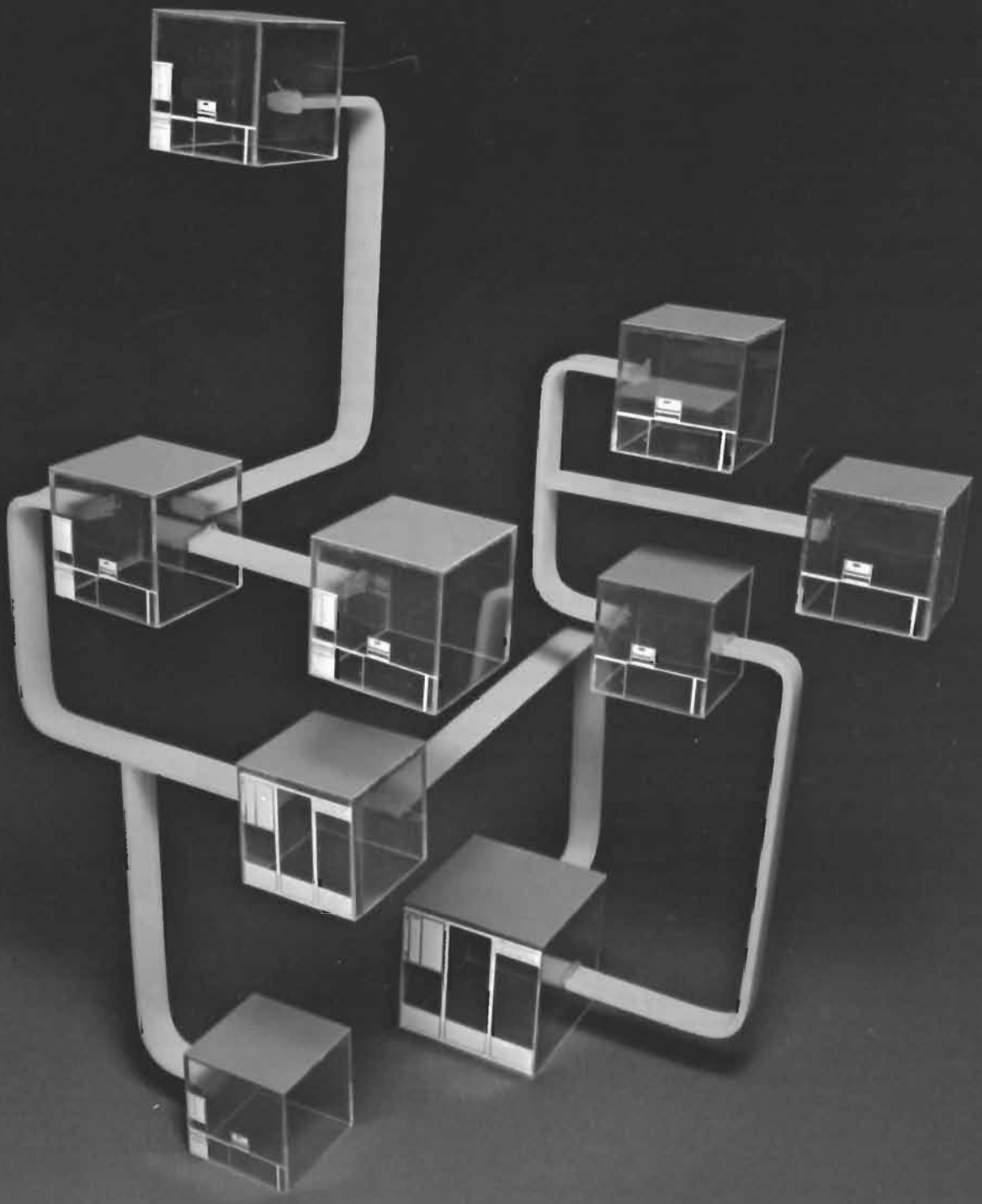
MARCH 1978

HEWLETT-PACKARD JOURNAL



The Hewlett-Packard Distributed System Network

HP-DSN is a set of distributed systems products and a set of design objectives that provide a framework for interconnecting HP computer systems to form a network.

by André O. Schwager

THE CURRENT computer-industry-wide move towards distributed computer networks and distributed processing is but the latest phase of a networking evolution that began with the first interconnection of two computer systems, with communication implemented at the user-software level. This was awkward and difficult to use, but it gave users the ability to share peripherals and special system services instead of duplicating them at each system.

The concept of easy-to-use distributed networks has been proposed for many years, but implementation has been slow, partly because of economic considerations and a lack of system network software, and partly because users' traditional ways of doing business are oriented towards centralized control. The availability in recent years of small computers with improved cost/performance has removed the economic impediment. Thus the potential exists for truly distributing processing power, through networking, into every department or functional area of a company, placing the processing power where the work is being done. System power may be increased as needed by adding low-cost systems to the network, thereby building "logically" large systems out of small-system building blocks. The sum of these smaller systems in a network configuration may equal or exceed the power and capability of a large central mainframe.

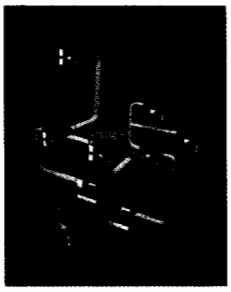
As soon as the hardware components became available, the transition from central systems to decentralized networks began. Because of the complexity and scale of the problem, this move has been cautious, tracking the availability of supportive network management software. Initially, most of the software was in the form of "tools" that could be used only by a specially trained, network-knowledgeable programmer at each node in the network. Now, as the issues of decentralized control are being resolved and network software is becoming available, total packages are available to users who see advantages in moving functions away from the central system and out to the network system members.

Hewlett-Packard Involvement

The first Hewlett-Packard distributed network products were designed to provide peripheral sharing and central programming services for multiple HP 2100s, distributed throughout factories and laboratories. This initial capability has developed into today's Hewlett-Packard Distributed System Network (HP-DSN).

