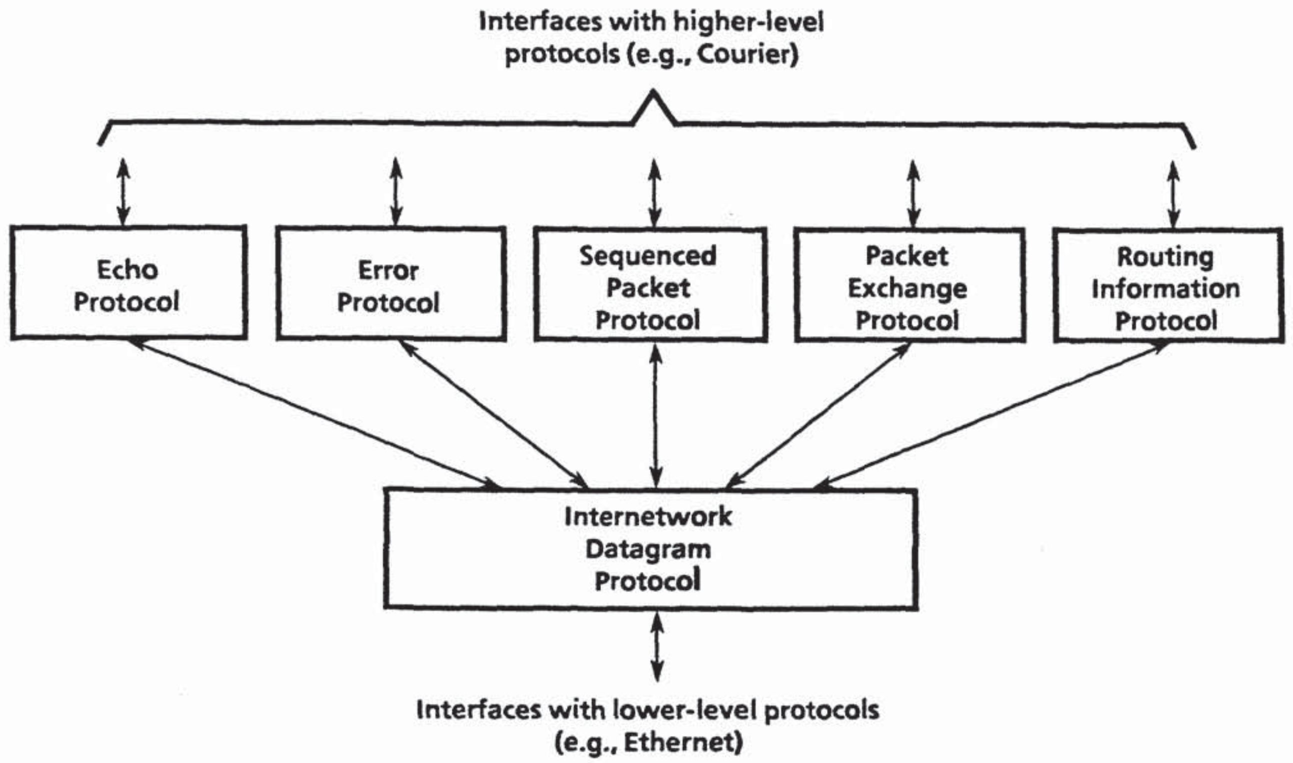

Interfaces with higher-level protocols (e.g., Courier)
Echo Protocol
Error Protocol
Sequenced Packet Protocol
Packet Exchange Protocol
Routing Information Protocol
Internetwork Datagram Protocol
Interfaces with lower-level protocols (e.g., Ethernet)

Figure 3-5 **The XNS Internet layer**

Sequenced Packet Protocol	This provides for reliable, sequenced, and duplicate-suppressed transmission of a stream of packets.
Packet Exchange Protocol	This supports simple transaction-oriented communication involving the exchange of a request and its response.
Routing Information Protocol	This provides for the exchange and dissemination of internetwork topological information necessary for the proper routing of datagrams.
Error Protocol	This standardizes the manner in which low-level communication or transport errors are reported.
Echo Protocol	This is used to verify the existence and correct operation of a host, and the path to it.

The work done by Internet involves interaction between one or more of the specialized protocols and the basic Internetwork Datagram Protocol. To simplify this discussion, functional distinctions between the operation of these various protocols have been avoided. The Internet documentation described in appendix D contains more detail about this relationship.

Datagrams

The XNS internet architecture identifies a fundamental unit of information flow called an internet packet or "datagram." It refers to a condition of "self sufficiency" for the transmission unit of information, the packet. A datagram, therefore, is a packet (typically several hundred bytes of information) whose movement through the system is individually controlled. (This is in contrast to systems that use the concept of a "virtual circuit" in which a logical relationship is constructed between source and destination for purposes of transmission; in such systems, the flow of packets is typically managed on a group basis.)

Packet structure

The Internet packet contains control information and data as shown in Fig. 3-6.

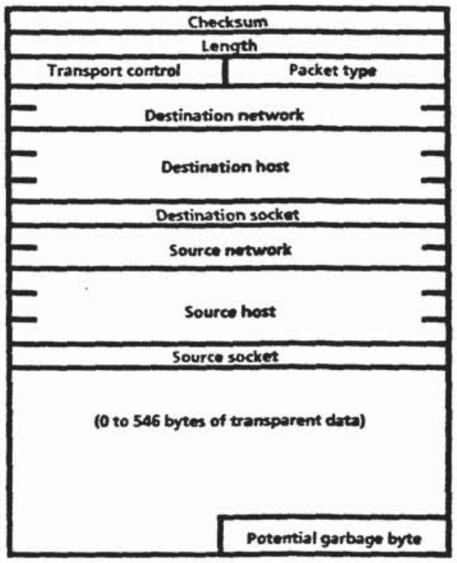


Figure 3-6 **Internet packet or "datagram"**

The Internet packet fields fall into three categories: adressing fields, which specify the address of the destination and source network addresses; control fields, which consist of checksum, length, transport control, and packet type fields; and data fields, which carry the data and consist of information that is interpreted only at the next higher Courier layer.

The function of the Internet is thus to manage the flow of these datagrams through the network. The "network" in this context consists of the local network, plus all other connected networks, however far-flung they may be. Internet is responsible for maintaining a global perspective. Regardless of how large or small the network may be, the process of managing datagram flow through it is uniform.

Source and destination addresses

The Internet packet has source and destination address fields which is more general than the 48-bit host number used on the Ethernet. It also includes a 32-bit network number and a 16-bit socket number for the source, as well as the destination address, as shown in Fig 3-6.

Host number

A host is any system element that supports the XNS communication protocols and is connected to a network. In XNS, every system element is assigned a unique 48-bit number independent of the network to which it is connected. Xerox chose an absolute host numbering scheme instead of the more conventional network-specific host numbering scheme. Absolute host numbers have many advantages when building large distributed systems. Operating systems and application software can use this number in generating unique identifiers. Also, when a host is moved from one network to another, its host number does not change, making alterations to the hardware or special bookkeeping unnecessary. Since such alterations are required when using network-specific host

0041

numbers, use of absolute host numbers substantially reduces field service overhead.

Xerox internets consist, for the most part, of Ethernets, which is the main reason that Ethernet addresses are identical to 48-bit host numbers. This structure is strictly for convenience and in no way compromises the generality of the architecture. When a host is connected to more than one Ethernet, its 48-bit Ethernet address on all Ethernets is equal to its 48-bit host number.

Xerox has implemented a procedure whereby other implementors of XNS hosts may reserve a block of host identification numbers. (Note that since the host number is 48 bits long, this represents an enormous address space: if all the numbers were used, approximately 281 trillion hosts would be identified! In practice, however, the number of hosts is limited to about four billion.)

Network number

Since a host number uniquely identifies a specific host, the network number field would seem redundant, but it is needed for internetwork routing. When the network number is included in the network address, each host has to know only the (partial) path to each network rather than that to each host—significantly reducing the amount of information that must be retained. A host may be connected to more than one network but still has a unique identity.

An internet packet addressed to a host contains the identity of the network to which the source believes the host is connected. Internetwork routers attempt to route the internet packet to the host via this network. If no route to the specified network exists, the packet is not delivered and client software must use another network address. The higher-level Clearinghouse supplies all network addresses for resources such as file servers, print servers, or a user's mailbox. All networks within an internet have unique network numbers.

Socket number

A socket is a uniquely identified data structure within a host, to which internet packets can be delivered, and from which internet packets can be transmitted. A socket is inherently a bidirectional structure, able to both send and receive packets.

The internet delivers packets (datagrams) among sockets in much the same way that the post office transfers letters between post office boxes. The sockets may be on the same host, on hosts on the same network, or on hosts on different networks. A host that receives an internet packet first delivers the packet to the approximate socket, and then the client of the socket demultiplexes the packet according to its transport protocol type. In this respect, the XNS approach differs from that used in other internetwork architectures such as the one defined by the ARPA Internet Protocol, which does not include a socket number in its network address—a host receiving an

ARPA internet packet demultiplexes it according to its protocol type, and the next higher level of protocol then has the option of defining a socket-like object.

Sockets are numbered within a particular host. The socket number is a 16-bit binary value (providing 65,536 possible socket numbers). The first 3000 socket numbers are reserved for well-known sockets; that is, the service performed by software using these sockets is well-known. Each host supplying a specific well-known service does so at the same well-known socket. All other socket numbers can be reused.

Multidestination addressing

Multicast is the delivery of a packet to more than one destination, and it can be performed at either the internetwork or intranetwork level if the transmission medium supports the concept. XNS supports internet multicast. Broadcast is a special case of multicast in which a packet is delivered to all hosts in the internet. The need for a generalized multicast capability arises from the anticipated need for a more general addressing capability. Broadcast is used in many situations to search for an object or to inform all hosts of an event. Although all applications can be designed without this capability, multicast provides some performance improvements.

Internet delivery and routing

One of the major responsibilities of the Internet layer is routing and delivery of information from a source to a destination. Since the network over which this is to be accomplished may be very large, involving many different subnetworks with a variety of interconnecting links, the routing procedure is very generalized. It presumes that intermediate functions, performed by internet routers, assist in the process. Internet routers move datagrams through the network based on the idea that each routing element in the network understands *the next place to send the datagram*. Depending on the scope of the network, many or all of the routing provisions in the chain may not "know" the entire network; but with the "next place to send" approach, the network can be arbitrarily large and the datagram will arrive at its destination.

Encapsulation and decapsulation

The internet packet is usually encapsulated for transmission through the various communication networks; the encapsulation specifies the addressing and delivery mechanisms specific to that network. Each communication network may have a different form of internal addressing. When an internetwork packet is to be transported over a communication medium, the immediate destination of the packet must be determined and specified in the encapsulation. The immediate destination is determined directly from the network address if it is the final destination, or through a routing table if it is an intermediate destination.

Routing

Basic routing is undertaken by the Internet Datagram Protocol. Each host connected to the network is equipped to execute this protocol. It uses a routing table containing the addresses of other elements in the vicinity of the host in question, specifically including the internetwork routers responsible for directing datagram flow between subnetworks. This routing information is kept up to date by the Routing Information Protocol.

Internet packets are routed through the internet using a store-and-forward algorithm that relies on the routing table. Routing information is maintained through the use of adaptive procedures. Neighboring internetwork routers periodically exchange routing information. Changes in internetwork topology may cause the routing tables in different internetwork routers to become momentarily inconsistent, but the algorithm is stable since routing tables rapidly converge to a consistent state and remain that way until the topology changes again.

A host that is not an internetwork router obtains routing information by polling internetwork routers on its directly connected networks. The host may obtain updates periodically if it receives the broadcast packets that other internetwork routers are exchanging; if not, then it may periodically poll internetwork routers for updates. If more than one internetwork router is providing paths to other networks, an internetwork router or host can merge the information it receives and thus select the best route for packets directed to any network.

When internet packets traverse other communication networks that do not support XNS absolute host numbers, like the Bell Telephone DDD network, Telenet, or other public or private data networks, translation tables will have to exist in the necessary hosts and internet routers to perform translation from absolute host numbers to internal addresses. This does not cause many operational problems other than setting up and maintaining these translation tables as appropriate.

Message integrity

While the basic unit of information in the Internet is the datagram, or internet packet, the overall purpose of communication is to deliver entire messages from source to destination. A message consists of an arbitrary number of datagrams—whatever is required to send the complete message. In some cases this can be a fairly large number, as in the case of a file containing a scanned photographic image, which might consist of as many as 1000 500-byte datagrams (a customary datagram size).

A basic characteristic of all packet-switching communication systems (including XNS) is that the freedom to optimize the routing of the individual packets can cause a transmitted string of packets to arrive at their destination out of sequence with respect to their order at the source. This happens as individual packets are routed through different links, with different transmission delays, and as some packets are retransmitted due to error conditions. It is also possible for packets to be duplicated and, in rare cases, to vanish altogether. Left uncorrected, this situation means that packet-switching systems have a number of opportunities for the integrity of messages to be violated. This is clearly unacceptable.

Sequenced Packet Protocol

The XNS Internet layer deals with this problem through the functions of the Sequenced Packet Protocol (Fig. 3-5). A sequence number is assigned to each transmitted packet within the Internet layer at the source host. This number is checked within the Internet layer at the destination host, and if aberrations (misordered packets, duplicates, missing packets) are detected, corrective action is taken on a negotiated basis between the source and destination Internet layers.

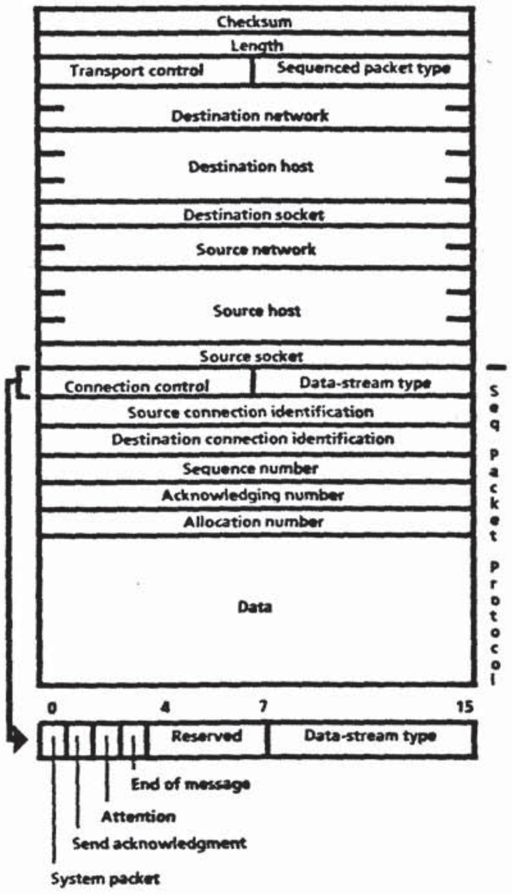


Figure 3-7 A sequenced packet protocol packet allows successive transmission of internet packets.

To achieve this result, Internet provides for associating the source and destination sockets in a connection, a temporary relationship in which various details concerning the operation of the exchange are negotiated and in which sequence numbers, acknowledgments, and related supervisorial information are exchanged.

This connection between source and destination can be thought of as establishing a virtual circuit between them, roughly analogous to what takes place in X.25 wide-area switched packet networks. The Sequenced Packet Protocol should be seen as using a datagram-based lower-level protocol, the Internetwork Datagram Protocol, to create a virtual circuit between source and destination, by taking responsibility for the integrity of information flow.

Arranging packets into messages and message sequences also circumvents the packet-size limitation at lower levels of the protocol architecture. The Sequenced Packet Protocol provides a mechanism to punctuate the stream of packets with end-of-message boundaries. The protocol specifies the format of packets as shown in Fig. 3-7, and the meaning of packet sequences.

The connection control field contains four bits that control the protocol's actions: system packet, send acknowledgment, attention, and end-of-message. The system packet bit enables the recipient to determine whether the data field contains client data, or is empty and the packet has been sent only to communicate control information required for the connection to function properly. If the send acknowledgment bit is set, the source wants the receiver to acknowledge previously received packets.

In a distributed environment, special procedures must be provided to bypass the normal flow control and interrupt a process. If the attention bit is set, the source client process wants the destination client process to be notified that this has arrived. If the end-of-message bit is set, then the packet and its contents will terminate a message and the next packet will begin the following message.

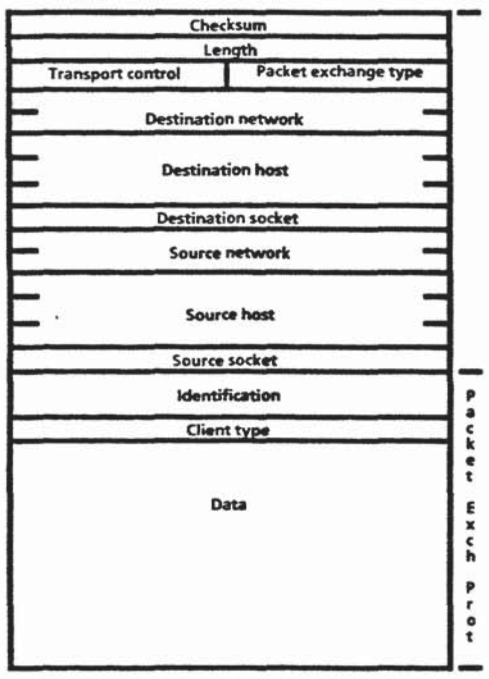


Figure 3-8 A packet exchange protocol packet simply transmits a request and receives a response.

The primary bridge between this protocol and the next layer is the data stream type field which provides information that may be useful to higher-level software in interpreting data transmitted over the connection. A connection must be created before it can be used and discarded when no longer required. One end of a connection is said to be established when it knows the address (host and socket number) and connection identification of both ends of the connection.

Connection-oriented communications, which is supported by the Sequenced Packet Protocol, involves an extended conversation by two machines in which much more information is conveyed than can be sent in one packet going in one direction. For simple transaction-oriented communication which involves single requests and responses, a simpler Packet Exchange Protocol is also provided in the internet architecture.

