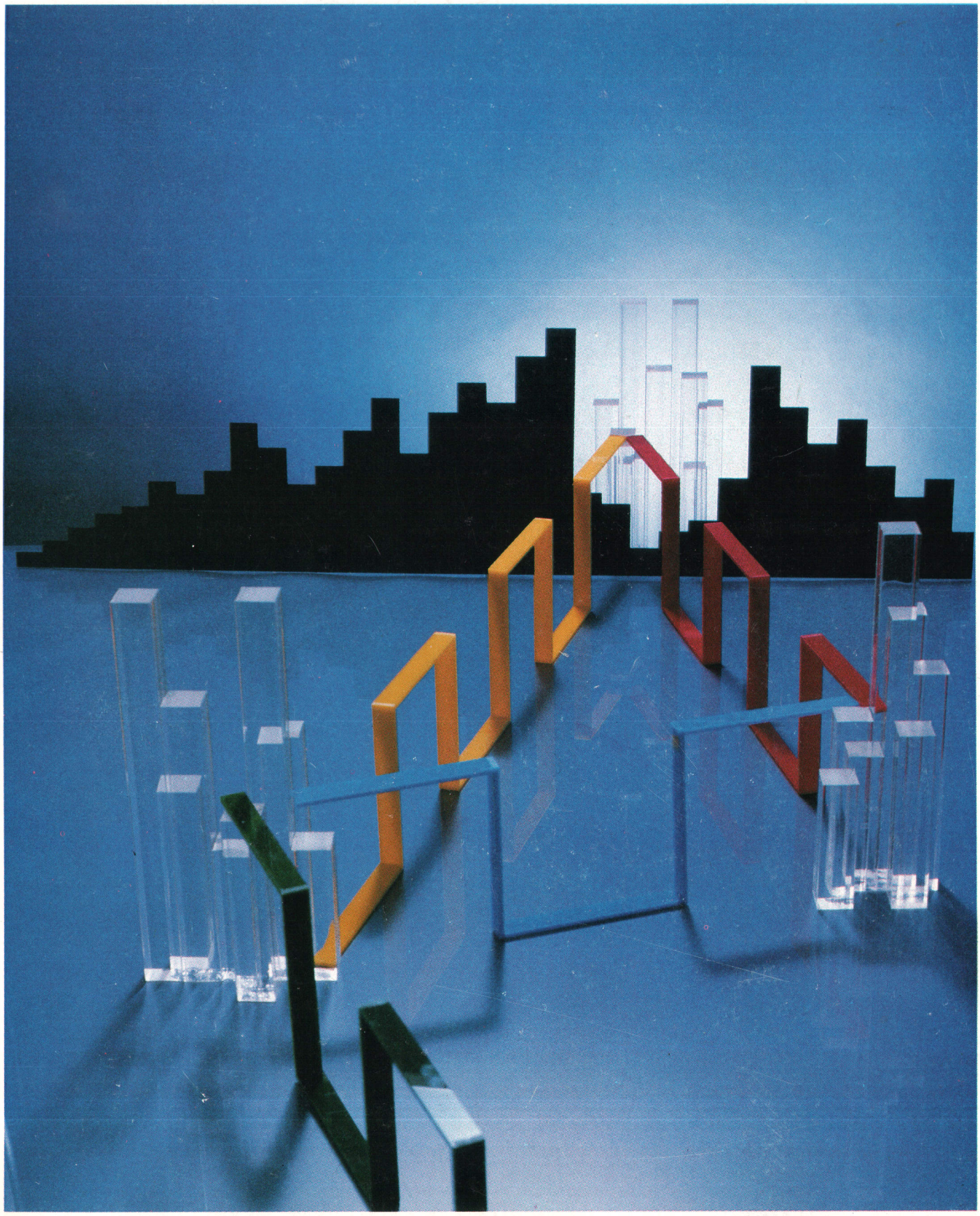
HEWLETT-PACKARD
JOURNAL

OCTOBER 1986



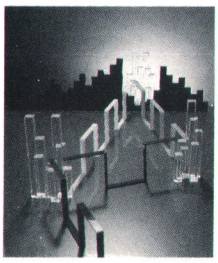
HEWLETT-PACKARD JOURNAL

October 1986 Volume 37 • Number 10

Articles

Departments

In this Issue



Since Hewlett-Packard released its first networking product for HP computers in 1973, the world has become steadily more dependent on computing, and the computer world has become considerably more complex. Instead of one computer product line, HP now has HP 3000 commercial systems, HP 1000 real-time automation systems, HP 9000 engineering workstations, Vectra and Touchscreen personal computers, the Integral and Portable Plus portable computers, and handheld computers and calculators. Other manufacturers' offerings are similarly diverse.

Computer users today want to be able to choose the best products for the job, and they want those products to fit easily into networks. Users are demanding that manufacturers provide extensive networking capabilities, not only between their own products, but also with other manufacturers' machines. Success in this effort demands worldwide standards. An important step towards bringing order to the variety of manufacturers, communication links, protocols, and applications is the Reference Model for Open Systems Interconnection of the International Organization for Standardization, published in 1983 and now adopted by many countries.

In this issue, network designers from several HP divisions tell us about their contributions to HP's overall networking scheme. On page 4, you'll find a statement of HP policy on the ISO OSI model, and on page 6 a description of HP AdvanceNet, which is HP's strategy for providing network services in today's diverse environment. The articles on pages 11, 22, and 28 discuss HP AdvanceNet-compatible network services products for HP 3000, HP 1000, and HP 9000 Computers, respectively. HP's implementations of two industry standards—the IEEE 802.3/Ethernet local area network standard and AT&T Information System's Digital Multiplexed Interface standard for PBXs—are described in the articles on pages 18 and 41, respectively. On page 36 is a discussion of HP capabilities with respect to CCITT Recommendation X.25, an emerging standard for packet switched wide area networks.

What you won't find in this issue is any description of the extensive networking services for HP's personal and portable computers. There's just no way to cover the entire networking area in one issue. So take this issue as a snapshot of one portion of this complex, rapidly evolving subject at one moment in time. Even in this limited view there's a clear message. The manufacturers are listening and the customer will be served.

-R. P. Dolan

Cover

A representation of a wide area network linking fanciful cities.

What's Ahead

The cover subject in the November issue will be the improvements in millimeter-wave device performance made possible by molecular beam epitaxy. Both the technology and some device applications are discussed in four separate articles. Four additional articles discuss applications of expert systems to configuring and troubleshooting HP computer systems and peripherals. Predictive support software, which alerts service personnel to problems in HP 3000 systems before they cause system crashes, is the subject of another article.

The HP Journal encourages technical discussion of the topics presented in recent articles and will publish letters expected to be of interest to our readers. Letters must be brief and are subject to editing. Letters should be addressed to: Editor, Hewlett-Packard Journal, 3200 Hillview Avenue, Palo Alto, CA 94304, U.S.A.

Hewlett-Packard and the Open Systems Interconnection Reference Model

The OSI Reference Model of the International Organization for Standardization is seen as the most significant tool for meeting HP's customers' needs.

by Gertrude G. Reusser and Donald C. Loughry

HEWLETT-PACKARD RECOGNIZES that customers have an urgent need to configure communication systems using components from various manufacturers that are compatible with each other. This means that users must be able to build simple or complex information networks by choosing the best components for them, regardless of who made those components, and have all those components communicate correctly and effectively with each other. How are we as an industry achieving this objective?

In the mid-1970s, the American National Standards Institute began to answer that question by defining open systems. They concluded that systems are open to each other by virtue of using standard protocols. For systems to be open to one another worldwide, these standards need to be globally accepted.

Communication takes place between open systems by transferring data among peer application processes of distinct end systems. This means that a user who has a set of equipment will be able to communicate with another user who has any other set of equipment if both users adhere to the standard protocols.

The ISO OSI Model

Because of the global nature of this requirement, the International Organization for Standardization (ISO) picked up the effort on a worldwide scale in 1977. A Reference Model for Open Systems Interconnection (OSI) was developed and was published in 1983 (Document IS 7498). Although there had been many standards development activities around the world since the early 1960s, until this point there had been no master planned approach that would make it possible for all aspects of open systems to be addressed and coordinated.

All of the major western countries, including their defense departments, have now committed themselves to OSI. Every year more countries are joining the effort to develop and commit to OSI protocols. The momentum toward open systems via OSI and toward multivendor systems is strong, and for good reason. OSI is the first major step in the direction of a global open systems environment.

OSI starts with a framework that organizes all intersystem communication functions into functional layers. Specific protocols that perform the functions assigned to each layer are developed within this structure. When all of these protocols are stable, any organization that wants to have open systems can achieve them by implementing these global protocols.

Fig. 1 shows the overall structure of the model and the names and numbers of the layers.

The primary concern of the application layer (7) is the semantics of the application. The main purpose of the presentation layer (6) is to allow application processes to be independent of differences in data representation. The session layer (5) provides the mechanisms for organizing and structuring the interactions between application processes. The purpose of the transport layer (4) is to provide transparent transfer of data between end systems. The network layer (3) provides independence from the data transfer technology and relaying and routing considerations. The purpose of the data link layer (2) is to provide the functional and procedural means to transfer data between network entities and to detect and possibly correct errors that may occur in the physical layer. The physical layer (1) has assigned to it those functions (mechanical, electrical, and procedural) that are needed to access the physical media.

Important strides are being made and important lessons are being learned in using the OSI protocols even though the current standards are not perfect and complete. The information gained in the implementation and long-term use of these standards will permit the completion and continuing improvement of these protocols.

OSI and HP

HP has carefully considered the importance of the OSI family of protocols in achieving multivendor systems for our customers. As a result, the company has made a firm commitment to having multivendor systems become a reality. We believe that the implementation of the OSI standards is the most important step toward achieving communicating systems that are practical, cost-effective, efficient, and reliable.

While the costs of OSI standards definition, development, implementation, and use (i.e., the overhead costs) are high, the long-term benefits are substantial. These benefits include the following:

- The existence of interconnection standards and the inevitable evolution of the quality of these standards
- Replacement of layer protocols can be done in a modular fashion, without affecting other protocols or layers
- Multiple protocols can be used, depending on need, for added performance and flexibility.

HP is now delivering initial OSI products, and will, in the next two years, be introducing many additional, specific products for the manufacturing and office automation

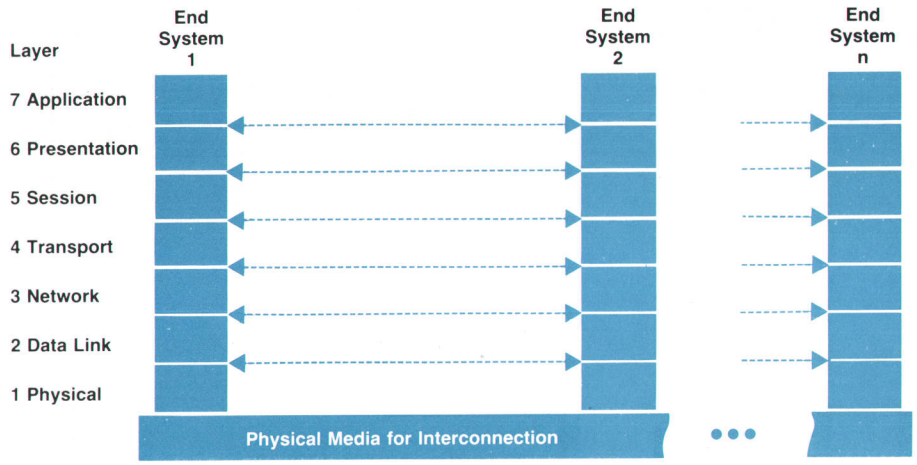


Fig. 1. *The Open Systems Interconnection Reference Model of the International Organization for Standardization.*

markets, domestically and in Europe. We are convinced that by the end of the decade, there will be a significant number of globally open systems, and in the early 1990s, practical multivendor systems will be the norm rather than the exception.

To reach that goal, HP is taking an active part in the process of further developing these standards while implementing them and developing action plans for the conversion of applicable current non-OSI products to OSI products. (For more details, see article on HP AdvanceNet, page 6.)

HP has taken the lead in some areas of standards development, such as OSI 8802/2 (Logical Link Control) and 8802/3 (Carrier Sense Multiple Access with Collision Detection). HP is also participating actively in the development of many other standards for all areas of OSI, including network management, directory service, and the network, presentation, and application layers.

HP was a founding member and is now a senior research member of the Corporation for Open Systems (COS), a consortium of all major North American vendors and many users of networks. COS is promoting the acceleration of standards development and implementation and will be performing certification of products as being proper implementations of OSI protocols. The HP commitment to support multivendor systems through the implementation and use of OSI networking standards and protocols is expressed and affirmed in a recent statement by John Doyle, Executive Vice President: "The adoption of national and international networking standards throughout the computer and communication industries is essential for transparent connectivity among open systems, thus enabling effective end user access to a broad spectrum of user-oriented applications. HP, in response to real and expressed customer needs, is committed to contributing to the birth and evolution of appropriate international OSI networking standards."

The alternatives to the OSI approach are not supportive of reaching the essential goal of open systems. One alternative is to develop our own protocols to obtain very lean, very efficient network performance. This approach is clearly contrary to the multivendor interconnection capability that our customers need. On the other hand, the implementation of other proprietary protocols would not allow for a broad-based multivendor solution. When specific gateways to other proprietary protocols are needed by specific customers, we will develop such solutions, as appropriate.

Cooperation among vendors on neutral territory, such as the standards development arena, is essential for the best quality of products in the industry and, therefore, for serving the larger good.

Experience with OSI

Many organizations in Europe have implemented various OSI standards since 1983. Their experience has been that the results do not come immediately or easily, but that the standards are effective toward achieving open systems. In the U.S.A., the Manufacturing Automation Protocol (MAP) and the Technical and Office Protocols (TOP) are based on the OSI standards and are being used satisfactorily. As more and more of these protocols are widely used, they can more effectively be refined to make them work as intended.

In July 1984, HP participated with other major manufacturers in an "Open Systems Interconnection Demonstration" sponsored jointly by the U.S.A. National Bureau of Standards, General Motors Corporation, and Boeing Computer Services Corporation at the National Computer Conference. This demonstration helped convince the world that OSI does exist, the OSI protocols are implementable, and they do work as advertised. Included in this successful event were the use of OSI protocols 8802/2, 8802/3, and 8802/4 (Token Bus), the ISO Class 4 Transport protocol, and an extended subset of the ISO File Transfer Protocol. In November 1985, HP participated in Autofact, a larger demonstration of the same type, with more participants. In this event, additional functions were demonstrated, including a bridge between 8802/3 and 8802/4 and communication between the previously mentioned OSI protocols and X.25 networks.

HP Strategy

HP announced the AdvanceNet strategy (see page 6) in 1985 as the mechanism to implement the OSI protocols. This architectural strategy is designed to have the necessary flexibility to accommodate a variety of protocols and to allow an orderly conversion to the OSI protocols as they continue to emerge.

HP AdvanceNet: A Growth-Oriented Computer Networking Architectural Strategy

Based on the seven-layer ISO OSI model, HP AdvanceNet accommodates old and new protocols in the same network, ensures migration paths to new systems, and provides ease of use and transparency.

by Robert J. Carlson, Atul Garg, Arie Scope, Craig Wassenberg, and Lyle A. Weiman

GIVEN BOTH today's diverse communications environment and tomorrow's promised changes, the task of developing a comprehensive networking strategy can be extremely challenging for a computer manufacturer. Using a variety of protocols, run over a variety of link types, to support a variety of applications, today's customers want to interconnect a variety of machines produced by a variety of manufacturers. The communications software required to meet today's needs is quite complex; tomorrow's software will likely be even more so. Perhaps the main reason for the increased complexity will be the need for tomorrow's networks to support both "old" and "new" protocols. Old protocols refer to those protocols currently in use, be they proprietary (e.g., IBM's SNA), or international standards (e.g., X.25). New protocols here refer primarily to the emerging industry standards, national standards, and international standards, but also to any proprietary protocols that customers or vendors might now be inventing. Often the literature on protocols seems to imply that somehow the old protocols will merely wither and vanish when the new standards finally do emerge. Unfortunately, this appears an unlikely scenario for a variety of reasons. We believe that "old" software neither dies nor fades away. The highest performance in protocols is achieved only when the needs are very narrowly defined. Conversely, satisfying a wide variety of user needs involves some sacrifice in performance. Therefore, we can expect that there will continue to be network applications whose performance requirements can only be satisfied by special-purpose protocols (i.e., nonstandard).

Some customers will not buy new hardware or software unless it is compatible with the gear they already own. They don't always believe that new is necessarily better, and they frequently are unwilling to change applications that already work.

Furthermore, there are often customers that, while they may want to upgrade their networks with new hardware or software, can't because of logistical problems. A customer with an automated factory floor, for example, might be unwilling to shut down long enough to equip all the nodes controlling production with new protocol software. And there is always the possibility that the old protocols will perform better than the new protocols.

We conclude from all of this that the single most important feature of present-generation network architectures is multiprotocol support.

Hewlett-Packard has been in the computer networking business since 1973 and has installed more than 30,000 computer-based nodes in networked configurations since then. At present, HP networks rely mostly on proprietary protocols. However, in October 1981, HP announced its strong support for interconnection of multivendor computer systems through compatibility with the Open Systems Interconnection (OSI) Reference Model of the International Organization for Standardization (ISO).

Objectives

To implement this support, HP has undertaken a major effort to develop new networking software products according to the OSI model, yet to retain the same end-user network services that existed before. This new architectural strategy is called HP AdvanceNet.

The objectives for these projects are:

Implementation Based on the OSI Reference Model. This is the cornerstone for the other objectives.

Flexible Architecture. This feature permits the exchange and interpretation of information, and the sharing of resources between HP's different computer families: HP 1000s, HP 3000s, HP 9000s, and HP personal computers. These machines vary widely in processor speed, memory capacity, and software operating environment.

Flexible architecture also provides the capability to interconnect networks that may differ markedly. The lower layers may, for example, interconnect a high-speed IEEE 802 local area network and a long-distance X.25 network. The middle layers allow interconnection of nodes with different protocols, like TCP/IP and ISO class 4, and so on for the other layers. The architecture should support several protocols in every layer.

In place today are networks employing a wide variety of protocol families. This condition will persist for the foreseeable future. Implementations based upon Xerox XNS, DARPA TCP/IP, Ethernet, SNA, and a plethora of other proprietary LANs exist now, and newly standard-

ized alternatives (such as IEEE 802) will be added without making the others simply go away. Many customers will expect newly purchased equipment to operate usefully in such an old/new environment.

Emerging international standards are also not converging into one simple protocol architecture. They offer many new alternatives, such as five classes of transport, two network convergence protocols, etc., and thus offer the user more choices, but these choices do not immediately replace the old ones. The picture becomes more complicated, not less. We firmly agree that this variety is justified by legitimate customer needs, and that the proper response is to choose an architecture that supports it rather than one that restricts it.

HP has made a major objective of this point, and it deserves some explanation. HP believes that the modification of protocols will continue. We also believe that one set of protocols will not satisfy all customers' environments. Some environments will need flexibility and comprehensive services while others will need high performance. This implies different protocols in the higher as well as the lower OSI layers.

One monolithic implementation of a specific set of protocols will not satisfy this objective.

An Assured Migration Path. Since HP has a large installed base that needs either to upgrade or to coexist with the new architectural strategy, this objective is an important one. Also, the architecture has to be able to accommodate future upgrades, in light of HP's commitment to support emerging standards. An architecture that allows for continuous evolution is a must. Most of the other implementations in the industry allow only the upgrade option, and cannot accommodate old nodes in the network. We believe that old applications using the old network services should not have to be rewritten, and they should use the same interface to the new architecture.

Ease of Use and Transparency. One aspect of ease of use is that a node, or an end user of a node, should not need to reconfigure its view of the network whenever the network changes and a new node or new protocols are added. The optimal case is that the node needs to know just its own name and the protocols it supports. It does not need to know other nodes' addresses and their supported protocols.

Users need not be aware of the protocols their node is using. Transparency means that names are completely independent of addresses, and named resources can be moved within the network without modifying the application programs.

HP AdvanceNet Architectural Strategy

HP AdvanceNet supports communications networks by distributing the intelligence among the various network elements. The intent of this network architecture is to provide a foundation for present and future development of data communications products within HP. In its layered approach, it provides a common network or protocol for exchange of data and a wide range of communications activities. It also increases the compatibility between various HP product lines, giving better performance and shorter development cycles.

The layer structure conforms to the basic principles of the ISO OSI Reference Model. Fig. 1 shows the correspondence between HP AdvanceNet layers and the ISO layers.

Such a layered approach allows for simple growth and enhancement in the future, since new capabilities can be added to one layer without requiring a redesign of the other layers. This flexibility will permit HP to take advantage of new data communication standards as they emerge, without altering the applications or the interface to the users.

In the functional description that follows, the modules described are product independent. The list is for illustrative purposes and does not include all of the modules in HP AdvanceNet.

Application-Layer Modules (Network Services)

Network File Transfer (NFT) Protocol. NFT defines a standard interchange format that permits files to be exchanged between all types of HP computer families and desktop systems. The standard interchange format can represent files from many different operating systems, such as AT&T Bell Laboratories' UNIX™, HP's MPE and RTE, or Digital Equipment Corporation's VMS.

Virtual Terminal (VT) Protocol. VT allows a program to access a remote terminal of similar type as if it were a local terminal. It provides a consistent terminal operating environment to an application program regardless of the actual terminal driver or the terminal-host operating system.

Remote File Access (RFA) and Remote Data Base Access (RDBA) Protocols. This module allows a program to gain access to files, peripheral devices, and data bases in remote systems as easily as on the local system.

Network Interprocess Communication (NetIPC). NetIPC is a generic interface to the protocols of various layers. It is protocol independent, and it is not an application-layer

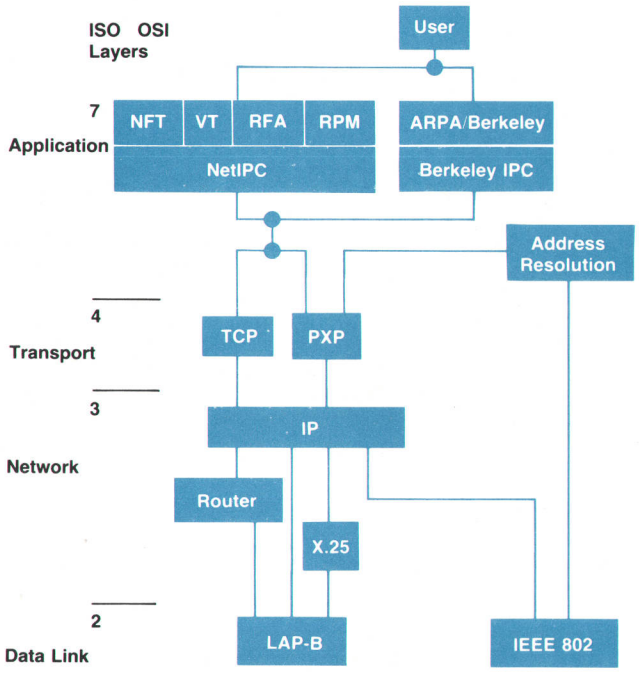


Fig. 1. *Protocols currently implemented in the HP AdvanceNet architecture, with corresponding ISO Open Systems Interconnection Reference Model layers.*

protocol in terms of the ISO OSI Reference Model. It is included here for simplicity. NetIPC provides a rendezvous facility and a facility in which processes can exchange data and control information. Both these services are provided locally and remotely in a transparent fashion. NetIPC also provides naive users, as well as system users, a consistent interface to the transport services.

Remote Process Management (RPM). RPM allows a program to create or terminate a remote process.

Address Resolution (AR)

Address Resolution provides user control of, and access to, addresses and configuration parameters. Access to all transport-layer and lower-layer modules is possible. Services provided by AR can be accessed by network-layer modules and applications (possibly through NetIPC). AR resolves names of objects into information about those objects. The most important information is location information (namely addresses), and the mapping of name to address is done dynamically.

Transport-Layer Modules

ARPA Transmission Control Protocol (TCP). TCP provides end-to-end reliable *connection-oriented* services with flow control and multiplexing. TCP relieves the users of any concern with the detailed way in which reliable and cost-effective transfer of data is achieved. It is intended to be used in association with connectionless network protocols (such as ARPA IP) which do not provide mechanisms for detecting duplicate, lost, or out-of-sequence packets. Applications that require in-sequence, reliable data transfer (such as file transfer, electronic mail, etc.) should use TCP.

Packet Exchange Protocol (PXP). PXP provides acknowledged service (request/reply). Requests are retried if the reply is not received in time. Reception of duplicate requests is not suppressed. PXP would be used by applications that require acknowledgments for receiving their messages, where duplication of requests is either taken care of elsewhere or is not a problem for the application.

Network-Layer Modules

ARPA Internet Protocol (IP). IP provides *connectionless* routing of user data to its final destination. It provides fragmentation/reassembly and internetting capability. When a packet arrives at a node, the IP internet address is checked to see if the packet is for that particular node. If it is, then the message may have to be reassembled before the packet can be given to TCP. If the destination address indicates that the packet has to be forwarded, it is given to the appropriate subnetwork module which determines the address for the final destination or intermediate gateway. If necessary, the packet is fragmented and forwarded to the appropriate node.

X.25 Packet Level. This module provides *connections* to systems through public or private packet switched networks (PSNs). X.25 packet level provides both permanent and switched virtual circuits. It provides mechanisms for call establishment and call clearing, multiplexing, procedures for resetting and restarting, and flow control. This protocol is used over a LAP-B data link to interface to a PSN.

Router. The router provides store-and-forward capability among HP machines. The router provides *connectionless* data transfer, message segmentation, error control, addressing, and routing. The protocol is functionally quite similar to IP, and it is used between HP systems over point-to-point networks.