HP-DSN consists of:

- An overall set of design objectives for the interconnection of various computer systems to form a network



Cover: *Our artist's representation of a distributed computer system network made up of HP 3000 Series II, HP 1000, and HP 2026 Computer Systems. HP distributed systems products provide for broad communications capabilities within such networks, and for remote job entry to large central mainframes.*

In this Issue:

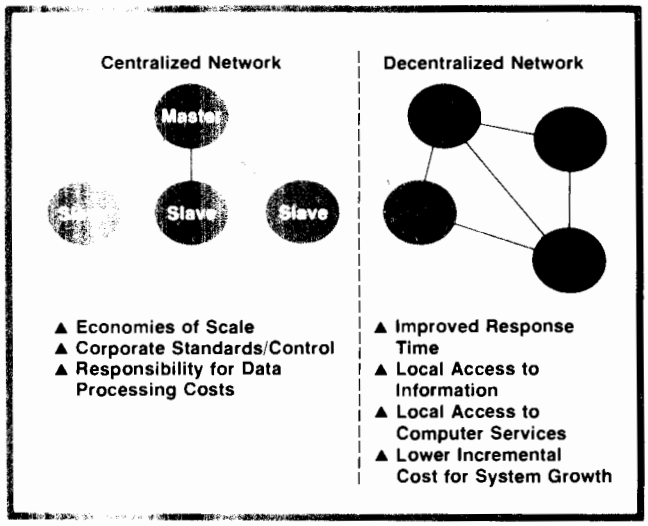


Fig. 1. *The Hewlett-Packard Distributed System Network is a set of design objectives that provide a framework for interconnecting HP computer systems. The framework of HP-DSN accepts a wide spectrum of network types, allowing the user to optimize the trade-off between total centralization and total decentralization.*

- A basic philosophy of distributed data processing
- A set of products that are available today.

The products include Distributed Systems/3000, Distributed Systems/1000, and the HP 2026 System. Articles on these products appear elsewhere in this issue.

The architecture of HP-DSN distributes both processing and control, but is sufficiently flexible to support hierarchical configurations and satellite processing for large, central mainframes via terminal access emulation (see Figs. 1 and 2). Ultimately, this will provide a user with an integrated family of HP computing products, (e.g., minicomputers, computers, calculating products, etc.) capable of spanning a wide range of applications in business, operations management, and instrumentation.

Distributed System Network Objectives

The objectives of HP-DSN may be summarized as follows:

- System and terminal capabilities that free the user from having to know communications protocols and network topology.
  - Remote file access
  - Remote data base access
  - Remote program operations
  - User access to any terminal in the network
  - Terminal access to any system in the network
- Provide connection to both public and private value added networks
- Interconnection of HP's family of computing products
- Connection with non-HP equipment
- Distributed computing with local control and processing
- Sharing of data and system resources
- Network diagnosis and recovery
- Data and communication security.

Distributed System Network Description

The Hewlett-Packard Distributed System Network is implemented as a number of functional layers in software (see Fig. 3). The layered approach provides

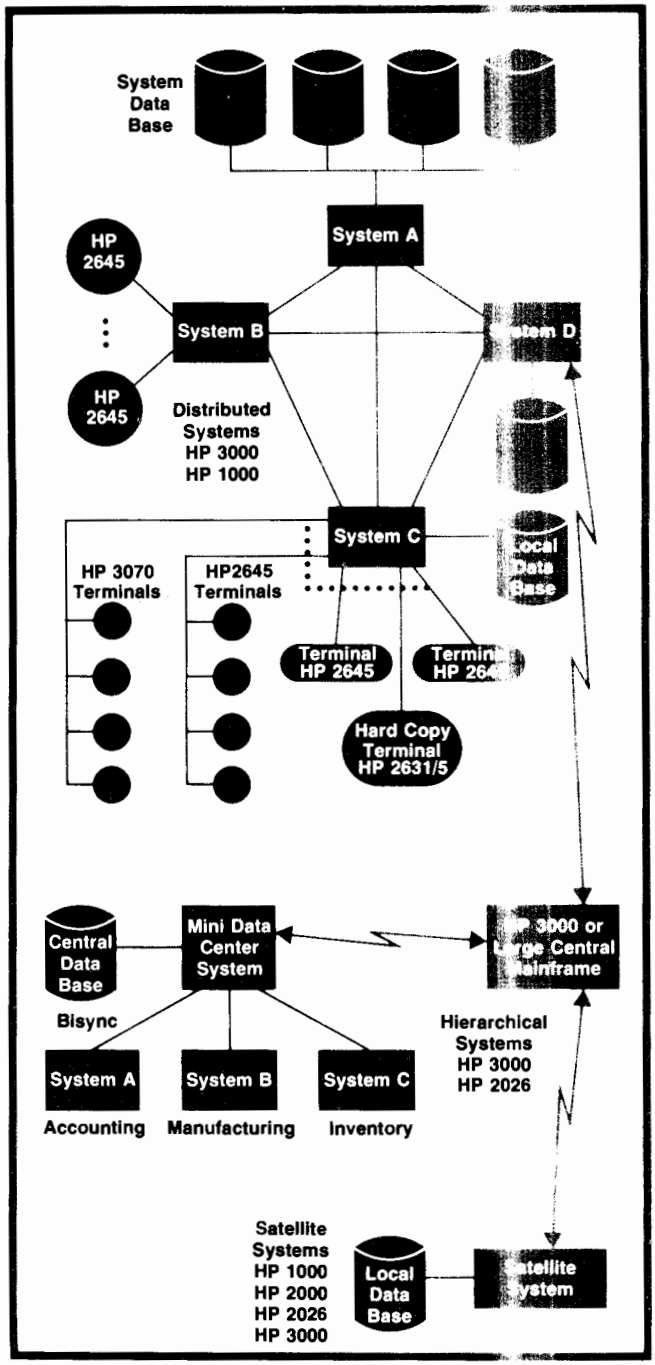


Fig. 2. *HP-DSN supports distributed processing and control as well as hierarchical configurations and satellite processing for large central mainframes.*