Transmitting a request in a packet and receiving a response via the Packet Exchange Protocol (Fig. 3-8) will be more reliable than transmitting internet packets directly as datagrams, but less reliable than the Sequenced Packet Protocol.

There are only three fields in the packet. An identification field, which contains a transaction identifier, is the means by which a request and its response are associated. A client type field indicates how the data field should be interpreted at higher levels. A data field contains whatever the higher-level protocols specify. Such a protocol might be used in locating a file server through a resource-location service, such as the Xerox Clearinghouse.

Other Internet protocols

As dominant as the Sequenced Packet and Packet Exchange Protocols are in the internet layer, they do not handle everything. Other internet protocols are required for specialized tasks of routing, error reporting, and diagnostics.

Routing Information Protocol

The Routing Information Protocol provides for the exchange of topological information among internetwork routers and workstations. Two packets are defined by the protocol: one of them requests routing information, and the other supplies it. The information supplied is a set of network numbers and an indication of how far away those networks are. This informa-

0046

tion is either sent on specific request or periodically distributed by all internetwork routers, which use the data to maintain routing tables that describe all or part of the internetwork topology.

Error Protocol The Error Protocol is intended to standardize the manner in which low-level communication or transport errors are reported. Moreover, it can be used as a debugging tool. If, for example, a machine receives a packet that it detects as invalid, it may return a portion of that packet by means of the Error Protocol, along with an indication of what is wrong. If the packet is too large to be forwarded through some intermediate network, the error protocol can be used to report that fact and to indicate the length of the longest packet that can be accommodated. If too many of these return, the system designer may conclude that something is wrong with his implementation.

Echo Protocol Another useful diagnostic and debugging tool is the Echo Protocol, which is used to verify the existence and correct operation of a host, and the path to it. It specifies that all Echo Protocol packets received shall be returned to the source.

Internetwork Routing Service

The Internetwork Routing Service (IRS) links geographically dispersed networks into a single internet by connecting a local network to other networks via telephone lines or public data networks, It thus enables communication with remotely-located XNS system elements.

The IRS implements the Internet Transport Protocols, and its functions include packet forwarding, routing decisions, and the gathering of routing information. Of particular importance is the media flexibility that allows XNS-supported systems to make use of the best suited transmission facilities: dedicated circuits, switched circuits, and public data networks. In addition, the Internetwork Routing Service makes it possible for isolated remote devices (i.e., those established on a standalone basis without their own Ethernet local area network, servers, etc.), or clusters of such devices, to interface with local networks via telecommunication circuits.

The Internetwork Routing Service implements one of the most important characteristics of Xerox Network Systems: geographic independence. This means that users of XNS in one location can communicate with users and network resources in a remote location, *without being aware of, or having to take special measures because of the distances between the two locations.* In other words, an XNS network appears as a single entity to a user, not a series of interconnected local area network segments.

How IRS works

The IRS is fundamental to all installations with multiple interconnected Ethernets. When a local network device sends information to another network, it sends the information to an IRS which forwards it to its destination (see Fig. 3-9).

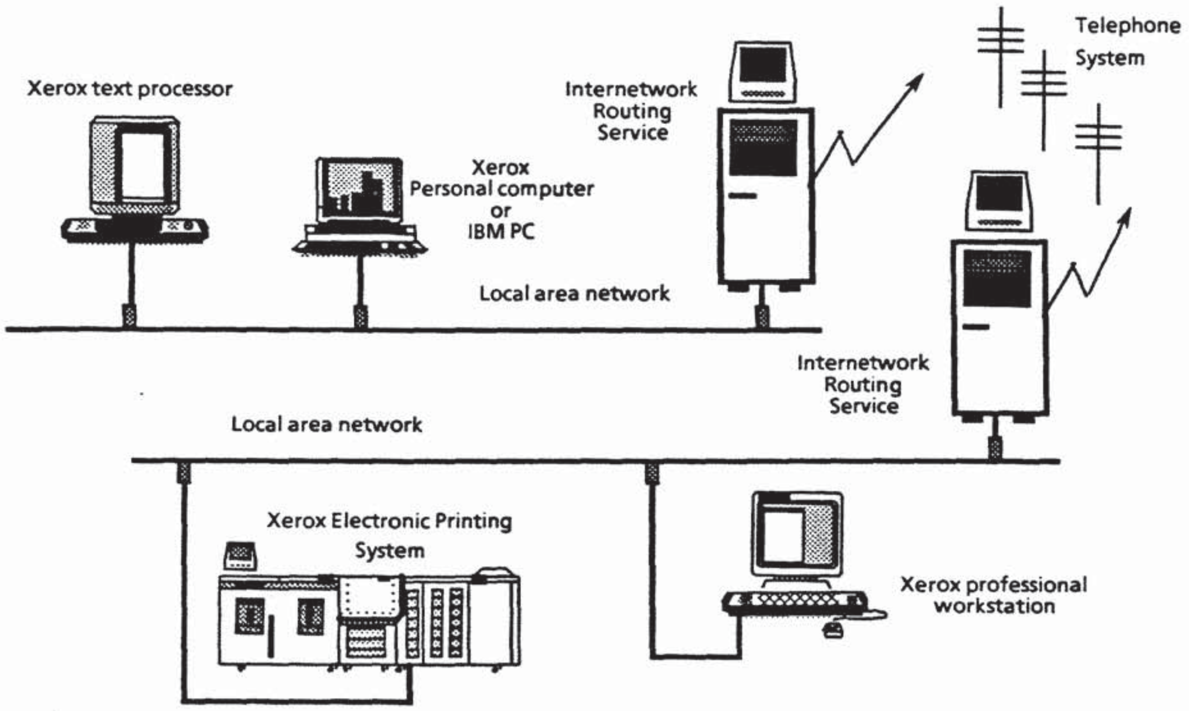

Telephone System
Xerox text processor
Internetwork Routing Service
Xerox Personal computer or IBM PC
Local area network
Internetwork Routing Service
Local area network
Xerox Electronic Printing System
Xerox professional workstation

Figure 3-9 **Two local area networks interconnected by a communication line**

To accomplish the forwarding function, each IRS holds a complete map (routing information tables) of the internet. This map contains the address of each remote network, how far away it is, and the next IRS along the path to get there. When an IRS receives a packet to be forwarded to a remote network, it uses its map to find the address of the network and sends the packet along its way using the shortest path to that network. IRS's on an internet exchange their maps on a regular basis. After an IRS is activated, it gradually learns of the complete internet map from neighboring IRS's by means of this exchange. Likewise, changes to the internet gradually propagate from one IRS to another until all IRS maps reflect the change. No manual intervention is needed.

The operation of the Internetwork Routing Service is invisible to the user of a local workstation. On the Xerox professional workstations, users access remote services through icons on the desktop in exactly the same way they access local services. For example, to access a file drawer on another network, a user would open the Directory icon and bring a copy of the file drawer icon to his desktop. The remote file drawer could then

0048

be accessed in the same way as a local file drawer. Although the user commands on other workstations may be different, the functioning of the IRS is still invisible. The user of any workstation never needs to communicate directly with the IRS.

Dedicated and switched circuits

Dedicated telecommunication circuits are generally available on a full-duplex basis at speeds to 56 kbps. The interface to these circuits is by means of the RS-232-C standard, or its higher-speed equivalents; use of these circuits requires corresponding modems. The Synchronous Point-to-Point Protocol is designed to operate over a dedicated circuit.

Switched circuits consist of telephone dial-up and similar facilities. In essence, a switched circuit provides a temporary point-to-point connection. The Internetwork Routing Service is capable of operating with switched connections made manually or of automatically dialing a connection. In both cases, subsequent operation over the circuit is similar to operation over a dedicated circuit.

X.25 public data networks

An alternative to the use of conventional telecommunication circuits is the use of public data networks (usually packet-switched), which provide user interfaces that comply with CCITT's Recommendation X.25. In some parts of the world, Europe in particular, X.25 public data networks are much more widely available than dedicated circuits and are able to overcome many of the network interconnection and trans-border data flow problems. The Internetwork Routing Service makes it possible for individual Ethernet systems to be linked by means of these public data networks. In such arrangements the X.25 Virtual Circuit Protocol, rather than the Synchronous Point-to-Point Protocol, is used at the link control layer.

An X.25 Internetwork Routing Service is a powerful facility. Means are provided for a local or remote system administrator to initiate and terminate individual X.25 links; to interrogate the status of a given link; to maintain a log of calls; and to automatically retry failed outgoing calls (as happens when a public data network's local ports are momentarily tied up). The Internetwork Routing Service also provides basic security checks on incoming calls to ensure that only authorized calls are accepted.

When information is transmitted using X.25 circuits, the IRS wraps the data in a special X.25 protocol and sends it to the local X.25 network. The network consults its routing table and sends the information to the destination IRS. The X.25 network serves as a pass-through medium for directing information. When the IRS receives the information, it unwraps the data and routes it to its final destination.

The use of X.25 networks does not preclude the use of other communication media as shown in Fig. 3-10. X.25 networks simply provide another, often more desirable, alternative for internet communication.

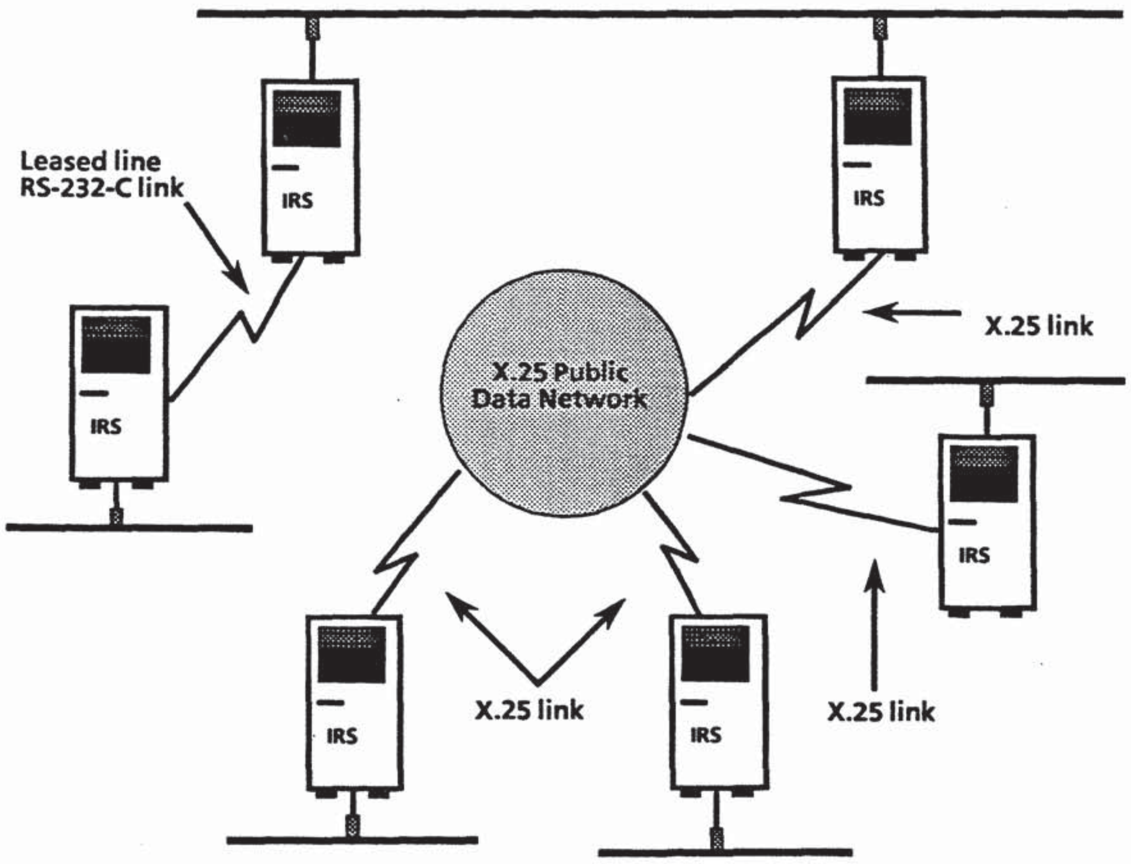


Figure 3-10 **Internet Routing Service (IRS) with X.25 links**

Clusternet communication

The IRS also makes it possible for individual workstations or a cluster of workstations (equipped with appropriate communication interfaces) to communicate with each other and the internet without the use of separate communication servers (see Fig. 3-11). The Synchronous Point-to-Point Protocol is used between Internetwork Routing Service and the workstations, and the resulting network is called clusternet.

The IRS includes a clusternet router that provides routing information to the clusternet. This router uses the network number of the clusternet and the host numbers of the remote workstations to route information to and from the clusternet's ports.

The remote workstation is attached to the IRS using an RS-232-C connection via leased lines or dial-up capability. In dial-up operations, the remote workstation becomes networked only when the user dials an IRS clusternet port. For the remainder of the dial-up session, the user can access all of the services normally provided by a direct connection to the network.

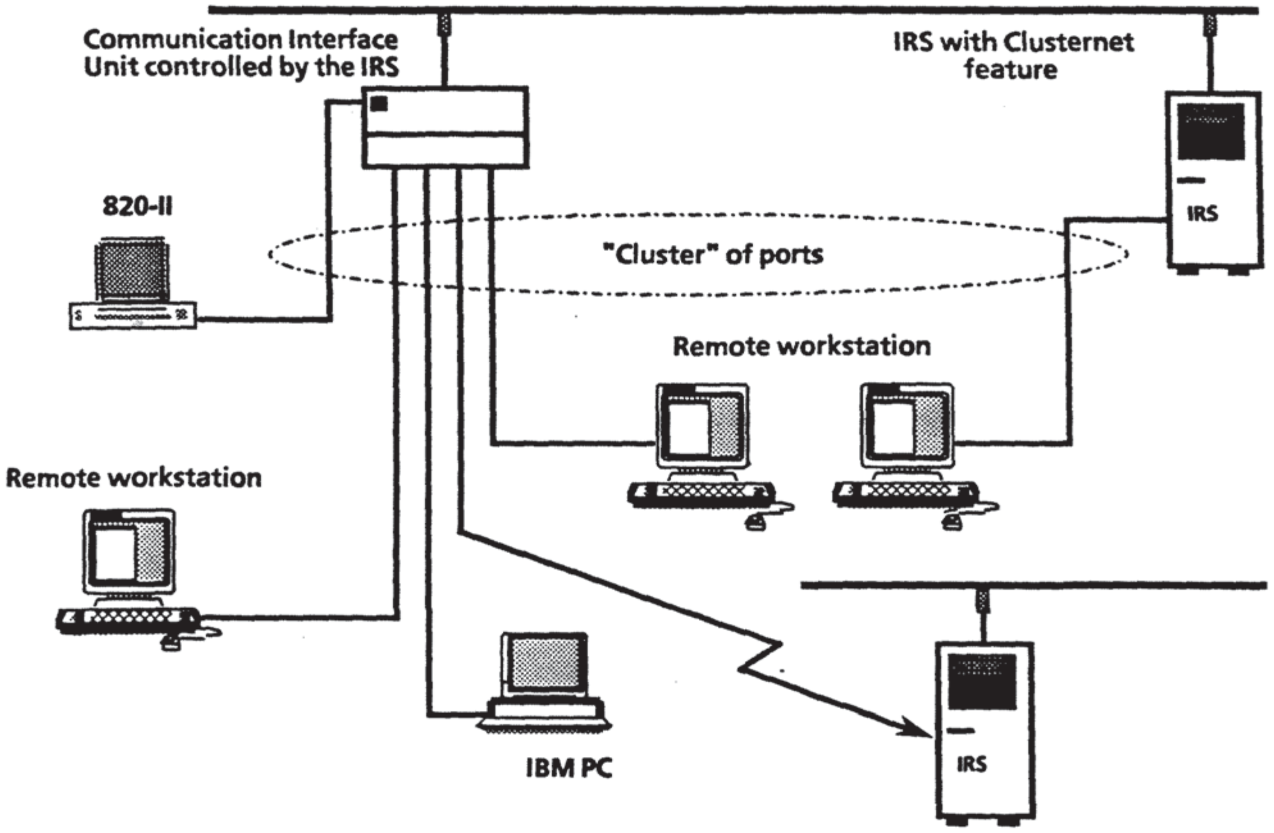

Communication Interface Unit controlled by the IRS
IRS with Clusternet feature
820-II
IRS
"Cluster" of ports
Remote workstation
Remote workstation
IBM PC
IRS

Figure 3-11 **Internet Routing Service (IRS) with clusternet feature**

Network management

Network management has traditionally been associated with managing telephone systems and modems. A network manager might be responsible for determining who gets what equipment and with what access rights, for publishing an internal telephone directory, diagnosing problems, and for reconfiguring the system to meet customer needs. The internet, including all its diverse workstations and services, must be similarly managed. However, the problem of managing an internet is more a problem of managing a distributed system. Managing the communication components is a subset of the problem.

Service monitoring

Network management in the distributed system involves monitoring the services and communication components of the internet. While most networks support monitoring only from a central network control center, entities in XNS can be monitored from any system element in an internet. Controlled access availability to the network management tools prevents their unauthorized use. The monitoring of services includes performance and load statistics, and unusual events. Monitoring communication components include statistics that assist in problem determination, plus network planning and configuration.

The Server Monitor Service (SMS) is a program that watches over servers on the internet, assesses their availability, and reports problems through mail messages. SMS runs on an XNS server and interacts with a specified set of servers. It maintains a database of information which includes the configuration of the target servers being monitored, the frequency with which a given server should be polled, and lists of interested users. These lists are used to send messages to interested users whenever a server's availability changes. All the information obtained about a server's condition is recorded by the SMS and can be examined.

System administrators

A large internet configuration (which may have thousands of users and their associated workstations, printers, file servers, mail servers, etc.) requires considerable management. To facilitate this, a system administrator function is provided in XNS that allows designated users to be registered as system administrators.

The XNS distributed system approach allows system administrators to provide localized system installation and reconfiguration. Alternately, the system administrator function may be performed from a remote network or a remote workstation. Network integrity is maintained by the strong emphasis on security in the XNS architecture. This gives great flexibility in network management and control.