Data-Link-Layer Modules

IEEE 802. This implements the IEEE Standard 802.2 Logical Link Control (LLC) and IEEE Standard 802.3 CSMA/CD Media Access Control Protocols.

LAP-B. This module provides *connection-oriented* data link service.

BSC. This is the *connection-oriented* Bisync (conversational Bisync on the HP 3000 and HP 1000) protocol. BSC is being supported on some systems for backward compatibility only.

Design Considerations

The OSI Reference Model and the different protocols used in its implementation leave design objectives unfulfilled. Some of the issues that were faced and how they were resolved are discussed in this and the following sections of this paper.

As previously stated, one of the design objectives was to allow interconnection of nodes with different protocols in the different layers. Fig 2 shows examples of multiple protocols that might be encountered in various layers.

One measure of a flexible network architecture is its ability to absorb changes at lower protocol layers without requiring corresponding changes to application layers. In theory, the protocols of a layered architecture should be modularly replaceable, but in practice such modularity can be difficult to achieve because of differences in address formats, supported service options, etc. One step taken by the HP AdvanceNet architecture to support layer modularity is the provision by NetIPC of a generic interface to the transport-level (OSI level 4) protocols.

NetIPC users (applications, processes) access transport services through communication endpoints called sockets. The most commonly used kind of socket is the connection-oriented variety. Having created a socket on one system, an IPC user can connect that socket to one on another system. Data can be transferred in full-duplex mode be-

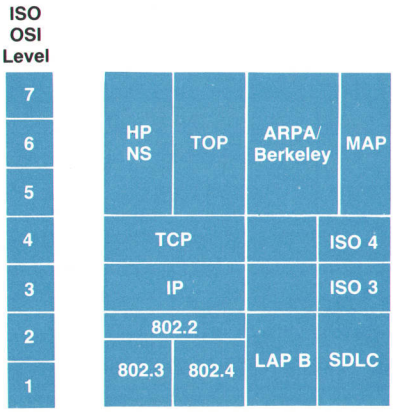


Fig. 2. *Examples of multiple protocols that might be encountered in various OSI layers.*

tween connected sockets. Transmitted data is guaranteed to be delivered reliably and in sequence. This kind of service can be supported by a variety of protocols: TCP, OSI level 4, NBS level 4, etc.

On a system supporting multiple-protocol "stacks," a socket might be bound to any of several different connection-oriented protocols. In being bound, such a socket is assigned an address (such as a TCP port number or an OSI level 4 service access point) where its owner can listen for incoming connection requests. The socket can become connected via any of the protocols to which it is bound.

NetIPC users can name the sockets that they own. The socket's name and path report are then registered in a data base called the socket registry. The path report describes the complete set of protocols and addresses through which connections can be formed to the socket. NetIPC users then specify by name the sockets to which they want to connect. Upon receiving a connection request, NetIPC attempts to resolve the name of the target socket by querying the socket registry. If the name is found to be registered, the path report associated with it will be returned. The path report contains enough information so that an NetPC running on one machine can figure out which protocols and addresses to use to access a particular socket on another machine. Protocols are defined to provide this information dynamically, without requiring the use of well-known addresses.

Because NetIPC is merely an interface and not a protocol, it is still possible for an NetIPC user to communicate with users on machines that do not support NetIPC and/or the socket registry. Any interface that provides equivalent access to the communication endpoints of a system's transport protocols can be used at the remote end. If a target system does not support the socket registry, NetIPC users at the initiating end of a connection have two options. Either they can specify which protocols and addresses should be used to access the remote system, or they can have their system manager configure proxy entries in the socket registry. Neither solution is ideal, but they both work.

Path Reports

To achieve the objective of flexible architecture, two concepts were introduced: path report and address resolution registry, which includes the socket registry.

Within this context, a path is a course that a message might take through the protocols of a node. A path report is a data structure that describes all of the possible paths by which a particular protocol or application service can be accessed on a particular node. In Fig. 3, for example, two paths are:

Path 1: NFT,NetIPC,TCP,IP,X.25
Path 2: VT,NetIPC,ISO4,NULL,802.3

The end user on node C who wants to connect to a service on node A does not need to know anything about these paths, but simply refers to this service by its name. The address resolution registry of the service node then tries to map that name into a path report. It fills in all of the available paths and sends it back through a standard protocol to the end user. The end-user node employs a set of heuristics to discover the best path. The heuristic procedure first eliminates paths that include protocols not supported in its node. Then a figure of merit is assigned to the remaining candidates. After choosing the path, the information is retained in memory for future use. Some variations of a standard can also be accommodated in a path report.

The path report concept has the following advantages:

- It allows the addition of new protocols as they become available without the need to reconfigure the entire network.
- It also meets the ease-of-use objective, since it does not require the configuring of path reports on every node.
- It enables old-architecture nodes to participate in the same network with new-architecture nodes, as will be explained later.

Name-to-Address Resolution

As mentioned previously, NetIPC users use names to reference remote sockets. These names consist of two major parts: a socket name and a socket registry name. The socket name part is the name actually bound to the socket. The socket registry name part is the name of the socket registry in which the socket name has been registered.

When queried with a socket name that it knows about, a socket registry will reply with the path report that describes how the socket can be accessed. The path report is not tailored in any way for the client that issued the query. It is always up to the client to decide which information in a path report is most relevant.

An interesting problem arises when a client wishes to query a remote registry: the client must decide which protocols to use for the transaction. Since the socket registries themselves sit atop NetIPC sockets, this means that determining viable query-transaction protocols boils down to a problem of obtaining the path report for the desired registry's socket. Special registries called nodal registries have been created to store path reports describing how to access a network's socket registries (and other well-known network services as well). Having nodal registries that contain information about a network's socket registries helps to minimize the amount of configuration input required for each network node. Rather than having to be told where all the socket registries are, a node need only be told where one (or possibly a few) nodal registries are.

Migration

To achieve the objective of assured migration to the new architecture, it was decided that the same interface to net-

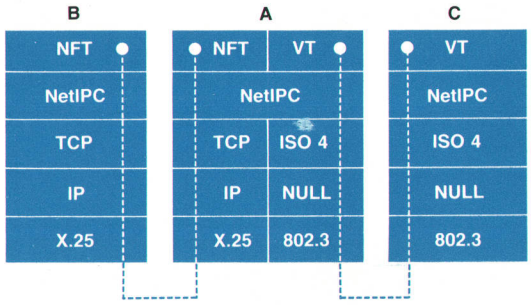


Fig. 3. *Examples of paths in internodal communications.*

work services that was used in the former proprietary architecture will be used in the new architecture. Also, participation of both old nodes and new nodes on the same network will be allowed.

There are several possible ways to achieve this objective. One is to use a gateway between an old network and a new network, and perform protocol conversion (for the different layers) in the gateway. This approach was not chosen because of the large number of protocol combinations that would need conversion. The number of combinations is expected to grow dramatically in the future. The gateway would also introduce performance degradation in the network operations.

Another possibility is to retrofit all of the old nodes into the new architecture. This approach was not chosen either, because in many cases, shutting down the network while the upgrade is being installed is unacceptable. Some nodes may not be upgradable if the new architecture will not run on old hardware.

A third approach is to allow the coexistence of the old architecture and new architecture in the same machine. This solves the problem of migration today and also permits the addition of new protocols in the future, without the need to stop the entire network.

In our architecture we chose the third approach, which gives the best flexibility. The path report concept makes its implementation feasible, since it does not require *a priori* information on the configuration of the various nodes. This approach also allows old hardware nodes that do not support the new architecture to communicate with new nodes that have both the old and the new protocols.

Another issue is how to preserve the software interface in existing applications. Doing so permits moving existing application software to new nodes without reprogramming and without being concerned with whether the other nodes are old or new. This is accomplished by translating the old service request intrinsics into the standard interlayer interface message called interprocess communication (NetIPC), and through use of the path report to select the appropriate protocol stack. NetIPC has been defined in such a way that it will keep the applications isolated from changes in the lower layers.

An additional avenue for adapting to changes in protocols (when they change in a minor way) is through negotiations, provided that the protocol supports this alternative. Negotiation allows nodes that have implemented common paths to agree on the use of options of each of the protocols used in that path. Application programs that can take advantage of protocol options can then adjust for non-use of those features if the peer does not support them, or the programs can provide a meaningful diagnostic message that the peer needs to be upgraded to support certain options. Provision of clear and understandable error messages assists those responsible for software maintenance in their function.

Summary

Hewlett-Packard's goal in this architecture is to provide users with a framework that will carry them from present *ad hoc* protocols to internationally standardized protocols as they evolve.

The basic premise in this implementation is that more and more networking will be used in the future. There will be thousand-node multivendor networks, and more efficient protocols will evolve in the future to accommodate the different physical environments.

A rigid architecture, one set of protocols, and a monolithic and hard-coded implementation of this set are not the right answers to the problem. Migration is, and will be, a major issue in the future. The HP architecture offers good paths for migration and for evolving protocols to accommodate the predicted growth pattern.

Acknowledgments

The ideas of many people are reflected in the HP AdvanceNet strategy, but the contributions of a few individuals stand out. Mike Wenzel was undoubtedly the greatest single contributor in the effort to hammer out the many parts of the HP AdvanceNet definition. Mike especially deserves credit for the decision to support multiple protocol families. John Bugarin helped to define NetIPC and also provided the germinal ideas leading to support of two of HP AdvanceNet's address resolution services. Rick Bartlett prototyped many of the HP AdvanceNet services on the HP 3000 as they were being defined and thereby helped others recognize and correct flaws.

Network Services and Transport for the HP 3000 Computer

NS/3000 provides network services for HP 3000 Computers attached to local area networks. It is compatible with older network products, it is expandable to new network topologies, and it can communicate with other HP computers.

by Kevin J. Faulkner, Charles W. Knouse, and Brian K. Lynn

THE HP ADVANCENET IMPLEMENTATION on the HP 3000 Computer family is a group of related software and hardware products collectively called NS/3000. The NS/3000 network services product provides a set of services used by applications and interactive users on an HP 3000 to use resources like data, programs, and devices located on computers in the network. Network services corresponds to the application and presentation layers (7 and 6) of the ISO OSI reference model (see article, page 4).

The LAN/3000 product allows an HP 3000 to attach to a local area network and to communicate with other computers on that network. LAN/3000 is composed of the NS/3000 transport and the LAN link. The NS transport provides an interface equivalent to the OSI session layer (5) and implements industry standard protocols for the transport and network layers (4 and 3). The LAN link is the software and hardware for the data link and physical layers (2 and 1) for the local area network.

Fig. 1 shows how the NS/3000 products relate to each other and the ISO OSI layers.

This article will discuss NS/3000 network services and transport. The LAN link is the subject of the article on page 18.

One of the primary objectives for NS/3000 is compatibility with its predecessor product, Distributed Systems/3000 (DS/3000).[1] Application programs and jobs that use DS/3000 can use NS/3000 without modification.

At each layer of the OSI model there are now several choices for industry standard protocols, and new protocols are under development. In consideration of the proliferation of protocols, NS/3000 has as an objective the addition of new protocols and services without major redesign.

Performance is always an objective in the development of computer products. The performance goals for NS/3000 are increased throughput and response time over the DS/3000 product. This will help customers who have reached the limits of DS/3000 performance.

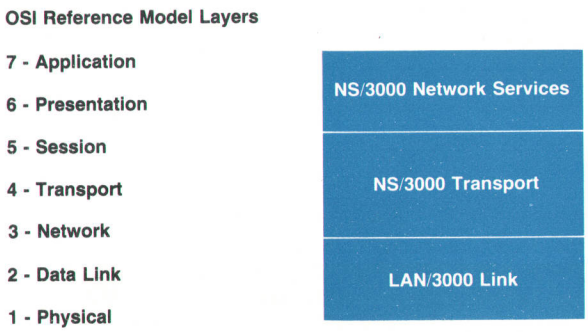


Fig. 1. *HP AdvanceNet products on the HP 3000 Computer.*

Virtual Terminal Service

The virtual terminal (VT) service allows a program on one computer to use a terminal attached to another computer. One way this is used is for remote command processing. Here the program is the Command Interpreter (CI) for a session on a remote computer, and the terminal is the session terminal for a user's local session (Fig. 2). Commands entered at the local terminal can be directed to the remote session for execution.

An application program can acquire control of a terminal attached to a remote system through the virtual terminal service. This use of VT is sometimes called reverse VT, because the program using the service is on the local node, and the terminal is on a remote node (Fig. 3). Reverse VT allows a program on the HP 3000 to access terminals on different types of systems (provided they offer the VT service). The reverse VT service is a new feature of NS/3000.

A key mechanism used for the virtual terminal service

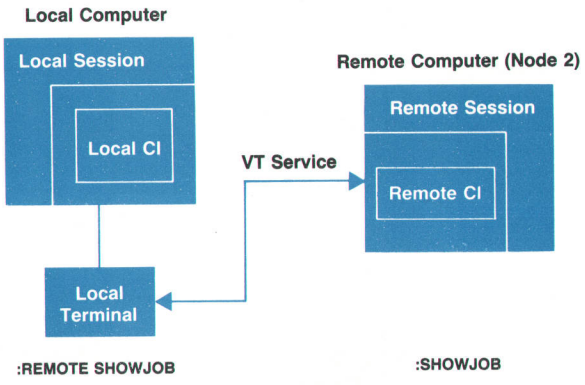


Fig. 2. *The virtual terminal service allows execution of commands on a remote system.*

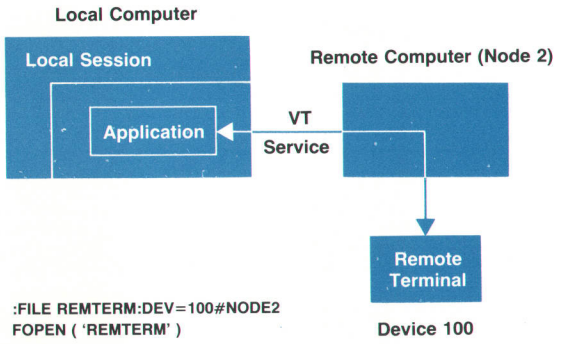


Fig. 3. *Reverse virtual terminal service allows an application on one computer to use a terminal attached to another computer.*

is the pseudoterminal (Fig. 4). A pseudoterminal is a device that is configured like a normal terminal in the HP 3000 configuration, except that it is associated with a special driver provided by NS/3000. On the computer with the program (the remote CI for remote command execution, or the local application for reverse VT), a pseudoterminal is associated with the program. When the program issues an I/O request to the pseudoterminal, the driver translates the request into a VT protocol message and sends the message to the computer where the real terminal is located. The VT software on that computer maps the VT protocol message into the appropriate I/O request for the real terminal. An advantage of the pseudoterminal mechanism is that the I/O system interface looks the same for a pseudoterminal and a real terminal, so there need be no modification to the many places where MPE performs terminal I/O operations.

Network File Transfer

Network file transfer (NFT) is an HP AdvanceNet service that allows users to copy files between nodes with similar and dissimilar file systems. The NS/3000 NFT user interface is an extension of the DS/3000 NFT interface.

When copying files between two computers, NFT executes in either interchange or transparent mode. Transparent mode is used by NS/3000 NFT when transferring a file to another HP 3000. Interchange mode is used when transferring a file to an HP host with a dissimilar file system.

Before transferring a file over the network, communication must be established between peer NFT processes on the source and target nodes. When this is done, the NFT processes exchange information, including the type of operating system each NFT process is executing under. This information is used by NFT to determine whether to transfer the file in interchange or transparent mode; the user does not need to supply this information.

If NFT is executing in interchange mode, a negotiation will take place between the NFT processes before each file is transferred. The NFT process at the source node (known as the producer) will send the target NFT process (the consumer) the file's characteristics, such as whether it contains ASCII or binary data, if it has fixed or variable length records, and so forth. The NFT consumer will determine if the specified attributes are acceptable for that file system. If any attributes cannot be supported on the consumer node, a counter offer with acceptable attributes is sent by the consumer to the producer. If the producer accepts the counter offer, a warning will be given to the user that the attributes have been changed and the producer will begin the file transfer using the negotiated attributes. The NFT protocol defines how the file's data is to be formatted in the NFT data packet, thus allowing record boundaries, if any, to be preserved.

When NS/3000 NFT is copying a file from one HP 3000 to another HP 3000, it executes in transparent mode. The NFT protocol does not define how a file's data is to be formatted when in transparent mode. Since the file's data can be written to the target file just as it was read from the source file, no massaging of the data is required. Because of this, NFT provides very efficient file transfer between homogeneous machines.

The NFT protocol is based on a three-process model. In addition to the consumer and producer processes, an initiator process is defined. The initiator process is the NFT process that interfaces with the user. The initiator is responsible for getting commands from the user, giving status to the user, and initiating the producer and consumer processes. In the NS/3000 NFT implementation, this does not imply that three distinct processes are created for all file transfers. A single NFT process can play one, two, or all three roles. For example, if a user on node 1 wants to copy file F on node 1 to node 2, the initiator and producer nodes are the same, so one NFT process would play both roles. A second NFT process on node 2 would play the third role, that of the consumer (Fig. 5).

If the same user wants to copy file F from node 3 to node 2, an NFT process would play the role of producer on node 3, another NFT process would be the consumer on node 2 and a third process would be the initiator on the user's node, node 1 (Fig. 6). Since only the producer and consumer are involved in the actual file copy, direct communication would be established between the producer and consumer for transferring the file's data. This is more efficient than DS/3000 NFT, where this type of transfer forces data to flow from the producer to the initiator, and then to the consumer.

Remote File and Data Base Access

The remote file access (RFA) service allows a program running on one HP 3000 Computer to use files located on another HP 3000 on the network (Fig. 7). There are two

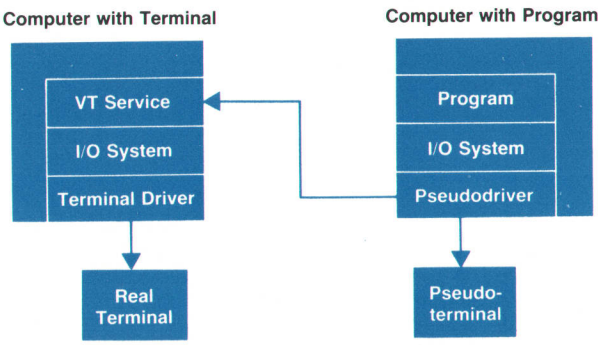


Fig. 4. *I/O operations from a program to a pseudoterminal are mapped by the virtual terminal service to the real terminal.*

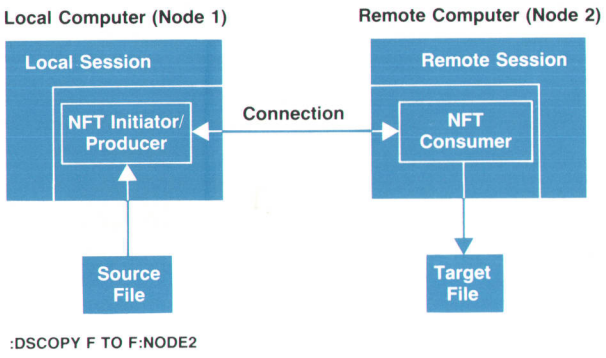


Fig. 5. *Transfer of a local source file to a remote computer.*

ways that the location of a remote file can be specified. The first way, which is compatible with the RFA service in DS/3000, is to specify the computer's node name in the DEV parameter of a FILE command or FOPEN call. For example,

```
:FILE REMFILE;DEV=NODE2#DISC
```

indicates that the file with the formal designator REMFILE is located on the computer named node 2. The second method, a new feature of NS/3000, is to include the node name directly in the file name. The extended file name REMFILE:NODE2 would accomplish the same result as the FILE command above. The advantage of extending the file name is that it allows the user to use a remote file wherever a file name can be supplied.

All types of MPE files are accessible through the RFA service. In addition to ASCII and binary data files, a remote file can be an MPE message file or a device like a printer. Both waited and nowait file access modes are allowed. Nowait file access, which was not available in the DS/3000

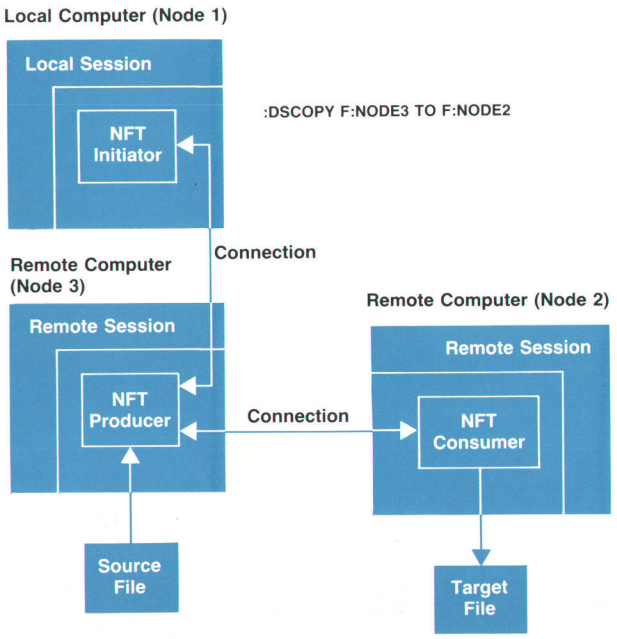


Fig. 6. *Transfer of a remote source file to another remote computer.*

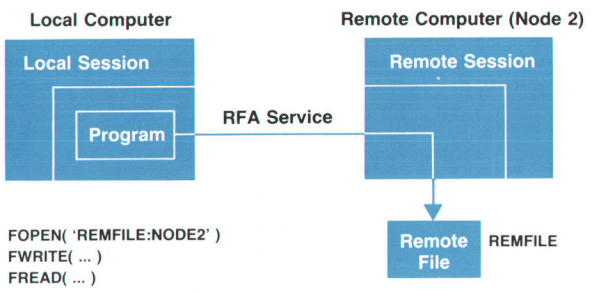


Fig. 7. *The remote file access service allows a local program to access a file on a remote computer.*

RFA, can be used to improve application performance by overlapping remote file access and other processing.

The remote data base access (RDBA) service allows a program running on one HP 3000 to gain access to an HP Image data base located on another HP 3000 on the network. RDBA is very similar to RFA in conception and execution. The HP Image data base management software invokes the NS software to set up the RDBA service to the computer holding the data base. All HP Image operations to the remote data base are directed by RDBA to the remote computer for execution.

Remote Process Management

The remote process management (RPM) service allows a program to create and destroy processes on any computer in the network. A typical use of RPM is to have a single process create a peer process with which it will communicate via the NetIPC service (see "Network Interprocess Communication," page 14).

NS/3000 RPM allows a program to create a local process (either as its child or in another local MPE session) or a process on a remote computer. RPM provides all of the process creation options available through the MPE CREATEPROCESS intrinsic.

A process created through RPM can be either independent or dependent. Independent processes remain active after the creating program terminates, while dependent processes are automatically terminated when the creating program expires.

Program-to-Program Communication

PTOP is a service that exists in DS/3000. It provides both interprocess communication and process management. PTOP allows a process to create another process in a remote session. PTOP provides a master/slave relationship between the creating process and the created process. All communication must be initiated by the master process. For instance, if the slave has data to send to the master, the master must first request the data. Slave processes are always dependent processes; if the master dies, the slave will be killed.

The designers of HP AdvanceNet felt that using NetIPC and RPM provided a better solution than PTOP. Using NetIPC and RPM allows processes to be independent of each other and communication is peer-to-peer, rather than master-to-slave. However, the NS/3000 implementers had a strong commitment to backward compatibility with DS/

3000. It was important that DS/3000 PTOP applications be able to work in an NS/3000 network, using HP AdvanceNet protocols. This was achieved by implementing another version of PTOP on top of NetIPC. Many of the problems encountered by PTOP were common to RPM, such as getting the CI to create the child process. PTOP and RPM share solutions in these instances.

Environments

A concept used by all of the network services is the environment. An environment is a session on a remote computer used by one or more services. Environments are frequently shared by several services. RFA, for example, will use the session initiated by the VT service. The NS software keeps track of the environments a user has set up on remote computers. The user can specify attributes of remote environments through the DSLINE command. The DSLINE command first appeared in the DS/3000 product. NS/3000 accepts the DS/3000 syntax and relevant options for DSLINE. NS/3000 has also enhanced the DSLINE command to include new features. The FILE command from MPE was used as the model for the new DSLINE. The FILE command associates a formal file designator with an actual file name and a set of attributes like record size, number of disc records, and extents. Similarly, DSLINE associates an environment ID with a node name and various environment characteristics.

Network Interprocess Communication

The objective of the network interprocess communication (NetIPC) team was to define a set of intrinsics that would allow HP AdvanceNet users to exchange data easily between processes. The goal was to define a service that could be used within HP and by HP customers as a foundation for building networking applications. This was achieved by defining a generic interface to protocols instead of an interprocess communication protocol to sit on top of the HP AdvanceNet transport protocols.

NetIPC is an architected interface giving users direct access to the transport's protocols. As mentioned above, it adds no protocol of its own. Because of this, the same NetIPC intrinsics available to customers are used internally by all of the NS/3000 network services.

The solution of providing an interface to the transport reduced development costs. NetIPC's functionality was required by NS/3000, so two birds were killed with one stone. No extra development effort was required to implement an interprocess communication protocol.

As a generic interface, NetIPC is defined to provide access to multiple protocols using the same set of intrinsics. The NetIPC intrinsics also provide option records for protocol specific information for users requiring the specific features offered by a particular protocol. The initial release of the NS transport provides NetIPC access only to the ARPA TCP protocol. However, NetIPC is implemented so that it can provide direct access to other protocols as they are added to the NS transport architecture.