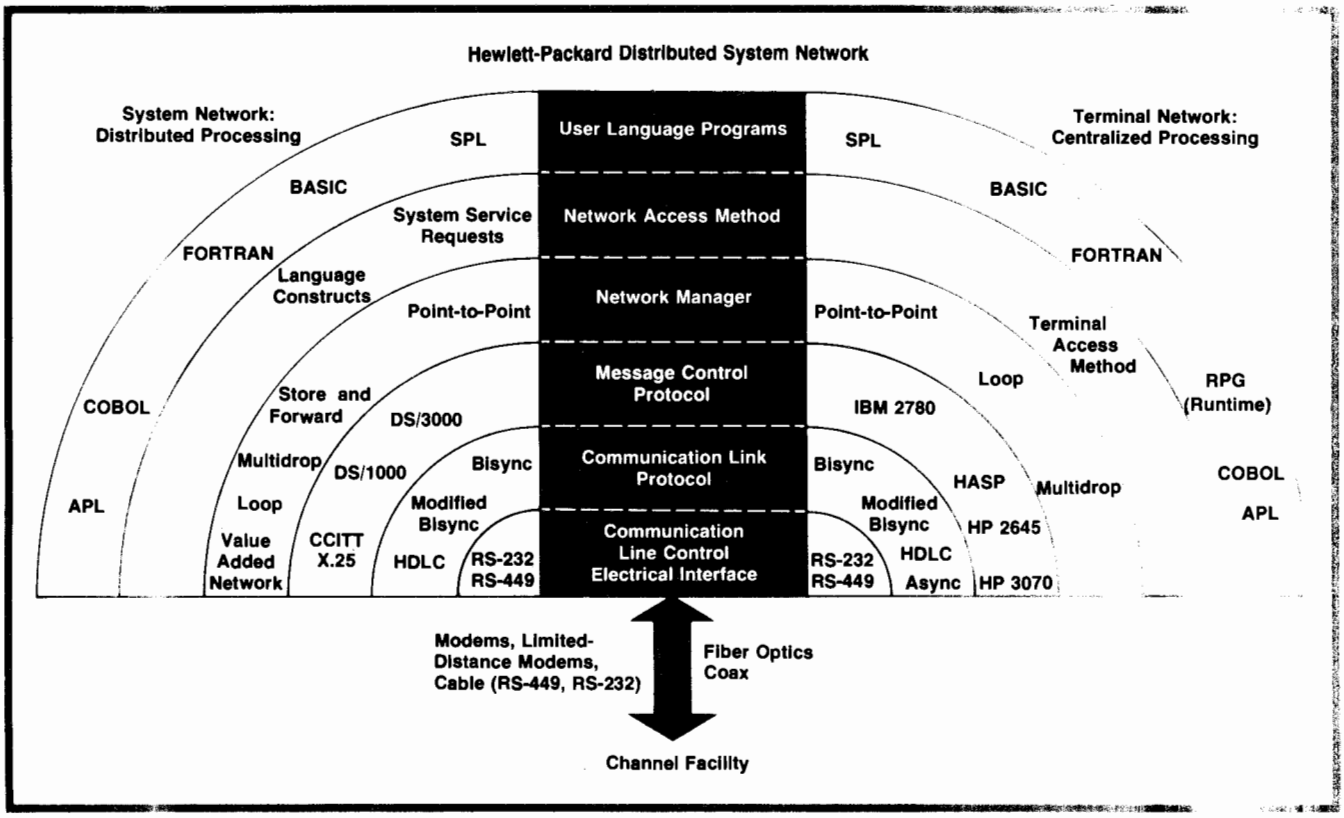


Fig. 3. *HP-DSN is implemented as a series of software layers. Improvements can be made in any layer without affecting the highest level, the user's applications programs.*

flexibility for additions and enhancements at each layer. Only the high-level system services are visible to users and applications programmers. Advantages of this approach include both stability and flexibility: stability because changes can be made to the internal layers without the user having to alter applications programs, and flexibility because the modular design of the network easily accommodates changes as a result of technological improvements.

The specific functions assigned to each layer are as follows (see Fig. 4).

Network Access Method. The network access method provides the network capabilities to the user at his application level via intrinsics, system services, or specific language constructs. The services include:

- Remote program management
- Remote file access
- Remote command processing
- File transfer
- Auto answer
- Auto dial
- Down-line loading
- Remote program development consoles
- Remote peripheral access

Network Manager. The network manager is the layer that is aware of the network topology and responsible for managing network error recovery and the various topology dependent functions such as polling and store and forward. For example, since the network access method views all transactions as logically point-to-point, the network manager of an intermediate node stores and forwards the message via the optimum path.

- Point-to-point
- Store and forward
- Alternate/secondary communication path
- Multidrop
- Loop or ring

Message Control Protocol. This layer provides control functions, addressing information, message type, and other requirements to effect end-to-end transmission (e.g., CCITT X.25).

Communication Line Protocol. The communication line protocol is the "grammar" or protocol by which two or more systems can exchange information in an efficient and reliable manner.

- Bisync
- Async
- ADCCP/HDLC

Communication Line Controller. Consists of the hardware that connects to the communications line and the software driver for this hardware interface.

Distributed System Network Implementation

The implementation of HP-DSN will be in phases, each adding to and supported by previous levels of

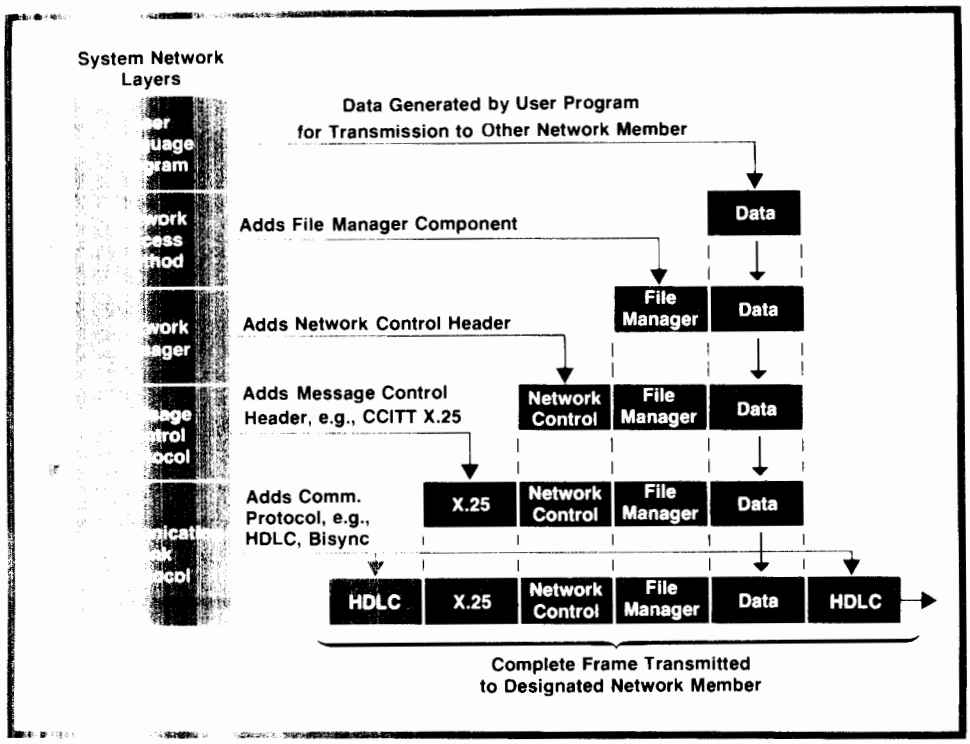


Fig. 4. *The building of an information packet, or frame, as data is passed through the various layers of HP-DSN. As the frame is received by the designated network member, the headers corresponding to each layer are peeled off, that is, the above process is reversed.*

HP-DSN capability. The recently announced Distributed Systems/1000 and Distributed Systems/3000 are the first steps. DS/1000 and DS/3000 provide high-level data communications among multiple HP 1000 and HP 3000 Computer Systems for remote command processing, remote file access, and program-to-program communication.