4. REMOTE PROCEDURES: COURIER

Xerox Network Systems uses a model in which a system exists to perform useful work. The purpose of the communication infrastructure is to facilitate doing that work. Although other forms of information exchange are also used, a key part of the XNS architecture deals with how the exchanges take place between a service provider and a service consumer that cause work to be done at a location. This location may be at a distance of thousands of miles or within the same processor. To the architecture, there is no significant difference.

It is crucial to the implementation of a modern distributed system that these work exchanges be accomplished efficiently, with maximum flexibility, and with no loss of generality.

This exchange is at a middle layer in the architecture in between the application-layer protocols and the network/ transport layer protocols. The functions involved at this point are more primitive than application functions (but still vital).

A special protocol called Courier (subtitled The Remote Procedure Call Protocol) defines the method by which directions for accomplishing work within XNS are sent and appropriate responses returned. The functions performed within Courier relate approximately to those in the session and presentation layers of the ISO Model.

The Courier model

Courier facilitates the construction of distributed systems by defining a single request-reply or transaction discipline for an open-ended set of higher-level application protocols such as printing, filing, and mail. Courier specifies the manner in which a workstation or other active system element invokes operations provided by a server or other passive system element (see Fig. 4-1).

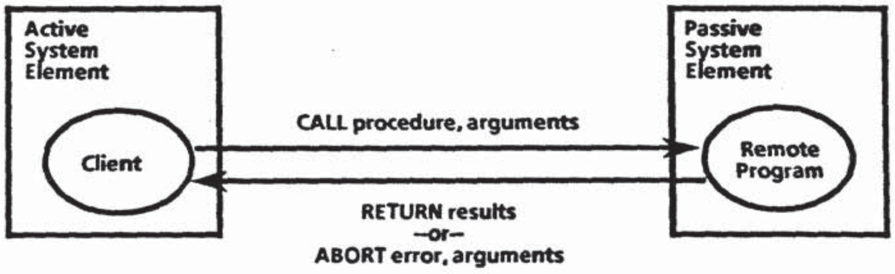


Figure 4-1 The Courier remote procedure call model

Courier is based on the concept that an active system element issues call procedures which contain "arguments" (data items or input specific to the requirements of the called procedure) necessary to get the work done. The remote procedure is undertaken in a passive system element and the result of that work is returned to the active element. If something goes wrong, the procedure is aborted and an error statement is returned. The error statement contains "arguments" (data feedback) that will help the calling element determine what went wrong.

Courier as a language

Courier does for distributed system builders some of what a high-level programming language does for implementors of more conventional, non-distributed systems. Pascal, for example, allows the system builder to think in terms of procedure calls rather than in terms of base registers, save areas, and branch-and-link instructions. So Courier allows the distributed system builders to think in terms of remote procedure calls, rather than in terms of socket numbers, network connections and message transmission. Courier also provides a rich set of predefined as well as constructed data types including boolean, integer, cardinal, string, array, and record among others.

Not all transaction-oriented communication may be best accomplished using Courier. Some applications may necessitate the use of datagrams rather than virtual circuits (upon which Courier is based). The XNS protocols at the Internet layer support applications for which Courier is inappropriate.

The internal Courier layers

Courier is internally divided into three hierarchical layers (see Fig. 4-2): The block stream at the lowest layer, the object stream at the middle layer, and the message stream at the highest layer. The block stream layer carries blocks of arbitrary binary data between system elements in accordance with the Internet Sequenced Packet Protocol. The object stream layer carries structured data (such as booleans and cardinals) between system elements. The message stream layer carries service requests (call messages) and replies (return and abort messages) between system elements.

Courier assumes that a higher-level function (such as an XNS application layer or a user process designed to interface with XNS at the Courier level) issues the appropriate remote procedure calls in the form of message streams. Courier's responsibility is to interface with the Sequenced Packet Protocol within Internet to move the request to the appropriate destination. Courier can thus be seen to have two primary functions: translating specific remote procedure calls into a common "language" (in the computer sense of the word) for subsequent action, and directing the communication system, as represented by Internet, to convey the required instructions to getting the work done. A third function of Courier is bulk data transfer, to which it has been adapted (as discussed in the next section).

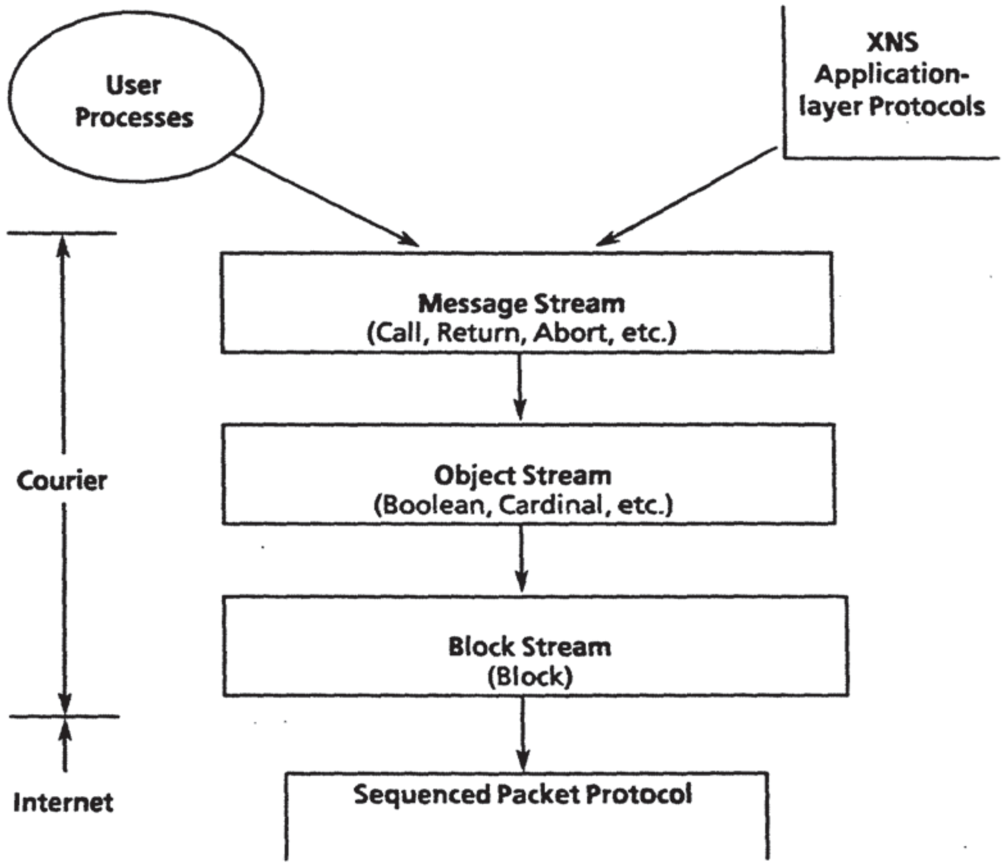

User Processes
XNS Application-layer Protocols
Message Stream (Call, Return, Abort, etc.)
Object Stream (Boolean, Cardinal, etc.)
Block Stream (Block)
Sequenced Packet Protocol
Courier
Internet

Figure 4-2 The layers within Courier

Because the top layer within Courier accepts application-generated messages and translates their content into a more primitive, but useful, form, the relationship between the *input* to Courier and the *content* of Courier may be considered similar to the relationship between a source and object computer language. In general, a source language is responsible for expressing the purpose of the program in terms particularly compatible with the purpose of the originator; the object language expresses that purpose in terms particularly compatible with the machine responsible for accomplishing that purpose. In this case, the "machine" is the Sequenced Packet Protocol and its underlying Internet and transmission protocols. Note that this source/object perspective is consistent with the functionality at the ISO Model's presentation layer which is encompassed by Courier in the XNS structure.

For these reasons Courier is considered a language as well as a protocol. The transaction-oriented expressions which Courier accepts are often spoken of as "written in Courier." The notation used for the Courier language is in Backus-Naur Form

(BNF). This is a formalized methodology for writing computer language statements produced in conjunction with the development of the ALGOL 60 language in the 1960s.

```
SimpleFile Transfer: PROGRAM 13 VERSION 1 =
BEGIN
- types
Credentials: TYPE = RECORD (user, password: STRING);
Handle: TYPE = UNSPECIFIED;

- procedures
OpenDirectory: PROCEDURE (name: STRING, credentials:
    Credentials)
    RETURNS (directory: Handle) REPORTS (NoSuchUser,
    IncorrectPassword, NoSuchDirectory, AccessDenied) = 1;

    Store File: PROCEDURE (name: STRING, directory: Handle)
    REPORTS (NoSuchFile, InvalidHandle) = 2;

    RetrieveFile: PROCEDURE (name: STRING, directory:
        Handle)
    REPORTS (NoSuchFile, InvalidHandle) = 3;

    CloseDirectory: PROCEDURE (directory: Handle) REPORTS
    (InvalidHandle) = 4;

- errors
NoSuchUser:                 ERROR = 1;
NoSuchDirectory:            ERROR = 2;
NoSuchFile:                 ERROR = 3;
IncorrectPassword:          ERROR = 4;
AccessDenied:               ERROR = 5;
InvalidHandle:              ERROR = 6;
END.
```

Figure 4-3 Example of Courier usage

To see how Courier is utilized, consider a user who wishes to retrieve a file from a file server on the Internet. Let us assume that the user is on a non-Xerox workstation and on a remote network thousands of miles away. The file system on the server contains named directions, each of which comprises one or more files. For this operation, the user would need a file transfer protocol. Utilizing Courier's standard notation, it is very easy to formally specify the hypothetical file transfer protocol shown in Fig. 4-3. Remote procedures are provided in this example for gaining and relinquishing access to directories and for storing and retrieving files.

To retrieve the file, the user's workstation locates and then establishes a connection to the file server. The workstation opens the directory, retrieves the file, and closes the directory. The workstation then terminates the connection. The workstation opens and closes the directory by calling the remote procedures named OpenDirectory and CloseDirectory in the file server. It requests retrieval of the file by calling the remote procedure named RetrieveFile, which tells the file server of the intention to retrieve. As soon as that procedure returns, the file server transmits the contents of the file on the connection, using the bulk data transfer mechanism.

Bulk data transfer and third parties

Courier supports applications whose communication requirements are primarily request/reply transactions. Of course, not all communication exchanges are transaction-oriented. An important alternative category involves the movement of comparatively large quantities of data (e.g., an entire document file) which would be inefficient to send as an argument within a procedure call. To allow for this, XNS provides a special adaptation of Courier called the Bulk Data Transfer Protocol which provides a single bulk data transfer discipline for an open-ended set of higher-level application protocols.

Bulk data is an arbitrarily long sequence of eight-bit bytes optionally interpreted as a single Courier data object. Although any data may be modeled as bulk data, bulk data transfer is intended primarily for transporting objects that may be too large to be reasonably modeled as arguments or results of remote procedures. For example, directory listings and the contents of files are among the entities that might be appropriately modeled as bulk data.

The Bulk Data Transfer Protocol is itself a Courier-based protocol, and it standardizes the manner in which the sender and receiver of bulk data make contact with one another; how bulk data is demarcated; and how the transfer can be aborted, if necessary, by either party.

The transfer of bulk data conceptually involves an initiator which requests the transfer, a sender which *produces* the data and a receiver which *consumes* the data. The initiator is often the sender or receiver. The Bulk Data Transfer Protocol supports three forms of bulk data transfer: Third party, immediate, and null. A *third-party* transfer is required for the most general case, in which the initiator is neither the sender nor the receiver. An *immediate* transfer suffices in the most common case, in which the initiator is either the sender or the receiver. A *null* transfer handles the degenerate case in which the data transfer that would normally take place must be suppressed.

Third Party transfers

Unlike simple remote procedure calls, bulk data transfers may involve third parties. For example, a workstation might wish to cause a file server to send a file to a print server for printing. Fig. 4-4 illustrates this third-party exchange and provides a general idea of its operation. In this example, a third party initiator wishes to cause information to flow from a sender to a receiver. It does this by issuing special procedure calls called *Produce* and *Consume* (Step 1, A and B). Assuming for a moment that the receiver is the active controlling party, it issues a Bulk Data Send procedure call to the passive (controlled) sender (Step 2). At that point, the sender actually transmits the bulk data (Step 3), and issues a return message to the receiver (Step 4). Finally, the sender and receiver issue returns to the initiator (Step 5, A and B). The Bulk Data Transfer Protocol permits either the sender or receiver to be active elements, depending on their characteristics and the nature of the data being transferred.

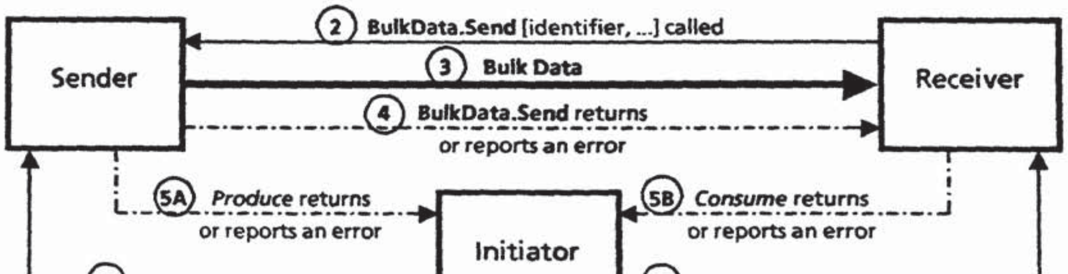

Sender
Receiver
Initiator
2 BulkData.Send [identifier, ...] called
3 Bulk Data
4 BulkData.Send returns or reports an error
5A Produce returns or reports an error
5B Consume returns or reports an error
1A Produce [... sink, ...] called
1B Consume [... source, ...] called

Figure 4-4 Third-party bulk data transfer (receiver active)

Immediate transfers

A third party is not always involved, of course. The initiator is often the sender or receiver, and this becomes a much simpler case. The initiator effects an immediate transfer by calling *Produce* or *Consume*, whichever is appropriate. As an argument to the procedure, the initiator supplies a description indicating that an immediate transfer is desired. *Produce* or *Consume* simply transfers the bulk data and returns. If the transfer fails for some reason, the procedure reports the error as shown in Fig. 4-5.

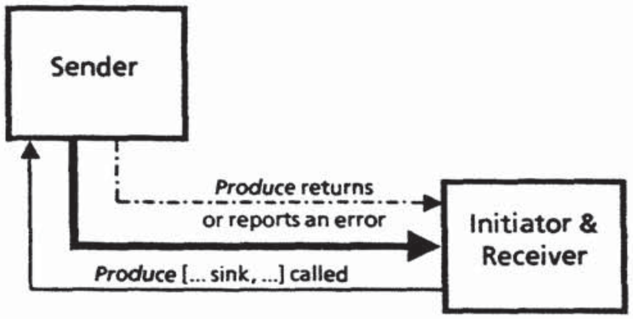


Figure 4-5 Immediate bulk data transfer (initiator the receiver)

0058

5. APPLICATION SUPPORT ENVIRONMENT

A number of important support services are required for the secure, reliable, and smooth functioning of applications in a distributed network environment. These services are often not visible to the user but they are used by nearly all the XNS application programs. Together, they constitute the application support environment base. XNS implements the application layer (Layer 7 of the ISO model) in a structure which embodies a series of interrelated applications that build upon this support environment and use standard information formats. It is this richness of applications built upon a solid foundation that makes the XNS architecture so useful and versatile. The application support environment functions include:

- Providing the means for a user to locate specific resources or individuals on the network.
- Ensuring that only authorized individuals have access to sensitive processes and files.
- Establishing a common time base for use throughout the (potentially worldwide) network.
- Providing a common character encoding system, standard use of fonts, and font services.

These support functions are used by the higher-level application services discussed in subsequent sections and are available to XNS users for user-written applications.

Clearinghouse

One of the problems that must be overcome in a distributed system is identifying the location of system resources and users. If a workstation does not know how to gain access to a printer attached elsewhere in the network, it can't get something printed. If another workstation cannot locate a communication service, it can't gain access to the mainframe connected to that service. To communicate with other users, their electronic mailbox addresses need to be known. The problem is made worse by the constant changes occurring in networks (adding or deleting resources from the network, removing failed elements from service, dealing with elements that lack the appropriate attributes to perform specialized types of work, etc.). For a large network, keeping track of addresses and key attributes of system elements is a major undertaking. The Clearinghouse Service is conceptually similar to a telephone directory service, but with more powerful capabilities.

Object names and addresses

In XNS this problem is solved by Clearinghouse, a protocol whose purpose is to provide clients with the addresses of important objects (including people's mailboxes). These addresses are used in the remote procedures through which clients get work done.

Clearinghouse is essentially a data base of objects. The entry for each object consists of a name and a set of one or more groups of data items that encode the object's properties. Clients use the Clearinghouse service by providing it with object names and properties, in return for which Clearinghouse provides the appropriate address information.

Why are names needed at all? Why not just refer to an object by its address? Why not just directly use the network address of a file server, mail server, or printer? The reasons are much like those for using names in the telephone system or in a file system. First, address numbers are unintuitive; we do not want to refer to a printer by its network address any more than we want to refer to a colleague by a telephone number. Second, distributed objects change locations much more frequently than they change names.

Objects in Clearinghouse are named unambiguously in a uniform manner with the same naming convention for every object regardless of whether it is a user, a workstation, a server, a distribution list, or whatever. The naming is in a three-level heirarchy, and of the form: LocalName:Domain:Organization.

This division into organizations and domains within organizations is a logical rather than a physical division. An organization will typically be a corporate entity which can choose domain names to reflect administrative, geographical, functional, or any other type of divisions. Very large corporations may choose to use several organization names if their name space is extremely large. In any case, the fact that two addressable objects have names in the same domain or organization does not necessarily imply that they are physically close.

The Clearinghouse naming convention allows great flexibility and permits a user's localname to be chosen as his legal name (or a name of their choice, instead of some computer selected name such as DJones or DMJ). The Clearinghouse also supports the use of aliases and default names within the same domain for user convenience.