The absence of a specific interprocess communication protocol can prevent multivendor lockout in the future. As HP implements and certifies industry standard protocols, NetIPC will provide interprocess communication through direct access to those protocols. If NetIPC had been a protocol, it would only be useful for communicating to other machines with implementations of the HP IPC protocol.

Another advantage of NetIPC's not being a protocol is performance. On the sender's side, data does not have to be copied to a buffer to add protocol information. On the receiver's end, data can be moved directly from the transport's buffers to the user's buffer. An intermediate move to a protocol module's buffer to interpret the data and strip the protocol information is not required. In addition, the overhead of transmitting a protocol's header is not incurred.

The cost of not having an interprocess communication protocol is the loss of value-added services. For example, program-to-program communication (PTOP) protocol allows the user to divide data into two parts, in essence allowing processes to send and receive two separate data buffers at the same time. Another feature of PTOP is that it allows messages to be accepted or rejected. However, the performance cost of these services is borne by all users, even the common case of users that simply want to exchange streams of data.

The NetIPC intrinsics can be used to communicate with processes on the same node, as well as other nodes in a network. Communicating with processes on the same node is achieved by the NS transport's software loopback capability. The NS transport's network interface module detects that the data's destination is the local node. Rather than giving the data to the LAN driver, the data is turned around in software. The NetIPC intrinsics manipulate entities called sockets. The HP 3000 NetIPC supports three types of sockets: call, virtual circuit (VC), and destination. Call sockets are addressable entities used to establish a virtual circuit between two processes. This virtual circuit is a full-duplex logical connection used by the processes for exchanging data. Each of the two processes has a VC socket, which identifies the endpoint of the virtual circuit owned by the process.

Destination sockets provide addressing information describing the location of another process's call socket. Given a call socket and a destination socket, a process is able to initiate the establishment of a virtual circuit. This is known as an active open. The peer process need only possess the call socket referred to by the other process's destination socket. Receiving indication that a virtual circuit has been established is known as a passive open.

NetIPC takes advantage of a name server process called the socket registry. When a process calls a NetIPC intrinsic to look up a socket name on another node, NetIPC sends a lookup request to the socket registry process on the destination node. The remote socket registry process receives the request and looks up the socket name in its tables of defined names. If the name is found, a data structure called a connect site path report is built. The path report contains the addressing information required to reach the referenced socket. This includes the socket's protocol and protocol stack. The socket registry, after building the path report, sends a reply message. From the path report, NetIPC is able to create a destination socket for the active process.

NS/3000 Transport Features

The HP AdvanceNet transport implemented on the HP 3000 Computer supports the NetIPC interface just described and the standard HP AdvanceNet transport protocol suite of TCP, PXP, IP, and Probe. The protocols are supported over the IEEE 802.3 LAN link as well as in software loopback mode. The latter offers the user the opportunity to make use of the NS services locally and to troubleshoot distributed applications before actually doing any remote communication.

PXP is a reliable datagram transaction protocol used to access a remote socket registry and resolve NetIPC socket name-to-address bindings. PXP, like TCP, is a level 4 protocol in the transport architecture. All PXP communication is in the form of requests and replies, the reply serving as the acknowledgment to the request. Reliability is achieved by retransmitting an unanswered request until the reply arrives. Duplicate requests and replies can therefore occur. Duplicate requests are not detected, but duplicate replies are detected and discarded. These limitations make the protocol suitable only for server query applications where quick response is required and duplicate requests are not harmful.

ARPA IP is the level 3 protocol in the transport architecture. IP treats all packets as internet datagrams, optionally providing datagram fragmentation and reassembly but not reliability of delivery. IP's major function is to provide nodal addressing and internet packet routing. Each LAN network is assigned a network number and each LAN node is assigned a node number within that network. A collection of interconnected networks is referred to as a catenet. The network and nodal address portions combine to form a unique 32-bit address for each node in the catenet. This address is referred to as an internet address. IP has the capability to act as a gateway between networks, using its address field and knowledge of the entire catenet to route packets to their destinations. For the first release of NS/3000, a subset of IP has been implemented that does not currently support the gateway functions.

While the transport identifies remote nodes only by their internet addresses, the user is allowed to reference nodes by alphanumeric names. The Probe is a multicast request/reply protocol which provides the dynamic resolution of node-name-to-internet-address bindings, contributing to ease of use without an additional configuration burden. The Probe also resolves internet-to-IEEE-802.3 address bindings, again lessening user configuration requirements.

The data communication policies implemented in the level 4 protocol are of particular importance in determining the performance characteristics of a transport implementation. Level 4 policies on matters such as retransmission, acknowledgment, and window use have a large impact on connection throughput, network loading, and individual system overhead.

The level 4 HP AdvanceNet protocol for reliable transmission is TCP. Although TCP is a *de facto* industry standard with a well-documented protocol definition, the policies in the above mentioned areas remain outside the standard and are therefore implementation dependent. The freedom from an all-encompassing protocol definition makes it possible to craft a TCP implementation to suit a particular system, network, or even application environment.

Given the requirements of communicating with HP 1000 and HP 9000 Computers and the objectives of eventually supporting connection to the U.S. Defense Data Network (DDN) and a variety of link types and network topologies, the NS/3000 TCP implementation combines some optimization with a large degree of flexibility. NS/3000 TCP assumes a reasonably reliable underlying network. Although it recovers from lost and duplicate packets, it is optimized for normal, error-free transmission. NS/3000 TCP functions optimally when the peer TCP adheres to the same policies, but it communicates effectively with TCP implementations using other reasonable policies.

NS/3000 TCP communication policies are designed to maximize throughput and minimize overhead through reducing the number of packets transmitted. These policies are:

- Packets received out of order are accepted.
- The retransmission strategy is a hybrid of the first-only and batch methods.
- Packet acknowledgments are minimized by piggybacking with data packets and the use of an acknowledgment delay timer.

Statistical sampling of various NFT file transfers shows the NS/3000 TCP protocol overhead to be as little as 15-20% of the total packets transmitted. This means that only a relatively small percentage of the network bandwidth and host CPU time are consumed processing packets that do

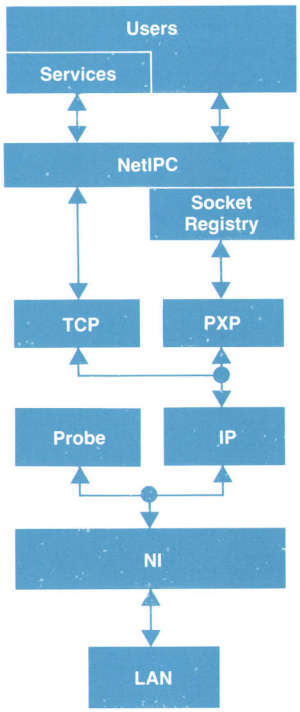


Fig. 8. *NS/3000 transport architecture. NetIPC is the interface between the services and users and the transport. TCP and PXP are level 4 protocols. IP and Probe are level 3 protocols. NI (Network Interface) provides the level 3 protocols with a generalized interface to the underlying local area network.*

not carry user data.

Configuration

There are many parameters that can affect the use and performance of NS/3000. Some examples are the maximum number of connections, buffer allocation characteristics, values for retransmission timers, and types of information to be logged. Also, the characteristics of the network links must be specified. Default values for most of these parameters can be chosen that will be reasonable for many types of networks. But to provide maximum flexibility, the parameters can be customized for each network and node.

NS/3000 parameters for a node are stored in a configuration file. A sample configuration file with default parameter values is supplied with the NS/3000 product. The network manager can then use the NMMGR program (part of the node management software) to customize the sample configuration for the node. The NMMGR program is also used for configuration of other data communications products, like the SNA transport and the LAN link.

Tracing and Logging

The NS/3000 services and transport take advantage of the tracing and logging facilities provided by HP 3000 nodal management. Both of these tools proved valuable in debugging and isolating problems during the development of NS/3000.

Tracing provides a history of events by recording user data, protocol headers, state transitions, internal messages sent between modules, and even some data structures. Tracing is provided by both the transport and NetIPC.

Logging is used to record specific events. Logged events include irregular events that require attention, such as internal software malfunctions and resource problems, and normal events, such as reporting statistics when a connection is shut down.

Transport Execution Environment

The NS/3000 transport is designed with a modular and expandable architecture, since it is intended to be the cornerstone upon which a wide variety of communication protocols and network topologies can be supported. The transport employs generalized data structures and interfaces that enable the support of emerging industry standard protocols at any level without major redesign.

Performance optimization was also a major design goal for the transport. However, a modular design can be especially susceptible to performance problems. The transport retains modularity of protocols and support functions without sacrificing performance by implementing its modules as MPE ports.

MPE ports provide a low-level message passing system. A port consists of several components: a set of message frames that are used to hold pending messages, a procedure that is called whenever a message is queued to the port, an individual context area of configurable size which the port procedure uses for its own data storage, and a control block that contains the message queuing data structures and port status information. The message queuing structures are composed of 16 prioritized subqueues, any combination of which may be selectively enabled or disabled. Port communication etiquette normally requires the adherence by all parties to a strict binding of message type (e.g. data, timer, control) to subqueue number. This allows the port procedure to receive messages in priority order and to control the type of messages it wishes to receive at any given time. All port data structures are contained in MPE data segments that are designated as port data segments.

Any process or port can send a message to a port. The port software will normally interrupt the sender's execution by invoking the appropriate port procedure immediately on the sender's stack. The port procedure receives pointers to the newly arrived message and its own context area. Unlike an MPE intrinsic, a port procedure executes as a critical section. Only one instance of a given port can be executing at a time. Asynchronous processing is provided by message queueing. If a message is sent to a port when it is already executing, the message is queued by the port software for later processing and the sender is freed to resume execution. When the port finishes executing the current message, it is immediately reinvoked to process the highest-priority message queued to an enabled subqueue.

The port concept encourages modularity within as well as across protocol boundaries. A level 4 protocol, for example, is typically implemented and treated as a single module, yet it performs most tasks as a logically separate entity per connection. Its protocol state information, such as sequence numbers and windows, must be kept separately for each connection.

The ports provide the opportunity to make the logical division of function into a physical division as well. Rather than a single port serving all connections, a protocol can be implemented as a series of ports, one per connection, with a private context area for each. The transport modules make use of separate ports per logical instance (socket, connection, directly connected network) wherever possible.

The modular and asynchronous characteristics of a design based upon ports requires shared access to the inbound and outbound data packets. All levels of the implementation require at least the ability to add headers to outbound packets and to strip them from inbound packets. However, port message frame size restrictions and performance considerations prohibit passing the actual data packets from port to port. What is needed is a packet buffering area accessible by all modules.

Rather than simply devising *ad hoc* space management

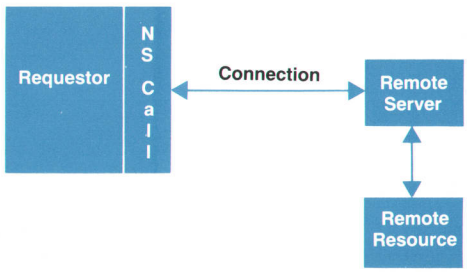


Fig. 9. *The one-server model is used for remote file access, program-to-program communication, and remote process management.*

routines, a generalized buffer management utility was implemented. The buffer manager allows creation of variable-sized pools of buffers. Access routines are provided which allow buffers to be obtained, released, written to, read from, and appended. The transport uses these buffers to hold both packet data and headers, passing a buffer identification string in the messages sent between modules.

Ports allow module execution in a process-like environment without the overhead associated with process switching. They encourage modularity of implementation and the development of generalized and well-defined message-based interfaces. The buffer manager provides a common data access method across modules and allows the transport to minimize its data copies to one each (from/to the user area) for outbound and inbound data packet processing.

Together the ports and the buffer manager form a general datacom environment allowing module execution without other low-level MPE dependencies. They were devised from a realization of common datacom needs, performance considerations, and a desire for some degree of operating system independence, and they form the design basis for the SNA XPort/3000 product as well as the NS/3000 transport.

Fig. 8 on page 15 depicts the transport architecture.

Services Architecture

The network services architecture supplies the framework in which the various services operate. Two central concepts are the requestor and the server. The requestor is a process that is requesting a certain service to be performed on its behalf. An example is an application program that opens a remote file. The server is an NS process that performs the task for the requestor. In the remote file example, the server would receive the request and open the file. The server is normally found on the computer that contains the resource (file, data base, program, or terminal) being used by the requestor.

Two server models are used by NS/3000. The first is the one-server model (Fig. 9), used by RFA, PTOP, and RPM. In this model, there is one server on the computer where the remote resource (file or program) resides. Between the requestor and the server there is a NetIPC virtual circuit connection. When the requestor issues a request to access the resource (for example, an FOPEN), NS software is called that executes within the requestor's process. This software builds a request message that indicates the requested action

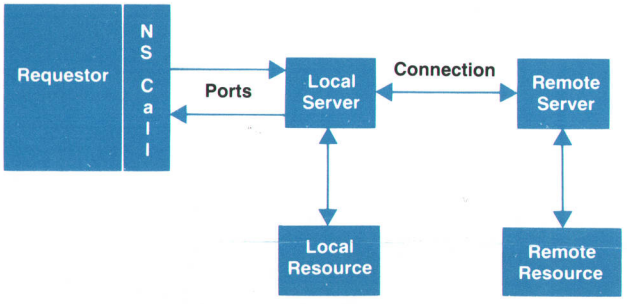


Fig. 10. *The two-server model is used for virtual terminal and network file transfer services.*

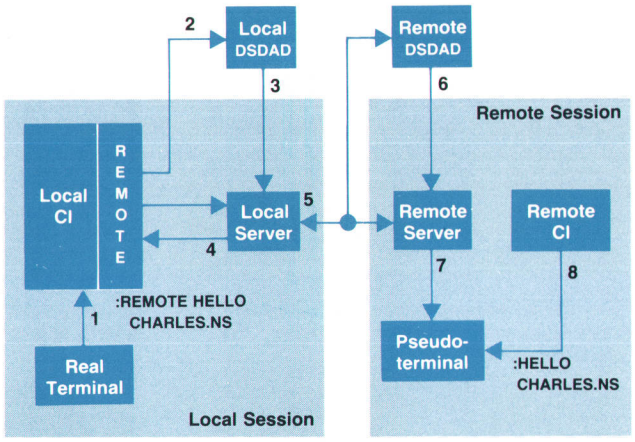


Fig. 11. *Actions during the execution of a* REMOTE HELLO *command. 1. The user enters a* REMOTE HELLO *command to the command interpreter (CI) for the session on the local computer. 2. The local CI invokes the* REMOTE *command executor, which sends a service initiation request to the port for the local VT service. 3. The local* DSDAD *process receives the service initiation request and creates a local server process for the VT service. 4. The local server sets up its own port, and* DSDAD *relays the port back to the local CI. The* REMOTE *executor sends command information to the local server. The local server adopts itself into the local session. 5. The local server establishes a connection to the socket on the remote node that is associated with the VT service. 6. The* DSDAD *process on the remote computer receives the connection request and creates a remote server process. The remote* DSDAD *passes the connection to the remote server. 7. The local server sends the logon command to the remote server. The remote server acquires a pseudoterminal and initiates the remote logon on the pseudoterminal. This results in the creation of a remote session with a CI. The remote server adopts itself into the remote session when it is available. 8. The remote CI issues a read to the pseudoterminal for a command. This signals the VT service that the* REMOTE HELLO *has completed. The VT service will retain the command read for the next* REMOTE *command.*

and sends the request message on the connection to the server. The server receives the request message from the connection, performs the requested action, and sends a reply message with the status of the action back to the requestor. Once the requestor receives the reply, it returns to the user's application.

The second server model is the two-server model (Fig 10), which is employed by the NFT and VT services. Here there are two resources to be managed: one on the local computer and one on the remote computer. For (normal) VT, the local resource is the local session terminal, while the remote resource is the pseudoterminal in the remote session. An example for NFT is copying a local file to a remote file. In this model there are two servers, one on the local computer to manage the local resource, and one on the remote computer to handle the remote resource. The requestor and local server are on the same computer. They communicate through a set of ports. There is a NetIPC virtual circuit connection between the local server and the remote server. Usually in the two-server model, the requestor starts up a request that results in a lengthy sequence of

messages between the local and remote servers. In response to the example DSCOPY request, the local NFT server (the initiator and producer) will transfer the entire file to the remote NFT server (the consumer).

The full three-node case of NFT, with an initiator on the local node copying a file from a remote producer node to another remote consumer node, can be viewed as two instances of the two-server model. The initiator and producer form one pair of servers, connected by a virtual circuit. The producer and consumer form a second pair of servers, connected by a second virtual circuit.

The network services architecture includes a control process, called DSDAD, that manages the servers. There is one DSDAD process on each computer that has started any services.

Fig. 11 shows the workings of the NS architecture during the execution of a REMOTE HELLO command.

Summary

The NS/3000 and LAN/3000 products provide an upward growth path for networking on the HP 3000. Existing applications using DS/3000 can be transported to NS/3000 systems with little or no modification. One example is the HP DeskManager mail system, which uses PTOP for its intersystem message transmission. Originally implemented on DS/3000, HP Desk was brought up on NS/3000 without any major problems.

NS/3000 and LAN/3000 offer improved throughput and data transfer rates over DS/3000. Performance measurements showed a two-to-sixfold increase in throughput for NFT, depending on the file blocking factors chosen. Transfer rates for NetIPC can average 40,000 bytes per second on an HP 3000 Series 42. The improvement in transfer rates results in part from the increased bandwidth of the data communications link (10 Mbits/s for the LAN versus 56 kbits/s for the DS point-to-point INP), and in part from improvements in the protocols.

Acknowledgments

The successes of the NS/3000 and LAN/3000 projects are the result of long and hard work by many people. In the services project, Doug Heath implemented network file transfer, John Hahn produced the remote file and data base access services, and Marg Chauvin and Jeff Orum worked on the virtual terminal service. In the transport project, Joe Devlin implemented the IP protocol and Rick Bartlett provided the network interface, the control process, and the Probe protocol. NS/3000 would have never become a reality without the heroic efforts of our test team, which included James Kao, Dave Kasberg, Karen Liebich, and Bill Medlin. Management for the projects included Bob Carlson, Connie Ishida, Ed Yang, Todd Reece, and Cherie McKinney.

Reference

1. P.M. Sakakihara, "Distributed Systems/3000," *Hewlett-Packard Journal*, Vol. 29, no. 7, March 1978.

A Local Area Network for HP Computers

LAN/3000 implements the IEEE 802.2 and 802.3 standards to interconnect HP 3000 Computers to other HP 3000s and to HP 1000 and HP 9000 Computers.

by Tonia G. Graham and Charles J. de Sostoa

THE FIRST HP LOCAL AREA NETWORK (LAN) product is an implementation of the IEEE 802.2 and 802.3 standards for local area networks. The LAN is an interconnection system intended to permit connection of up to one hundred "intelligent devices" within a facility (see Fig. 1). Data is exchanged via unacknowledged datagrams. Transmission is bit-serial at 10 megabits/second. The transmission medium is coaxial cable. Communication is half-duplex baseband using carrier sense multiple access/collision detect (CSMA/CD).

A system is connected to the network through a combination of hardware and software components (see Fig. 2). Each system connection must include a medium attachment unit (MAU), an attachment unit interface (AUI) cable, a LAN interface controller (LANIC), a LANIC driver and diagnostic, and transport software. The MAU and the AUI cable are the same for all systems. The LANIC and the LANIC driver and diagnostic are system specific; this paper discusses their implementation for HP 3000 Computers.

Medium Attachment Unit (MAU)

The MAU has three major functions:

- Receive data from the coax and pass it up to the DTE (data terminal equipment)
- Transmit data received from the DTE onto the coax
- Detect and report collisions on the coax.

The receive function provides a high-impedance load to the coax to prevent excessive current loading and interference from reflections caused by impedance discontinuities. In addition, it provides electrical and mechanical isolation of the coax from the DTE. Finally, it translates the coax signaling parameters to values that are appropriate for the

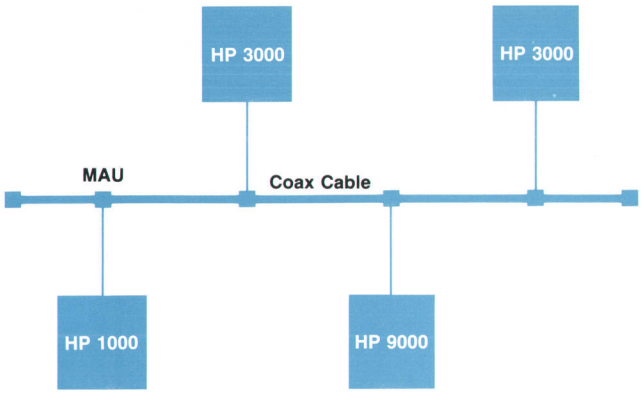


Fig. 1. *HP local area network.*

AUI; the most notable of these are rise and fall times, dc levels, and peak-to-peak magnitude.

The transmit function provides signal translation from AUI to coax parameters and wave shaping to limit the high-frequency content of the signal. To support collision detection (see below), the transmitter also adds a dc component to the transmitted signal and presents a high-impedance load so that it does not significantly impact the collision detection mechanism.

The collision detection function reports to the DTE the presence of a collision on the coax. A collision is defined as two or more transmitters active at the same time. Its presence is detected by the fact that the dc level on the coax exceeds a threshold not achievable by a single transmitter. Collisions occur during normal operation because of physical separation of the transmitters on the coax, finite propagation speeds of the coax signals, and the carrier sense multiple access protocol of the IEEE 802.3 standard. Recovery is also specified by the standard at the link level.

Other ancillary MAU functions include:

- Jab — Disables the transmitter if a transmission (activity between idle periods) exceeds a predefined time limit or excessive leakage current is detected over a prolonged period of time.
- SQE (heartbeat) — Verifies proper collision detection and reporting by the MAU after each transmission.
- dc-to-dc converter — Provides electrical isolation between the DTE and coax grounding systems.

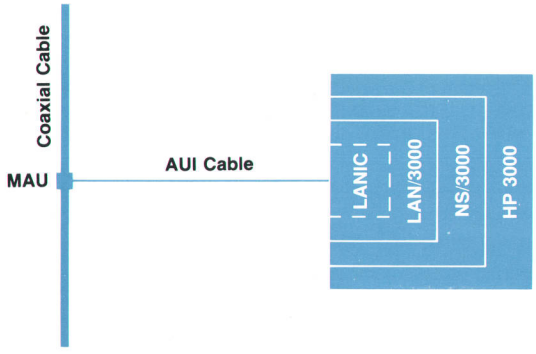


Fig. 2. *System connection for the HP 3000.*

MAU Product Verification and Defect Analysis

The design effort for the medium attachment unit (MAU) was closely tied to the development of the IEEE 802.3 standard. A great deal of time and effort was spent to design a product that is in strict compliance with the standard. A significant portion of the development focused on the many aspects of the manufacturing process, two of which were product verification and defect analysis.

Product verification does all of the testing that is necessary before a production unit can be shipped. The work of repairing the units that fail in product verification is the job of defect analysis. This moves the task of repairing the cards, and more important, analyzing the defects, to an area that is not in the production flow. Any cards that fail and are sent to defect analysis are immediately replaced by a card from the buffer area. This flow keeps production moving despite the occurrence of failures, and it permits careful analysis of the defects in an off-line environment.

The MAU test station (see Fig. 3) consists of an HP 9000 Model 226 controller manipulating two HP 8116A Function Generators, an HP 1980B Oscilloscope Measurement System, an HP 6034A HP-IB-controlled power supply, an HP 2671G Graphics Printer, and an HP 59307A VHF Switch. Using this combination of equipment and menu-driven software, the station performs a 3-minute comprehensive functional analysis of the MAU with no user intervention. The test requires one minute of setup time and can be set up by a person with no knowledge of how the assembly functions. The software runs a collection of 12 functional tests (e.g., AUI driver, coax receiver, collision detector, etc.) and informs the operator when they have all completed successfully or as soon as any one of them fails. Upon successful completion of the test the unit is ready for shipment. On failure, the station generates a list of the specifications that the unit did not meet and contrasts them with corresponding limits.

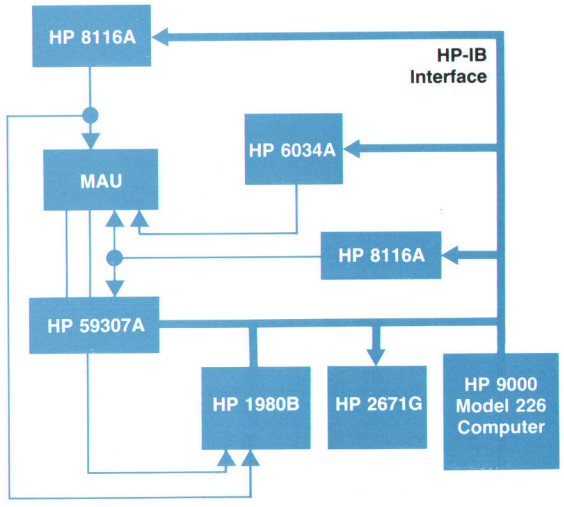


Fig. 3. *MAU test station.*

In the defect analysis area, the software allows the technician to run each of the individual tests independently. In addition, it provides a built-in troubleshooting tree which leads the technician, by means of a menu-driven map, guided probing, and digitized nominal waveforms, to a failed functional block. The guided probing acts as an instructor to the technician as it logically proceeds along a functional path in the MAU. At every branch of the tree the card is in a quiescent state with some function being exercised on a very basic level. In this manner the failing area can be quickly identified and isolated.

LANIC

The LANIC is an intelligent DMA channel that communicates with the host system via the system backplane. It consists of a backplane interface, a Z80 microprocessor for fundamental control of the LANIC, a LAN coprocessor (which provides the link-level protocol activities), and attachment unit interface (AUI) hardware.

