OpenNET: A network architecture for connecting different operating systems

by LEONARD H. MAGNUSON and MICHAEL SZABADOS
Intel Corporation
Santa Clara, California

ABSTRACT

A major goal of local area networks is the connection of different systems. This paper discusses Intel's OpenNET, which provides transparent interoperation of different operating systems on the same network, based on industry standard protocols. OpenNET encompasses all seven layers of the OSI network model. The focus of this paper is on the upper-layer software, which implements the network file access capability.

A major goal of local area networks is connecting different systems. Intel's OpenNET family of products is the first to provide transparent interoperation of different operating systems on the same network, based on industry standard protocols. OpenNET is compatible with IBM's PC Networking Software and Microsoft Networks (MS-NET) and also supports Xenix and iRMX 86. OpenNET's implementation of the OSI model offers a high degree of design flexibility for system integrators and system users.

The OpenNET architecture allows transparent, heterogeneous interoperation between PC-DOS, MS-DOS, Xenix, iRMX 86, and other operating systems from vendors implementing the protocols, making possible a wide range of diverse local-area network (LAN) applications in manufacturing, transaction processing, and office automation environments. Unlike many available LAN systems, OpenNET makes use of existing file access systems, so the user's existing applications software can be used without change on the network.

The OpenNET product family consists of products that encompass all seven layers of the OSI network model. Based on Intel's advanced VLSI LAN technology, the key elements in the system are board and system-level hardware for Ethernet, transport layer software, and software that supports OSI Layers 5 through 7. Specific products in the OpenNET family include SMX 552 Multibus Transport Engine; the 186/51 Multibus COMMputer Board; iNA 960 transport layer software; and Intel's RMX Networking Software and the Xenix Networking Software, developed in a partnership between Intel and Microsoft.

The lower four layers in the PC environment are supported by the Ungerman Bass Personal NIU running a preconfigured version of the iNA 960 transport software. The upper layers for DOS operating systems are implemented by MS-NET. The products provide a set of flexible network building blocks that can be used to link different microcomputers, work stations, PCs, and peripherals. (See Figure 1.)

The focus of this paper is on the upper-layer software implementing the novel heterogeneous network file access capability on top of an ISO 8073 compatible transport capability.

The upper-layer protocols are implemented in software initially configured to run on the host processor. Layer 5 provides a logical, name-based connection capability; so systems and users can be accessed at all times, regardless of their physical location on the network. Layers 6 and 7 provide network file service. These two capabilities combine to offer transparent file access to the user from any remote or local node.

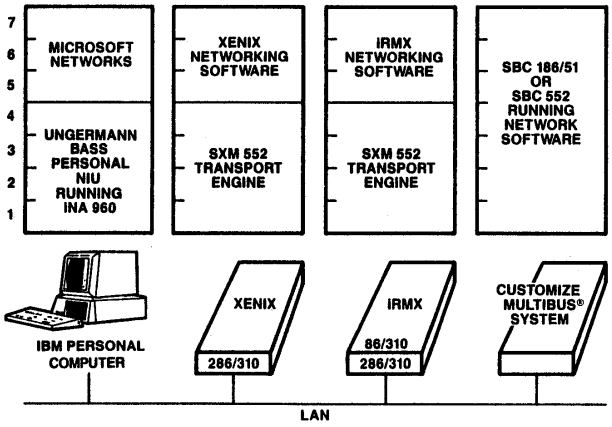


Figure 1—How Ethernet/OpenNET connects different systems on the same network

SESSION LAYER

Layer 5, the session layer, will translate the symbolic name supplied by the user into a physical address and will then use the transport layer to establish a virtual circuit between the requestor and the target system. Names are assigned by the user so that nodes can be easily accessed with familiar commands.

PRESENTATION AND APPLICATION LAYERS

The network file access (NFA) protocol lets users read, write, open, close, and otherwise manipulate files from a remote location and thus implements true transparent remote file access, as opposed to the disk-sharing and file transfer methods used by other networks. This transparency permits users to continue working with existing applications software and lets them make the transition from a stand-alone to a network environment without having to learn a new interface. On the network, each node is either a server or a consumer or both. The consumer at a given node transmits the user's commands across the LAN to the server at the access node, which executes them. Unlike file transfer (a separate capability that is not transparent to the network user), network file access generally eliminates the need to move entire files before working with the data. Network file access thus results in less movement of data along the network and higher overall network performance.