Remote Command Processing. Remote command processing allows users of a local system to access a remote system using exactly the same set of operating system commands as are available to a user at the remote system. All of the capabilities of both the local and the remote systems for program preparation, program management, subsystem access, and utility program use are available to the local user. In this mode, the user's local terminal becomes a "virtual terminal" to the remote system.

Remote File and Peripheral Access. Remote file and device access gives the local user full access to the data files and peripheral devices on a remote system. To access a remote file or device, the user need only make a minor change to the operating system file declaration to indicate that the resource is at the remote site. User programs need not be changed.

Program-to-Program Communication. Program-to-program communication allows user application programs being run in separate systems to exchange data and control information. This capability has provisions for remote program initiation and termination, data exchange coordination, and dynamically variable master/slave relationships.

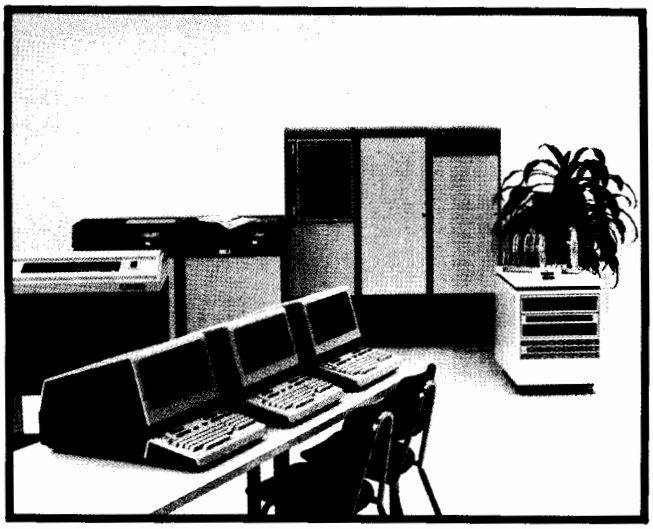
Future Enhancements

HP-DSN provides customers with a long-range, coherent plan for the interconnection of HP's diverse computing products and reaffirms HP's commitment to distributed processing. Currently, enhancement of the distributed system network capability is taking place in four dimensions (see Table 1).

- The already comprehensive, high-level set of user commands and procedures will be expanded to provide additional tools for high-level interfacing of HP systems. This enhanced set of system services will provide continued advantages to the customer with an all-HP system network.

André O. Schwager
André Schwager received his BSEE degree from the University of Utah in 1967 and his MSEE from the University of Santa Clara in 1972. He joined HP in 1973 after five years as a development engineer and served as project manager for 21MX data communications hardware. He's now an engineering section manager at HP's General Systems Division. André was born in Zürich, Switzerland and now lives in Cupertino, California. He's married and has three children. He enjoys skiing, tennis, soccer, and camping, and is active in local youth soccer activities.

- The network capability will be expanded to support additional system links in a fashion consistent with the HP Distributed System Network discussed above.
- The message and link protocol (transparent to the user) will be expanded or replaced to provide methods for HP networks to communicate with ADCCP/HDLC and industry standard CCITT X.25 packet switching value added networks.
- Electrical interfaces will move towards implementation of RS-449.



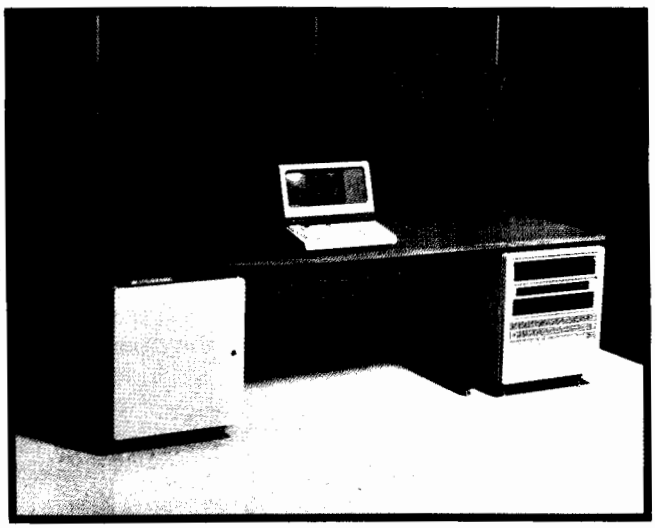


Fig. 5. *Typical configurations of HP computer systems that have network capabilities. Top: HP 3000 Series II Model 9. Middle: HP 1000 Model 30. Bottom: HP 2026.*

TABLE I

Current and Future Hewlett-Packard Distributed Systems Network Capabilities

Layer	Current Capability	Direction of Expected Additional Capabilities
NETWORK ACCESS METHOD	SYSTEM Remote Command Processing (Virtual Terminal) Remote File Access Program-to-Program Communication	SYSTEM Remote Data Base Access
	TERMINAL Terminal Access Method	TERMINAL User Access to any Remote Terminal Terminal Access to any System in the Network
NETWORK MANAGER	SYSTEM Point-to-Point Store and Forward (DS/1000)	SYSTEM Value Added Networks Multidrop
	TERMINAL Point-to-Point Multidrop (DS/1000)	TERMINAL Loop Multidrop (DS/3000)
MESSAGE CONTROL PROTOCOL	SYSTEM DS/3000 DS/1000	SYSTEM CCITT X.25
	TERMINAL HP 2645 HP 3070 IBM 2780/3780 RJE HASP	
COMMUNICATION LINE PROTOCOL	SYSTEM Bisync Async	SYSTEM ADCCP/HDLC
	TERMINAL Async Bisync	TERMINAL ADCCP/HDLC SDLC
COMMUNICATION LINE CONTROL	SYSTEM RS-232	SYSTEM RS-449
	TERMINAL RS-232 Multidrop Cable	TERMINAL RS-449

Distributed Systems/3000

DS/3000 makes it possible for the user of an HP 3000 Computer System to communicate with remote HP 3000, HP 1000, and HP 2026 Computer Systems.

by Philip M. Sakakihara

DISTRIBUTED SYSTEMS/3000 is a communications system, consisting of both hardware and software, that resides on an HP 3000 Computer System. DS/3000 allows an HP 3000 user to communicate with other HP 3000s and with HP 1000 and HP 2026 Computer Systems (see Fig. 1). Communication between these systems occurs in a bidirectional interleaved fashion using hardwired coaxial cables for HP 3000 to HP 3000 and HP 3000 to HP 1000 communications and modem lines for HP 3000 to HP 3000 and HP 3000 to HP 2026 communications.