The backplane interface is implemented with a state machine. It provides the required handshaking, input/output registers, function decoding, and DMA functionality of the LANIC. Its hardware implementation is system specific because of differences in backplane architectures, yet to the system software the differences are totally transparent.

The microprocessor system controls activities on the LANIC such as initialization of hardware and software, processing of interactive and batch commands from the host, self-test, and supervisory control of the LAN coprocessor/AUI hardware. Interactive commands and completion status are communicated between the host and the LANIC via registers associated with the backplane interface. Batch commands and their responses, on the other hand, are communicated via command and response queues physically located in host memory. The self-test resides in PROM and is an exhaustive test of all LANIC hardware, excluding the backplane interface, which is tested by the LANIC diagnostic. The self-test is executed at power-up, before downloading the LANIC operating system, and during the diagnostic. Supervisory control of the LAN coprocessor includes starting the receiver and transmitter, configuring such parameters as maximum collision retry limit, preamble length, and type of frame check sequence to use, and informing the LAN coprocessor where transmit and receive buffers are located. Supervisory control of the AUI hardware includes such things as transmission loopback from the LAN coprocessor and resetting the MAU power.

The LAN coprocessor provides the link-level protocol functionality. This includes:

- Node address filtering
- Collision backoff and retransmission
- Handling of preamble, start-of-frame delimiter, and frame check sequence fields of the transmit frame
- Deference to frames present on the network
- Verification of frame check sequence on received frames.

The AUI hardware provides Manchester encoding and decoding of the transmitted and received frames, interpretation of activity on the control lines from the MAU (i.e., collision detection), carrier sense reporting to the LAN coprocessor, and control of the MAU power.

LANIC Driver

The LANIC driver for the HP 3000 consists of three major modules: the LANIC monitor, the interrupt handler, and the initialization procedure. The initialization procedure performs certain functions that need only happen at system startup and powerfail recovery times. For example, it allocates and freezes an extra data segment for use by the monitor, locks the driver code segment in memory, and initializes the appropriate device reference table entries and the device linkage table entry. The interrupt handler is invoked by the microcode when LANIC interrupts are detected. The LANIC interrupts upon completion of self-test or a diagnostic step, completion of interactive or batch commands from the monitor, or detection of a catastrophic failure. The interrupt handler is responsible for disabling all interrupts from channels with lower priority when it is invoked and reenabling them when it is finished servicing the interrupt. In some cases the interrupt handler is able to service the interrupt completely, such as updating various status fields in the driver extra data segment. In other cases it must call the monitor to complete the interrupt service request, such as for interactive command completion.

The LAN monitor is by far the largest part of the LANIC driver. It is responsible for interpreting requests made by higher-level software to the LANIC and ensuring that these requests are processed in an orderly fashion. It is responsible for interfacing directly to the board either when it is in an uninitialized state (interactive commands via hardware registers) or after its operating system has been downloaded (batch commands via batch command/response queues maintained in host memory).

The LAN monitor is also responsible for implementation of the logical link control (LLC) sublayer of the data link layer as specified by the IEEE 802.2 standard. The im-

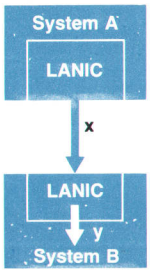


Fig. 4. *Two-node network.*

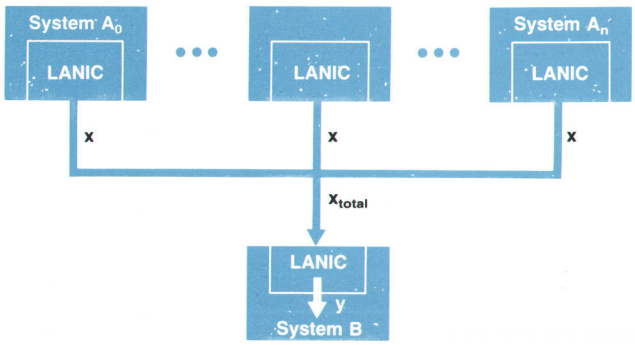


Fig. 5. *Multinode network.*

plementation is class I which implies type 1 operation (i.e., datagram). For outbound data, the LAN monitor builds the LLC header (destination and source service access points—DSAP and SSAP—and control field) in the protocol data unit before sending it to the board. For inbound data, the LAN monitor checks the control field of the LLC header to determine the appropriate action (i.e., pass data up to the network layer or reply to a test unit or process an exchange identification data unit) and either reports the DSAP and SSAP to the next higher layer or assigns the appropriate values to a test or exchange identification data unit reply.

LANIC Diagnostic

The purpose of the LANIC diagnostic is to provide a tool for determining the health of the LANIC board. The diagnostic is designed for use by both manufacturing and field service. It is run on-line; however, the network cannot be active during testing since the state of the board is altered. The diagnostic does an exhaustive test of the backplane interface and programmatically invokes the PROM-resident self-test to test the remainder of the board. In addition, tests have been included to test the AUI cable and MAU, and to perform a remote loopback test from a node specified by the diagnostician.

LAN Receive Buffer Requirements

As with most projects, many challenging problems had to be addressed during the development of the LANIC boards for the HP 3000. During the design of the architecture of the boards, determining the number of receive buffers that would be needed became an interesting problem that significantly impacted the performance, cost, and reliability of the product.

It was necessary to determine if received frames should be buffered on the board or stored directly in system memory. The architecture of the board was dependent on this decision. If only a small number of buffers were required, the right answer might be to put the buffers on the board. If a large number were required, the right answer would be to receive directly into system memory.

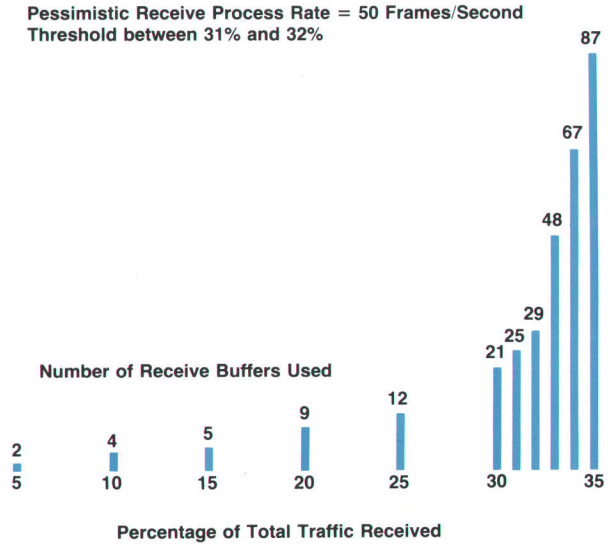


Fig. 6. *Results of a simulation done to determine the number of receive buffers required.*

Consider the case of a two-node network (see Fig. 4). If system A can process transmit frames at the rate of x frames/second and system B can receive frames and pass them to upper-level software at the rate of y frames/second, then the reservoir of receive buffers in system B must be sufficient to hold $(x-y)$ frames/second times the normal burst length in seconds.

If, on the average, $x \geq y$ for a prolonged period, an indefinite number of buffers is required. If x is always less than y, then a maximum of two buffers are required. Since the LAN medium does not contribute significantly to the transmission overhead, the transmission rate is a function of the work done by the LANIC driver and firmware to prepare the frame for transmission. This is slightly more than the work done to process a receive frame. Therefore, for the case of a two-node LAN, $x<y$.

Now consider a multinode network. The LAN topology gives rise to configurations in which multiple transmitting systems may be sending to the same receiving system. The receiving system might be a print server, shared resource manager, or gateway. Since each transmitting system can send at rate x and the input at the receiver of system B is the total of the outputs from systems A_0 through A_n, this is clearly a case of $x>y$ (see Fig. 5). Assuming that this transmission rate to system B is not maintained indefinitely, then it can be thought of as a burst of traffic. A reservoir of receive buffers will be required to hold the received frames until system B can process them. The size of this reservoir is a function of the duration of the burst.

The problem is, what sort of burst length can reasonably be expected? We decided that with the complexities of the CSMA/CD algorithm, pencil and paper calculations might be misleading. A small team was formed to simulate the behavior of a heavily loaded network. Using traffic estimates based on a typical office automation application model and a measured peak-to-average ratio of traffic in an IEEE 802.3-like environment, the team developed a traffic model to estimate the worst-case burst duration. Based on the protocol overhead, we decided that acceptable performance would be achieved if we could handle one third of the worst-case burst traffic at one receiving node.

Since the anticipated receive process rate strongly affects the number of buffers required, simulations were performed using a range of receive process rates. Using a particularly pessimistic receive process rate, the simulation indicated that indefinite reception of 31% of the frames sent on the network during peak traffic loading could be obtained with 25 buffers (see Fig. 6). A more realistic figure for receive process rate indicated that with around 25 buffers, 50% of the traffic could be directed to the receiving system with no loss of data.

This was the information the design team needed. To put 25 buffers on the LANIC would require 38K bytes of RAM, additional DMA controllers, and a triple-port memory controller. The decision was made to receive directly into system memory, thereby reducing the board space requirements and parts cost.

Acknowledgments

Information Networks Division represented HP at the initial standards work with IEEE, worked with the cable vendors to develop the coaxial and AUI cables, and developed the MAU and the two LANICs used for the HP 3000 Series 37 and Series 39, 40, 42, 44, 48, 64, and 68. Responsibility for developing networking hardware has moved to Roseville Networks Division and Grenoble Networks Division.

Network Services for HP Real-Time Computers

HP 1000 Computers have had networking capabilities for nearly 15 years. The latest network services product conforms to HP AdvanceNet standards and supports local area networks and very large networks

by David M. Tribby

THE HP 1000 FAMILY OF COMPUTERS controls manufacturing processes for HP customers throughout the world. For nearly 15 years, HP has provided the hardware and software to link these computers into networks, giving them the ability to automate entire factories.

As more customers purchase a mixture of HP computers to solve engineering, manufacturing, and office problems, they require a greater degree of interconnectivity between the computer families. HP AdvanceNet, the architectural strategy encompassing all HP computer families, provides the required set of common network services and protocols.

Network Services/1000 (NS/1000), a component of HP AdvanceNet, allows HP 1000 Computers to communicate with other HP AdvanceNet computers and with the older DS/1000-IV nodes. NS/1000 is designed for HP 1000 A-Series Computers (A600, A700, A900) running the RTE-A operating system.

Advances in HP 1000 Networking

In 1973, Hewlett-Packard released its first distributed systems product, DS 1, for the Real-Time Executive (RTE) and Basic Control System (BCS) operating systems. The only supported configuration was a disc-based central node controlling memory-based satellites. DS 1/B (and its follow-on, DS 1/B′) updated HP's network offering and provided more user-callable services, such as program-to-program (PTOP) communication.[1] The basic configuration remained a star.

In 1977, HP released DS/1000.[2] It provided many new network-level features, such as nodal addressing, store and forward routing of messages, and communication with DS/3000, the equivalent product for HP 3000 Computers. Connections between HP 1000s were no longer limited to star topologies. Memory-based and disc-based nodes could be linked together in arbitrary configurations. It was possible to address any HP 1000 in the network from a particular node; the DS/1000 software routed messages through intermediate nodes.

When DS/1000-IV was released in 1980, many new HP 1000-to-HP 1000 link-level services joined the existing DS/1000 feature set. Dynamic rerouting allowed the network to detect a broken link and use the next best path until the original link was fixed. Message accounting kept track of which messages were sent and received, assuring that each was delivered exactly once. The data link layer was moved to microprocessor-based interface cards (running up to 230 kilobits/second), freeing the mainframe from handling line protocols. The HP 1000-to-HP 3000 link added a modem link.

Several other products have added value to DS/1000-IV since its release. In 1981, the HP DataLink product allowed a master node to communicate with several slave nodes at up to 19.2 kilobits/second. X.25/1000 was released in 1982 with full DS/1000-IV compatibility. This product is an important step in Hewlett-Packard's support of recognized international data communications standards. In 1984, the RTE file system added support for remote files. A user can edit a remote file, for example, simply by adding a > and the node's number to the end of a file name when supplying the name to Edit/1000.

NS/1000, released in early 1986, fully integrates the DS/1000-IV set of user and transport services, and adds the following contributions:

- Support of HP AdvanceNet user services
- Support of communication to other HP computer families through the HP AdvanceNet transport standards
- Integration with the high-speed IEEE 802.3 local area network product, LAN/1000
- Support of large networks

Fig. 1 shows that NS/1000 includes the ISO OSI protocol layers 3 through 7. Different data communication products for the HP 1000, such as LAN/1000, provide layers 1 and 2.

The subnet layers (LAN, Gateway, and Router) perform tasks associated with controlling the data link layer. All translate between IP (ARPA internet protocol) and link-level addresses, move messages between NS memory and system memory, perform error recovery, and log error messages. The Router subnet layer, which uses DS/1000-IV message formats, manages dynamic rerouting over HDLC links.

Although X.25 is a network protocol, X.25 addressing is invisible to NS/1000. Under the current implementation, X.25/1000 manages the virtual circuits and presents NS/1000 with an interface equivalent to a data link layer.

NS/1000 includes the protocol layers for both HP AdvanceNet and DS. The three protocol stacks shown in Fig. 1 are quite distinct: HP AdvanceNet, DS/1000-IV, and DS/3000 use different user interfaces, message formats, and communication protocols. Yet all are available to NS/1000 users.

HP AdvanceNet User, Transport, and Link Services

For the first time, Hewlett-Packard has a common networking product for all its computer families. This is a tremendous advantage to users with mixed networks, and gives other customers flexibility for future expansion.

The NS/1000 implementations of HP AdvanceNet standards match the functional specifications described elsewhere in this issue (see article, page 6). The functional descriptions will not be repeated in this article, but several important distinctions for NS/1000 will be described here.

Currently, NS/1000 supports user interfaces for the following HP AdvanceNet application services:

- Network file transfer (NFT)
- Socket registry (SR)
- Network interprocess communication (NetIPC).

NS/1000 includes implementations of all HP AdvanceNet transport protocols:

- ARPA transmission control protocol (TCP)
- Packet exchange protocol (PXP)
- ARPA internet protocol (IP)
- Probe address resolution protocol.

NS/1000 is the first HP AdvanceNet implementation to include IP gateway functionality, allowing messages to pass between networks. Fig. 2 shows three local area networks connected by HDLC links. All nodes on the LANs can talk to the entire catenet. IP determines the route when a message must cross a network boundary, segmenting and reassembling messages when necessary. To minimize the amount of segmentation, IP informs TCP of the proper message size whenever possible.

The local area network connection gives HP 1000 users a much higher line throughput than the HDLC link (10 megabits/second versus 230 kilobits/second). The bus nature of the LAN reduces connection costs compared to multiple point-to-point links. Because there is no store-and-

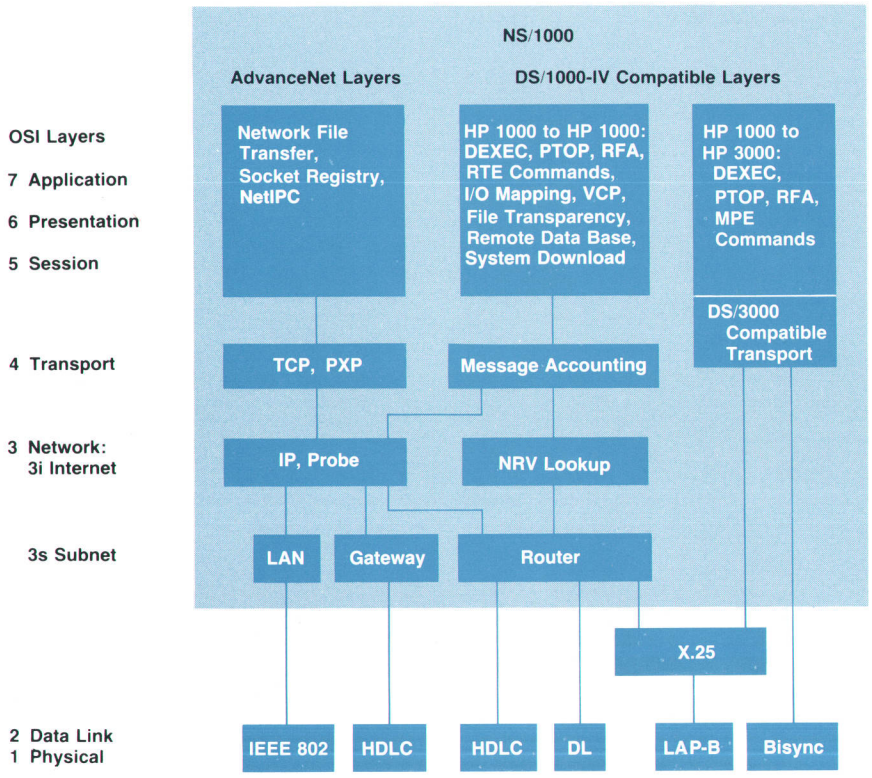


Fig. 1. *NS/1000 includes ISO OSI protocol layers 3 through 7 for communication with other NS/1000 nodes or with the older DS/1000 nodes. The data link and physical layers are implemented in separate products, such as LAN/1000 and X.25/1000.*

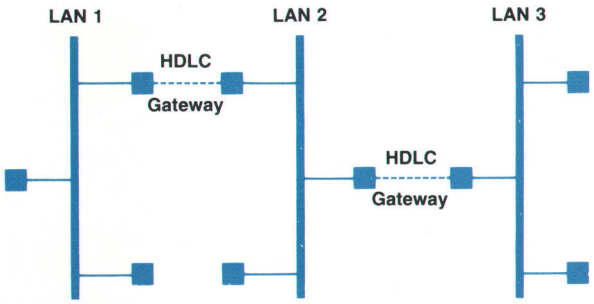


Fig. 2. *NS/1000's IP layer can route messages from nodes on one network to another network. Here three local area networks are connected by two HDLC gateways.*

forward delay, end-to-end throughput improves dramatically when a configuration goes from multihop HDLC links to a single-hop IEEE 802 link.

Fig. 3 shows the increase in PTOP performance when a link is upgraded from HDLC to IEEE 802. The lowest line on the graph indicates the number of bytes per second a master-slave pair can exchange over an HDLC link. The two programs are designed to transfer buffers as quickly as possible, and were run in dedicated systems. The middle line shows how many bytes the same two programs can transfer when the HDLC link is replaced by IEEE 802.

The top line in Fig. 3 shows the dramatic increase in throughput achieved when the PTOP programs were recoded to transfer data using NetIPC. This increase is a result of fundamental differences between DS and HP AdvanceNet services and transports. DS services are request-reply oriented. When sending multiple messages, the first reply must be received before the second request can be sent. The new HP AdvanceNet services take advantage of the transport layer's flow control and reliability, and can deliver multiple messages before an explicit reply is required.

Backward Compatibility

Compatibility with DS/1000-IV is a fundamental part of NS/1000. This is extremely important because customers need to

- Move working DS applications to NS nodes without recoding
- Mix DS and NS nodes in a network
- Communicate from NS to older HP 1000 M-, E-, and F-Series nodes
- Upgrade to use IEEE 802 links a few nodes at a time
- Add new NS nodes to an existing DS network without changing hardware in the existing network.

NS/1000 achieves all these compatibility goals.

Although HP AdvanceNet services offer attractive alternatives to existing DS services, many customers want to run current applications without modification. As shown in Fig. 1, NS/1000 maintains the DS user interface. DS services (RFA, PTOP, DEXEC, VCP, I/O mapping, file transparency, remote downloads, and remote operator commands) work to NS/1000, DS/1000-IV, and DS/1000 nodes. No change to user code is required to move an application from DS/1000-IV to NS/1000.

NS/1000 continues support of HDLC and X.25 links as they are used in DS/1000-IV. Multidrop DataLink is also supported, but only as master. Any HP DataLink slaves must continue to use DS/1000-IV. Rerouting continues to work over HDLC connections.

HP 1000 Computers other than A-Series can continue to use DS/1000-IV services to communicate with NS/1000 nodes. All DS/1000-IV-to-DS/3000 services continue to work over the DS/3000-compatible transport.

Fig. 4 shows the upgrade of a ten-node DS/1000-IV network to NS/1000. The network hub (the A900) controls three computers, and those three each control two others. Under DS/1000-IV, all links are HDLC. The diagram on the right shows all A-series nodes upgraded to NS/1000 and

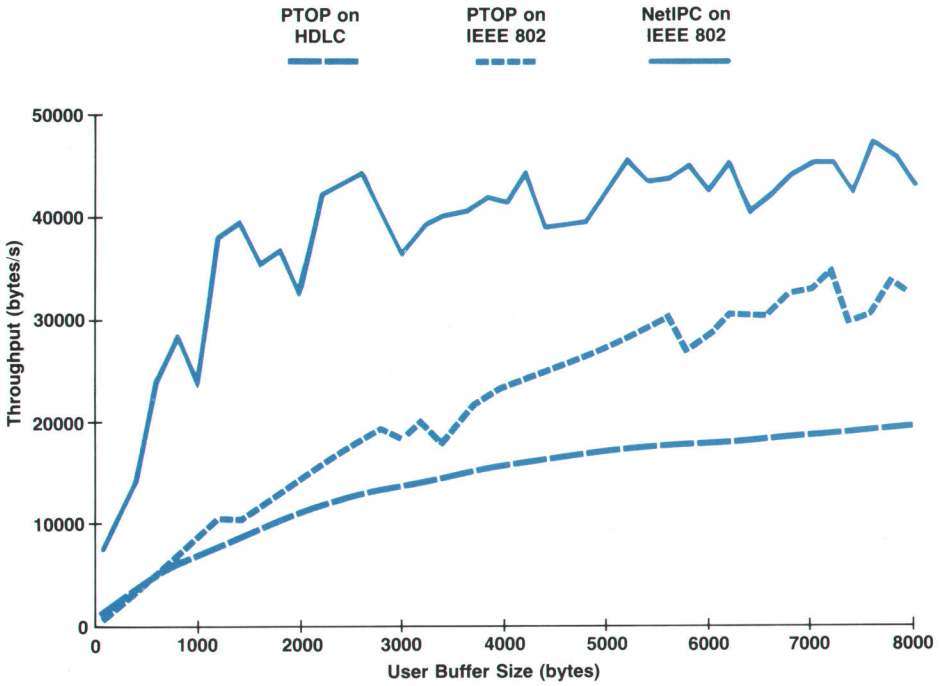


Fig. 3. *NS/1000 can increase the throughput of program-to-program (PTOP) communications applications either by changing the link from HDLC to IEEE 802.3 or by recoding to use NetIPC. These performance measurements were made on A900 computers running only NS programs.*

LAN/1000. (The nodes would not have to be converted all at once.) Any applications that run in the first network will run in the second. Communication between the A900 and the A600s will be much faster, because of the faster IEEE 802 link and the removal of store-and-forward through the A700s. The older E- and F-Series nodes are unaffected by the changes.

Protocol Flexibility

NS/1000's design provides crossover paths between the HP AdvanceNet and DS/1000-IV transport layers. This allows mixing and matching of user services and physical links.

The first crossover path in Fig. 1, between message accounting and IP, is taken when DS messages are sent over HP AdvanceNet links. The second crossover path, between IP and Router, is for HP AdvanceNet messages transmitted over DS links. Take the example of an RFA message sent over a LAN connection. The originating node prepares the message in DS/1000-IV format, but must hand the message over to IP before it can be transmitted on the physical link. At the destination node, IP hands the request back to the DS transport layer and the message continues up the DS protocol stack. The reply makes the same trip in reverse. When an HP AdvanceNet message is sent over a DS link, it is given a temporary DS/1000-IV header.

A particular physical connection is an HP AdvanceNet link if it is tied to the IP layer; IEEE 802 falls into this category. X.25 and HP DataLink must be DS links, but HDLC can be either. The HDLC link is shown twice in Fig. 1 as a reminder that a particular interface card can handle only one type of traffic.

When requests are destined from an NS/1000 node to a DS/1000-IV node (e.g., from an A600 to an E-Series in Fig. 4), the IP layer determines which NS/1000 node is the gateway to the DS network (the A900 in Fig. 4). At this guardian node, the NS routing information is translated to DS format, and the message is sent on to its destination. When the reply is received at the guardian, the message is put into NS format and sent back to the originator.

There is no crossover path for DS/3000 services or the Bisync link.

Internetwork Examples

Fig. 5 shows a complex catenet that includes several types of connections. Nodes within a site are linked on a LAN (if they are NS/1000 nodes) or HDLC router links (if they are DS/1000-IV nodes). When a message moves from one node to another, it not only has to travel the physical links shown in Fig. 5, but also the various protocol layers shown in Fig. 1.

The following two examples describe message handling by protocol layers in each node. These examples were chosen to show the flexibility of routing between different networks. The software in each node must make routing decisions, but all of this complexity is hidden from the application programmer, although much of it must be dealt with by the network manager when each node is initialized.