- THE FILE ACCESS PROTOCOLS PROVIDE FOR A CONSISTENT VIEW OF FILES THROUGHOUT THE NET
  - — THE LOCAL DIRECTORIES ARE EXTENDED ACROSS THE NET

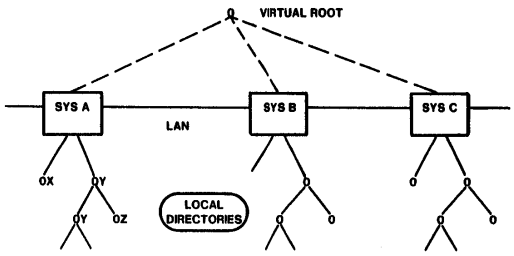


Figure 2—The network hierarchical file system

File access control is another feature of NFA. Different operating systems naturally have different protection models, and NFA honors the protection schemes specified by every type of server for its files. NFA resolves the differences between server protection models, however; so the consumer system will always see these access permissions expressed in its own operating system's format.

FILE SYSTEM

The protocols support a network hierarchical file structure, as illustrated in Figure 2. The hierarchy has been extended "upwards" through the addition of a network root (indicated by a double slash, either // or \\ to the file name space. The file names are prefixed with the name of the system on which they reside within the network. This scheme—a logical extension of the file systems used by DOS, Xenix, and iRMX 86—ensures that file names are both unique and consistent throughout the network.

User applications access remote files through the same system call interface used to access local devices. Since this interface is not modified by the network, applications transparency for remote file access is ensured. Because most services are based on the file system, many of them will work across the network with little or no change. For example, Xenix mail can exchange mail with Xenix users anywhere on the network, and the "at" command can be used to initiate batch jobs on other Xenix network systems.

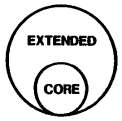
EXPANDABLE PROTOCOLS

The OpenNET file access protocols were jointly developed with Microsoft and are compatible with the protocols used by IBM in PC-NET. The application layer protocol sets are expandable to provide transparent file access among transparent interoperation between MS DOS and PC DOS systems. The extended protocols allow Xenix and iRMX-based systems to interoperate transparently. As illustrated in Figure 3, the extended protocols are a superset of the core protocols.

IMPLEMENTATIONS

The network service software consists of separate server and consumer modules. The iRMX, Xenix, and PC DOS implementations permit every node to be a server, a consumer, or both. MS DOS nodes can be either a consumer or a server.

- THE PROTOCOLS ARE EXPANDABLE



  - — THE CORE PROTOCOLS SUPPORT DOS-TO-DOS INTEROPERATION
  - — THE EXTENDED PROTOCOLS SUPPORT MORE SOPHISTICATED FILE SYSTEMS (E.G. XENIX, RMX)

Figure 3—Protocol set and heterogeneous interoperation

Implementation Under PC DOS

A PC DOS or MS DOS consumer can access iRMX, DOS, or Xenix remote files as well as remote printers connected to DOS or Xenix servers. A DOS user who wants to access a remote file or printer first connects to the file server with a NET USE command. This command equates the remote directory with a local drive identifier. Any subsequent reference to that drive identifier will automatically be routed to the redirector. A command to equate a drive identifier with a remote directory would take this form:

NET USE c: //System A/Directory Name

Once this connection is established, the user can access the remote directory by simply specifying drive c.

Implementation Under Xenix

A Xenix system may perform concurrently as both a network file consumer and a network file server. The Xenix consumer implementation offers complete file access transparency when accessing Xenix or iRMX 86 servers. Existing Xenix (or Unix) applications can generally use the network file system without change. The consumer software is embedded in the Xenix kernel and automatically directs file accesses to the appropriate local device or remote system.

The Xenix file server provides full, transparent access to PC DOS, MS DOS, iRMX 86, and Xenix systems. The server software consists of a number of kernel processes. A process is created for each remote process that makes network accesses to the server. These processes are transitory and are terminated when the remote process terminates.

Implementation Under iRMX

Like Xenix, an iRMX 86 may be configured as a file server, a file consumer, or both. When configured as an OpenNET consumer, an iRMX system may transparently access files residing on both iRMX and Xenix servers. As noted before, this implies that many iRMX applications may operate on remote files without change. If configured as an OpenNET

server, an iRMX system can service PC DOS, MS DOS, Xenix, and iRMX 86 consumers.

Implementation Under Other Operating Systems

OEMs and system integrators using other operating systems and system backplanes will need customized implementations of the hardware and protocols in order to connect onto an Ethernet OpenNET LAN. The task can be considerably simplified by using the original equipment manufacturer (OEM) building blocks available from Intel.

To implement the lower four ISO layers for non-Multibus environments the OEM can develop a transport engine based on Intel's 82586, 82501,and 80186 VLSI components and the iNA 960 software.

The upper-layer software can be developed from scratch on the basis of the file access protocols, which will be published by Microsoft. In most cases, however, OEMs will save development time and resources by acquiring the source code (written in PLM for iRMX and in C for Xenix) and porting it over to their operating systems.