Mapping and binding

The Clearinghouse maps each name into a set of any number of properties to be associated with the name. A property consists of a Property Name, Property Type, and a Property Value. There are two types of property values: Type item which is an uninterpreted block of data, and the type group which is a set of names. Thus mapping a name into a network address is an example of a type item mapping, as in:

Daisy:SDD:Xerox → {<Printer, item, network address of the printer named Daisy>}.

A distribution list in electronic mail is an example of a mapping of type group, as in:

Authors:SDD:Xerox → {<Distribution List, group, {"Author 1:SDD:Xerox", "Author 2:SDD:Xerox"}>}.

Many properties may be associated with a name, as in:

John D. Smith:SDD:Xerox → {
<User, item, descriptive comment such as V.P. Marketing>,
<Password, item, password to be used for user authentication>,
<File Server Name, item, name of file server containing user's files>,
<Mailbox, item, name of mail server where user's mail is stored>,
<Printer Names, group, set of names of local printers any of which may be used>}.

In this example, the Clearinghouse is used to store the user's "profile." Note that the user's name was mapped into the *name* of his local file server (and mailbox and printer) rather than directly into its network address. This is because the name of the file server will perhaps never change but its location will occasionally change. We would not want a change in a server's location to require a major update of the Clearinghouse's database.

When a network object is referred to by name, the name must be bound to the address of the object. The later a system binds names, the more gracefully it can react to changes in the environment. If client software binds names statically, the software must be updated whenever the environment changes. On the other hand, binding takes time. Static or early binding increases runtime efficiency since, with either, names are already bound at runtime. A useful compromise combines early and late binding, giving the performance and reliability of the former and the flexiblity of the latter. XNS clients generally use early binding wherever possible and late binding

0061

only if any of these (early) bindings becomes invalid. Thus, software supporting printing stores the addresses of print servers at initialization and updates these addresses only if they become invalid.

The Clearinghouse naming and property declaration provisions are enormously flexible, making it possible for the most specialized aspects of a resource to be encoded in the Clearinghouse data base for future reference by clients. Clearinghouse also provides for pattern searches, in which a client can specify an object by indicating only those properties of interest to it. Clearinghouse will then ignore the other properties associated with the object in its database.

Generic names and Yellow Pages

The set of property names known to the Clearinghouse enables it to provide a Yellow Pages-like service. Client software can request a service in a standardized fashion, and need not remember what named resources are available. For example, each user workstation generally has a piece of software that replies to the user command "Help!" Suppose the generic name "Help Service" is agreed upon as the standard property name for such a service. To find the addresses of the servers providing help to users in domain:organization, the workstation software call asks to list all objects of name *"*:domain:organization"* with propertyname *Help Service*. The wildcard character "*" matches zero or more characters. This can be used by any workstation regardless of its location.

The "wildcard" feature allows clients to find valid names where they have only partial information or can only guess the name. It is particularly useful in electronic mail and in other uses of user names. If looking up *"Smith"* with propertyname *Mailbox* fails, because "Smith" is ambiguous, the electronic mail system may choose to list all names of the form *"*Smith*"*. with propertyname *User* to find the set of user names matching this name. It presents this set to the sender of the mail and allows him to choose which unambiguous name is appropriate.

Database replication

In all except the very small systems, the Clearinghouse (and its associated database) is decentralized and replicated. That is, instead of one global Clearinghouse, there are many Clearinghouse servers scattered throughout the internet (perhaps, but not necessarily, one per local network), each storing a copy of a portion of the global database. Decentralization and replication increase efficiency (it is faster to access a Clearinghouse server physically nearby), security (each organization can control access to its own Clearinghouse servers) and reliability (if one Clearinghouse server is down, perhaps another can respond to a request). However, conceptually it is assumed that there is one global database and each Clearinghouse server contains a portion of this database. No assumptions are made about how much of the database any particular Clearinghouse server stores. The union of all the local databases stored by the Clearinghouse servers is the global database.

The architecture allows a high degree of flexibility in what these distributed data bases contain. These can range all the way from single domain databases to the full global data base. The more domains a local Clearinghouse database contains, the higher would be the speed of response, because typically the local queries are answered faster than non-local queries. The updating of the multiple Clearinghouse databases is also automatic in the network. When a domain Clearinghouse updates its own domain data base, it also propogates the update if the database for this domain is replicated on more than one server. This update is done via electronic mail messages (which are time-stamped) so it is possible to have servers with temporarily inconsistent databases. The XNS architecture permits this; the Clearinghouse resolves these issues satisfactorily in the context of the services it provides.

Because many of the Clearinghouse exchanges are routine, they are often accomplished using the simpler Internet Packet Exchange Protocol, rather than the Sequenced Packet Protocol which, though more reliable, is more time-consuming.

The preceding discussion identified the Clearinghouse service as a separate system element as pictured in Fig. 5-1. While this may usually be so, on a small network the Clearinghouse can co-exist with other services on a single system server. (Such servers are called multifunction servers.) The Clearinghouse server is also a key contributor to network security in providing authentication and access control, the subject of the next section's discussion.

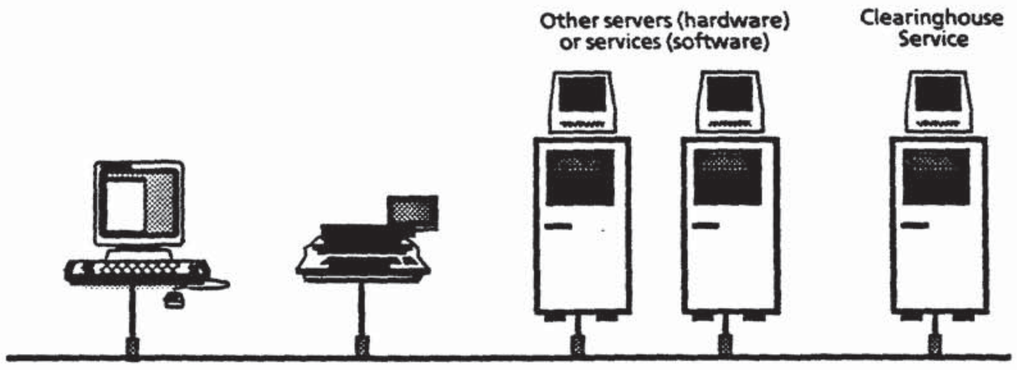


Figure 5-1 The Clearinghouse Service

Authentication

One of the difficulties with an open, distributed system is that access to system resources (files, printing facilities, etc.) is easy for any network user to obtain. Most of the time this is a

0063

distinct asset, but sometimes circumstances require that only certain users be able to gain access to a specific resource.

In XNS this problem is addressed by two provisions: access control mechanisms designed into appropriate workstation and services (e.g., filing, printing), and an Authentication Protocol which helps clients and services determine each other's identity in a reliable and secure way.

The solution provided by the Authentication Protocol assumes a secure Authentication Service which all clients and services trust to know their specially encrypted passwords, and that the clocks in the system elements are reasonably well synchronized. The Authentication Protocol provides for both a strong and a simple level of security. The goal of strong authentication is to make it practically impossible for one user to impersonate another, whereas simple authentication merely makes it difficult.

Password encryption

Every user has two passwords, strong and simple (a service has only a strong). The password used depends on the workstation encryption capability and the security environment. Passwords are for human users to identify themselves to the system. When entered into a workstation, the passwords are immediately encrypted according to a specific algorithim (hashing for simple, and NBS Data Encryption Standard for strong) to form a strong key or a simple key, thus ensuring that a user's password is never transmitted unencrypted.

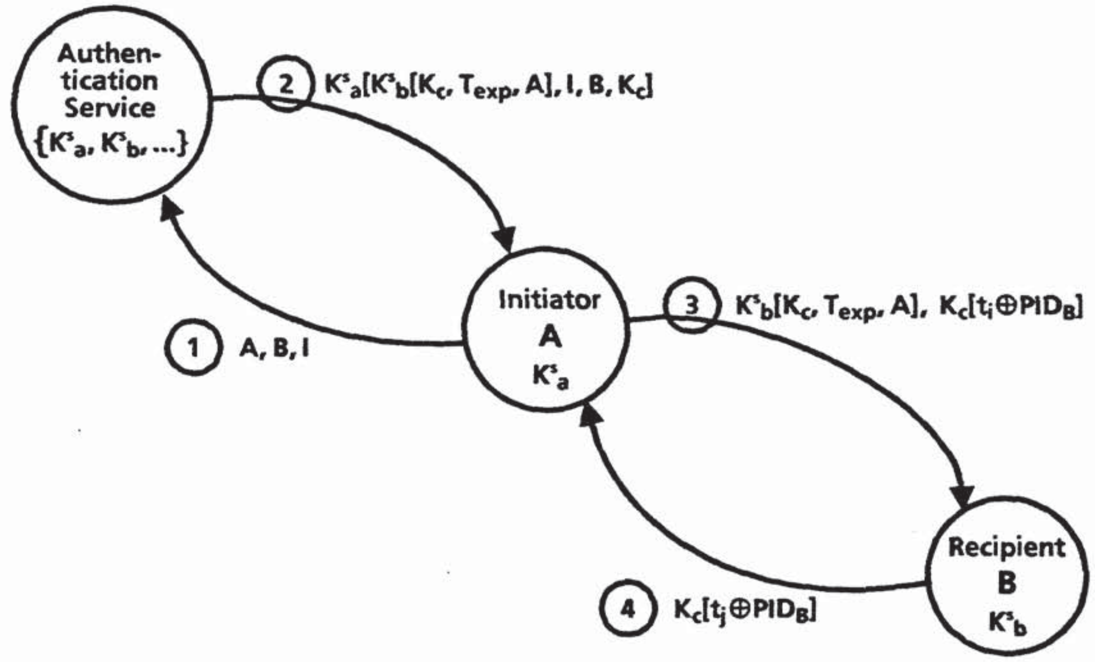
Strong authentication

In slightly simplified terms, the strong authentication works in the following way:

Assuming that Party A (the initiator) wishes to identify itself to Party B (the recipient), Party A first uses the Authentication Protocol to contact a server on which an Authentication Service has been implemented, telling it the names of both parties and supplying a random number called the nonce (Step 1, see Fig. 5-2). The Authentication Service returns several pieces of information to Party A, all of it encrypted so only Party A can decrypt it (since the Authentication Service knows Party A's encryption key). Included in the return are the nonce, the name of Party B, a conversation key, and another set of information called credentials, which are further encrypted so that only Party B can decrypt them (Step 2).

Party A checks the nonce and Party B's name, comparing it to what had been sent. If it is the same, Party A can be reasonably certain the transaction is not a replay of a previous transaction. The credentials, encrypted with Party B's key, contain the conversation key, the name of Party A, and an expiration time that defines how long the credentials will be valid. The conversation key (which was also sent encrypted with Party A's key) is used to create a verifier by encrypting a time value (derived from the Time Service by exclusive OR of time stamp

0064


Authentication Service
$\{K^s_a, K^s_b, \ldots\}$
2 $K^s_a[K^s_b[K_c, T_{exp}, A], I, B, K_c]$
1 A, B, I
Initiator A K^s_a
3 $K^s_b[K_c, T_{exp}, A],\ K_c[t_i \oplus PID_B]$
Recipient B K^s_b
4 $K_c[t_j \oplus PID_B]$

Legend

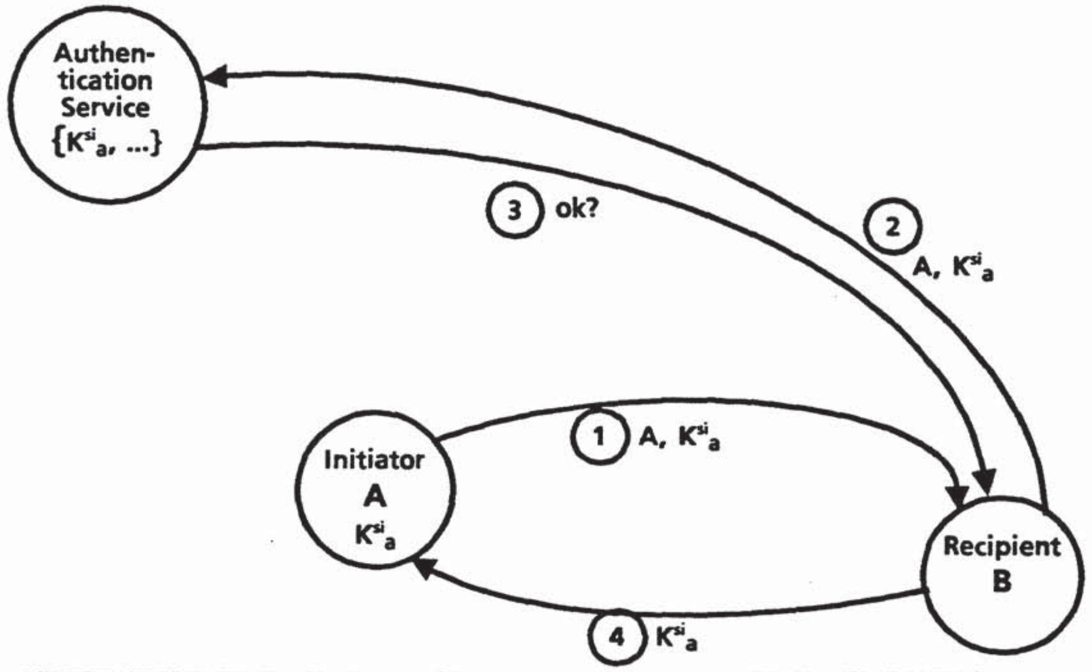
A, B	=	the fully qualified names of the initiator, **A**, and recipient, **B**
I	=	the nonce
K^s_a, K^s_b	=	the strong keys of **A** and **B**, respectively
K_c	=	the conversation key
T_{exp}	=	the expiration time for a set of credentials
$K_y[x]$	=	the value x encrypted with the key $\mathbf{K_y}$
t_i, t_j	=	a time stamp obtained from the system clock at time i, j
PID_X	=	48-bit processor ID of X, padded after the least significant bit with 16 bits of zero

Figure 5-2 **Strong authentication model**

with recipient processor ID). Party A sends this verifier, along with the encrypted credentials originally supplied by the Authentication Service, to Party B (Step 3). Party B decrypts the time value and, by comparing it to the current time, can ascertain that the transaction is not an old one, and that the credentials (which identify Party A) can be trusted. Finally, Party B returns a verifier to A to ensure that Party B is not an imposter (Step 4).

Simple authentication

The Authentication Protocol also provides for a far simpler level of security. This acknowledges the fact that not all system resources are properly equipped to implement the tightest possible security, nor is it necessary or economically reasonable for them to do so. Since simple authentication is encoded by hashing, an initiator requires no interaction with the Authentication Service to manufacture the simple credentials. The simple authentication process is shown in Fig. 5-3; the initiator A sends its name and simple key to the recipient B (Step 1) which verifies A's simple key with the Authentication Service to assure of A's identity (Steps 2 and 3). Unlike strong authentication, the simple authentication verifiers are always the same, which means that an eavesdropper may obtain a valid credential-verifier pair and pose as the initiator. Also, the returned verifier (Step 4) provides no assurance that the recipient B is not an imposter.



Legend	
A	= the fully qualified name of the initiator, **A**
K^{si}_a	= **A**'s simple key
ok?	= a boolean which indicates whether or not **K^{si}_a** is **A**'s hashed password

Figure 5-3 Simple authentication model

The Authentication Protocol also provides for a privileged user, one capable of managing other users' keys and undertaking certain sensitive administrative procedures.

There also exists an Authentication operation called BroadcastForServers, which is used to locate instances of the Authentication Service in the internet. It is invoked using the Packet Exchange Protocol. A broadcast is made on the designated network and each Authentication server on that network returns its network address.

The security provisions of the Authentication Protocol, coupled with the access control mechanisms designed into the appropriate resources, ensure the necessary protection while allowing XNS to retain its openness and availability of resources. The Authentication Service usually co-exists with the Clearinghouse Service on the same server hardware.

0066

Time

Various components of a distributed system must frequently obtain the current time. For example, file services need to record the time when a file is read or written, electronic mail messages need time stamping, and time may be needed for authentication purposes. To facilitate the acquisition of such information in a reliable and unambiguous way with a world-wide scope, a single time protocol and time standard is used throughout Xerox Network Systems.

Time Standard

The Time Protocol uses the Time Standard which defines time as a Courier datatype, available for use by any of the Courier based remote procedures. Its standard representation is also used in the definition of other non-Courier based protocols. A data object of the type Time is a 32-bit number which represents the current time unambiguously in seconds. This representation gives the time starting from 12:00:00 AM (the beginning of the day), 1 January 1968. The 2^{32} seconds (about 136 years) represented by Time extend into the 22nd century before any ambiguity with a past date is encountered.

Time Protocol

The Time Protocol specifies the manner in which a Time Service makes the current time available to its clients on other system elements. (These clients may be workstations, terminals or servers.) It is built upon the Internet Packet Exchange Protocol. The response packet from the Time Service provides the current time, along with its time zone, and information about when Daylight Savings Time is observed at its location, as shown in Fig. 5-4. The additional information is for client convenience which may (but need not) use this information in formatting the time for human consumption.

This global approach to the handling of time is one of the reasons why XNS systems can be implemented uniformly across geographical boundaries which would restrict systems based on architectures designed with a less global perspective.