Example 1: Socket name lookup from node B to node D. The protocol path of this example is diagrammed in Fig. 6. A NetIPC program in node B wants to find out if there is a socket named MONITOR3 in node D and, if it exists, what socket descriptor to use when setting up a connection. The program calls the socket registry intrinsic IpcLookup with the socket and node names. IpcLookup builds a name query request and instructs the PXP transport to send the message to node D. PXP adds its header to the name query request and passes it to the IP layer. IP determines that D is not on the same network as B and so it must route the message via some other node on the local LAN. Because IP's routing tables send all Chicago traffic through node C, IP instructs LAN to send this message destined for node D to gateway node C. LAN adds its header to the message and sends it over IEEE 802. The protocol path within node B is

IpcLookup → PXP → IP → LAN → IEEE 802 → {out B, destined for C}

The message is sent, and the IEEE 802 interface card at C picks it up. LAN software in node C determines that the message is destined for IP, so the LAN header is removed and the message is passed to the IP protocol handler. IP's header indicates that the message is destined for node D, so IP consults its tables for the route. IP finds that node D

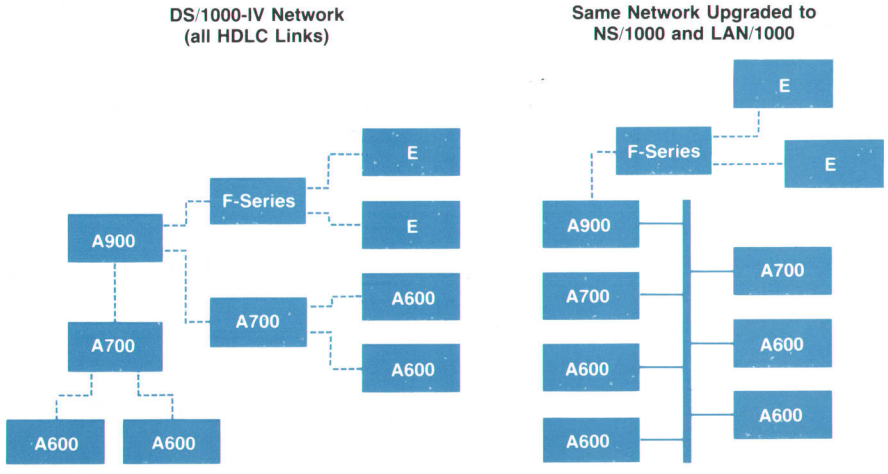


Fig. 4. *Existing DS/1000-IV networks can easily be upgraded to NS/1000 and LAN/1000. Older E- and F-Series nodes cannot be upgraded, but can access any NS/1000 node by using DS services.*

is reached via the Router network, using X.25. IP passes the message to Router software with instructions to send it to node D. Router adds a DS header to the request and transmits it over the X.25 network. The protocol path in C is

{in C} → IEEE 802 → LAN → IP → Router → X.25 → LAP-B → {out C, destined for D }

At node D, Router removes the DS header and passes the message to IP. IP removes its header and hands the message up to PXP. PXP removes its header and hands the request to the socket registry. The protocol path in the destination node (D) is

{in D} → LAP-B → X.25 → Router → IP → PXP → socket registry

The socket registry does the lookup, formats a reply to node B, and transmits it out the same path the incoming request just used.

Example 2: I/O mapping request from A to E. An operator on a node in San Jose wishes to establish a mapped logical unit (LU) in Chicago. The operator runs REMAT at node A to schedule IOMAP at node E. REMAT generates a DEXEC request, looks in its nodal routing vector for the communications link to E, and sends it out the link to B. The protocol stack path in A is

DEXEC → message accounting → NRV Lookup → Router →HDLC → {out A, destined for B}

The message arrives at B, an NS/1000 node. The Router software determines the destination node is not connected by another Router link, so it hands the message to IP. IP finds that all traffic to the Chicago network must be routed to C, so it adds its header and passes the message to LAN. The protocol stack path in B is

{in B} → HDLC → Router → IP → LAN → IEEE 802 → {out B, destined for C}

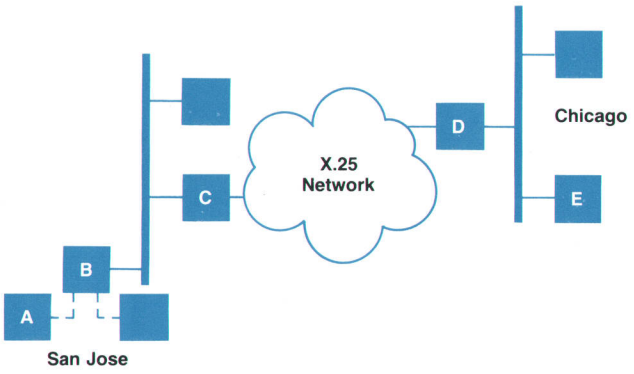


Fig. 5. *Two different sites connected by an X.25 network. On each site the NS/1000 nodes are connected by a local area network. In San Jose, two DS/1000-IV nodes are connected by HDLC links. The article describes how various messages are passed between nodes.*

At C, IP looks at the destination address and determines that the next step is node D. The message is handed over to Router, and Router hands the message to X.25 for transmission. The protocol stack path in C is

{in C} → IEEE 802 → LAN → IP → Router → X.25 → LAP-B → {out C, destined for D}

At D, Router hands the message to IP, which forwards the message to the LAN. The protocol path is

{in D} → LAP-B → X.25 → Router → IP → LAN → IEEE 802 → {out D, destined for E}

When the message arrives at the destination node (E), LAN removes its header and passes the message to IP, which removes its header and passes the message to the message accounting software for processing as a DS message. The request is given to the DS monitor which executes remote scheduling requests. The protocol stack path in node E is

{in E} → IEEE 802 → LAN → IP → message accounting → scheduling monitor

After the monitor acts upon the request, it sends a reply to A. This reply takes the same path through protocol layers and links as the request, but in the reverse direction.

Both of these examples are simplified because the requests and replies do not have data buffers. If data were present, the IP layer would have to segment and reassemble messages of more than 1490 bytes when they are transmitted across IEEE 802 links.

Execution Environment

As the internetwork examples show, NS/1000 requires efficient methods to pass messages between protocol layers and to add or subtract protocol headers. A significant challenge in implementing NS/1000 was fitting this architecture within the RTE-A operating environment without taking too many system resources, and maintaining a real-time response level for other subsystems.

The heart of NS/1000 message passing is the memory manager. When users build their systems, they can reserve up to two megabytes of memory for DSAM, the NS/1000 memory area. The NS initialization program allocates this memory for global variables, network tables, and data buffers.

In DS/1000-IV, network tables are maintained in the RTE system memory block. Because this area is typically less than eight pages and must be shared with RTE and other subsystems, DS networks are limited to about 100 nodes (depending upon which network options are being used). With the NS memory area, several hundred nodes can be accommodated.

The DSAM data buffers are implemented as linked lists. When a header is added to a message, it is put into DSAM and linked onto the head of the list. No additional processing is required for the existing portion of the message. Similarly, at the receiving node the headers are removed efficiently.

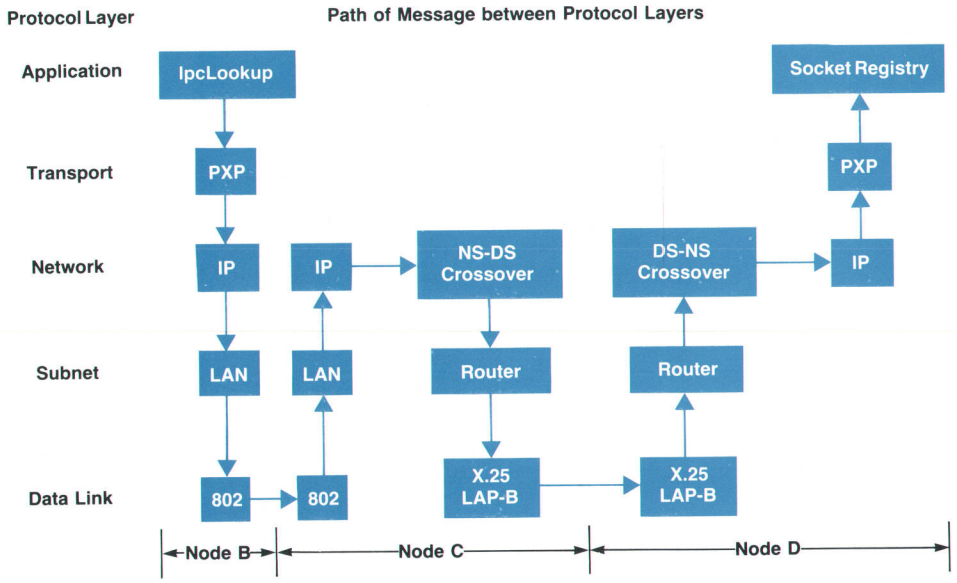


Fig. 6. *Example 1 in the text describes the path of an* IpcLookup *request sent from node B to node D. This diagram summarizes which protocol layers are involved at each node.*

In the RTE operating system, the code that provides the protocol layers has to run outside the kernel. The architecture in Fig. 1 suggests a straightforward implementation: write each protocol processor (TCP, PXP, Probe, IP) as a separate program. However, to minimize the number of process switches, we chose instead to implement all the inbound protocol handlers in one program (INPRO), and the outbound handlers in a second (OUTPRO). This has the added advantage of using only two RTE-A program ID segments.

Both INPRO and OUTPRO have control code which switches between the protocol handlers. The protocol switching code uses path reports and the information provided by the protocol modules to pass messages to the proper handler. This flexible structure allows the addition of new protocols to the NS/1000 architecture.

In internetwork example 1, the user program running in node B uses the IpcLookup subroutine to format the name query request and pass it to OUTPRO. The protocol switching code in OUTPRO hands the message to PXP, then to IP, and finally to the LAN link interface, all without the overhead of additional program dispatching.

Nodal Management

NS/1000 includes several important nodal management tools.

Because NS/1000 allows complex networks of computers, the initialization of each node can be a significant task. NS initialization utilities have been designed to tell the network manager what options are available and (whenever possible) to suggest a default value. These utilities permit shutting down NS, then restarting it with new parameters.

Once the node is initialized, users can run the NS information program to see the status of various network parameters and variables. By looking at how many networking resources are in use, a network manager can tune a node's initialization to match its application.

NS/1000's trace facility records at two levels: data entering and leaving NetIPC sockets and messages entering and leaving link interfaces. The captured data can later be analyzed to verify application programs.

NS/1000 logging records unusual events. At the choice of the network manager, the logging file can include warnings and recovered errors as well as serious problems. These messages can point to areas where the node was not configured properly or needs additional network resources.

Acknowledgments

The lab development team overcame many challenges to define the HP AdvanceNet standards, and then design, implement, and test NS/1000. Craig Wassenberg led the architecture design, implemented NetIPC, and managed much of the configuration testing. Several engineers each designed, implemented, and tested a significant piece of the product: Clark Johnson (IP), Alex Lau (TCP and PXP), Jack Repenning (memory manager), Lissa Martin Sesek (initialization), Tom Strom (NFT), and Anne Hathaway (LAN interface). Eric Wilford, Reza Memar, and Marjo Mercado concentrated on implementing and testing backward compatibility modules, while Chye Lin Chee and Jerry Chu tested portions of the HP AdvanceNet services. Frank Fiduccia helped design the memory manager and later assisted in system debugging. Lyle Weiman, Dennis Allen, and Joan Lawler were loaned to the NS/1000 project to implement several vital pieces. Section-level management was provided by Ed Yang, Wim Roelandts, and Roger Cheung.

Key contributors from other functional areas include Tim Temple and Jeff Williams (product assurance), Doug McLean (product marketing), Liz Tam and Jacquie Toth (manuals), and Colleen Dillon and Zenia Latoff (training).

References

1. S. Dickey, "Distributed Computer Systems," *Hewlett-Packard Journal*, Vol. 26, no. 3, November 1974.
2. R. Shatzer, "Distributed Systems/1000," *Hewlett-Packard Journal*, Vol. 29, no. 7, March 1978.

Networking Services for HP 9000 Computers

Developing a networking service for the members of the HP 9000 family presented a number of design challenges, among them dealing with different microprocessor architectures, the introduction of new members to the family, and being able to communicate with other HP computer products.

by J. Christopher Fugitt and Dean R. Thompson

THE HP ADVANCENET IMPLEMENTATION for Hewlett-Packard's engineering workstations is a group of transports and services known as NS/9000. NS/9000 services allow a user of an HP 9000 Series 200, Series 300, or Series 500 HP-UX workstation to exchange files freely with another HP 9000 workstation, an HP 3000 system, or an HP 1000 system. In addition, NS/9000 provides transparent file access between HP-UX systems and direct access to the Ethernet, IEEE 802.2, and IEEE 802.3 protocols, allowing a sophisticated user to write high-speed network applications. NS/9000 also provides a set of diagnostic utilities and other aids for diagnosing network configuration problems.

HP 9000 Networking Requirements

In designing the NS/9000 products, we had to satisfy several requirements.

- The HP 9000 machines had to be members of the HP AdvanceNet family. This implied implementation of the interchange format of the network file transfer protocol, the HP AdvanceNet transports, and the name-to-address resolution protocol.
- The various members of the HP 9000 family of HP-UX-based machines had to communicate in a smooth, reasonably transparent fashion. The HP-UX standard remote file access (RFA) service provides this capability.
- A straightforward, low-level access to the Ethernet and IEEE 802.3 protocols was necessary, since many of our customers wanted to develop high-speed custom applications around this LAN. We satisfied this by developing the link-level access service interface.
- A set of tools was required to help the user diagnose common hardware and software configuration errors. To this end, the landiag and linkloop utilities were developed along with an extensive chapter on troubleshooting in the manuals.

Network File Transfer

The HP AdvanceNet architecture allows HP 9000 machines to exchange files with other HP computers via the network file transfer (NFT) protocol. HP 9000 machines can participate in an NFT transaction in any of three roles: the initiator (the process that requests the transfer), the producer (the process that reads and transmits the source file), or the consumer (the process that receives and stores the transmitted file).

The dscopy command provides the user interface for initiating a file transfer. The basic syntax for this command is:

```
dscopy source_node#login:password#filename dest_node#login:
                    password#filename
```

Degenerate cases are allowed when either the source or destination node is the local machine. The login parameter identifies the user to the remote systems for validation of access rights and specification of ownership of the destination file. If the user provides only the login followed by a colon for this parameter, dscopy will prompt for the necessary password with echoing suppressed. Other parameters (not shown here) allow the user to specify file characteristics, such as whether records are fixed or variable-length, and whether the file contains ASCII or binary data.

NFT was originally implemented as part of the LAN/9000 product.[1] This original implementation provided only the homogeneous system (transparent mode) portions of the NFT protocol, since the specification was still evolving. This was implemented in MODCAL (a Pascal derivative) in the kernel space of the Series 500. For the NS/9000 products, this original code was converted to C and placed in user space. The code was then augmented with the nonhomogeneous system portions of the protocol (interchange mode). Correct operation across the various systems was verified by an extensive set of test scenarios (see box, next page).

Remote File Access

Remote file access (RFA) provides smooth, transparent access to files on other HP-UX systems. The basis of RFA is the network special file. This is a special file, similar to a device file, that identifies the remote component of a path to the file system. A network special file is created for each remote machine on the network.

To access one of the machines on the network, the user first establishes access credentials with the remote machine

Connecting NS/9000 and NS/3000

Connecting HP computer products together is a vital part of the HP AdvanceNet strategy. Part of the connection was made possible with the release of the NS/3000 and NS/9000 products, providing network file transfer as the common service for transferring data. However, even with these releases, the NS/9000 and NS/3000 connection was not free.

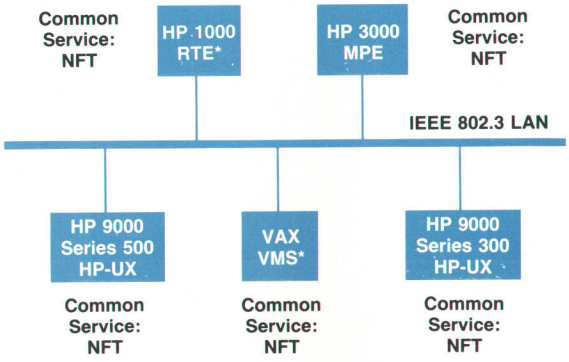
Before connecting the NFT service between the HP 3000 and the HP 9000, major problems had to be addressed. The first was the file system differences. The HP 9000 contains the HP-UX hierarchical file system in which there is only one type of file: a stream of bytes. These bytes can be ASCII characters or binary data. The meaning of these bytes depends on the programs that interpret that file. There are no record structure formats imposed on the file, and only sequential access is possible. The HP 3000 contains the MPE file system, which has multiple file types. The file type identifies the data contained in the file and the record structure of the file. The HP 3000 also allows file locking.

After investigation of the file system differences, it became clear the NFT service had to grow from a file transfer service into a file type translation service as well. The HP-UX files were defaulted to the variable ASCII (VA) file type on the HP 3000. New options were created on both the HP 9000 and the HP 3000 to allow for the HP-UX translation to fixed ASCII (FA), variable binary (VB), and fixed binary (FB) file types. These file types are the only ones supported for transfer to and from the HP 9000. The newline character is used to denote the end of record in ASCII HP 9000 file transfers to the HP 3000.

In transferring binary files, the NFT service does not perform any interpretation of data. For instance, if a data file that contains floating-point numbers is transferred to a different system type, there is no guarantee that the remote system can read the representation as floating-point. Consequently, the usability of target files must be determined by the applications that use them.

When an HP 9000 file is transferred to the HP 3000, the MPE operating system preallocates space for the file. To do this, the MPE system must get the maximum number of logical records in the file from the HP 9000. The HP 9000 approximates this value by dividing the number of bytes in the file by 15. In creating fixed files, the HP 9000 defaults the file size to 160 bytes. These values work fine for the majority of the transfers but can cause problems in some transfers. Special NFT options have been provided to change these values if needed. The implications of these assumptions and defaults, and the options used to change them, are fully described in the NS/9000 and NS/3000 manuals.

Other problems centered around various design decisions made for the HP 9000 and HP 3000 transports. The HP 9000 transport design consisted of batch retransmissions, inability to receive out-of-order packets, and retransmission timeout values in the range of 200 ms. In contrast, the HP 3000 transport design consists of single-packet retransmission, ability to receive out-of-order packets, and retransmission timeout values in the range of 2 seconds. This design mismatch created problems under stress conditions. Eventually the design for the HP 9000 transport was changed to that of the HP 3000.



*Not yet tested with NS/3000 and NS/9000.

Fig. 1. *Current and future network file transfer (NFT) service connections between NS/9000 and other HP network services products.*

To integrate NS/3000 and NS/9000, a separate group was formed to test the connection. This group consisted of personnel from the Colorado Networks Division, Roseville Networks Division, and Information Networks Division. Testing was divided into functional testing and system testing.

A definition document was created to define the functionality supported in the NS/3000 and NS/9000 connection. This document was used as the basis for manual extensions in both NS manuals. Functional testing was executed from both the HP 9000 side and the HP 3000 side on the NFT, transports, LAN driver, and Probe modules. System testing was executed to test that the connection could withstand stress for long periods of time. Testing of this heterogeneous connection relied heavily on the fact that these products had already been thoroughly tested for the homogeneous connection. Defects found were fixed and new releases for NS/9000 and NS/3000 provided the heterogeneous functionality.

Connecting NS/9000 and NS/3000 was one step in implementing the HP AdvanceNet strategy. The NFT service between NS/9000 and NS/1000 and NS/VAX will soon be added, as shown in Fig. 1.

Tim DeLeon
Project Manager
Colorado Networks Division

by means of the netunam command:

```
netunam /net/network_special_file login:password
```

This creates a logical connection with the remote system specified by the network special file for this user. Other users on the system can also perform netunam commands to the same system with different logins. Each user's logical connections are kept separate, thus maintaining the proper level of security.

After the netunam, the user can access files on the remote system as if its root directory were mounted on top of the network special file. For example:

```
ls /net/network_special_file/users/lan
```

will list the contents of the /users/lan directory on the remote system, and

cd /net/network_special_file/users/lan

will establish /users/lan on the remote system as the user's current working directory. Access to files in that directory on the remote system can then proceed as if the files were local. Any path names that begin with a leading slash (/) are always considered relative to the local root even if the current working directory is remote.

The specification for the RFA service was developed as a part of the original LAN/9000 product for the Series 500. The Series 200/300 implementation was developed to adhere to that specification. This extends to both the protocol itself (the packet formats that are actually transferred across the network) and the syntax of the associated commands and the error information that is returned by them. The second part, consistent error reporting, is especially important in that it makes applications using RFA (shell scripts, etc.) fully transportable between the systems.

Link-Level Access

Link-level (data link layer) access (LLA) allows a user to transmit and receive Ethernet or IEEE 802.3 frames with a minimum of overhead. This access is provided through the file system in a fashion similar to raw access to other devices such as discs.

LLA activity is centered around a device file. This file references the appropriate LAN interface card and also specifies the protocol to be used. The HP 9000 supports communication via both the IEEE 802.3 and the Ethernet protocols through the same interface simultaneously. Fig. 1 shows the header formats of the two protocols. For outbound traffic, there is no ambiguity regarding which protocol should be used since this is specified by the device file. For inbound traffic, however, the choice of how to interpret the frame is more complicated. To do this, we rely on some interesting alignments between the headers.

When a frame is received, it is treated initially as if it were an IEEE 802.3 packet. The length field is examined and checked for validity. If this field carries a valid length (0 to 1500), then the frame is assumed to be an IEEE 802.3 packet. If the length is invalid, the frame is assumed to belong to the Ethernet protocol. (Since Xerox has restricted the type field use in this range, this is a safe assumption.) Using the LLA service is as straightforward as using any other device file. The following is pseudocode for a simple LLA command sequence:

```
descriptor = open (/dev/LAN_device_file)
ioctl (descriptor, LOG_SSAP, s_sap)
ioctl (descriptor, LOG_DESTINATION_ADDRESS, d_sap)
write (descriptor, data)
read (descriptor, data2)
close (descriptor)
```

To initiate the operation, the user opens the proper device file. This establishes the interface to be used and the protocol to be spoken. The open call returns a file descriptor that identifies this particular LLA instance. This descriptor is then passed to an ioctl call to specify the source service access point (SSAP) to be associated with this instance. Any inbound frames whose destination service access point (DSAP) field corresponds to this value will be queued for reading on this file descriptor. A second ioctl call is made to specify which remote machine is of interest. When a frame is transmitted, this address will be placed by the driver in the destination address field of the packet. After these ioctls, data can be transmitted and received by making write and read calls. When the user is finished, the file is closed, returning any resources that have been dedicated to it by the driver.

Additional ioctl calls are available to control other packet header fields or the amount of caching associated with a file descriptor and to provide access to statistics kept by the card and driver. Control of certain other card configuration parameters is also performed via ioctl.

Diagnostics

NS/9000 provides two diagnostic functions to help the user troubleshoot any hardware or software configuration problems that may arise. The first of these is the landiag utility. This allows the user to obtain statistics about a network interface and to provide some control over the interface. Landiag also allows the user to bounce packets off a remote HP 9000 at the service level, verifying that the transports and addressing information are correct.

The second diagnostic utility is the linkloop program. Linkloop uses link-level-access to bounce packets off the driver level of the remote system. It performs this by transmitting IEEE 802 test frames, which are looped back by any remote system that supports the IEEE 802 protocol.

The use of these functions is tied together by a comprehensive troubleshooting guide in the manuals. With it, a user can quickly isolate and correct most network configuration problems.

Using the Network during Development

The network was used very heavily during the development of the NS/9000 product. Since the Series 500 already had homogeneous network facilities in place, these were used from the very beginning of the project. Source code

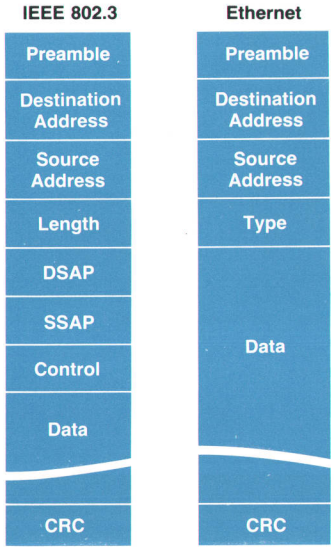


Fig. 1. *IEEE 802.3 and Ethernet frame formats.*

was kept on a large central Series 500. All system releases were built there and transferred via the LAN, using the network services, to the test machines.

Since the Series 300 initially had no networking capability, a different strategy was employed. The Series 300 networking product is implemented on the Leaf Node Architecture and much of this code is shared with other implementations (see box below). Common access to the source code had to be provided, so it was also placed on a large central Series 500. To build a system, the source code had to be moved to a Series 300. Initially this was accomplished by staging the code through a shared resource manager (an earlier form of proprietary local area networking, providing a dedicated file server, available for HP 9000 workstations) that was accessible by both systems. As the network became stable on the Series 300, a series of shell scripts was provided that retrieved the files directly from the source data base on the Series 500 via the RFA service.

Since all HP 9000 family members share common network interface definitions, much of the test code and other tools were also shared. These tests were also distributed to the various test machines via the LAN.

Acknowledgments

We would like to thank the following people who helped make NS/9000 a reality. Steve Booth was responsible for the LLA definition and implementation of the name-to-address mapping mechanism. Bill Mowson developed the transports for the Series 300. Mike Shipley, Dave Joslin, and George Feinberg implemented Series 300 RFA. Jeff Wu developed the Series 300 driver and the linkloop utility. Mike

Leaf Node Architecture

After the implementation of the networking code for the HP 9000 Series 500 Computers in 1983, a plan was formulated to place the entire range of the computer product line on HP's strategic network, HP AdvanceNet. At that time the HP 150 Personal Computer (also called the Touchscreen) was just being introduced, and it was clear that it and its successors would have to network to the HP 3000 Computer.

We looked at a number of alternatives, including reusing code from the HP 9000 Series 500, HP 3000, and other places. However, it was determined that none of these would provide an appropriate solution for a personal computer. In particular, these alternatives required a robust multiprocessing operating system as well as large amounts of memory. Since such resources are not available on the HP 150, it was evident that a totally new network transport would have to be designed. It seemed that the best thing to do would be to design for maximum portability and minimum size. With this in mind, we designed a system that requires very few capabilities from the operating system, but instead implements the required support itself. This became known as the Personal Computer Networking Architecture. The reason for calling it an architecture was that it defined the way all of the rest of the network code was to be constructed, and the rules that it must obey.

Several benefits are derived from this new architecture. The first is that, because the network code is minimally dependent upon the operating system, it is maximally portable between a wide variety of machines. This spans several operating systems and processor types. The second benefit is that it is much more efficient than some previous implementations. This is not only because we learned from our previous experience, but also that we were able to tune the architecture routines specifically for networking rather than use the general-purpose operating system equivalents.