Each of these computers can perform different data processing tasks. With DS/3000 the HP 3000 user has the benefits of the combined capabilities of the local HP 3000 and the remote computer, which may be another HP 3000, an HP 1000, or an HP 2026. The HP 3000, for example, with its Multiprogramming Executive (MPE) operating system, provides on-line applications, such as transaction processing, interactive program development for timeshare users, batch job processing, remote job entry, and now DS/3000. These capabilities can all be used concurrently.

The HP 1000, with its Real-Time Executive (RTE) operating system, is a high-performance computer system designed for real-time computation and instrumentation applications. HP 1000s with RTE-III, RTE-MII and RTE-MIII operating systems can be nodes of a DS/3000 network.

The HP 2026 is designed to meet users' data entry and data communications needs. Data can be entered, stored, and retrieved locally and then communicated using batch transfer techniques to the larger centrally located HP 3000 for further processing.

Layered Software Implementation

The principal design philosophy that guided the development of DS/3000 was that it be easy to use, requiring little or no user programming effort. As soon as the special hardware and software are installed the user can begin to use all of the features without any knowledge of system level communication functions.

An equally important design consideration was that the design be structured in functional layers, so that technological improvements can be made in any layer without affecting the user.

As Fig. 2 shows, Distributed Systems/3000 consists of five functional layers. Additions and enhancements can be provided at any layer without impacting the highest level, the user. For example, as technological advances are made in hardware and line protocol software, only the lower layers, 1 and 2 in Fig. 2, would be subject to change. Another advantage of the layered implementation is that network project design responsibilities are clearly defined, and hence easily manageable.

Network Access Method (Layer 5). This layer of software allows the 3000 user to access another computer using the following functional capabilities:

- Remote Command Processing (RCP). This is the ability to initiate a command on the HP 3000 and have it executed on a remote computer system.
- Remote File Access (RFA). This is the ability to access a remote file on a remote computer on a logical record basis.
- Remote Data Base Access (RDBA). This is the ability to access a remote data base on a remote computer system on a transaction-by-transaction basis.
- Program-to-Program Communications (PTOPC). This is the ability to have two applications programs running simultaneously in separate computers and have them communicate directly with each other.

The network access method software causes these user functions and requests to be formatted into DS/3000 messages, which consist of a message header, a block of control information, and optional data for requests. These messages are communicated to the second layer of software, the network manager, by the following set of DS intrinsics (callable routines).

DSOPEN	Validates user capabilities, builds appropriate DS table entries, and returns line number (equivalent to a file number returned by file system)
DSWRITE	Used by slave process to initiate a reply
DSWRITECONV	Used to write and receive data from the remote process
DSCHECK	Return status of layer 4
DSDEVINFO	Used to obtain DS device characteristics
DSCLOSE	Deallocates DS table entries, based on DSOPEN line number.

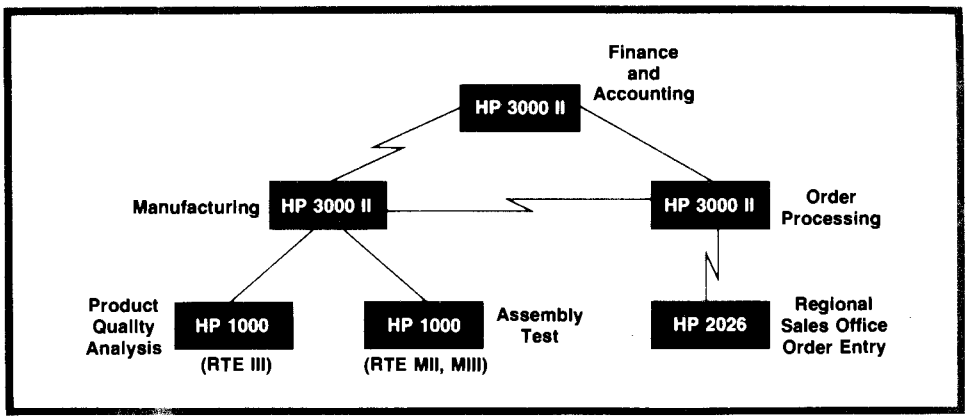


Fig. 1. *A possible DS/3000 network. DS/3000 allows high-level communication between a local HP 3000 and remote HP 3000, HP 1000, and HP 2026 Computer Systems.*

Slave processes also make use of these intrinsics for returning the replies that originate at the remote computer. The network access method layer is responsible for awaiting the arrival of the corresponding reply from the network manager.

Network Manager (Layer 4). The network manager layer consists of the DS monitor and related procedures, and the set of DS intrinsics listed above. The network manager receives the formatted request message from the network access layer, and is responsible for directing the formatted request to the appropriate communication link or remote computer via the message protocol controller, layer 3. The network manager is also responsible for returning the reply corresponding to the user request to the network access layer. Since this layer may receive many user requests and replies simultaneously, the network manager also provides the necessary buffering and bidirectional multiplexing and demultiplexing of user requests and replies to and from the message protocol controller. User requests and replies are communicated to the message protocol controller by a set of communications system intrinsics (CS/3000).

COPEN	Builds appropriate CS table for the particular communication hardware and selects the communication protocol
CREAD	Reads data
CWRITE	Writes data
CGETINFO	Obtains CS device characteristic
CCHECK	Returns status of layer 3
CCONTROL	provides status messages to the line protocols, e.g., abort a given I/O request
CCLOSE	Deallocates CS table entries.

Message Protocol Controller (Layer 3). The message protocol controller consists of the CS/3000 intrinsics listed above. These allow the network manager to communicate with the appropriate remote computer. This layer is protocol-independent, and controls the opening, reading, writing, and closing of any communication line controllers (layer 1). This is done by communicating these functions to the communication line protocols (layer 2) using the standard HP 3000 software I/O driver interface ATTACHIO.

Communication Line Protocols (Layer 2). This layer receives instructions from the message protocol controller, Layer 3, to send or receive user requests or replies using IBM's binary synchronous communication protocol. It is at this level that error recovery is done. This layer is typically called a communication protocol driver.

Communication Line Controller (Layer 1). This layer consists of the necessary hardware to establish the physical line connections between computer systems. Presently this is the synchronous single-line controller (SSLC) for communication over modem lines and the hardwired serial interface (HSI) for high-speed hardwired coaxial connections.