Character code and fonts

Information interchange on a world-wide internet requires a fundamental rethinking of the encoding of characters, the basic information element in written languages. In the United States and other English-speaking countries, the 7-bit ASCII (American Standard Code for Information Interchange) is widely used in all varieties of workstations, terminals, and computers. ASCII is capable of representing basic English upper- and lower-case letters, numerals, punctuation, com-

0067

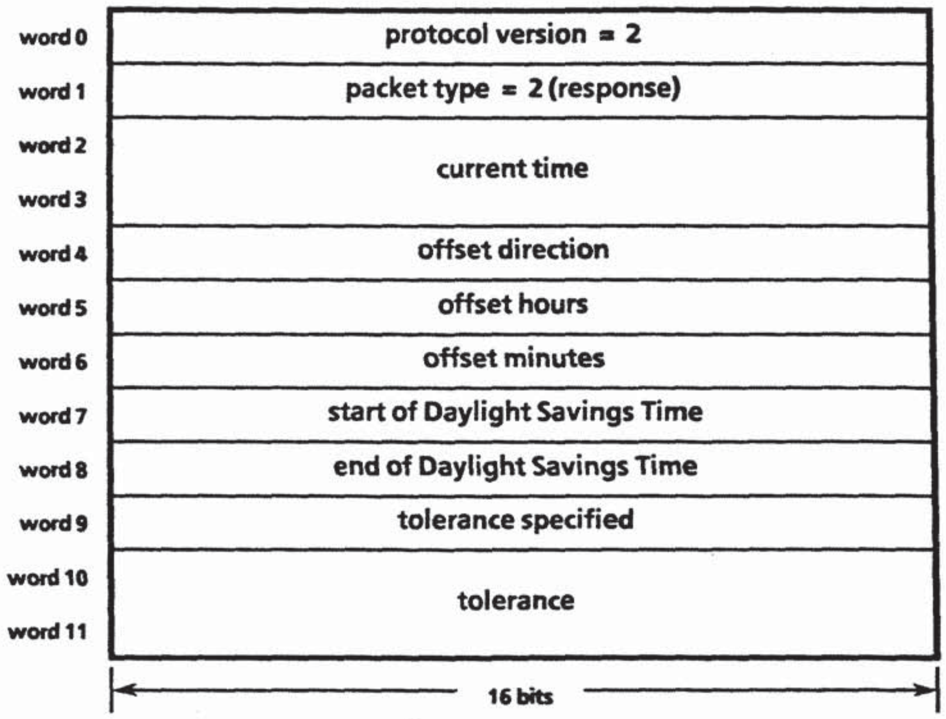


Figure 5-4 **Layout of data field of Time Service response packet**

monly used graphic characters and communication control codes totalling 128 different characters.

But 128 characters are hardly enough to deal with the printing needs of English, let alone other languages. To correct this situation, ASCII has been extended to 8-bits to define an additional 128 characters. The International Standards Organization (ISO) has also adopted a similar 8-bit character code standard commonly referred to as ISO 646. These 8-bit character codes, while a vast improvement over 7-bit ASCII, are still inadequate. Special accents used in many European languages, Greek, mathematical symbols and printing-oriented characters quickly exceed the 256-character capacity of any 8-bit code. When the needs of the many different languages in the world are taken into consideration, any hope for using a restricted code space vanishes. Chinese and Japanese each have requirements for thousands of characters.

The ASCII and the ISO 646 standards, as well as the numerous other national and international character codes that are being used, permit only limited information interchange. However, it is vitally important to use these standards to communicate with the millions of workstations, terminals, and computers that use them. A global information system must intelligently deal with all of these standards in a uniform and unambiguous way, while retaining compatibility with existing codes.

0068

Character code standard

The Xerox solution to this problem is a character encoding system which normally conforms to the ASCII and ISO 8-bit character codes, but expands to a 16-bit code when necessary. The 16-bit coding scheme permits 65,535 different character codes which is sufficient for encoding all of the commonly used human languages. However, should future requirements warrant, mechanisms exist to expand the character code space beyond 16-bits.

Multilingual capabilities

The April 1984, published version of the Character Code Standard (also known as the Xerox Character Code) specifies the character code assignments for Greek, Cyrillic, and Japanese characters in addition to the Latin character set defined by ISO 646. Additional characters are being added to this set for new technical domains, and the alphabets of more languages, including Arabic/Farsi, Hebrew, Hindi (Devanagri), Chinese, and Korean. Updates to the standard will be published periodically to reflect such additions. The fact that all the different languages can be conveniently represented in a single character code standard simplifies the design of multilingual document generation and printing systems. Such multilingual capabilities are available in many Xerox products. They allow a user to mix text in different languages, use any of the Greek symbols in mathematical equations, and put everything in the same electronic document in a uniform and consistent manner.

Character Code model

The Character Code Standard may be used worldwide for any number of applications including communication protocols, electronic printing, electronic mail, keyboard input, document editing and document interchange. The standard assigns a unique, unambiguous and absolute numerical code to each semantically different character to permit efficient storage and processing while also ensuring proper interpretation of information. To understand how this assignment is made, the 16-bit character code space may be viewed as a series of 256 blocks of 256 codes each. Each such block is called a character set. Each 16-bit character code may be viewed as consisting of two 8-bit bytes, the first of which is the character set code, and the second an 8-bit character code within that character set, as shown in Fig. 5-5. The character codes are assigned so that characters within a single character set tend to be related to each other by traditional usage. Thus, all of the characters in Character Set 0 are for the Latin set. Some alphabets, such as Japanese Kanji, which involves well over six thousand characters, require many character sets, as shown in Fig. 5-6.

High-quality typography

The Character Code Standard also allows for high quality electronic typography by providing rendering characters which are variant representations and special sequence of graphic characters, such as ligatures. Other entities, such as logos and

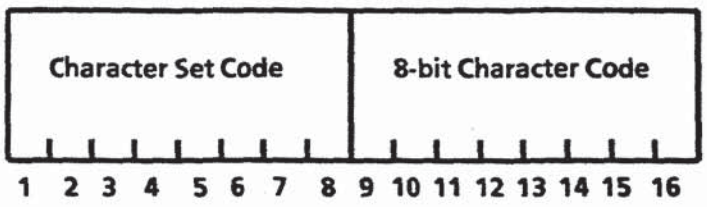


Figure 5-5 **16-bit character code**

people's signatures, are not defined by the standard. A portion of the rendering code is reserved for private use to accomplish their conversion into hard-copy on an electronic printer. These provisions make high quality typography easier on a system-wide basis.

Encoding efficiency

It would seem that 16-bit codes would take twice the storage and transmission time as 8-bit codes. Normally, this would be true, but the Character Code Standard also defines a string encoding standard which compresses the 16-bit codes into 8-bit bytes on a one-for-one basis (i.e., one 8-bit byte for each 16-bit entity), thus providing versatility with little or no loss in efficiency. Moreover, all sequences of 8-bit ISO 646 characters may be stored or transmitted as they are. For languages such as Japanese, which normally require the use of a 16-bit code, text may be stored and transmitted as a sequence of 16-bit codes.

Compatibility with other codes

Every effort has been made by Xerox to be compatible with the large number of national and international code standards. For example, the Character Set 0 assignment is fully compatible with the ISO 646 and the ASCII standards, and the Japanese Kanji assignment is in accordance with the Japanese Industrial Standard Code JIS-C-6226. However, it should be recognized that the various national and international standards are not consistent among themselves, so some compromises must be made for the goal of full compatibility. Nevertheless, the Xerox Character Code Standard represents one of the most consistent and successful international implementations available.

Font architecture

A character's numerical code expresses its identity, its content, and its semantics, but this does not fully determine a printed or displayed character's visual appearance. For example, the character with the same unique code can be printed in a different size, style or typeface, and look very different. The term font is used to define a particular collection of characters of a typeface with their unique orientation, size, posture, weight, and other attributes. Together, the character code and the font define the visual appearance of a character. A font architecture, defining font representations, font file formats, font names, and font services is important to assure that documents are displayed and printed with high quality and uniform consistency; that is, a document "looks" the same

0070

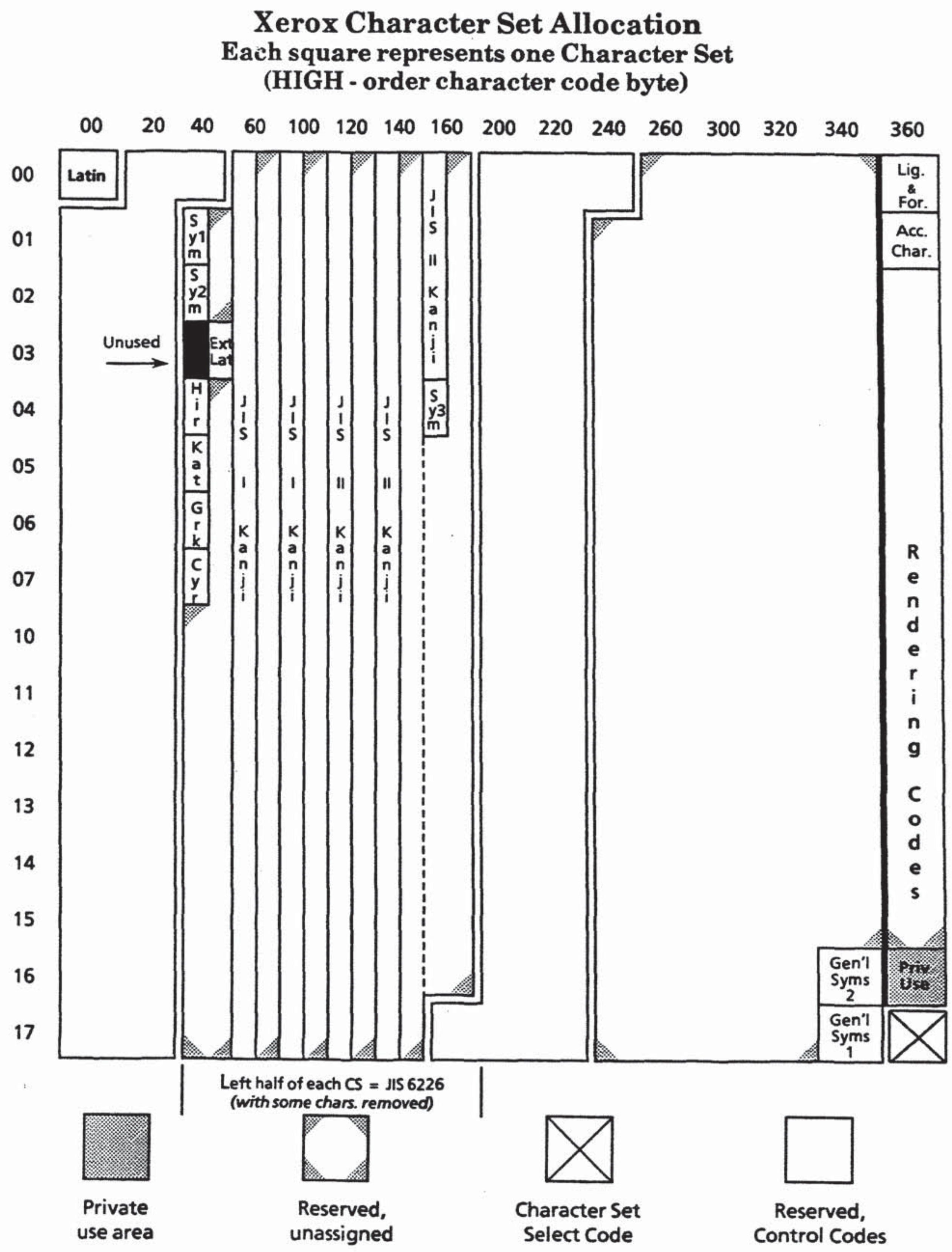


Figure 5-6 **Xerox Character Set allocation**

whether it is displayed on a workstation screen, typeset on a typesetter, or printed on any number of electronic laser printers available from Xerox and others.

Font representation

Fonts may be represented and stored (for later printing or display) in a number of different forms. Two of the most common forms of representation are bit map representation and contour representation. The bit map representation generally offers higher quality and greater processing efficiency, while the contour representation (often stored in spline form) generally offers greater flexibility with regard to choice of size, orientation, and resolution ("dots per inch"). The Xerox font architecture allows for both types of representations and gives the user the flexibility to choose the representation that best suits the needs of a particular application. Most Xerox products use the bit map font representation.

Font file format

A font file is a set of font records including the digital representation for the characters and control information for processing. A standard font file format provides for easy interchange of fonts between all font-using products (work-stations, printers, etc.) and thus increases font consistency between products.

Font names

Xerox has standardized a font name structure for its own use which is defined in the Print Services Integration Standard. In this structure, used in Interpress, the first three identifiers of a font name vector uniquely identify a font. These identifiers are: Naming authority (Xerox, NBS, etc.), character code (e.g., Xerox Character Code, EBCDIC, etc.), and typeface (e.g., ItcGrammd-Demibold-Oblique-DesignSize9pt). Xerox also pro-vides a Font Name Registry for those wishing to register their fonts with Xerox. The typeface identifiers themselves have no mandated structure, but guidelines are suggested to help in achieving greater uniformity.

Font service and protocols

A font service supplies printers, workstations, composers and other font using devices with fonts and/or font metrics upon request. A font service may also be able to initiate a distribution of fonts to font-using network devices. A font server is a network service which provides this function using a font protocol. A schematic of the use of a font service by workstations and printers is shown in Fig. 5-7. According to this model, an electronic printer or electronic printing system may request bitmap or contour fonts from the font service (Step 1A), which supplies these fonts from its database (Step 1B). The printer (or any other system element) can (if so desired) use the font service for archiving bitmap fonts (Step 1C), which it may have generated from contour fonts. Alternately, the font service may be used by a workstation to obtain screen fonts (Steps 2A and 2B). A composition system such as a mainframe or publishing system connected to the network can also use the font service to obtain the required font metrics (Steps 3A and

3B). The interactions with the font service use the font protocol, and fonts and metrics are transmitted using standard font file formats. The use of this model enables greater availability of fonts and consistency in their usage.

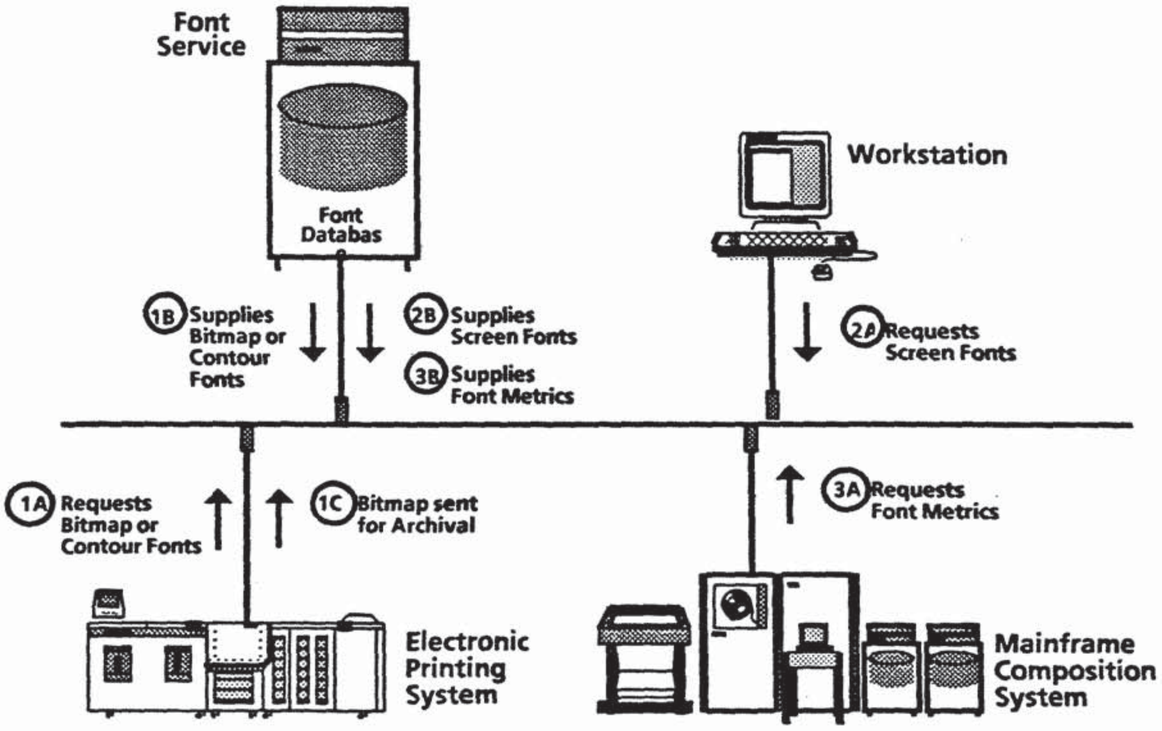

Font Service
Font Databas
Workstation
1B Supplies Bitmap or Contour Fonts
2B Supplies Screen Fonts
3B Supplies Font Metrics
2A Requests Screen Fonts
1A Requests Bitmap or Contour Fonts
1C Bitmap sent for Archival
3A Requests Font Metrics
Electronic Printing System
Mainframe Composition System

Figure 5-7 A font service

0074

6. GATEWAY ACCESS

One of the more challenging problems in a network architecture is interfacing other systems that were designed according to different, incompatible architectural standards. Yet the problem must be solved because with the proliferation of all kinds of computers and networks, users are demanding the integration of diverse systems, including office information systems, conventional data processing systems, manufacturing and engineering information systems, and retail and financial transaction processing systems.

If one system is to integrate successfully with another, there must be a way to convert the communication protocols, document and data formats, file structures, commands, and control functions used by another system. In XNS, the responsibilities for these conversions are located in a gateway, a hardware/software subsystem intended to bridge the differences between two incompatible systems.

Recognizing that the problem of interconnecting with other systems will continue to arise indefinitely, the XNS designers chose to implement a variety of gateway services. These gateway services provide access to and from an XNS system for a variety of other systems and devices, including those that use TTY, VT-100, and the IBM 3270 protocols. The Gateway Access Protocol is used in communicating with other non-XNS systems.

Gateway Access Protocol

The basic functions performed by the Gateway Access Protocol (GAP) are to move information, support other communication models, and support terminals.

Communication support

GAP supports many communication models, including the document transfer model for electronic mail, transaction-oriented model for remote data base access, and an interactive model for interface to a mainframe data processor. To achieve this, GAP provides flexible control of the unit of data transfer and the frequency of transmission activity. GAP does not, however, deal with the content of data which is application dependent. It provides *information transcription* but **not** *information translation*. Information translation may be provided by either party using GAP, or via a separate conversion service, such as a Document Interchange Service in gateway or workstation products. In supporting other communication models, GAP adapts to the other protocols when different from XNS. Thus many of the commonly used communication protocols are supported by the XNS gateway services.