At the time this architecture was being designed, the HP 9000 Series 200/300 Computer designers were also looking for a design that might be better adapted to their needs than the original Series 500 implementation. After studying the possibility extensively, it became evident that the PC architecture would be easy to use with very few adaptations. One was the use of the C language as the primary development language, since the code would have to execute inside the HP-UX operating system kernel. C was also supported on more machines than any other available language. Another adaptation was the capability of handling multiple users, something that MS™-DOS is not concerned with. It was renamed the Leaf Node Architecture, primarily to denote that it was intentionally not designed for network gateway operations, which are particularly expensive to design and implement.

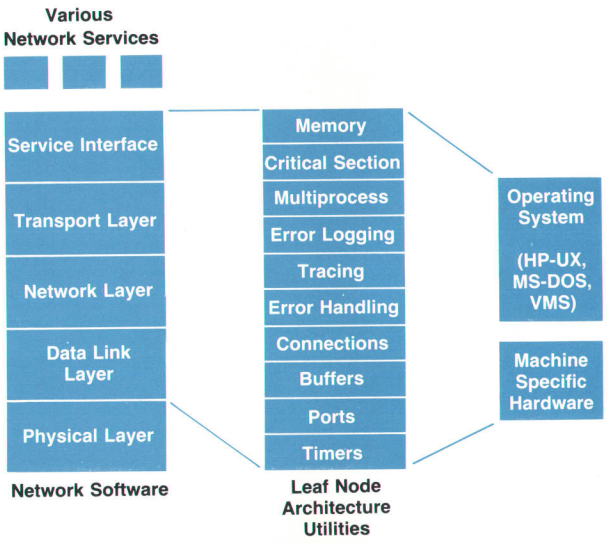


Fig. 1. *The Leaf Node Architecture provides a standard interface between the network software and the operating system.*

Details

The Leaf Node Architecture performs several functions:

- It isolates the rest of the network code from operating system and hardware specifics by providing a standardized, portable interface.
- It provides a highly tuned set of utilities for general use by the network protocols.
- It standardizes the method used by the protocols to communicate. In so doing, it also helps to streamline and modularize the network protocol software.

The architecture consists of the following modules (see Fig. 1): memory management, critical section protection, multiprocess management, error logging, tracing utilities, utility macros, error handling, and network-specific operations such as connection management, buffer management, port number alloca-

tion, timer management, and interprotocol communication. All of these are tuned to network use. For example, the memory manager is not general-purpose, and therefore requires a mere 1140 bytes even when coded in the C language on the Series 200 and 300. The interprotocol communication module consists of no code at all; it is merely a set of rules, conventions, and data structures. Other modules such as multiprocess management may in fact use the HP-UX operating system facilities when available, but emulate those facilities in an MS-DOS system.

What Have We Learned?

The objectives for the leaf node architecture were portability, overall small size, and execution efficiency. These were achieved to a large extent.

The portability of this code is demonstrated by the fact that this architecture to date is functional on numerous machines, including the HP 9000 Series 200 and 300 Computers, Digital Equipment Corporation VAX/VMS systems, and the HP 150, Touchscreen II, and Vectra Personal Computers. The portability includes not only the network product itself, but extends to the test harnesses that were used to exercise the code. In addition, because much of the code shares a common source between multiple products, we were able to find and fix defects once in the common code rather than rediscovering similar defects in the different products.

The size of the leaf node implementation on the Series 300 is less than half the size of the previous Series 500 networking implementation. This is primarily because it was designed for a restricted set of functionality and implemented in C.

The efficiency is harder to measure, but experiments have shown that the Leaf Node Architecture designs are substantially faster than previous implementations.

One other thing learned in this project was that particular services such as network file transfer and remote file access should not necessarily be implemented within the constraints of the Leaf Node Architecture. One reason is that these services are highly integrated with the operating system. Another is that they are not sensitive to real-time constraints as is the transport. These services are, however, designed for portability in themselves.

Carl Dierschow
Project Manager
Colorado Networks Division

Robinson designed the services interface to the network, Thom Bartz implemented NFT across the family, and Christian Brunet developed the landiag utility. We would also like to thank the test team of Dennis Freidel, Alan Burke, Ken Aird, Tim Remple, and Arun Chandra for their efforts. Finally, we would like to thank the other members of the management team: Jim Willits, John Bugarin, and our section manager, Sandy Chumbley.

Reference

1. J.J. Balza, H.M. Wenzel, and J.L. Willits, "A Local Area Network for the HP 9000 Series 500 Computer," *Hewlett-Packard Journal*, Vol. 35, no. 3, March 1984.

Authors

October 1986

4 OSI Model

Donald C. Loughry

With HP since 1956, Don Loughry has held numerous engineering and engineering management assignments and is currently the standards manager for the Information Networks Group. Over the past 15 years he has been an active participant in the development of information processing and instrumentation systems standards. Committees under his leadership have developed nine United States and international standards related to networking and communication systems. He was instrumental in initiating and elaborating IEEE Standard 488 and has received two IEEE awards for his contributions to standards development. He's the author or coauthor of numerous papers on communication topics. Don's BSEE degree was awarded by Union College in Schenectady, New York in 1952 and he served in the U.S. Army before joining HP. He and his wife live in Los Altos, California and they have two married children. He's active in his church and enjoys working on his bonsai plant collection.

Gertrude G. Reusser

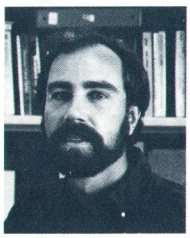
Trudy Reusser is a specialist in networks and networking standards who joined HP in 1986. She is responsible for OSI liaison and represents HP in national and international network standards development activities. Her other professional experience includes work for the MITRE Corporation, Fireman's Fund, Geico, and General Motors. She was also cofounder of a consulting firm and has consulted for many U.S. government agencies. She founded the ANSI national committee on OSI, is the author of a book chapter on the management of standards, and is working on another book. She was born in Cologne, Germany and attended Marquette University, as well as other institutions. She completed work for a BS degree in business administration in 1972. She also earned an MBA degree from the University of Pennsylvania in 1985 and is currently working on an MS degree in computer science at Stanford University. Trudy and her husband live in Irvine, California. She's active in her church, likes working with young people, and enjoys music and plants.

6 HP AdvanceNet

Robert J. Carlson

Bob Carlson is a native Californian who was born in San Francisco and attended Stanford University (BS mathematics 1971). He also has an MS degree in computer science from the University of Wisconsin (1975). He worked as a software engineer at two other companies before joining HP in 1978. He's a specialist in computer networking and communications and is the project manager for OSI. He has also contributed to the development of HP NS/3000 and HP DS/3000. Bob lives in Palo Alto, is married, and has two children. For recreation, he likes volleyball and skiing.

Craig Wassenberg

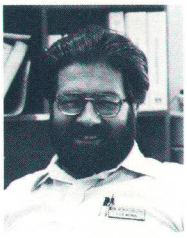


An R&D project manager at HP's Information Networks Division, Craig Wassenberg has been with the company since 1980. He contributed to the development of several standards for HP AdvanceNet and to networking products for HP 1000 Computers and is now working on networking for HP Precision Architecture Computers. A native of Wisconsin, Craig was born in Milwaukee and attended the University of Wisconsin (MSCS 1980). He's now a resident of Sunnyvale, California and enjoys tennis, skiing, and travel.

Atul Garg

Born in New Delhi, India, Atul Garg earned a bachelor's degree in electrical engineering from the Indian Institute of Technology in 1979. He continued his studies at the University of Hawaii, completing work for an MSEE degree in 1981. After coming to HP the same year, he worked on HP AdvanceNet and on HP's design of the Internet architecture and IEEE Standard 802.3. He is one of the principal authors of HP's network management architecture and protocol and is currently developing applications for network management. He's also the author of a conference paper on LAN Internet protocols. Atul lives in Sunnyvale, California and enjoys badminton, sailing, and bridge.

Lyle A. Weiman



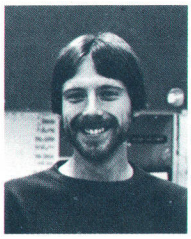
Born in Cleveland, Ohio, Lyle Weiman attended the University of California at San Diego and completed work for his BS degree in computer information science in 1972. He also worked at the University's Marine Physical Laboratories. His MS degree in computer information science was awarded by San Jose State University in 1978. With HP since 1971, his contributions include work on HP DS/1000, on HP's implementation of Manufacturing Automation Protocol, and on HP-UX networking software. Lyle lives in Cupertino, California, is married, and has three children.

Arie Scope

Arie Scope has been with HP since 1980 and manages the HP AdvanceNet program in the Information Networks Group. He is responsible for developing strategies, sales tools, merchandising, and leading the product marketing council for ING. In his first HP position in Israel, he was the country sales manager. He has also been a system engineering manager for the IBM Corporation and has written a book on data communication systems. Arie was educated at the Israel Institute of Technology and holds a BS degree in electronics engineering (1967) and an MS degree in computer science (1974). He's married and has three children.

11 Services and Transport

Kevin J. Faulkner

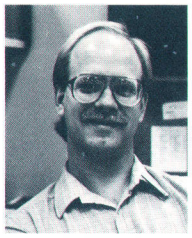


With HP since 1980, Kevin Faulkner is project manager for HP NS/3000 Transport. He has worked on HP IMF/3000 and has implemented TCP and PXP modules for NS/3000. He was born in Hammond, Indiana and educated at Indiana University. He completed work for his BA degree in psychology in 1978 and for his MS degree in computer science in 1980. A resident of Santa Clara, California, Kevin's interests include running, music, travel, and political satire.

Brian K. Lynn

Brian Lynn is an alumnus of Indiana University, receiving his BA degree in computer science in 1980. He came to HP the same year. He was responsible for NetIPC and program-to-program communication for HP NS/3000 and is now a project manager for HP NS/3000 XL. Brian was born in Trenton, New Jersey, but grew up in Pennsylvania. He's now living in San Jose, California and enjoys basketball, music, and reading in his spare time.

Charles W. Knouse



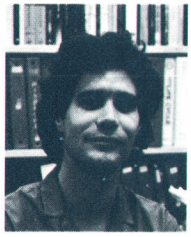
With HP since 1978, Charles Knouse is an R&D project manager for HP NS/3000 at the Information Networks Division. His BA degree in mathematics and physics was granted by Coe College in 1976 and his MS degree in computer science was awarded by the University of Iowa in 1978. His HP contributions include work on HP NS/3000 services architecture and on the RPM service. Charles was born in Davenport, Iowa and now lives in San Jose, California. He's married and likes running, hiking, reading, and playing piano and guitar.

18 Local Area Network

Tonia G. Graham

Toni Graham joined HP in 1978, the same year she got her BSEE degree from California State Polytechnic University, Pomona. She's an R&D project manager and has contributed to the development of the HP 4955A Protocol Analyzer and to HP's implementation of IEEE Standard 802.3. She has also been responsible for the LANIC for several HP 3000 Computer series and is now working on wide area networking for HP Precision Architecture commercial systems. For recreation, Toni likes scuba and skin diving as well as small- and large-format photography.

Charles J. de Sostoa



With HP since 1979, Chuck de Sostoa is an R&D engineer at the Information Networks Division. He has contributed to the development of the HP 4945A Transmission Impairment Measuring Set, the HP 30241A LAN Medium Attachment Unit, and HP LAN/3000. He has also worked on NetIPC for the HP 3000 Computer. A Michigan native, Chuck now lives in Sunnyvale, California. He was born in Grosse Pointe Woods and studied electrical engineering at Michigan State University. His BS degree was awarded in 1977 and his MS degree in 1979. He enjoys skiing and scuba diving.

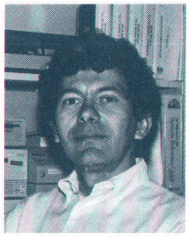
22 Real-Time Computers

David M. Tribby

Dave Tribby has been with HP since 1975 and is a network architecture engineer in the Information Networks Group. He started at HP as a technical writer and later became a software engineer and project manager. He has worked on HP DS/1000, HP DS/1000-IV, and HP NS/1000. He holds a 1975 BS degree in mathematical sciences from Stanford University. A resident of Sunnyvale, California, Dave was born in St. Petersburg, Florida. He's married and is active in his church. His favorite pastime is amateur printing. He is president of the American Amateur Press Association and publishes *The Tribby Tribune* several times a year. He also likes writing software on his home computer and rooting for Stanford at football games.

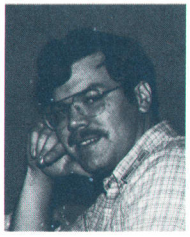
28 HP 9000 Computers

J. Christopher Fugitt



With HP since 1977, Chris Fugitt spent several years in Colorado and in Böblingen, Germany working on the HP 250 Business Computer and then developed asynchronous data communication software for the HP 9000 Series 500. He has also been a project manager for HP 9000 local area networking. His BSEE degree was awarded by Northwestern University in 1970 and before coming to HP he developed software at the University of Illinois. Chris is married and lives in Fort Collins, Colorado. He's interested in astronomy, photography, and amateur radio (N0GFM) and likes listening to classical music.

Dean R. Thompson



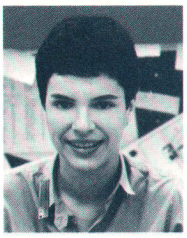
Born in Rochester, New York, Dean Thompson has a 1979 BS degree in computer science from Brigham Young University. He joined HP the same year and has contributed to the development of data communications software for the HP 250 Business Computer and the HP 9845 Desktop Computer. He has also worked on the LAN for HP 9000 Computers and is currently an R&D project manager. Dean lives in Fort Collins, Colorado, is married, and has three sons. He's active in the Boy Scouts and in his church and enjoys wood turning in his spare time.

35 X.25 Networking

Pierry Mettetal

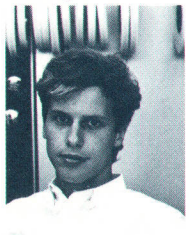
Pierry Mettetal studied computer science at Grenoble University and completed work for a BS degree before joining HP in 1981. A software engineer, she has contributed to the development of X.25 for the HP 1000 and HP 3000 Computers and is now a network consultant for the HP European network marketing center. She's interested in X.25 and other OSI standards. Pierry lives in Grenoble, France, likes to travel, and enjoys outdoor activities such as skiing, tennis, and hiking.

Nancy L. Navarro



With HP since 1985, Nancy Navarro is a senior technical writer at HP's Information Networks Division and has worked on several reference manuals for HP NS/3000. She holds a BA degree in English from the University of Pennsylvania (1981) and an MS degree in technical writing from Rensselaer Polytechnic Institute (1983). She was a technical writer at Intel Corporation before joining HP. She's a member of the Society for Technical Communication and coauthor of a paper on the use of statistics in fiscal audits. Nancy is married and lives in Santa Clara, California. She was born in Orange, New Jersey. An outdoor sports enthusiast, she enjoys sailing, skiing, hiking, and tennis.

Timothy C. Shafer



With HP since 1983, Tim Shafer is the product manager for ISDN at the Roseville Networks Division. He has previously worked on software for manufacturing automation, on HP's implementation of the IEEE 802.5 token ring standard, and on high-performance networking. He has written several articles on ISDN and is active in developing ISDN standards. Tim was born in Fort Hood, Texas and completed a BS degree in electrical and computer engineering at the University of California at Davis in 1983. He lives in Sacramento, California and is a community art supporter. He's married and enjoys wine tasting.

Deepak V. Desai

Deepak Desai is a project manager for office networking products at HP's Roseville Networks Division. With HP since 1976, he has worked on a compiler for automatic test systems, data communications products for HP 1000 Computers, the HP 27140A Terminal Multiplexer, HP DMI/3000, and the HP 30276A/30277A PBX Interfaces. Born in Poona, India, he attended Poona University, receiving a BSc degree in physics in 1968 and a BE degree in instrumentation in 1971. He continued his studies at Michigan State University and was awarded an MSEE degree in 1973 and an MSCS degree in 1976. Deepak and his wife live in Auburn, California and have three sons. His outside interests include gardening, running, backpacking, cross-country skiing, and travel.

X.25 Wide Area Networking for HP Computers

HP provides access to X.25 packet switched networks for HP computers, and in cooperation with another company, can provide entire networks.

by Pierry Mettetal

AROUND 1977, DISTRIBUTED DATA PROCESSING began to displace mainframe batch data processing. Since that time, computers have become smaller, cheaper, and more numerous, and people have become more and more interested in connecting them together to form data communications networks and distributed systems.

A data communications network is a data transmission system that interconnects data processing equipment at separate locations. When these locations are geographically remote from each other (in different cities, states, or even countries), then the network is generally referred to as a wide area network (WAN). When the geographic coverage of a network is much smaller, either within a building or connecting buildings within a distance of a few kilometers, it is referred to as a local area network (LAN).

Hewlett-Packard has been involved in distributed data processing and data communications networks for many years. With the introduction of HP AdvanceNet in 1984 Hewlett-Packard has committed itself to implement international standard protocols compatible with the ISO Open Systems Interconnection (OSI) model as soon as this becomes practical. Compatibility with the OSI model brings several benefits, but a major one is that OSI compatibility favors peer-to-peer communications among multivendor equipment, which is part of the HP AdvanceNet strategy. Hewlett-Packard started the program by implementing two international standards: the X.25 protocol for wide area networking and the IEEE 802.3 (and 802.4 in the future) protocols for the local area networking.

Wide Area Networking Alternatives

Today we can talk about four main alternatives for wide area networking:

- Circuit switched (dial-up communications)
- Dedicated circuits (leased lines)
- Packet switched public data networks
- Packet switched private data networks.

Other alternatives such as satellite channels and microwave links are now used mainly as parts of such networks rather than as total solutions.

The circuit-switched option, also commonly called dial-up communications, is tariffed based on both time and distance. Therefore, it is chosen for networks where communication sessions are relatively local and of short duration. This option can use voice-grade lines of a telephone network. The telephone line is the cheapest method for low-volume and low-speed (up to 2400 baud) data transfer. Digital lines can be used for higher-speed data transfer (up to 1024 kilobaud). However, digital lines are quite expensive and are usually used only between a few selected sites.

The dedicated circuits, or leased lines, option is commonly used in point-to-point networks and multipoint net-

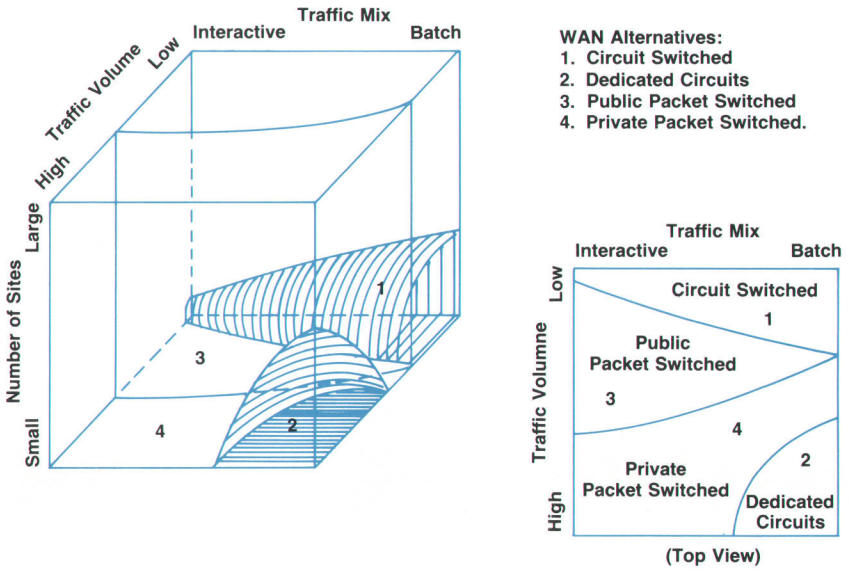


Fig. 1. *Relative coverage of wide area network alternatives.*

works in the United States. This service is tariffed based solely on distance. Therefore, it is preferred when the user sites are few and traffic is heavy and constant (high-volume batch traffic).

The packet switched public data network option is a good choice for medium-to-low-volume traffic. This service is tariffed based on both connection time and traffic volume, but not on distance. It is therefore best suited for a large number of sites spread over great distances and with sporadic traffic.

The packet-switched private data network option is a good choice for medium-to-high-volume traffic. Tariffed based on distance, it combines the multiplexing of packet switching with the time and volume independence of dedicated circuits.

Fig. 1 shows the relative coverage of these networking alternatives versus the traffic mix, the traffic volume, and the number of sites to be interconnected.

X.25 Packet Switched Network

The basic principle of packet switching consists in splitting data into small pieces called data packets, adding to each packet some information such as the sender and destination addresses, and finally sending these packets through a transportation facility called a packet switched network (PSN). The X.25 protocol defines the communication interface between the user equipment and the transportation network. In other words, it defines the access to the packet switched network. More details on CCITT Recommendation X.25 are presented later in this article.

A PSN is made up of switching nodes and data communication links. A switching node is a data communication processor that routes the data packets coming from different senders and going to different addressees through data communication links. Some data communication links (mainly dedicated circuits and/or telephone lines) are dedicated to connections between the user equipment and the switching nodes. The others (dedicated circuits, satellite channels, digital lines, etc.) are used to connect the switching nodes, as shown in Fig. 2. A separate processor called the network control center (NCC) handles network management (access control, statistics, billing, failure tracking, etc.).

One of the main benefits of packet switching is a better sharing of both switching nodes and data communication links as a result of multiplexing. Fig. 2 illustrates this principle: the user equipment A1 sends data packets to A2 while B1 sends data packets to B2 and C1 sends data packets to both C2 and C3. It is assumed that A1, B1, and C1 send data packets concurrently. Fig. 2 shows that data packets from A1 and C1 are multiplexed on link L1 and data packets from B1 and C1 are multiplexed on link L2. From the users' point of view it looks as if the pairs A1-A2, B1-B2, C1-C2, and C1-C3 all have their own private circuits. These are called virtual circuits (see Fig. 3). The PSN is said to be private when the user equipment and the PSN equipment (inside the cloud) belong to the same company. Otherwise, the company that owns and maintains the PSN equipment provides other companies with a tariffed transportation service and the PSN is then said to be public.

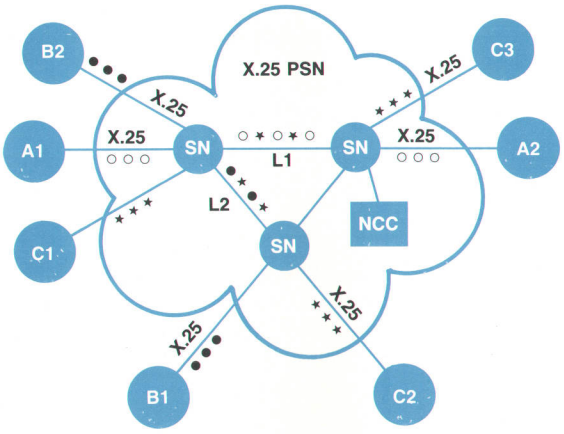


Fig. 2. *X.25-based packet switched network configuration.*

Why X.25?

As already mentioned, X.25 is an international standard protocol. It has been adopted today by many manufacturers, including IBM, Digital Equipment, Data General, Wang, and others, and therefore, it favors communication among multivendor equipment.

X.25 offers high connectivity. By using only one data communication link to the PSN, one user system is able to connect concurrently to many remote systems and/or workstations.

Almost all packet switched public data networks worldwide use X.25 as the PSN access protocol. Furthermore, almost all these networks are compatible and can communicate with each other because they use a common addressing scheme based on another international standard, X.121. In other words, any X.25 user can access an already existing worldwide public PSN.

Network availability is crucial because the information carried is essential for the user companies. Most of the public networks maintain a network uptime factor better than 99%. It can be assumed that a very similar figure is achieved by private networks.

The closed user group facility for the public network and the network control center for the private network ensure the same security level on PSNs that is achieved on point-

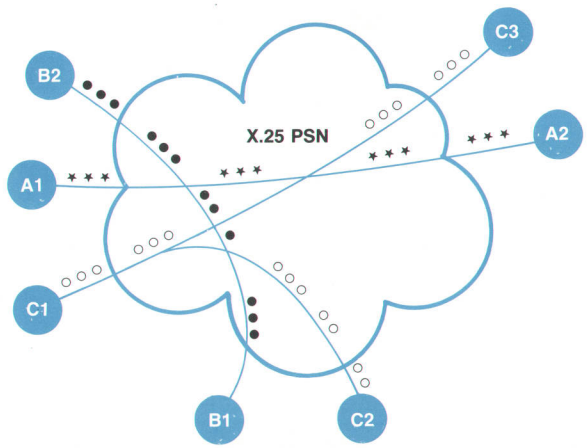


Fig. 3. *X.25-based packet switched network virtual circuits.*

to-point networks.

On private networks especially, the NCC offers efficient control of the equipment by providing statistics, billing, investment forecasts, and other management data.

Because of the high connectivity they provide, X.25-based PSNs offer a great deal of flexibility for growth and configuration changes.

Last but not least, the X.25-based PSN is a very attractive performance/cost alternative for several reasons. It is a single solution for a wide range of applications: interactive and batch traffic, low-to-high-volume traffic, medium-to-large numbers of sites, and multivendor data processing equipment. It reduces the number of data communication links required by means of dynamic adaptive routing, high connectivity, and compatibility with existing networks. It allows better use and sharing of the data communication equipment (switching nodes and links) by its intrinsic multiplexing capability. Finally, it aids network optimization by offering increased control, flexibility, and reliability over other alternatives; this is particularly true for private networks.

Hewlett-Packard X.25 Capabilities

We can distinguish two different types of HP X.25-based product capabilities.

The first type of product provides customers with X.25 access for existing HP computer systems such as HP 3000s, HP 1000s, HP 9000s, and future generations of HP computer systems. In addition, HP offers dedicated data communication equipment (HP 2334A) for connecting workstations to X.25-based PSNs (see Fig. 4).