DS/3000 Messages

All DS/3000 requests that are received by an HP 3000 from another HP 3000, an HP 1000, or an HP 2026 have the following format:

Message Length	Message Class
Stream Type	
From Process Number	To Process Number
Reserved	
Reserved	
Request Length	
Calling Parameters for RFA, RDBA, RCP, etc.	
User Data if Remote Request Is a Write	

All DS/3000 replies that are returned by the HP 3000 to another HP 3000, an HP 1000, or an HP 2026 have the following format. Each reply corresponds to a request.

Message Length	Message Class
Stream Type	
From Process Number	To Process Number
Reserved	
Reserved	
Reply Length	
Status Information and User Data if Request Was a Read.	

Message class is a preassigned number that specifies the type of network service to be performed: program-to-program, remote file access, remote data base access, or remote command processing. Stream type is a number that reflects the specific function to be executed in one of the categories of network services: for instance, in remote file access, a specific HP 3000 file system request. Process number is used in a multiprogramming environment to identify the process to which a particular request or reply belongs. For example, when two HP 3000s are communicating this number will correspond to the local or remote session main process number.

The flow of user requests and replies is described in Fig. 3, the life of a DS/3000 message. This functional flow is transparent to the user. The local terminal user first (Step 1) logs on to the remote HP 3000 using Remote/3000 to ensure that the proper account, group, and user are referenced. This causes a DS/3000 request message to be formatted and passed to the network manager. The request message is passed through successive layers and eventually is received at the remote computer's network manager. The arrival of the request message (Step 2) causes a remote session main process to be created in the appropriate user account. A reply is then sent back to the local session main process user attesting to the validity of the request (Steps 3 and 4).

The local user can now run a local application program (Step 5) to access a remote file, data base, or slave program. If the operation is remote file access or remote data base access, the corresponding standard HP 3000 access method, FS/3000 or IMAGE/3000, will generate the request message. The request message is received at the remote session main process and executed (Step 6) on the remote system. The remote session main process then returns the corresponding reply (Steps 7 and 8) to the user application program as if the request had been executed on the user's local system.

The network manager, message protocol controller, communication line protocol, and communication line controller are all symmetrical when HP 3000s are communicating with each other. If the computer in-

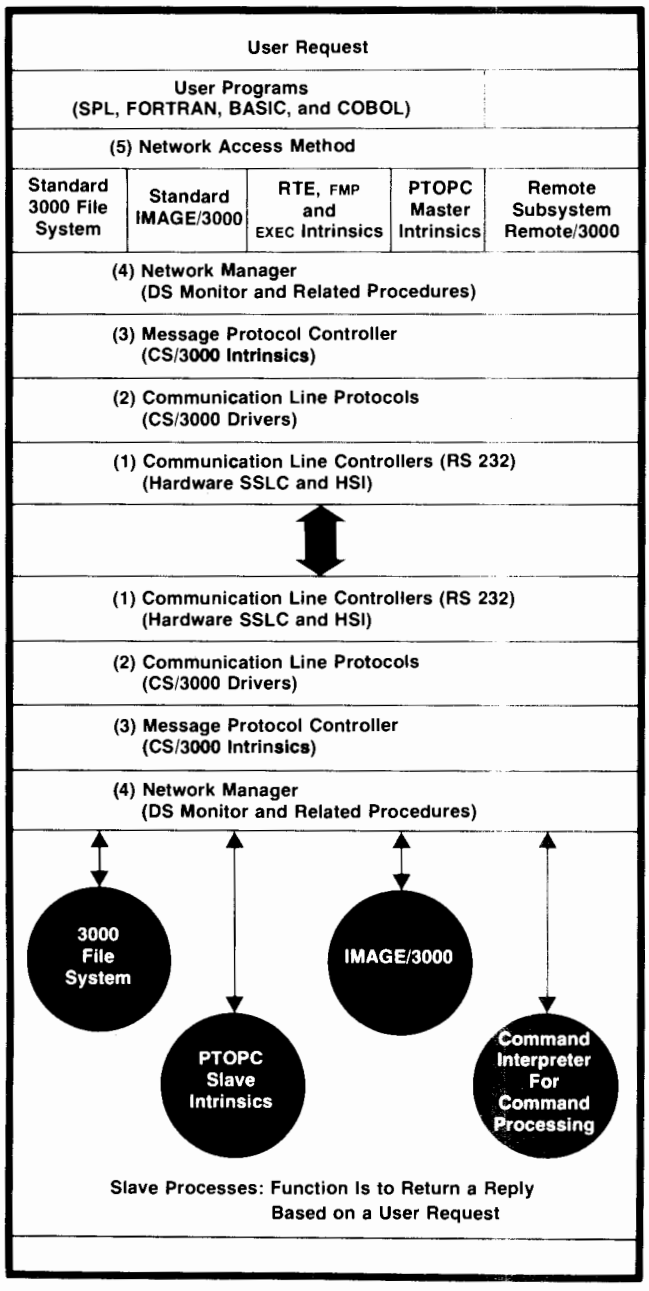


Fig. 2. *DS/3000 layered software implementation allows technological advances at any level without affecting the user. Corresponding layers in local and remote HP 3000s are identical, and communicate with each other through the intervening layers.*