The Gateway Access Protocol (GAP) is an application-level protocol that makes use of the Courier and Internet Protocols (specifically, Sequenced Packet Protocol) to interconnect an XNS system with a non-XNS system or device. GAP uses these protocols to issue or receive customized command and data sequences that exactly replicate the command and data sequences used by the target system or device. Courier is used to establish a session with the target and the Sequenced Packet Protocol is used to transmit and receive the appropriate bit and character sequences.

Virtual terminal support

The GAP creates a logical appearance of communication compatibility to the other system or device. For interconnecting with a remote mainframe system, this appearance emulates particular devices, such as TTY, VT-100, or IBM 3270 terminals, which the mainframe supports. GAP also makes it possible for non-XNS terminal devices to interconnect with an XNS system and access XNS services. For this reason, the Gateway Access Protocol is also referred to as the Virtual Terminal Protocol (VTP)

Gateway services

In order to integrate successfully with other devices and systems, XNS is equipped with a variety of gateway services that use the Gateway Access Protocol. For example, the External Communications Service is used for integration with mainframe computers and the Interactive Terminal Service provides access to non-network terminals.

Fig. 6-1 illustrates how a communication server is used to make the physical and logical connections to a remote terminal and to a large mainframe. A workstation on the Ethernet to which the server is connected is then capable of creating documents or records and sending them to the mainframe and/or accessing information stored at the mainframe. The remote terminal can input document content to the XNS system for subsequent editing, storage, or printing; it can also send and receive mail. The communication server can initiate and maintain many simultaneous connections between the internet and the remote systems. A Communication Interface Unit may also be used in conjunction with the communication server to provide additional communication ports.

0076

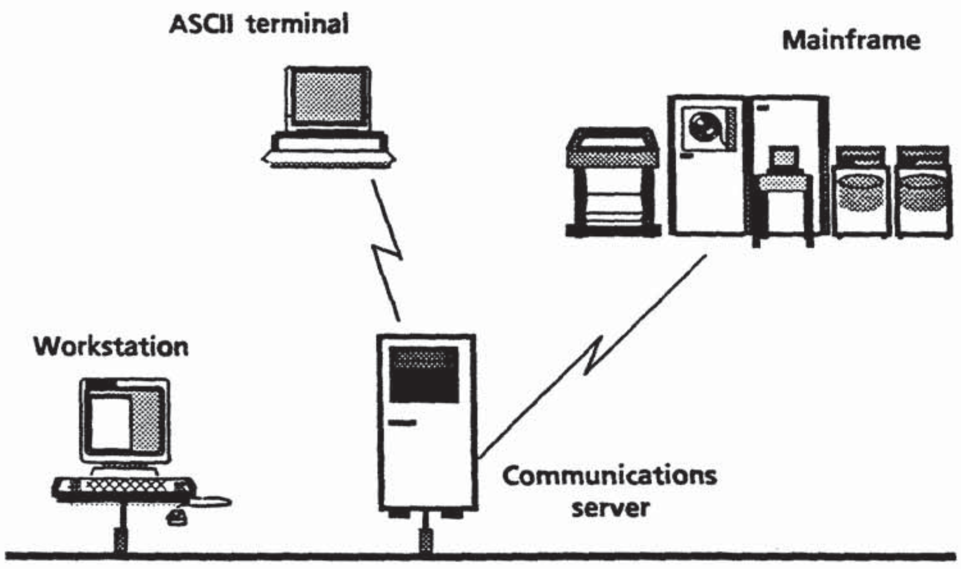


Figure 6-1 Gateway access to/from non-XNS systems

The many different types of gateway services are generally implemented as separate software products which may be co-resident on the same server or have separate server hardware.

External Communication Service

Access to mainframe computers

XNS workstations use the External Communication Service (ECS) to access non-XNS mainframe computers. This service works in concert with terminal emulation capabilities in Xerox workstations which simulate the appropriate terminal protocol. To support these terminal emulation capabilities, the ECS performs three functions: It communicates with the host in the host's native protocol; it communicates with an XNS workstation in XNS protocols; and it provides the sharing of communication resources to connect the host to all XNS workstations.

Terminal emulation in XNS

When a workstation user requests an emulation session with a host computer, the workstation uses the Clearinghouse to locate an External Communication Service that supports connections with that particular host. The workstation then connects to the ECS which initiates a session with the remote host and performs conversions between XNS and the remote host protocols. The workstation presents a user interface equivalent to that of the terminal being emulated. The protocol conversion provided by the ECS allows information originating either in the mainframe computer or anywhere on the XNS internet to be transferred and integrated into the mainframe's environment for use in various applications.

A workstation is not restricted to using only local External Communication Services. Since the XNS protocols support communication anywhere on the internet, a workstation that supports the XNS protocols can use any ECS on the internet.

IBM 3270 emulation

Major portions of IBM's extensive repertoire of mainframe computer software are designed to work with IBM's 3270 display terminal systems. Hundreds of thousands of these terminals, and various plug-compatible equivalents, have been manufactured in both clustered and standalone versions.

During IBM 3270 emulation, the ECS supports the IBM 3270 Binary Synchronous Communication (BSC) or the IBM 3270 System Network Architecture (SNA) protocols. An IBM 3276-2 cluster controller is emulated when using BSC protocol and an IBM 3276-12 cluster controller is emulated when using SNA protocol.

Like the IBM 3276 cluster controllers, the ECS can support up to eight concurrent emulation sessions. Unlike IBM cluster controllers which require their attached devices to be within the vicinity of the controller, the XNS emulation sessions may originate either from a single workstation or from several workstations, anywhere on a worldwide internet. Software options enabled at the XNS workstations allow their users to interact with an IBM host computer in an IBM 3270 emulation mode. This feature is also available for the non-IBM mainframe and large minicomputer systems which are compatible with the 3270 terminals. Thus by emulating the IBM 3270 terminal, an XNS workstation can gain access to a very large number of mainframe computers.

VT-100 and TTY emulation

The ECS also contains the software option that supports emulation of VT-100s and TTY-type terminals. TTY emulation is generally via a physical dial-up connection for each session. The protocol used between ECS and the host is Asynchronous ASCII. Workstations emulating teletype (TTY) terminals present interactive displays that are equivalent to the standard teletype terminal. The ECS maintains the bi-directional communications necessary for the emulation session by transforming the ASCII protocols into XNS protocols, and vice versa. This capability makes it possible to interconnect XNS workstations with most non-IBM general-purpose computers.

Support of incoming calls

The ECS also supports incoming calls from personal computers or TTY-type terminals that are not part of XNS. Users can access the Interactive Terminal Service through a dial-up connection to an ECS port.

Interactive Terminal Service

The Interactive Terminal Service (ITS) allows users of workstations and terminals not directly connected to the XNS internet to access XNS print, file, mail, and other resources. Any personal computer or standard ASCII terminal that can initiate a dial-up communication session using the asynchronous protocols can communicate with the ITS and use the network services on the internet. Nearly all computers with standard RS-232-C communication ports are able to operate that way. This includes Xerox Memorywriters, personal computers, professional workstations, and text processing systems.

ITS provides an interactive user interface to give users access to XNS resources. Users at remote devices interact with this user interface to store and retrieve documents on a file service, print documents on electronic printers, or send and receive electronic mail. During a communication session a user has the illusion of being directly coupled to the internet. That's because ITS communicates with the internet using XNS protocols once it determines what a user wants to do.

ITS document interchange

To enhance the exchange of messages and documents between terminals and workstations, ITS supports a variety of document formats including 860 document format and plain text.

If the intended recipient is a workstation on the internet, ITS can be instructed to convert the document into the format suited for that workstation.

ITS mailing

After the user establishes a connection and logs on to the ITS, he or she can issue commands through the mailing interface and request various mail operations. Messages can be composed and sent to different individuals and distribution lists, or the user can retrieve and edit the messages received.

Messages are not limited to text entered while the user is connected to the ITS. The user can include a previously prepared document in a message. This means a user can work on a document and send it over the phone network to anyone who has direct or indirect access to the internet. When a document is sent to a Xerox workstation, the recipient can improve it with the workstation's advanced text and graphics capabilities and then print, file, or mail the document to other users on the internet.

ITS filing

ITS also supports file transfer between non-XNS devices and the File Service, using the XModem asynchronous communication protocol. Users can create files in any format, using their own word processing or spreadsheet software. These files can then be transmitted from the device to ITS and stored in their original format on the File Service. The stored files can be displayed and edited by XNS as well as non-XNS devices.The storage and retrieval operations using the XModem protocol allow output from multi-vendor computing devices to be shared on the network.

ITS printing ITS supports the use of XNS printing for documents in Interpress, the Xerox electronic printing standard. Conversion to Interpress is also available for a limited set of formats.

850/860 Gateway Service

The 850/860 Gateway Service allows users of non-networked communicating Xerox 850 and 860 text processors and the Kurzweil Intelligent Scanning Systems, to exchange information over telephone lines with users on the internet. By dialing the 850/860 Gateway Service, non-networked users can exchange documents with users of workstations on the internet, or with other remote workstations. This flexible extension of the internet is particularly valuable in organizations with 850, 860, or compatible workstations located in small branches of the organization.

The Gateway Service communicates in both 850/860 and XNS protocols. Because XNS can handle information in 860 format, the 850/860 Gateway Service acts as a transfer agent between the 860 and the internet while preserving the enhancements in the original document. If documents originate at an 850, the 850/860 Gateway Service converts the 850 format to 860 format before making the transfer.

Remote Batch Service

The Remote Batch Service (RBS) enables XNS users to exchange documents and transfer files with devices that implement the IBM 2770, 2780 and 3780 Binary Synchronous Communication protocols used by IBM Remote Batch Terminals, as well as by word processing and data processing terminals that emulate them.

The RBS basically performs the following tasks: It translates between document formats during the interchange of documents, providing compatibility with XNS; it acts as a third-party transfer agent for formats which it does not understand; and it transmits XNS files to a mainframe computer for storage and archiving without any loss of structure or information.

The RBS can be used for document interchange, document transfer, or document archiving applications with a mainframe. Tasks sent to a mainframe require an additional document that specifies the Job Control Language to execute the task on the mainframe.

7. FILING

Word processing equipment and most personal computers store documents in electronic form on flexible diskettes holding several hundred thousand characters. Since diskettes can be removed from a machine and stored off line, there is almost no limit to the number of characters that can be recorded this way. But storing documents on diskettes is inconvenient when they must be shared with others, especially those in remote locations. And off-line storage can be troublesome if no one can locate a particular diskette.

Documents stored locally at an XNS workstation may also be stored remotely in a file service. High-capacity disk files are able to store documents to make them available to other authorized users on the internet. This internet-wide file sharing capability provides many benefits to a user such as access to the most up-to-date information, ability to work jointly on a document even from remote locations, and creating document databases pertaining to activities of a small group or an entire organization.

File service requests

A file service deals with action requests from two different kinds of sources. The first represents the user who wishes to access the file server. This access must be provided via a protocol; in this case, the Filing Protocol, which is layered above Courier, as are all XNS applications protocols.

The second source of action requests comes from system administrators, persons who desire information about the current state of the File Service or who wish to make administrative changes (e.g., creating a new file drawer) authorized by their credentials. System administrators may perform their actions through a local TTY terminal or remotely via the Gateway Access Protocol, also layered above Courier. System backup operations are initiated through this interface as are many operations which authorize system access. These back-up and archival features of a file service are generally not visible to the client, and are not specified in the Filing Protocol.

Filing Protocol

In XNS, the interaction between clients and file services is defined by the Filing Protocol. The Filing Protocol is both a guide for using a file service and a specification for the implementation of such a service. It is not a description of a particular implementation of the protocol. This protocol provides a general filing facility to support a wide variety of applications. However, it is not intended to directly support network administration functions, printing, electronic mail, or other distributed activities. These are subjects of other specifications.

Filing Protocol model

The Filing Protocol follows a session-oriented model in which a client interacts on behalf of a user (which may be a human or other entity such as another service); and must log on before using the file service. If the log on is successful, the service establishes a session during which the client interacts with the service. When interaction is complete, the client logs off to terminate the session. The client must log on again before any further interaction may occur.

Sessions may vary greatly in duration. In some patterns of use, a session is established to perform a single operation and then terminated. In others, a session may last a very long time even though it is largely inactive. There may be several sessions simultaneously in existence for the same user whether or not they were established by the same client. The file service reserves the right to terminate a session at any time that a remote procedure call is not in progress. This might occur if a session remains inactive for a long period or if the system element supporting the file service has to be shut down.

The file service stores and operates on files. It does not control the content of what it stores. Documents in any format may be stored in or retrieved from a file service as shown in Fig. 7-1. If a document in ASCII format is stored and then later retrieved by a non-ASCII device, the document will essentially be unintelligible. A document conversion service must be used to convert information encoded in one format into the format used by a dissimilar device.

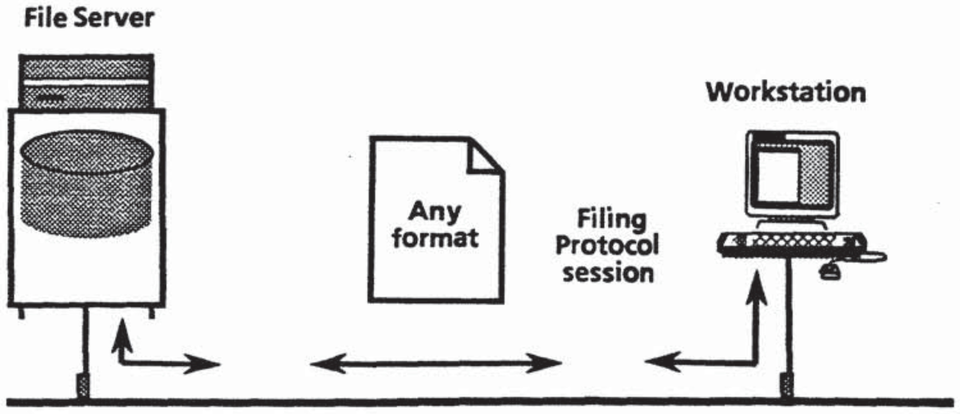


Figure 7-1 Transmitting documents between a workstation and a file service

0082

Filing Protocol structure

The Filing Protocol may be viewed as being composed of four sub-layers, one above the other, which progressively provides additional functionality (see Fig. 7-2). These four sub-layers, from lowest to highest, are:

1) Session; 2) File; 3) Directory; 4) Search

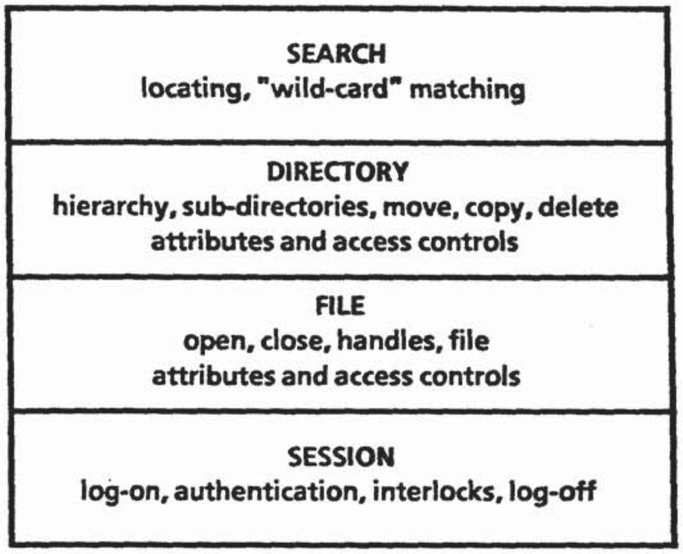

SEARCH
locating, "wild-card" matching
DIRECTORY
hierarchy, sub-directories, move, copy, delete
attributes and access controls
FILE
open, close, handles, file
attributes and access controls
SESSION
log-on, authentication, interlocks, log-off

Figure 7-2 Filing Protocol structure

Session sub-layer

The session sub-layer is responsible for providing a context within which a sequenced set of related action requests from the client can take place.

It begins with a process which authenticates to the File Service that the client is who he claims to be and which authenticates to the client that the service is indeed the one intended (to prevent impersonation of a service). This process utilizes the Authentication Service via the Authentication Protocol. Encryption (DES) is utilized to make this authentication process secure.

The authentication determined at the beginning of the session is subsequently utilized to gain permission to access files.

The session also provides a context for a system of software interlocks which coordinate the access of two sessions competing for access to the same information. Without such interlocks, shared information can easily become inconsistent.

File sub-layer

The files sub-layer is responsible for the *file* data structure and, within the context of a session, for implementing those operations which deal with the individual file. The Open and Close commands provide a handle (an identifier used in client-service interaction) on a file within the context of the session.

A file consists of content and attributes. The content is a sequence of octets (an octet is a group of 8 bits of binary information not necessarily representing a graphic character).

The File Service does not attempt to place any interpretation upon the octets of a file's content (with the exception of when the file is a directory, as explained below for the directory sub-layer). The Retrieve and Replace commands transfer all or part of the content of a file.

Attributes

Attributes are additional information about the file: they are partitioned into interpreted and uninterpreted attributes. Uninterpreted attributes are only for the use of the client and are not processed or acted upon by the File Service other than to record them and to return them to the client. The interpreted attributes, on the other hand, have meaning to the File Service and are, in general, maintained by the File Service. While interpreted attributes are all defined at the files sub-layer, some are maintained by the files sub-layer and others by the directory sub-layer.

At the files sub-layer, some of the interpreted attributes are:

— a binary file identifier
— a file name as a human sensible character string.
— a flag indicating temporary or permanent
— a version number
— a checksum
— the file size (in octets)
— access control information
— the time and identity of the client performing the operation.

Directory sub-layer

The directory sub-layer organizes the files into a hierarchical tree as is customary. Some additional attributes do the actual function of binding the files together into the tree structure.

Many commands deal with a whole sub-tree, manipulating a directory and all of its files including its sub-directories at all levels. These commands include Move, Copy, and Delete. These commands are extremely powerful because of the large scale reorganization of the files which a single command can invoke.