The second type of product is the X.25-based PSN itself. Hewlett-Packard can provide its customers with switching nodes and the associated network control center for them to build an X.25-based packet switched private data network. The key to this capability is a joint marketing agreement with M/A-COM, a large telecommunications equipment company. Hewlett-Packard has selected M/A-COM equipment for its own private X.25-based PSN. Ultimately, this network will consist of 30 switching nodes across the United States, Europe, and the Far East, a redundant network control system (two NCCs), and 45 dedicated circuits (leased lines) connecting up to 2000 computers. Fig. 5 shows a typical HP M/A-COM private X.25-based multivendor transport network and Fig. 6 shows typical applications.

CCITT Recommendation X.25

CCITT (International Telegraph and Telephone Consultative Committee) study group VII was formed in 1972 to develop standards for the new public data networks. One question was to determine if use of packet switching technology was appropriate for public data networks (PDN). In 1976, a proposal was submitted to the final meeting of study group VII. That was the birth of CCITT Recommendation X.25. The X.25 Recommendation became a technically sound, practical standard in 1980.

The ISO (International Organization for Standardization) has developed the Open Systems Interconnection (OSI) Reference Model to define a standard architecture for system interconnections. This model provides a universally applicable structure, serves as a reference to position exist-

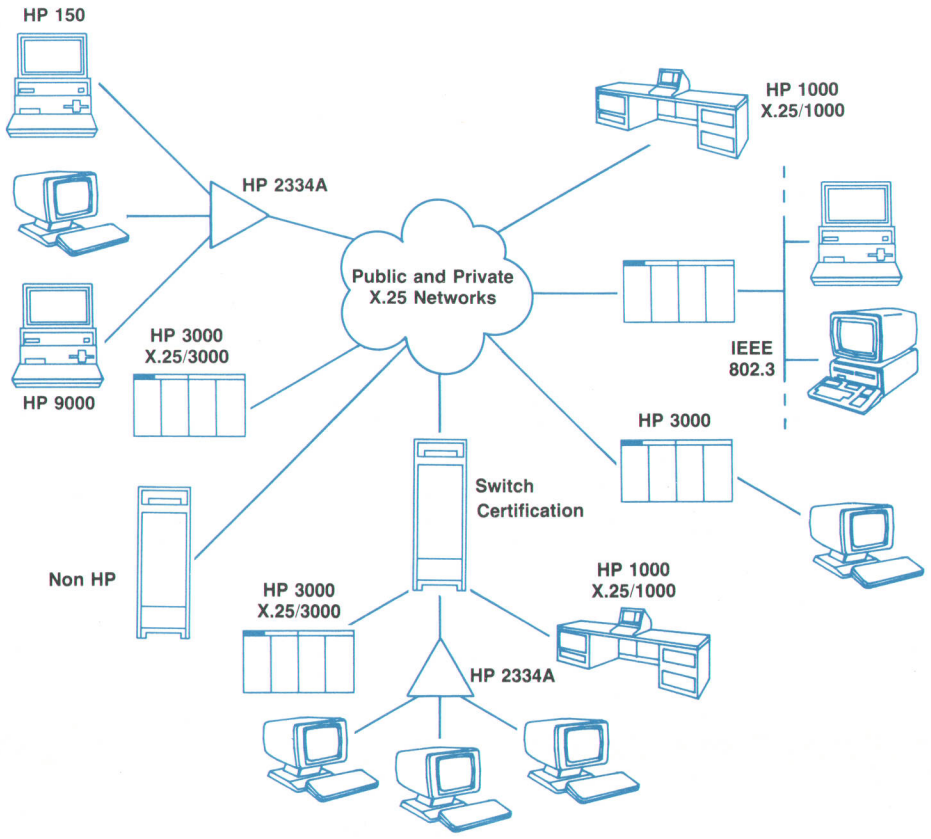


Fig. 4. *Hewlett-Packard X.25 access capabilities.*

ing standards, facilitates compatible interconnections, and will enable an evolution of advancing technology into the future.

The Reference Model has seven layers. The application layer directly serves the end user, which is the application process. The presentation layer provides the services to allow the application process to interpret the meaning of the information exchanged. The session layer supports the dialog between cooperating application processes, binding and unbinding them into a communicating relationship. The transport layer provides end-to-end control and information interchange with the level of reliability that is needed for the application. The network layer provides the means to establish, maintain, and terminate the switched connections between end systems. The data link layer provides synchronization and error control for the information transmitted over the physical link. The physical layer provides the electrical, mechanical, functional, and procedural characteristics to activate, maintain, and deactivate the physical connection. Collectively, the lower four layers can be considered a "bit pipe" for transferring information between communicating end systems.

How does Recommendation X.25 fit into the OSI Reference Model? The X.25 Recommendation defines the operations for the three lower layers of the OSI Reference Model (physical, data link, and network layers).

Recommendation X.25 references others recommendations for the physical-level characteristics. These are X.21, which defines a general-purpose interface for synchronous operations, and X.21 bis, which is the equivalent of IEC RS-232-C/V.24. These standards specify the electrical characteristics. The physical access to the X.25 network is specified for a dedicated access circuit.

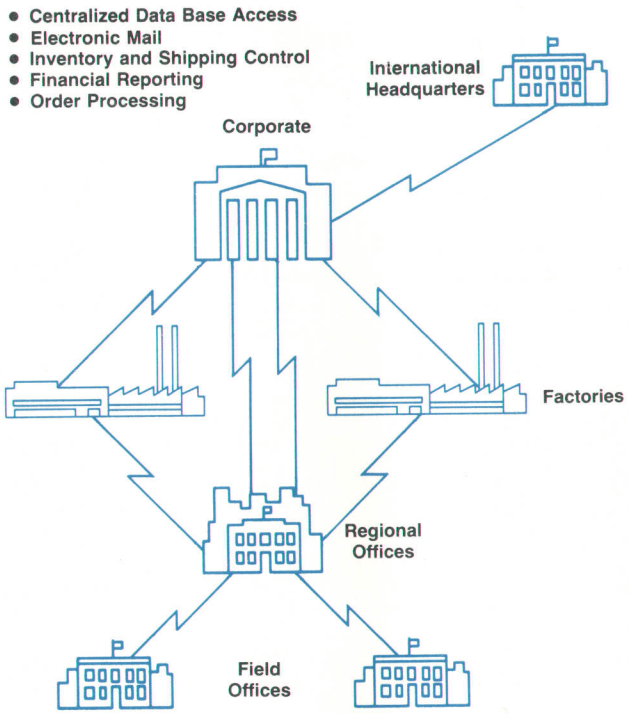


Fig. 6. *Typical applications of wide area networks.*

X.25 link-level procedures are responsible for the transfer of frames, without loss or duplication, over the physical link between the user (the data terminal equipment or DTE) and the network (the data circuit terminating equipment

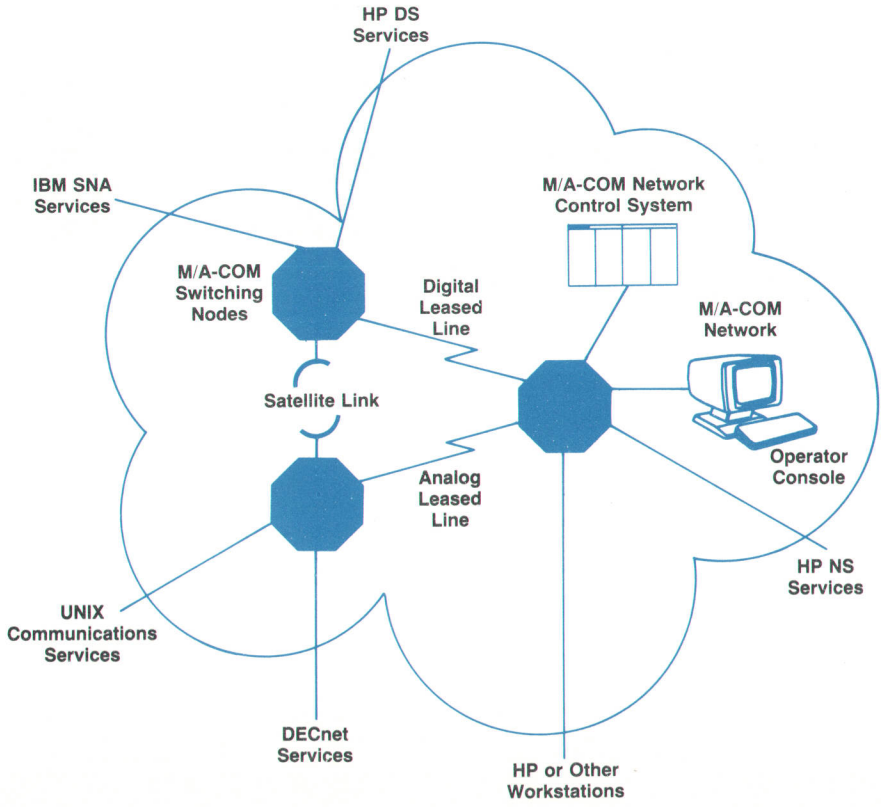


Fig. 5. *Private X.25 PSN offered by HP and M/A-COM.*

or DCE). LAP (link access procedure) was designed first, followed later by LAP-B (link access procedure balanced). LAP-B is now the preferred implementation and is available in all networks. Two link-level entities exchange frames, which contain a header, information, and a trailer.

X.25 packet level provides the virtual circuit service. Permanent virtual circuits can be compared to dedicated leased circuits. Switched virtual calls can be associated with dial-up circuits. X.25 procedures apply at the DTE/DCE interfaces (see Fig. 7). They are independent of any public data network internal interfaces.

Two packet-level entities exchange packets. Packets are carried in information frames, as shown in Fig. 8.

A packet contains a header (3 bytes) which is followed either by user data or by control fields. The packet header contains various pieces of protocol information:

- The logical channel number is used to differentiate between virtual circuits.
- The packet type ID identifies the specific packet (data packet, call packet, etc.)

A logical channel number (LC1) exists between the local DTE and the local DCE. Another logical channel number (LC2) exists between the remote DTE and the remote DCE. It is the responsibility of the network to establish the relation between LC1 and LC2. This is a very important point to understand. The X.25 procedures are defined at the interface between a DTE and a DCE. But the network itself performs actions like establishing relations between logical channel numbers, routing packets, and so on. Connecting two systems with a hardwired line is completely different from connecting two systems through a public data network. When system A and system B are connected with a hardwired line, what is generated by A is received by B without any changes. So, B must understand what A says. When A and B are connected through a public data network, if A sends some information to B, this information travels through the network and the network may add or remove some information before giving it to B. B must understand what is generated by the DCE, regardless of what A generated. In other words, systems only have to follow the interface rules. They do not need to know how the network works.

The X.25 packet level takes over after the physical and link levels are up and running. Packet level uses the service provided by the link level, which itself uses the service provided by the physical level. The main actions the packet level performs are :

- Call establishment
- Data transfer
- Call clearing
- Call reinitialization
- Control and recovery actions.

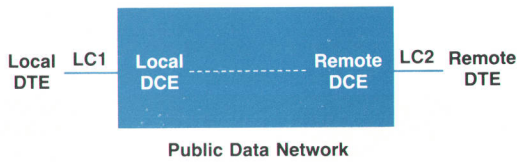


Fig. 7. *X.25 procedures apply at the DTE/DCE interfaces.*

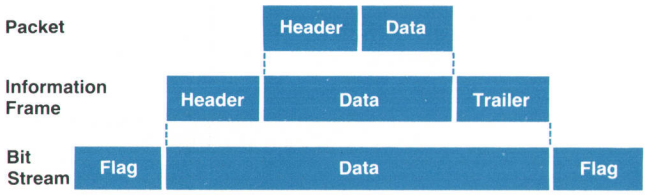


Fig. 8. *A packet contains a three-byte header followed either by user data or by control fields. Packets are carried in information frames.*

To perform these actions, different types of packets are available along with the procedures to follow. For example, to establish a virtual circuit, the X.25 recommendation tells which packet to send to the network, which packet to expect from the network, and how to react to the received packets. Let's say that DTE A wants to establish a virtual circuit to DTE B to exchange data. DTE A sends a call request packet to the local DCE (the call request packet contains B's address). The local DCE (A) routs the packet to another DCE (close to DTE B) and waits for an answer. This DCE accepts the packet, sends an incoming call packet to DTE B and waits for an answer. The incoming call packet contains A's address. DTE B decides to accept the call and sends back a call accepted packet to its local DCE, which forwards the packet to its counterpart DCE. This DCE sends DTE A a call connected packet. A virtual circuit is now established between A and B. They can both use it to exchange data. When A or B decides that it has no more data to transfer, it can clear the virtual circuit by using the appropriate packets.

Conclusion

The X.25 procedures define the packet that a DCE expects from a DTE to perform a precise action and the packet that the DCE must send back to the DTE. In other words, the X.25 procedures are written from the point of view of a DCE; they specify the actions and reactions of a DCE. Many issues are left for further study by Recommendation X.25, and some behaviors have not been standardized yet.

Since DTE actions are not specified at all, some choices have to be made to do an X.25 implementation on a particular system. An HP system connected to a public data network acts as a DTE, so it must have a DTE behavior, which is unspecified. Therefore, designers try to look at Recommendation X.25 through a mirror to determine how a DTE should behave. For this reason, different X.25 implementations are sometimes not able to communicate with each other. Presumably these issues will be resolved in the future.

DMI/3000: A Move Toward Integrated Communication

This product implements AT&T Information Systems' Digital Multiplexed Interface standard to provide communication between computers and terminals or other computers over private digital telephone networks.

by Nancy L. Navarro, Deepak V. Desai, and Timothy C. Shafer

USING EXISTING TELEPHONE WIRES to provide communication among host computers and terminals is by no means a new idea. Modems have made this possible for some time, and data is carried by phone lines in voice and data PBX (private branch exchange) systems.

However, using a standard, nonproprietary interface to provide computer communication over private digital telephone networks is a relatively new idea. Hewlett-Packard has participated in the development of a standard for one such interface, called the Digital Multiplexed Interface (DMI).[1] This standard, announced by AT&T Information Systems and supported by over 60 companies along with Hewlett-Packard (see box, page 47), promises to lead to more efficient and cost-effective PBX-based terminal-to-host communication.

Some History

In recent years, two factors have pointed to the need for a new means of connecting multiple terminals to hosts—first, the increasing complexity and cost of PBX-based terminal-to-host communication, and second, evolving international standards for data networking using existing telephone wiring.

PBXs were originally designed to provide economical telephone service on a single business premise and to consolidate connections from a single premise to the public or long-distance telephone network. As such, they were designed for carrying analog (voice) signals. Advances in the last decade or so have led to digital PBXs and CBXs (computerized branch exchanges), which carry digital, rather than analog signals, and which can, therefore, carry both voice and data over standard telephone wires.

The advantage realized through the use of PBX-based data switching is primarily that telephone lines, rather than specially installed dedicated links, can be used to carry information. When changes to the data wiring system must be made fairly often (to accommodate additional users or for other reasons), maintaining dedicated links can be more costly and more difficult than maintaining the same data connections using existing telephone wiring.

Despite this advantage, however, PBX-based data communications has not been without its drawbacks.

Although they can use existing wiring, PBX-based terminal-to-host and host-to-host connections also require the use of proprietary interfaces available from various PBX vendors. Different PBXs require different proprietary interfaces, which are provided by devices called data modules. Two data modules are needed for every terminal-to-host connection (see Fig. 1a). This system of proprietary interfaces, while initially satisfying a site's need for terminal-to-PBX-to-host communication, can in many cases become problematic as more and more terminals are added to a given system, creating a multiplicity of data modules and wiring needed to connect data modules with terminals and hosts.[2]

Also, despite the development and use of digital PBX systems for data communication, such systems are not op-

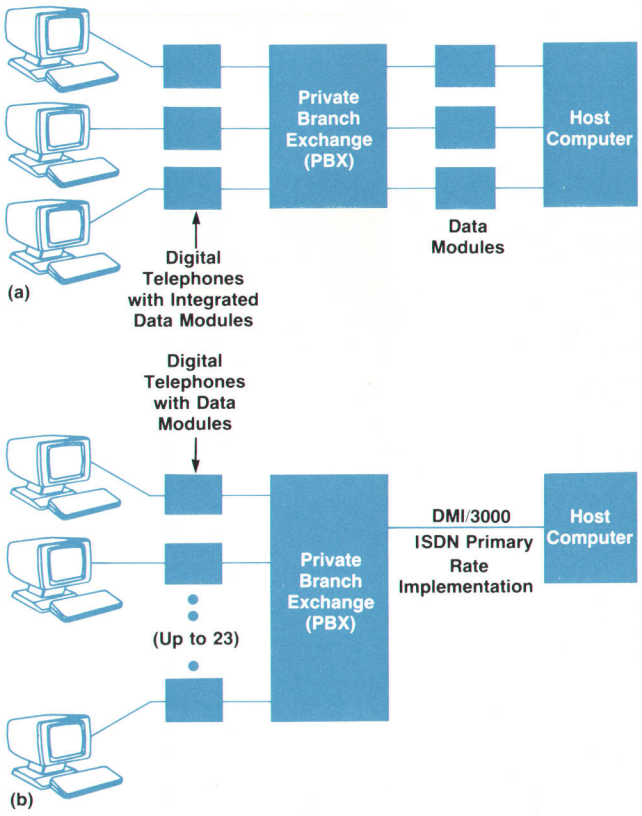


Fig. 1. *PBX-based host-to-terminal connections. (a) Typical interfacing. (b) DMI/3000 interfacing.*

Glossary of DMI Terms

Bipolar signal. Another name for alternate mark inversion signaling, the type of signal used in T1 transmission. Pulses are transmitted with a 50% duty cycle and with the polarity of alternate 1 bits inverted.

CCITT. International Telegraph and Telephone Consultative Committee, an international body formed under the auspices of the United Nations to develop worldwide standards for data communications protocols and methods.

CEPT. Conference of European Postal and Telecommunications Administrations, the European body that recommends and approves standards for European telecommunications methods.

D4. The framing technique currently used on 1.544-Mbit/s digital transmission facilities (used in North America and Japan). D4 frames consist of 192 bits plus one delimiter known as the frame bit for a total of 193 bits/frame.

Data access module. The interface between digital PBXs and host computers and between PBXs and terminals. Currently, data access modules usually provide a standard interface (such as RS-232-C) for attachment to a computer, and a proprietary protocol for passing information through the PBX.

DS1. Digital signal level 1, the term widely used in North America to describe the standard 1.544-Mbit/s multiplex signal used for digital transmission across telephone wire.

Dual-rail data. Data transmitted in such a way that alternate bits are present on two separate signal lines. In DMI/3000, the DS1 card converts bipolar data received from the T1 carrier to dual-rail data.

F_e. Extended framing format, which is expected to replace D4 framing for DS1 transmission in North America. F_e framing incorporates error measurement and a data link in addition to providing framing.

Framing. The technique used in digital transmission to divide data into segments to make synchronization possible.

HDLC. High-level data link control, a protocol designed to prevent transmission errors from being interpreted as error-free by the receiver. LAP-D, a variation of HDLC that incorporates HDLC and additional features, has been incorporated as part of ISDN recommendations for signaling.

ISDN. Integrated Services Digital Network, a currently evolving worldwide standard for transmission of digital data.

LAP-D. Link access procedure, D channel. A data link protocol that is a modification of the HDLC protocol. LAP-D includes HDLC, and adds functions such as error correction. LAP-D has been incorporated in ISDN recommendations.

OSI Reference Model. Open Systems Interconnection Reference Model, a hierarchical description of data communications network functions established by ISO, the International Standards Organization.

T1. The name for the digital transmission carrier used in North America. T1 carrier can accommodate 24 64-kbit/s channels, at a total rate of 1.544 Mbits/s.

Time division multiplexing (TDM). Allowing several separate signal sources to use a single channel for separate short periods of time to share the capacity of the channel among them.

timized for data transmission, since they were originally designed as voice communication systems.

At the same time that the drawbacks of past PBX-based networking were becoming more apparent, the advent of digital telecommunications spurred the International Telegraph and Telephone Consultative Committee (CCITT) to develop standards for digital telecommunications to ensure that worldwide telecommunications, such as international calling, would still be possible as different countries changed their previously analog networks to new, digital technology. A direct result of the CCITT's efforts is a set of standards called ISDN, for Integrated Services Digital Network. ISDN proposes standards for a broad set of digital communications requirements, including transmission and switching methods, equipment interfaces, and service functions. Ultimately, it is thought that when implemented around the world, ISDN will enable digital communication between virtually any systems that can transmit digital information—whether it be voice, computer data, or even video information.

The ISDN Standard

The Integrated Services Digital Network standards, parts of which have been recently finalized and parts of which are still under discussion, will eliminate the need to convert digital information to analog form before it can move over phone lines—a function that has been provided in the past by modems.

The CCITT has defined two types of interfaces to an ISDN network. One type, called the basic access interface, consists of three channels: two ISDN B channels, used for data, and one ISDN D channel, used for signaling. For this reason the basic access interface is also called the 2B+D standard.

The second type of access, called the primary rate interface, provides either 23 or 30 data channels, depending upon where it is implemented, and one signaling (D) channel. It is this primary rate interface that provides the basis

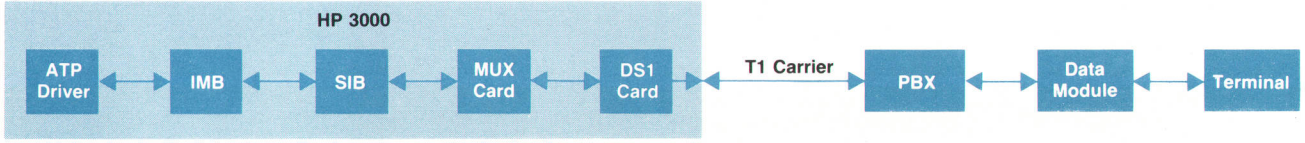


Fig. 2. *DMI and HP 3000 components. (ATP = Advanced Terminal Processor. IMB = Intermodule Bus. SIB = System Interface Board.)*

for the DMI.

The ISDN primary rate interface is designed to take advantage of existing telephone data transmission rates. These differ slightly in North America and Japan, on the one hand, and in Europe, on the other. In North America and Japan, the transmission rate of the standard carrier, known as T1 carrier, is 1.544 Mbits/s; in Europe, this rate is 2.048 Mbits/s, to correspond to the transmission rate of the standard European carrier known as CEPT carrier.

This transmission rate difference is the reason that ISDN can provide a different number of data channels in Europe than it can in North America and Japan. In Europe, the primary rate interface provides 30 data, or B channels, and one signaling, or D channel. In North America and Japan the primary rate interface provides 23 B channels and one D channel.

The DMI Specification

DMI addresses the problem of increased costs of current PBX-based switching systems by eliminating the need for data modules and corresponding wiring between a PBX and a host for each terminal attached to the PBX (see Fig. 1b). Instead, the same function will be served by three twisted-pair telephone wires—one for transmitting data, one for receiving data, and one for diagnostic functions. Hewlett-Packard estimates that its first DMI product will enable customers to reduce the cost of adding terminals to a system by about 25%.

As a result of adherence to the ISDN primary rate interface, DMI implementations will operate over existing line, whether it is the 1.544 Mbit/s T1 carrier currently used in North America and Japan or the 2.048 Mbit/s CEPT carrier used in Europe. Hewlett-Packard's first implementation of DMI, DMI/3000, connects computer equipment to T1 carrier.

Major elements of DMI that reflect the emerging ISDN standards are its use of common-channel signaling, in which one channel is used for all communications signals, and clear (fully available to the user) 64-kbit/s transmission.

For DMI's 1.544-Mbit/s (T1) implementation, signaling information from the 23 data channels (ISDN B channels) is multiplexed into the 24th channel (ISDN D channel) that is used solely for signaling. Putting signaling bits on one channel is an important improvement over the traditional "robbed bit" technique (which originated with analog voice transmission) for including signaling information.[3] In robbed bit signaling, bits used for signaling are periodically written over user information being carried by the communications channel. This method is fine for voice communication, because people conducting a voice call cannot detect that the voice information is being corrupted by the signaling bits. However, computers cannot tolerate the periodic corruption of data that robbed bit signaling imposes; complicated schemes must be implemented to overcome the resulting data loss.

The DMI specification includes two types of signaling: bit-oriented, which will allow early implementation and is based on currently used tie-trunk signaling techniques, and message-oriented, which is consistent with ISDN recommendations.

In bit-oriented signaling for DMI, the first two bits of each byte received through the 24th channel indicate the status (on hook or off hook) of one of the data channels. This signaling method uses the same signals as does traditional robbed bit signaling, but places the signals on the 24th channel rather than over user information carried by a data channel.

Message-oriented signaling, on the other hand, places individual bytes, each conveying the status of a single data channel, within an HDLC (high-level data link control) frame. Message-oriented signaling is part of the ISDN recommendation, and when implemented, will enable the signaling channel to carry more information about each con-

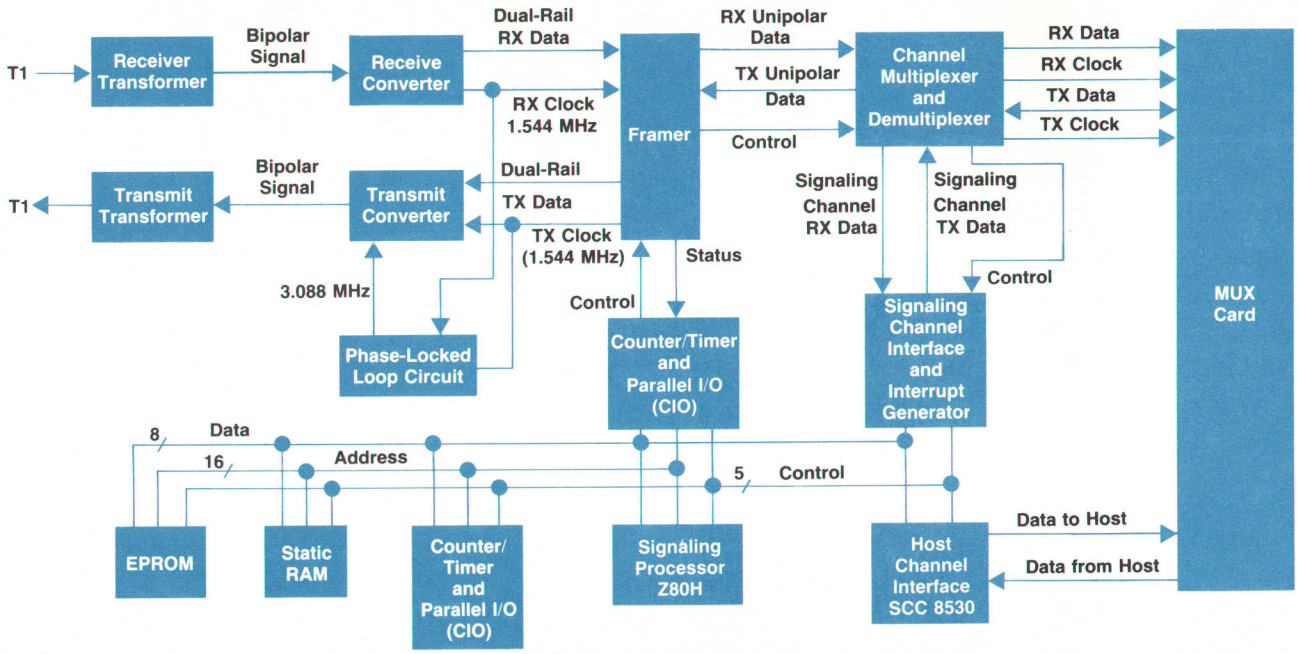


Fig. 3. *DS1 card structure.*

nection. While bit-oriented signals can only support pulse dialing and indicate whether a channel is ready to receive incoming data (off hook) or not (on hook), message-oriented signals will also contain information about call setup, determination of call status, address redirection, and host status. These added capabilities of message-oriented signaling are expected to improve network resource management, security, and diagnostics.

DMI Modes

AT&T Information Systems' DMI specification includes four different data channel protocols, called modes, which support data transport at different standard rates. The specification allows the choice of data mode to to be made on a per-channel, per-connection basis.

Mode 0 supports clear, 64-kbit/s, full-duplex synchronous transmission. Mode 1 supports 56-kbit/s, full-duplex, synchronous transmission that is compatible with dataphone digital service (DDS), the public data switching facility currently used in North America.

Mode 2, which is supported by the first release of DMI/3000, provides synchronous or asynchronous transmission of full-duplex or half-duplex data using RS-232-C terminal connections. Using Mode 2, data can be transmitted at a variety of rates up to 19.2 kbits/s, a rate that ensures that Mode 2 DMI implementations can be used with existing data terminal interfaces. Mode 2 uses the HDLC protocol to detect transmission errors at the data link level.

The Mode 3 protocol provides reliable (error-protected) data transmission in either circuit-switched or virtual circuit-switched environments in which statistical multiplexing allows a single physical channel to carry more than one virtual channel. To transmit data, Mode 3 uses the LAP-D protocol, a protocol that incorporates HDLC (used for Mode 2) but adds other features. LAP-D provides error correction as well as the error detection provided by HDLC. The use of LAP-D also promotes compatibility between PBXs and packet switching networks.

DMI for the HP 3000

By obtaining a license to use the Digital Multiplexed Interface specification, Hewlett-Packard has been able to design a product that implements DMI for HP 3000 computer systems.

This product, called DMI/3000, consists of two printed circuit boards, or cards, that fit into the backplane of the HP 3000, and accompanying software and firmware. The cards contain components that work with an existing HP 3000 board, the system interface board (SIB), to provide communication between the HP 3000 and up to 23 terminals or personal computers. This number of connections allows the first offering of DMI/3000 to work with North American/Japanese T1 transmission facilities.

One card, named the DS1 card after the widely used term for the kind of digital telephone signal commonly used in North America, provides the interface to the PBX and multiplexes and demultiplexes the 24 channels.

The second card, named the MUX card, processes the data sent through each of the 23 data channels, and then sends it to the HP 3000 host through the SIB. The cards are connected by two 64-pin edge connectors.

The software components of DMI/3000 consist of a driver that controls communication between the HP 3000 and the two cards, and a signaling channel monitor that controls and monitors the switched connections between the HP 3000 and the PBX. Firmware that is downloaded upon

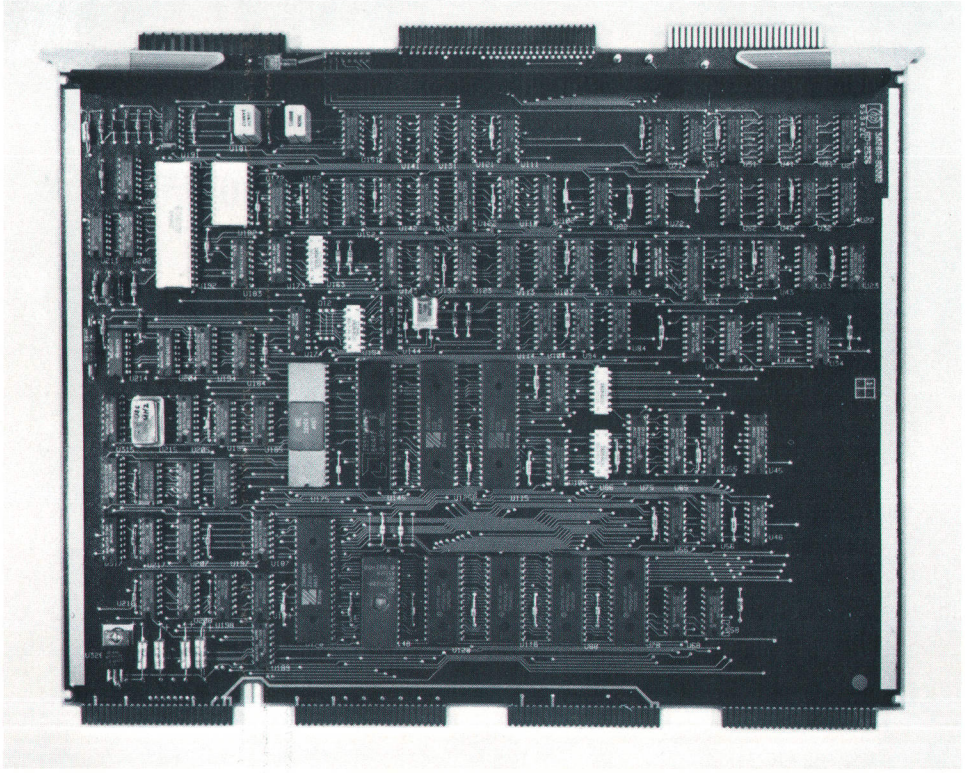


Fig. 4. *DS1 card.*

system startup controls signal processing and implements the link-level protocol used on a particular data channel.

Fig. 2 on page 42 shows the relationship among various DMI/3000 and HP 3000 components.

The DS1 Card

The DS1 card has three main functions. First, it multiplexes and demultiplexes the 24 data channels, using time-division multiplexing. Second, it provides a physical-layer interface between the T1 line and the DMI link. Finally, the DS1 card processes the signaling (24th) channel at the link level.

Fig. 3 shows the structure of the DS1 card. Physical details can be seen in the photograph of the card, Fig. 4.

The physical interface is provided mainly by three VLSI chips provided by AT&T Information Systems: the receive converter chip, the framer chip, and the transmit converter chip.

The receive converter obtains bipolar data from the receive transformer, which receives a signal from the T1 line linked to the PBX. The receive converter then extracts clock information from the incoming data and converts it to dual-rail data before sending it to the framer chip. In addition, the receive converter includes an equalizer that provides a fixed pulse amplitude to timing and data recovery circuits.

The framer chip decodes framing information from received data and inserts framing information into transmitted data. Framing is a method of dividing data into fixed segments to permit error detection. DMI supports two framing formats. One, called D4, has been used for T1 transmission for many years and is a current standard. The other format, called F_e or extended framing, is a newer format that the CCITT recommended in 1984. F_e framing is expected eventually to replace D4 framing in North America.

The transmit converter chip takes dual-rail data from the framer, converts it to bipolar form, equalizes the T1 output signal, and sends the data to the transmit transformer chip, which then passes it to the T1 line.

After incoming data is passed through the framer chip, and before outgoing data is sent to the framer, the data passes through a channel multiplexer/demultiplexer circuit. This circuit multiplexes transmissions from the 23 data channels into one channel for transmission across T1 lines, and demultiplexes received data into 23 individual channels. In addition, the multiplexer/demultiplexer circuit uses a buffering scheme to slow the data burst rate from 1.544 Mbits/s to 386 kbits/s to comply with a limitation of the USART (universal synchronous/asynchronous receiver/transmitter) chips used on the MUX card. Once incoming data is demultiplexed and its rate is slowed, it is passed to the MUX card.

A final function of the DS1 card is to provide common-channel signaling. To accomplish this, the card uses a Zilog Z80H microprocessor as a signal processor. Additional components needed for signaling are a Zilog serial communications controller (SCC 8530 in Fig. 3), two Zilog counter/timer and parallel I/O (CIO) chips, a serial-to-parallel converter, a parallel-to-serial converter, a 16K-byte EPROM, and an 8K-byte static RAM.

In accordance with the DMI specification, DMI/3000 is designed to accommodate both bit-oriented and message-oriented signaling. The choice of signaling method is made when code that specifies which method is to be used is downloaded from the HP 3000 host to the DS1 card at system startup. At present, bit-oriented signaling is avail-

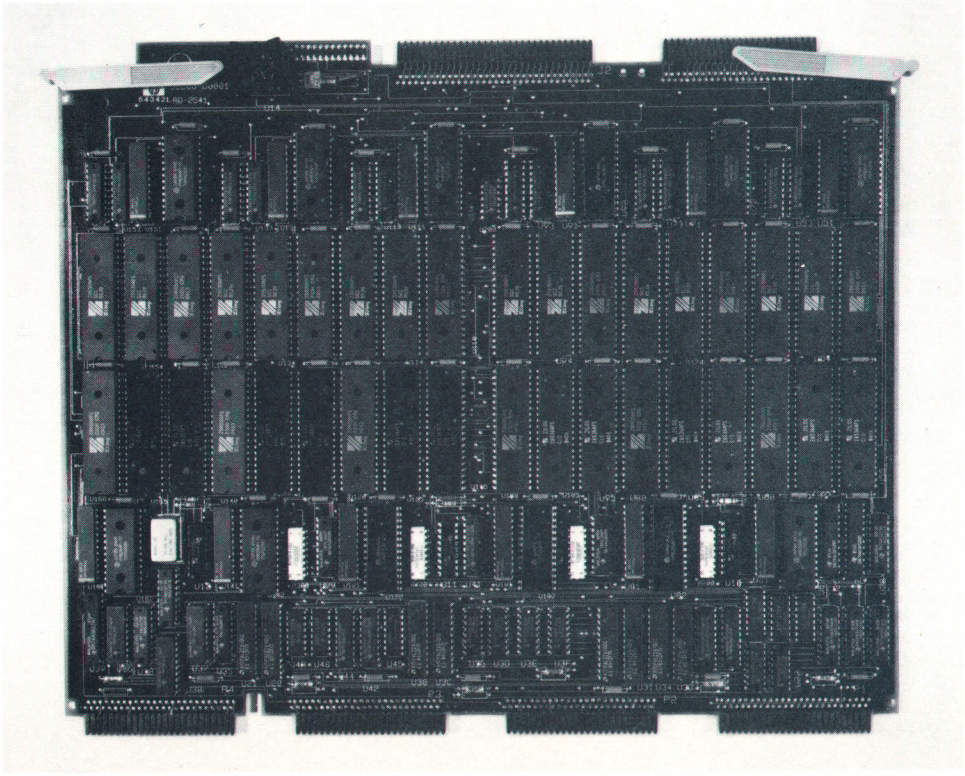


Fig. 5. *MUX card.*

able; in the future, the host will be able to request message-oriented signaling instead of the bit-oriented default.

The MUX Card

The MUX card is the other major hardware component of DMI/3000. Shown in Fig. 5, the MUX card implements OSI data-link-layer protocols on the data passing through each data channel. It performs terminal character processing, such as special character recognition and handshaking. It meters outbound data and adapts the baud rate to 64 kbits/s, and it furnishes an interface to the HP 3000 SIB (system interface board).

The OSI data-link-layer protocol, terminal character processing, rate adaptation, and SIB interfacing are all performed by a set of 12 dual port controllers on the MUX card. Each dual port controller (DPC) controls the data flow for two of the 24 available channels. Fig. 6 shows the circuitry of each DPC. The components include a Z80H microprocessor, a serial communications controller or SCC (this is the USART), a counter/timer parallel I/O (CIO) chip, a 4K-byte PROM, a 32K-byte RAM, and a PAL (programmable array logic).

The 32K-byte RAM included with each DPC performs a significant function linked to the flexibility and upgradability of DMI/3000. When an HP 3000 with DMI/3000 installed is powered up, the HP 3000 downloads code to each DPC RAM. This code implements the data channel protocol to be used on the channels associated with the particular DPC. Code to perform this downloading as well as a self-test resides in each DPC's PROM.

The first release of DMI/3000 implements DMI Data Mode 2. The Mode 2 protocol was chosen because it is the most suitable for supporting terminals that attach to a PBX via data modules using an RS-232-C interface (the type of connector in use today).

The use of downloaded code to implement the data channel protocol is a significant DMI/3000 design advantage. Because the data modes will be implemented through code distributed as part of HP 3000 operating system software, an upgrade from one mode to another—for example, from Mode 2 to Mode 3—will be relatively simple. A monitor program will enable those installing the system to choose the desired protocol for each set of data channels.

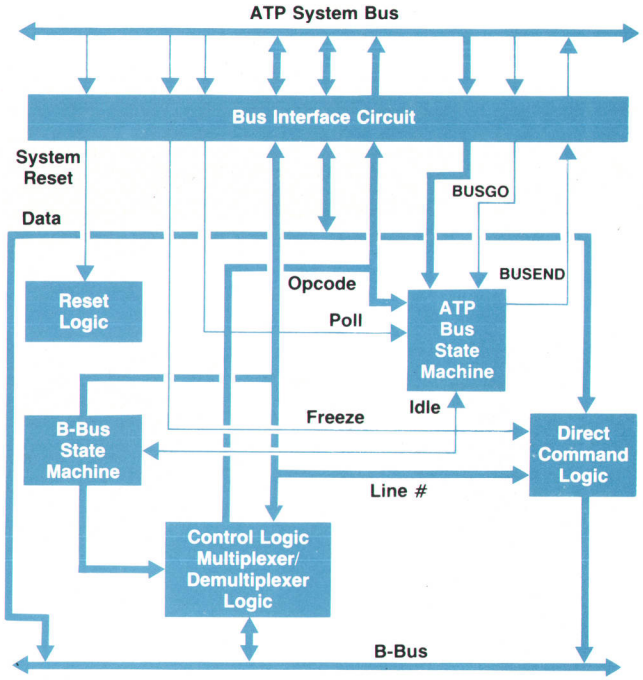


Fig. 7. *MUX bus adapter structure.*

The interface between the HP 3000 SIB and the 12 DPCs is controlled by a group of MUX card components known collectively as the MUX bus adapter. The MUX bus adapter emulates the functions of two HP 3000 boards known as asynchronous interface boards. As Fig. 7 shows, the bus interface circuit of the MUX bus adapter connects the HP 3000 ATP (Advanced Terminal Processor) system bus to the internal bus of the MUX card, the B-bus. Arbitration and control of the B-bus are achieved by a B-bus state machine composed of three 256 × 8-bit bipolar PROMs, an eight-bit counter, one PAL, and two eight-bit latches.

An additional PAL and three more latches are used to create a bus state machine, which controls the information flow between the bus interface circuit and the ATP system bus.

Control logic and multiplexer/demultiplexer logic im-

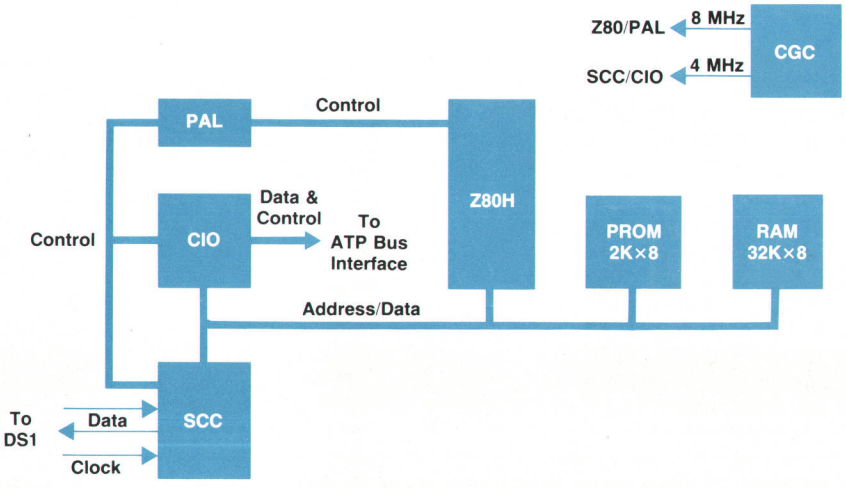


Fig. 6. *Dual port controller structure.*

Companies Supporting the DMI Standard

Advanced Computer Communications
Advanced Micro Devices Inc.
Amdahl Corporation
American Telephone and Telegraph Inc.
BBN Communications Corporation
Bose Associates Inc.
Burroughs Corporation
California Microwave Inc.
CASE Communications
Compaq Telecommmunications Corporation
Control Data Corporation
CXC Corporation
Dallas Semiconductor Corporation
Data General Corporation
Datapoint Corporation
Davox Inc.
Digital Sound Corporation
Distributed Solutions Inc.
DMW Group Inc.
Edward K. Bower Inc.
Four C Enterprises Inc.
Gandalf Data Limited
Gold Star Tele-Electric Co. Ltd.
Harris Corporation
Hewlett-Packard Company Inc.
Hitachi America Ltd.
Honeywell Inc.
Idacom Electronics Inc.
Infotron Systems Inc.
InteCom Inc.
Intel Corporation
ITT Corporation
Jeumont-Schneider
Jistel Inc.
LP Com Inc.
Micom Interlan
Mitek Corporation
Mitel Corporation
NCR Corporation
NEC America Inc.
Nixdorf Computer
Nokia Corporation Electronics
Prime Computer Inc.
Radio/Switch Inc.
Raytel Systems Corporation
Rockwell International Corporation
Scitec Corporation
Seimens Communications Systems Inc.
Societe Anonyme de Telecommunications
Sonecor Systems Inc.
Soron Company
Spectrum Digital Corporation
Speech Systems Inc.
Syntrex Inc.
Tadiran Electronic Industries
Tandem Computer Inc.
Tekelec Inc.
Telettra SpA.
Thompson-CSF Telephone
Timeplex Inc.
Tricom Konsortium
Ungermann-Bass Inc.
Wang Laboratories Inc.
Western Digital Corporation
Ztel Inc.

plemented with a combination of PALs and demultiplexer chips interfaces the bus state machines to the DPCs and the buses.

Finally, direct commands to individual data channels that originate from the DMI driver (see next section) are issued through direct command control logic.

Software and Firmware

A combination of software and firmware gives DMI/3000 the flexibility needed to change as ISDN moves toward complete definition and as current transmission methods are gradually replaced by newer ones.

DMI/3000 control code is of two types: firmware already burned into the DS1 card EPROM and MUX card PROM, and software downloaded from the HP 3000. The MUX PROM firmware downloads the data channel protocol for each DPC, and the DS1 card EPROM firmware downloads signaling software. In addition, both the PROM and EPROM contain self-tests and the EPROM contains a debug monitor.

The signaling software downloaded to the DS1 card manages call setup and teardown. For the first release of DMI/3000, the signaling software implements procedures that process bit-oriented signals. For future implementations that are expected to be available when ISDN signaling recommendations are in place, the signaling software will implement message-oriented procedures. In addition to making DMI/3000 easily upgradable to newly implemented standards, another advantage to downloaded software is that other software changes can also be readily made.

Two pieces of software continually residing on the HP 3000 are necessary for DMI/3000 to function: the ATP driver that provides I/O between HP 3000 applications and the DMI cards, and the signaling channel monitor. The driver code resembles HP 3000 terminal drivers, but accommodates the downloading of code to the DMI cards. The signaling channel monitor software controls and monitors the switched connections between the PBX and the HP 3000.

Diagnostics

In addition to the self-tests provided by the DS1 and MUX cards, more extensive DMI/3000 testing can be accomplished by diagnostic software provided to HP field representatives.

The interactive diagnostic tests the SIB, the DS1 and MUX cards, and individual channels. In some cases, the

diagnostic invokes the DS1 and MUX firmware self-tests. In addition, the diagnostic software can abort DMI/3000 jobs or I/O, reset individual ports and associated HP 3000 operating system tables, and display (or write to disc) failure information.

Part of the diagnostics for the DS1 card consist of 7 LEDs that indicate which self-test is active when a self-test is executing and at other times indicate various detected error conditions.

Future Directions

Expected future upgrades to DMI/3000 include the availability of message-oriented common-channel signaling and the DMI Mode 3 protocol. The increased functions of message-oriented signaling over bit-oriented methods should improve overall network operation. In a related arena, Hewlett-Packard is developing network management capabilities that will be controlled via message-oriented signaling HDLC frames.

Benefits associated with the use of Mode 3 will be greater transmission speed—64 kbits/s rather than Mode 2's 19.2 kbits/s—and the ability to use a packet switch on either the terminal side of the PBX or within the PBX itself to concentrate traffic from multiple terminals to a single ISDN B channel. In addition, the use of the LAP-D protocol will enable DMI systems to receive data from X.25 packet switched networks.

New VLSI circuit designs, some already announced, will enable replacement of some of the off-the-shelf integrated circuits used in the current version of DMI/3000 with fewer circuits that implement specific DMI functions. For example, it is expected that the DPC USARTs and the multiplexer/demultiplexer circuit will be replaced by a single VLSI circuit. This will substantially decrease the board space needed for DMI implementation.

It is also expected that replacements will soon be forthcoming for the data modules that must currently be used to connect terminals to PBXs. Some integrated circuit manufacturers have already announced integrated circuits that will support the ISDN basic access interface. This could lead the way for the introduction of inexpensive terminal-to-PBX interfaces to substitute for the data modules, which are still relatively expensive today.

Cooperation among DMI Developers

To ensure that DMI product development, DMI specification additions, and ISDN recommendations all evolve consistently, AT&T Information Systems has formed a DMI users' group composed of companies interested in developing DMI-based products. Involvement in this group has enabled HP to design DMI/3000 for maximum compatibility with other vendors' planned products. For example, HP has been able to design DMI/3000 to work with AT&T Information Systems System 75 and System 85 PBXs. Users' group members have also agreed on concurrent scheduling for interface upgrades and cross-vendor certification mechanisms. The continued existence of this group and its spirit of cooperation will ensure that the HP's DMI products, as well as those of other manufacturers, will remain consistent with current standards.

Acknowledgments

Thanks are due the many DMI/3000 product team members: Phil Taylor, research and development section manager, Dave Langley and Bob Gudz, the MUX project managers at various times, Jon Hewitt, the ATP driver project manager, Terry Gong, Liz Poteet, and Greg Snider, who worked on firmware design, Bharat Singh, who worked on the overall hardware design of the DS1 card, Bill Hayes, for the DS1 PAL and multiplexer/demultiplexer design, Bob Bortolotto and Denton Anderson, for MUX card hardware design, Joel Dunning, for the design of the signaling channel monitor, John Nivinski, for diagnostics design, Terry Perkinson, for modifications to the ATP driver, Tim Carlson, for technical marketing assistance, and John McHugh, for production engineering assistance. Also, special thanks to Felicia Choy and the many others who aided in DMI/3000 development at HP's Roseville Networks Division, Information Networks Division, Office Systems Division, and Computer Systems Division, whose names are too numerous to mention here.

References

1. *Digital Multiplexed Interface Technical Specification, Issue 3.0*, AT&T Information Systems, 1984.
2. T.G. Shafer, "Packing Data for the ISDN Migration, and Avoiding Costly Baggage," *Data Communications*, November 1984.
3. A.R. Severson, "AT&T's Proposed PBX-to-Computer Interface Standard," *Data Communications*, April 1984.

Hewlett-Packard Company, 3200 Hillview Avenue, Palo Alto, California 94304

HEWLETT-PACKARD JOURNAL
October 1986 Volume 37 • Number 10

Technical Information from the Laboratories of Hewlett-Packard Company
Hewlett-Packard Company, 3200 Hillview Avenue
Palo Alto, California 94304 U.S.A.
Hewlett-Packard Central Mailing Department
P.O. Box 529, Startbaan 16
1180 AM Amstelveen, The Netherlands
Yokogawa-Hewlett-Packard Ltd., Suginami-Ku Tokyo 168 Japan
Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive, Mississauga, Ontario L4V 1M8 Canada

Bulk Rate
U.S. Postage
Paid
Hewlett-Packard
Company

CHANGE OF ADDRESS: To subscribe, change your address, or delete your name from our mailing list, send your request to Hewlett-Packard Journal, 3200 Hillview Avenue, Palo Alto, CA 94304 U.S.A. Include your old address label, if any. Allow 60 days.

5953-8553