Search sub-layer

Finally, a Set command exists which can identify a whole set of files related by similarities in their human understandable file names. This set is identified by providing a file name-like pattern containing various "wild card" characters. The search sub-layer generates a list of all files whose name matches the pattern and provides this list to the client, who is free to do what he chooses with it.

This mechanism is often utilized by a user interface to support a "see and point" mechanism of file identification. By displaying a list of candidates, a user can select the proper file.

Printer Subset of Filing

It is sometimes desirable to allow remote access to a printer's file system so that files may be stored there and used at a later time. Forms overlays, masters to be referenced from other masters, and font files are examples of objects that a client may want to store on a printer. The Printer Subset of the Filing

0084

Protocol is designed so it can be implemented as an adjunct to a Print Service. Thus, it provides much more limited functionality than the Filing Protocol. It is a strict subset of the Filing Protocol. This means any behavior defined in this subset protocol is identical to the behavior defined in the Filing Protocol. Clients of this protocol are guaranteed that they may talk to services implementing the full Filing Protocol simply by using a different service identifier.

Data transfer

For those filing procedures that intrinsically require the transmission of a large amount of data, the Bulk Data Transfer Protocol is employed. Basically it works as follows: Between the call to a remote procedure and the return from that procedure, the sender (either client or service) uses the bulk data transfer mechanism to send to the receiver the attributes or the content of the designated file(s). Note that the Bulk Data Transfer Protocol (with its third party transfer) allows the data to be sent to, or retrieved from, a system element different than that of the client.

File Service

The XNS file services give users access to shared files anywhere on the internet. This sharing of information is not limited to XNS workstations. Any authorized user may access the file service from any terminal, personal computer or even an electronic typewriter (such as the Xerox Memorywriter) which has the required communications interface. This access is enabled via the Interactive Terminal Service.

File organization

File sharing is made easy by the organization of the file service. Files in the file service are organized hierarchically; a major directory (file drawer) at the top can contain several sub-directories (folders), each of which can contain additional layers of sub-directories or documents (record files, spreadsheets, mail notes, etc.) with no limit to the number of descendant directories. (There is, however, a limit on both the file drawer size and the total amount of storage available on the file service.)

All file drawers, folders, and documents are given names, and all levels of hierarchy contain the names that are descendants of the preceding level. The file service also attaches a unique identifier to each file which indicates its position in the hierarchy and provides efficient access to the files.

In addition to a name, the file service associates a number of attributes with each object in the hierarchy. Attributes describe the current characteristics of a file and help to distinguish it from other files, but are not part of its content. Examples of attributes are the date/time stamp associated with the file at the time it is entered into the file service, the name of the creator, the access privileges associated with one or more individuals or groups and the number of pages allotted to the file drawer. These attributes are used by the file service to provide a number of benefits to the user. For example, files can be sorted in a folder by creation date or by name according to the user's needs.

The file service automatically sorts the contents of folders according to the attributes established for each. For example, files in a folder might appear in descending alphabetical order by name or in numeric order. A user can sort by ascending or descending alphabetical order, or by creation date, or have the folder unsorted. This flexibility makes it easy for users to organize and locate a particular folder or document.

Access controls

Since XNS is a distributed system (allowing users to access all services across the internet), access controls must be available to protect private or restricted information. The File Service allows access controls to be placed on file drawers. These controls determine who is permitted access and the type of access permitted.

Each file drawer has an owner registered in the Clearinghouse. The owner has complete access to the file drawer. The system administrator or the owner establishes access rights for other users to each file drawer. Only users listed in the file drawer's access list can read or alter a file. The owner can designate that some users be given only partial access rights (such as the right to read files), or more extensive rights (such as the right to delete, move, or add new files). Any person not listed will be denied all access to the file drawer and its contents. The controls placed on the file drawer apply to all of their descendant folders and files.

File service back-up

A regular back-up of the contents of a network file service is essential. Backup is a procedure in which the file service contents are copied onto other media or to another file service. This backup occurs at intervals determined by the system administrator and can take place unattended. If information is lost or accidentally deleted, it can be restored from the backup copy.

8. DOCUMENT INTERCHANGE

Document compatibility

Paper documents exist in many forms and so do electronic documents. Document-oriented information systems are typically made up of various kinds of editing devices such as word processors, advanced workstations, terminals of different kinds, and automated composition systems. Extensive distributed information systems also frequently provide interconnection to external devices and systems which are themselves capable of document creation and editing. This type of information system risks accumulating documents created by a wide variety of devices over a long period of time. Although such documents can be stored and retrieved by means of XNS Filing (in the case of XNS) without regard to their origin or content, any given document cannot normally be subsequently edited except *by an editor compatible with the editor that initially created the document.*

To "edit" a document means both to modify it and include any unmodified portion of it in other documents. Usually, either of two incompatible editors is unable to perform any operation on the other's documents.

For new systems this situation is usually only a nuisance. But as systems grow, new editors are introduced, existing editors are modified to the point where they will no longer accept documents produced by earlier versions, and documents are introduced into the system through exterrnal communication links. As the demand for cross-editing grows, a situation that began as a nuisance will become intolerable.

The reasons that documents produced by different editors are incompatible include the use of different data codes, different structural conventions, different techniques for indicating special conditions such as underlined text, the provisions made for explicitly declaring output characteristics, etc.

One potential solution to this dilemma would be to design a translation service that directly converts the private form native to one editor into a form that can be understood by another editor. This has two drawbacks, however. For even a modest population of different editors, this suggests a potentially large number of different translation services. And whenever design changes are made to one editor, corresponding changes must be made to each translation service that understands that editor's format.

Interscript

The Xerox approach to this problem is to adopt a document encoding standard that is sufficiently general so any document can be described in its terms, and to provide a way to convert the private forms of idiosyncratic documents to/from this new encoding. The encoding standard is thus an interchange standard that exchanges (not edits) documents between editors.

The encoding standard designed by Xerox is called Interscript. Fig. 8-1 shows document interchange using Interscript. Within the domain of a given editor, documents exist in the private form known to that editor. Examples include the Xerox 860 form, the Xerox Star form, or any of the innumerable forms generated by various word processing programs operating in a personal computer. Each of these private forms deals differently with the expression of a document's content and structure, and generally cannot be used by any other editor.

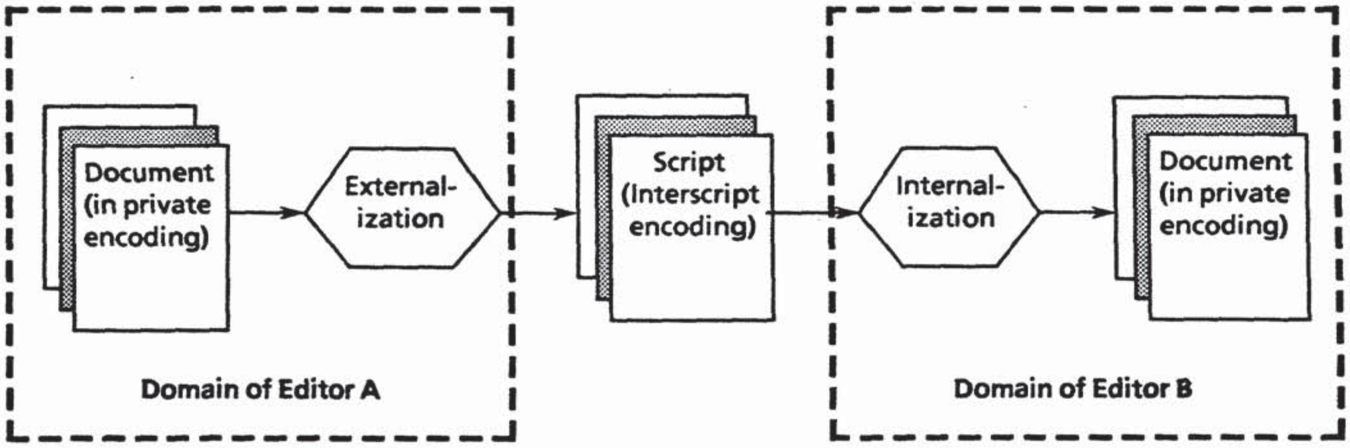

Document (in private encoding)
External-ization
Script (Interscript encoding)
Internal-ization
Document (in private encoding)
Domain of Editor A
Domain of Editor B

Figure 8-1 Document interchange using Interscript

The conversion of a document from its private form to a script is called "externalization." Similarly, the conversion of a script to a private form is called "internalization." These two processes are accomplished by computer programs typically (but not necessarily) associated with the domain of the corresponding editor.

One immediate advantage of this is economy of conversion. If a system has 15 incompatible editors and complete compatibility is desired, only 15 pairs (internalization plus externalization) of conversion routines must be written, rather than the

0088

105 pairs that would be required for the everything-to-everything conversion.

Another advantage is that by careful design, no restrictions are placed on document complexity because of interchange encoding. Problems may exist when a document produced by a feature-rich editor is edited by a basic editor. This, however, is not a problem caused by the interchange encoding.

The design of Interscript is particularly appropriate for a globally-distributed document management system in which documents will exist for years and where continuing changes —upgrades, extensions, new applications—are the norm.

Interscript is a kind of computer language for representing the content part, logical structure, and layout structure of documents (see Fig. 8-2). All three are important.

Content part: What you see when you look at a document.

Logical structure: The way a document is organized, typically but not always in a hierarchical structure (paragraphs subordinate to sections which are subordinate to chapters).

Layout structure: The way a document is to be rendered (headings and footings on every page, multi-column dimensions, margin settings, and the like).

Interscript as a document interchange standard

Some Interscript qualities that make it a general-purpose document interchange standard include:

Encoding efficiency A script encoded according to the provisions of Interscript does not take up a particularly large space. Such a script is well-suited for transmission and intermediate storage.

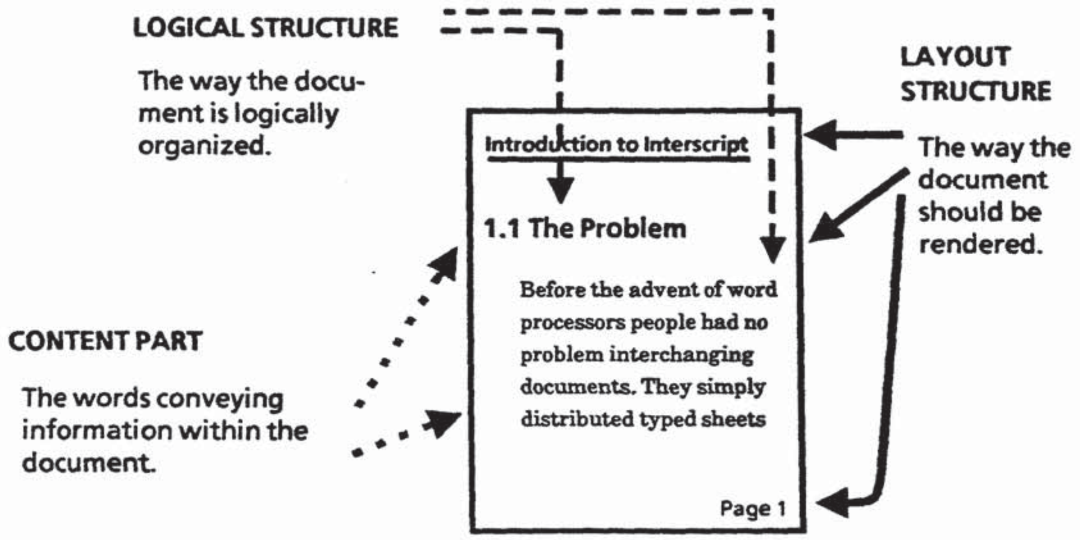

LOGICAL STRUCTURE
The way the document is logically organized.
LAYOUT STRUCTURE
The way the document should be rendered.
CONTENT PART
The words conveying information within the document.
Introduction to Interscript
1.1 The Problem
Before the advent of word processors people had no problem interchanging documents. They simply distributed typed sheets
Page 1

Figure 8-2 A document's constituent parts

0089

Open-ended representation

The power of the document interchange service is only as great as the comprehensiveness of the form into which documents are forced for interchange purposes. Interscript, therefore, is comprehensive relative to forms of data, types of edit functions, etc. But it is also extensible, so that as new constructs come into widespread use, it will be possible to incorporate them into the interchange coding without perturbing existing encoding provisions. This means that as Interscript is extended, old editors will still be able to work on old material (internalized from the Interscript encoding) without modification.

Document structure

A careful association is maintained between the content of a document and the form attributes applicable to various parts of the content. If original documents are hierarchical (e.g., a book with sections, chapters, paragraphs, etc.), that relationship is maintained; but where a hierarchical relationship does not originally exist, none is imposed. Some documents contain indirect (non-hierarchical) structural relationships between separate parts, such as the practice of printing informal comments in a formal specification in some specially-reserved type face, or the practice of interlocking certain graphical elements according to their geometries. These indirect structural relationships are preserved.

Transcription fidelity

Interscript-encoded documents are capable of generation and consumption in such a way that information is not changed. A document can be externalized into Interscript, then internalized back into the original private form, without incurring any change.

Interscript also makes it possible for the simple parts of complex documents to be understood by simple editors. This means many simple workstations may be used for routine editing chores. Although a basic editor might not be able to handle all the constructs of an internalized document, the user can do whatever the editor is capable of doing.

With Interscript's design, the internalization/externalization processes can be structured to allow the characteristics of an internalized document (to which the editor has made no changes) to flow through the editing process and remain intact when the document is once more externalized.

Interscript base language

The Interscript base language defines the representation of documents in a byte stream. This digital representation encoded using the Interscript base language is called a script. A script is an encoding of the document, not the document itself.

Interscript allows the definition of symbols with which some formal meaning is associated. These symbols are then interpreted by Interscript systems according to their precise

0090

meaning. The Interscript base language also has these features which together make it unique:

- Complex data structures may be represented in the form of a data stream;
- The contents of a document may be interpreted according to meaningful symbols recognized in the scripts;
- New abstractions that can be associated with symbols may be formally defined. One does not need a meta language to describe the constructs used in scripts; the Interscript base language is sufficient.

Simplicity and extensibility

Simplicity is achieved by a single language. Interscript has another major advantage: one can *insert the definition of new constructs* into a script's data stream using the Interscript base language. This makes it possible for a system to define its own abstractions in terms of the Interscript language, then communicate those definitions together with the scripts which use them. Other systems can then understand these non-standard constructs.

Interscript provides a comprehensive and simple binding mechanism. A binding associates a name with some value; the value is then accessed through the name. This very general operation may also be used to achieve compactness. A form might contain, for example, the same item repeated 25 times. A compact script would bind it to a name and repeat that name 25 times.

The part of the Interscript standard concerning the base language is called Layer 1 of the standard. The entire grammar for the base language may be described in Backus-Naur Form in just 22 rules (see Fig. 8-3).

In addition to the base language, the Interscript standard also provides a set of abstractions that are expected to be widely used. These abstractions are described using the Interscript base language. Since those abstractions have their own significance and are build on the top of Layer 1, this part of the Standard is called Layer 2. Convenient abstractions such as appendix, glossary, codicil, tables, spreadsheets, graphics, etc. can be designed for a particular application using the Interscript base language. The Interscript structure also allows other standard forms to be included within it, such as the Raster Encoding Standard for pixel arrays, and the GKS standard for graphics.

The advent of Interscript and the internalization/externalization processes for editors will launch a new era for modern office systems. A widespread exchange of work products will facilitate new levels of productivity. This is another example of integration in the XNS architecture leading to improved forms of document management.

```
Rule 1   identifier ::= letter idTail
Rule 2   idTail ::= empty | idTail ( letter | digit )
Rule 3   script ::= "INTERSCRIPT/1.0"  node  "ENDSCRIPT"
Rule 4   node ::= "{" items "}"
Rule 5   items ::= empty | items item
Rule 6   item ::= tag | localBinding | staticBinding | expression | reference | formula | scope
Rule 7   basicObject ::= INTEGER | "<" STRING ">" | ATOM
Rule 8   tag ::= identifier"$"
Rule 9   localBinding ::= name " = " expression
Rule 10  name ::= identifier | qualifiedName
Rule 11  qualifiedName ::= identifier "." identifier
Rule 12  expression ::= term | operation term | expression operation term
Rule 13  operation ::= "+" | "-" | "*" | "/" | "!" | "<" | ">" | "<=" | ">=" |
                "equal" | "not" | "or" | "and"
Rule 14  term ::= basicObject | reference | node | "(" expression ")"
Rule 15  reference ::= ( path | reference) " |"
Rule 16  path ::= name | path ":" name
Rule 17  staticBinding ::= name " = = " (expression | selection | formula | quotedExpression)
Rule 18  selection ::= "OneOf"  " (" alternatives " )"
Rule 19  alternatives ::= empty | alternatives identifier
Rule 20  formula ::= expression"%"
Rule 21  scope ::= "["  items  "]"
Rule 22  quotedExpression ::= """ expression """
```

Figure 8-3 Interscript base language grammar

Document Interchange Service

The XNS architecture includes Document Interchange Service that can convert documents to and from different standard formats. These services are available as part of Gateway Services (such as Interactive Terminal Service and the 850/860 Gateway Service) and in workstations.

The Xerox Professional Workstations provide software to convert to/from the format of the older Xerox 860 Information Processor. They can also convert a document in their format to and from the DIF format used by many vendors of word processing equipment. Although this helps alleviate some document interchange problems, it is not a complete solution.

9. MAILING

One of the most useful aspects of a distributed network system is the ability of users to send and receive electronic mail. Electronic mail may be defined as a non-interactive document or message communication between people that is transported electronically, not physically. The usefulness of an electronic mail service depends largely on its wide availability and on its flexibility.

Computer based electronic mail systems offer many benefits that are not available in other forms of communications such as the telephone, telex, and conventional hard copy mail. Some of these features that increase productivity and effectiveness are:

Speed and flexibility

Mail is delivered almost instantaneously whether they are going to a user in the next office or in another city. This eliminates the delay present in conventional mail.

The users also have the advantage of being able to send and receive communication at a time and place of their own choosing. No more telephone interruptions or "telephone tag."

Multiple distribution

Mail may be sent to distribution lists with no extra effort. It is easy to be effective in communicating to a single person, a group, a department, or a whole organization.

Precision and productivity

The written word itself leads to precision. The document is a record of the communication and may be acted upon, increasing the effectiveness of individuals and groups. In addition, powerful electronic mail systems make it very easy to respond to mail, forward it, use parts of it or all of it in some other communication, and file it electronically. It saves time for an individual worker and makes it easy to build upon the work of others.

Standards and interconnection

There are a large number of electronic mail networks in existence today, most of which are incompatible with each other and offer very limited features. To make electronic mail as universal as the telephone or the postal service, it is essential that there be standards for electronic addressing, delivery, verification, and other mail functions, as well as for mail and message formats. The great variety of electronic mail systems also need to be interconnected into a unified network. For electronic mail, internetworking is vital because of the *critical mass effect*; that is, how effective an electronic mail system is depends primarily on how many people it can reach. Since many potential correspondents will in practice be users of different mail systems, interconnection is vital for maximum utility.

Xerox has been a leading proponent of such industry standardization and had a leading part in the development of the CCITT X.400 series of standards for message handling systems. The Xerox mail protocol and mail format standards are aligned with the CCITT standards.

Mailing standards

The Xerox mailing standards consist of two protocols and a format standard. The Mail Transport Protocol and the Inbasket Protocols address the functions of sending and receiving mail, and the Mail Format Standard defines the format of messages transported using the two protocols. Together these three standards specify a carefully layered mailing architecture that promotes compatibility while also allowing flexibility through transparent multi-level encapsulation.

Mailing protocol model

The relationship of the two mailing protocols can best be explained in terms of the CCITT X.400 architectural model which defines two layers: Message Transport and User Agent, as shown in Fig. 9-1. The model envisions an originator and a recipient user agent, ordinarily a human/workstation client although entirely automatic processes might also originate or receive mail.

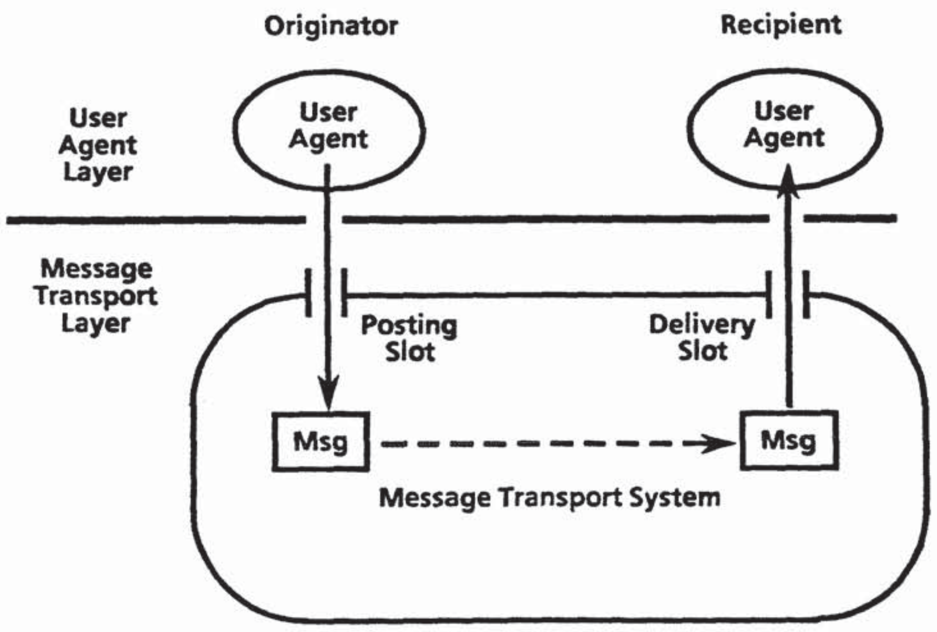


Figure 9-1 **CCITT X.400 architecture model**

0094

Mail Transport Protocol and Inbasket Protocol

The XNS Mail Transport Protocol corresponds to the boundary between the Message Transport layer and the User Agent layer, and provides operations for sending and receiving mail using the posting and delivery slots (see Fig. 9-1). Accessing the XNS Mail Service according to this protocol thus equates the physical boundary between the workstation and the server with the architectural boundary between the two layers. However, there are pragmatic reasons for departing slightly from this model. A typical user agent (client) will transfer incoming mail from the delivery slot to an "inbasket" container (mail file) for perusal by the recipient. Using the Mail Transport Protocol for delivery implies that the mail is transferred to an inbasket container residing on the client machine (e.g., workstation). Since users may wish to gain access to their mail from any of several workstations, it is preferable that the inbasket resides on the server, making it globally accessible. To this end, the Inbasket Protocol is defined, which corresponds architecturally to an internal interface of the user agent layer.

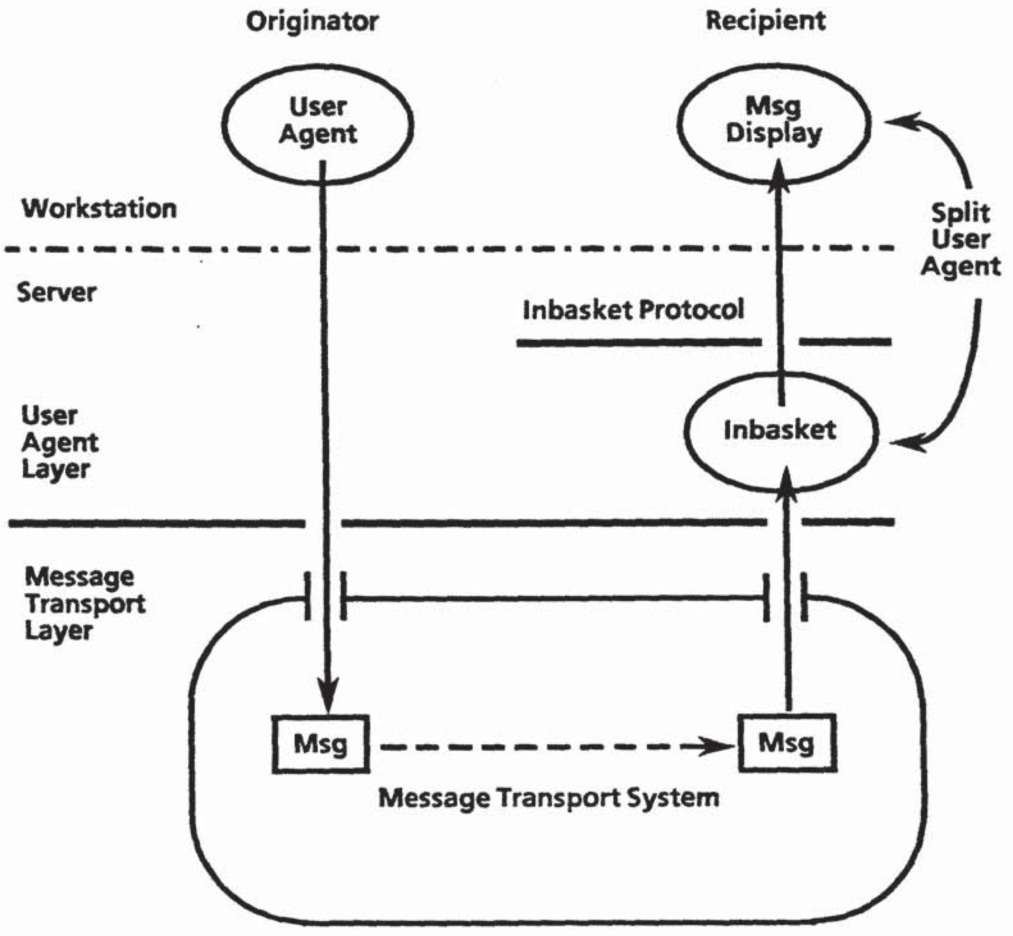


Figure 9-2 The Inbasket Protocol provides an internal interface for user convenience

0095

Workstation/service interface

The interface between the workstation and the service thus consists of both protocols: The Mail Transport Protocol is used for posting messages, while the Inbasket Protocol is used for receiving them.

To send a properly formatted message, the originator invokes the Mail Transport Protocol, handing the message through the "posting slot." The Mail Transport Protocol is responsible for calling the appropriate Courier procedures that will deliver the mail to the intended recipient(s) through "delivery slots." The Mail Transport Protocol then delivers the mail to a holding facility managed by the Inbasket Protocol. This protocol acts at the request of the receiving user by fetching messages that have arrived at the holding area since the last access.

Together, the two mail protocols provide a transport medium for messages (including all variety of electronic information) that is totally transparent to the content and format of the messages. Specifically, a message at the Mail Transport level is defined to consist of two parts: envelope and content, as shown in Fig. 9-3.

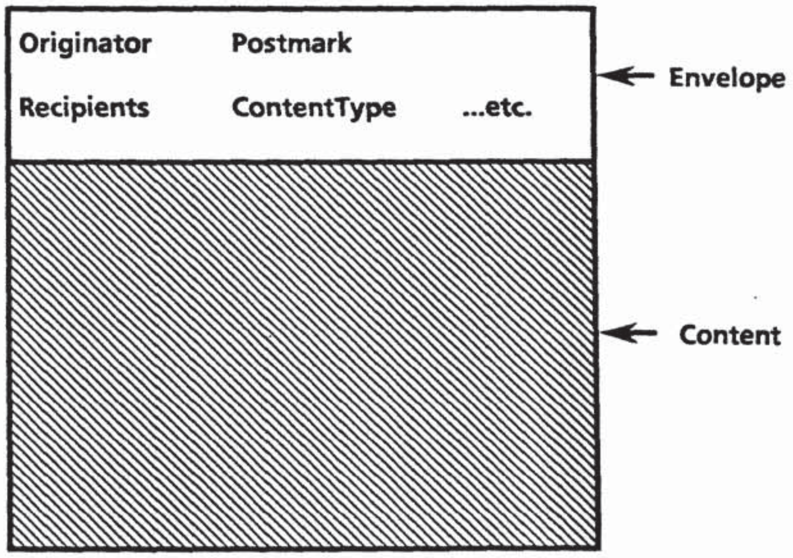


Figure 9-3 **Envelope and content of a mail message**

Content and envelope

The crosshatching in Fig. 9-3 indicates that the Mail Transport level is completely transparent with respect to the message content. The envelope contains information of two kinds:

1. Information that is related to the functioning of the Mail Transport level, including information passed from the user agent to the Mail Transport system (e.g., recipients) and information passed from the Mail Transport system to the user agent (e.g., postmark).

2. Information that is passed from the originator's user agent to the recipient's user agent and is required on all messages (e.g., ContentType).

Mail format standard

While the transparency of the Mail Transport system provides layering and flexibility, most usage of Mail Transport will adhere to a single format for the content. This format is defined in the Xerox Mail Format standard, which is functionally aligned with the CCITT X.420 P2 specification for interpersonal messages. The presence of such a message is signaled by the appearance of the corresponding ContentType on the envelope. Messages of this type are thus encapsulated within the outer level defined by Mail Transport and provide, in turn, a second level of encapsulation by defining a two part structure consisting of a message *heading* and a message *body*, as shown in Fig. 9-4.

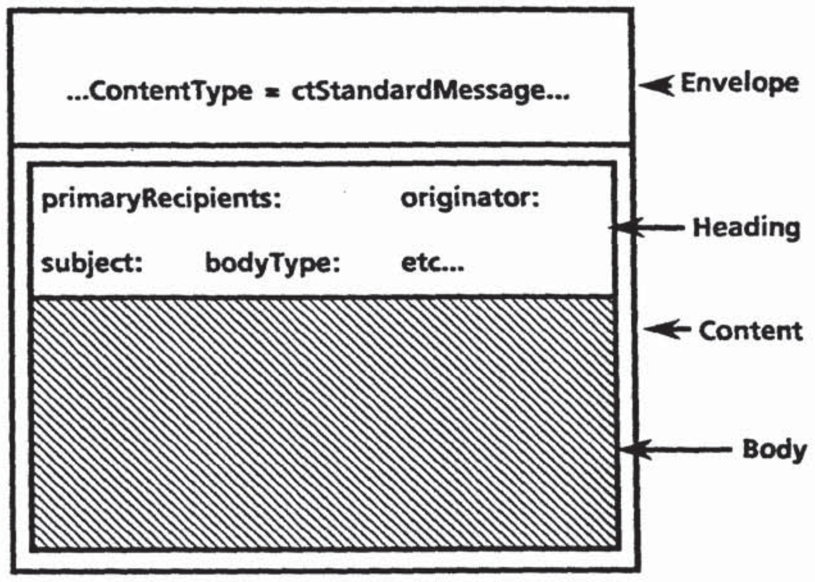


Figure 9-4 Heading and body of a mail message

Message body The crosshatching in Fig. 9-4 indicates that the Mail Format standard is transparent with respect to the format of the encapsulated message body. Examples of message body types would include document formats for various document preparation systems (e.g., Xerox 8010, 860, etc.); these correspond to subtypes of the Nationally Defined body type of CCITT X.420 model. In all cases, the heading information (originator, subject, etc.) is represented in a uniform way, independent of the format of the body. At the innermost level, the mail message body must be interpreted by appropriate format-specific software, selected according to the type indicated in the bodyType field of the heading.

Mail Service

The XNS Mail Service allows users to send and receive electronic mail varying from brief messages to long documents in a variety of formats. Mail is delivered almost instantaneously whether it is going to a user in the next office or in another city. The Mail Service uses the Mail Transport Protocol, the Inbasket Protocol, and the Mail Format standard in its operation.

Accessibility

The Mail Service is directly accessible to all XNS workstations and to many non-networked, non-XNS devices that can dial up the appropriate gateway service.

Mail originating from geographically dispersed networks can also be exchanged across the internet via the External Mail Gateway or the Internetwork Routing Service.

Flexibility in content

Unlike most electronic mail systems, XNS Mail Service supports not only messages but full attachments which can include typeset text, graphics, record files, or any other electronic information. Information originating from a workstation can be sent through electronic mail and be received at any other workstation with all formatting retained. For example, when a document containing tables, various font sizes and faces, and illustrations is mailed, it can be viewed, printed, and filed with all the formatting information retained.

Because the Mail Service can handle all types of information, it provides a convenient means for people to exchange documents as well as data.

Distributed mail system

The Mail Service is a distributed system that allows multiple Mail Services on an internet to cooperate actively to form a unified mail system. It does not restrict the number of Mail Services allowed on an internet or the number of individuals who can communicate, but works as a large cooperative mail system.

Multiple Mail Services act much like multiple post offices within the U.S. Postal Service. Each Mail Service is capable of accepting mail, holding mail for pickup by nearby recipients, and forwarding mail to the recipient's Mail Service.

A distributed mail system has advantages over an independent mail system because of its reliability and potential for growth. It provides constant communication availability because mail can be sent using any Mail Service if a local Mail Service is not operating. A distributed system also allows for smooth expansion from a small configuration to a large system with many users, since additional Mail Services can be added as the number of users grows.

Distribution lists

The recipient list determines who will receive the message. In the XNS Mail Service this list can contain the names of individuals and/or the names of distribution lists. Generally, mail is addressed and sent to an individual, but in many cases documents or their copies are also sent to many individuals. This can be accomplished either by entering the names of all the recipients, or by addressing mail to a previously created distribution list.

A distribution list is composed of the names of a group of individuals who are registered in the Clearinghouse and are designated as a user group. User groups are generally created by a system administrator, but users can be granted administrator privileges when a group is created, allowing them to add and remove individuals from the group. A distribution list itself may contain other distribution lists.

User group distribution lists can also be used to control access, such as access to file drawers. When the name of a user group is entered as a recipient, it becomes a distribution list and all the individuals associated with the list receive the message. All registered users can use any of the established distribution lists when sending mail. This feature saves time and assures consistency and accuracy when mailing to a group.

When a distribution list is used, the Mail Service generates a list of individuals by interacting with the Clearinghouse to determine the users on the list. If several lists are used and there are duplicate names, the Mail Service sends the message only once to each intended recipient. After the recipients are determined, the Mail Service begins distribution of the message. Local recipients with mailboxes located on the distributing Mail Service receive a copy of the message in their mailboxes. Remote recipients with mailboxes on another Mail Service, require forwarding of the message to their own local Mail Service, which places the mail in their mailbox.

Mailboxes and user names

Separate mailboxes are maintained for every authorized user on a Mail Service. By holding mail in a mailbox, the Mail Service allows authorized users to read their mail from any workstation or terminal with access to the Mail Service through the Interactive Terminal Service. Once the message has been copied to a recipient's workstation disk, it can be deleted from the Mail Service mailbox.

Mailboxes are generally identified by the name of the user and the name of the Mail Service where the user is registered. If no Mail Service name is given, the default domain is assumed. XNS permits great flexibility in the choice of user names (all characters except ":" and the wildcard character "*" are allowed). It permits users to choose their own full legal name (e.g., John Q. Public) or any name of their choice. XNS also allows use of aliases and assists clients to find valid names when they have only partial information.

External Mail Gateway and Teletex Gateway Service

The External Mail Gateway feature of the Mail Service allows multiple Mail Services on different internets to exchange mail, but not share other services. This feature is useful when limited access is desired (e.g, a company wishes to send orders to a supplier via electronic mail, but neither party wishes to allow the other any access to its File Service, which would contain private information).

Mail can be sent from one internet Mail Service to another over autodial telephone connections established by the sending gateway. When a user sends a message, it is placed in a special queue which the Mail Gateway software polls periodically. When a message is found, an attempt is made to establish a telephone connection with the External Mail Gateway in which the recipients are registered. These connections are attempted according to an established schedule and are always made by the sending Mail Gateway. Once a connection is made, the message is transmitted and placed in the recipient's mailbox.

Another XNS facility is the Teletex Gateway Service which interconnects the Mail Service to the Teletex services offered by many common carriers (PTT's) in Europe. This allows XNS workstations and other Mail Service clients to send or receive documents from Teletex and Telex terminals.

Interconnection with other non-XNS private and public electronic mail services supporting the CCITT X.400 standard protocol will also be possible in future.