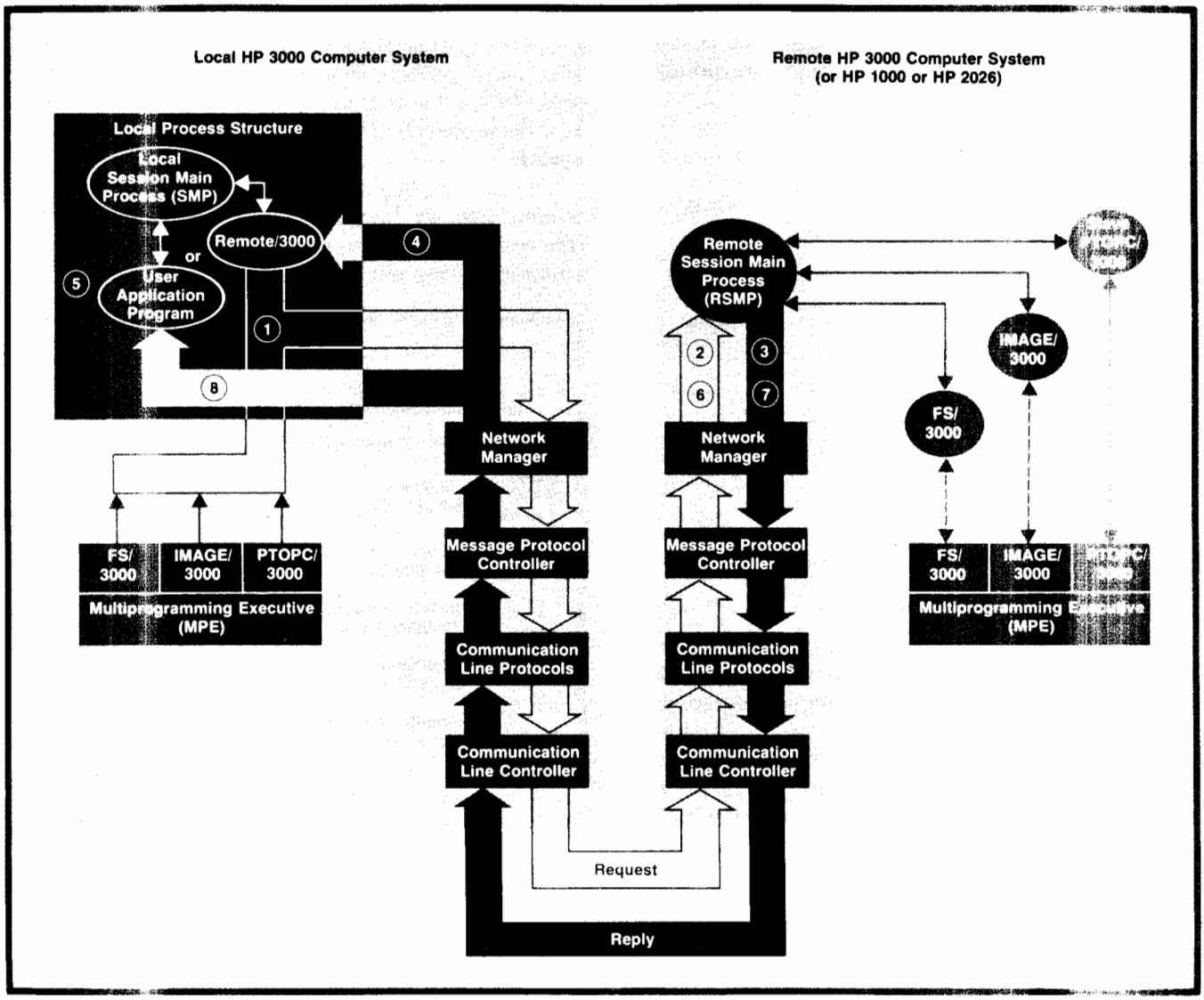


Fig. 3. *The life of a DS/3000 message. 1) Local terminal user logs on to the remote HP 3000 using Remote/3000. 2) Resulting DS/3000 request arrives at remote computer, causing remote session main process to be created. 3-4) Reply is sent back accepting or rejecting request. 5) If accepted, local user runs application program to access a remote file, data base, or slave program. 6) Resulting request message, generated by FS/3000, IMAGE/3000, or PTOP/3000, arrives at remote system and is executed. 7-8) Reply is returned to user program as if the request had been executed locally.*

itiating the request to the HP 3000 is an HP 1000 or an HP 2026, the HP 3000 message flow is still the same.

New Network Commands

Two new commands enable the HP 3000 user to make use of DS/3000 network services to other network nodes. The command DSLINE identifies the remote computer for remote command processing, remote file access, remote data base access, and program-to-program communication. Telephone number, identification list, maximum transmission buffer size, and the activation of data compression may also be specified. The other new command, REMOTE, is used to direct locally entered commands to a remote computer identified in the DSLINE command. This command is for remote command processing.

Remote Command Processing

All standard HP 3000 commands can be issued to both local and remote HP 3000s. The same restrictions that apply to the local use of these commands also apply to the remote use.

An HP 3000 user at a terminal can access a remote HP 3000 by simply typing the word REMOTE. The local system then assumes that the commands that follow are to be directed to and executed on a specified

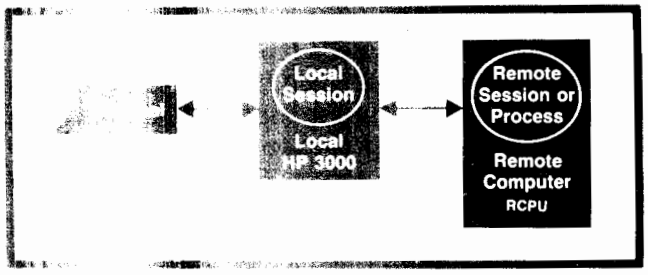


Fig. 4. *Remote command processing allows a local user to issue commands to a remote system as if the local terminal were connected directly to the remote system.*

remote computer (specified by DSLINE). The reply generated at the remote computer is returned to the local terminal user. This capability is called "virtual terminal".

The following example shows how a local user issues commands to a remote system (also see Fig. 4). The underlined text in this example is all the user needs to access the remote system.

:HELLO USER.ACCT	Local log on
:DSLINE RCPU	Designates the remote system for remote processing
:REMOTE	
# HELLO RUSER.RACCT	Log on to the remote system.
# LISTF	The user can now issue any HP
# EDITOR	3000 command to be executed on
# SHOWJOB	the remote computer
.	
.	
.	
# BYE	Log off the remote HP 3000.

Issueable HP 3000 to HP 3000 commands include all user commands, system supervisor commands, account manager commands, and system manager commands.

User commands allow the user to initiate, control, and terminate processing programs:

ALTSEC	EOF	RELEASE
		REMOTE
BASIC	FILE	RENAME
BASICOMP	FORTGO	REPORT
BASICPREP	FORTPREP	RESET
BASICGO	FORTRAN	RESETDUMP
BUILD	FREERIN	RESTORE
BYE		RJE
	GETRIN	RPG
COBOL		RPGGO
COBOLGO	HELLO	RPGPREP
COBOLPREP		RUN
COMMENT	JOB	
CONTINUE		SAVE
	LISTF	SECURE
DATA		SEGMENTER
DSLINE	PREP	SETDUMP
	PREPRUN	SETMSG
EDITOR	PTAPE	SHOWDEV
EOD	PURGE	SHOWIN
SHOWJOB	SPL	STREAM
SHOWOUT	SPLGO	
SHOWTIME	SPLPREP	TELL
SPEED	STORE	TELLOP

System supervisor commands provide for the general operation of the system:

ALLOCATE	RESUMELOG	SWITCHLOG
DEALLOCATE	SHOWLOG	SYSDUMP
QUANTUM	SHOWQ	

Account manager commands are used to define groups and account attributes:

ALTGROUP	NEWGROUP	PURGEGROUP
ALTUSER	NEWUSER	PURGEUSER
LISTUSER		

System manager commands are used to define account structures:

ALTACCT	LISTGROUP	PURGEACCT
LISTACCT	NEWACCT	RESETACCT

The HP 3000 user can use the same REMOTE command to issue standard HP 1000 commands to be executed on a remote HP 1000 Computer System. Issueable HP 3000 to HP 1000 commands include:

AB	Abort current batch program
BL	Set buffer limits
BR	Set break flag in named program's ID segment
DN	Declare I/O device unavailable
EQ	Examine status of I/O device
GO	Restart programs out of suspension
IT	Set time intervals for programs
LU	Examine/alter device logical unit assignments
OF	Turn programs off (abort)
ON	Turn programs on
PR	Change priority of programs
RU	Start a program immediately
RT	Release program's disc tracks
SS	Suspend programs
ST	Examine the status of programs
TI	Display the current time
TM	Set the RTE real-time clock
TO	Examine/alter an I/O device's time-out parameter
UP	Declare I/O device available

Remote File Access (RFA)

HP 3000 to HP 3000 RFA. RFA allows the local HP 3000 user to access a remote HP 3000's files or peripherals using the standard HP 3000 file system. This feature allows the standard languages—COBOL, BASIC, SPL, FORTRAN—and HP 3000 subsystems to access remote files with their standard I/O calls, transparently. This transparent feature was made possible by modifying File System/3000, which is used by all languages and subsystems, to recognize a new DS device, or remote computer, specification. The standard device specification of the file system

and the FILE command was expanded to include the DS device in a manner that would not impact the file system.

For example, to transfer a file LOCFILE from the local HP 3000 to a remote HP 3000's line printer (Fig. 5), the standard HP 3000 program FCOPY may be used. The local HP 3000 user enters

:HELLO USER.ACCT	Local log on
:DSLINE RCPU	Specify remote computer
:REMOTE HELLO RUSER.ACCT	Remote log on to remote computer
:FILE LIST; DEV = RCPU#SLP	File equation to specify the list file to be on the remote system's line printer.
:RUN FCOPY	
>FROM=LOCFILE; TO=*LIST	FCOPY running on the local system lists LOCFILE on the remote system

The same commands with the underlined text omitted would cause LOCFILE to be printed on the local printer if one existed.

Issueable HP 3000 to HP 3000 file system intrinsics include:

FCHECK	FREAD	FSPACE
FCLOSE	FREADDIR	FUNLOCK
FCONTROL	FREADLABEL	FUPDATE
FGETINFO	FREADSEEK	FWRITE
FLOCK	FRELATE	FWRITEDIR
FOPEN	FRENAME	FWRITELABEL
FPOINT		

The standard HP 3000 KSAM intrinsics, which are extensions of File System/3000, may also be issued, to access remote KSAM files.

HP 3000 to HP 1000 RFA/DEXEC. To access files on an HP 1000's moving-head disc, flexible disc, or tape mini-cartridge from an HP 3000, the HP 3000 user must write an HP 3000 user program that makes use of a new set of intrinsics, called remote FMP, that allow access to the remote HP 1000 file system. These intrinsics have a direct correspondence to the standard RTE* file management package (FMP). The new remote FMP intrinsics to access HP 1000 files are:

DAPOS	DNAME	DREAD
DCLOS	DOPEN	DSTAT
DCONT	DPOSN	DWIND
DCRET	DPURG	DWRIT
DLOCF		

To access remote HP 1000 peripherals, such as line printers, magnetic tape units, and scientific instrumentation, the HP 3000 user must write an HP 3000 program using remote DEXEC calls. The new remote DEXEC intrinsics to access HP 1000 I/O devices are:

DEXEC (1) Read a Record
DEXEC (2) Write a Record
DEXEC (3) I/O Control

DEXEC (10) Program Schedule
DEXEC (11) Time Request
DEXEC (12) Execution Time
DEXEC (13) I/O Status

These DEXEC calls may also be used to schedule or terminate programs, request system time, or inquire about the status of a program or an I/O device. These intrinsics have a direct correspondence to standard RTE EXEC calls.

To access HP 1000 files or peripherals, the local HP 3000 enters

:HELLO USER.ACCT	Local log on
:DSLINE RCPU	Specifies the remote computer (HP 1000)
:RUN PROG	This is a program that executes on the HP 3000 that accesses an HP 1000 disc and/or instrument using the intrinsics listed above.

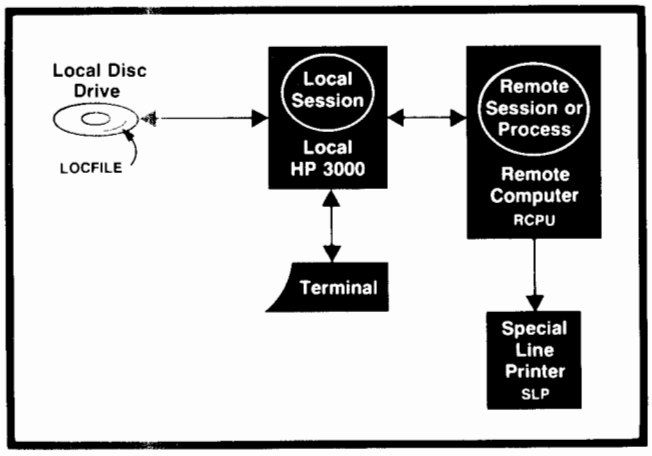


Fig. 5. *Remote file access makes it possible to read remote files or transfer local files to a remote system. Here a local file LOCFILE is being transferred to a remote line printer.*

Remote Data Base Access (RDBA)

RDBA is presently possible only between two HP 3000s. RDBA allows a local HP 3000 user to access remote data bases, that is, DS/3000 will direct standard local IMAGE program calls to be executed on a remote computer. IMAGE is HP's data base management system.

For example, an application program on the local computer can access and modify a data base on a remote computer. All IMAGE intrinsics remain the same, and the execution of IMAGE calls to access the remote data base is transparent. The use of IMAGE for RDBA was made possible by modifying IMAGE to recognize an expanded device parameter in the same way as the file system was modified for remote file

*RTE is the HP 1000 operating system, the Real-Time Executive system.

access, as described above. Standard IMAGE intrinsics that can be used for RDBA are:

DBOPEN	DBUPDATE	DBUNLOCK
DBFIND	DBDELETE	DBINFO
DBGET	DBLOCK	DBCLOSE

There are presently two methods by which an HP 3000 user can access a remote data base. The first method requires the user to establish a communication line and a remote session and enter a FILE equation for each remote data base. The FILE equation specifies which data base is to be accessed on which remote system and device. A local IMAGE application program can now be run to access the remote data base.

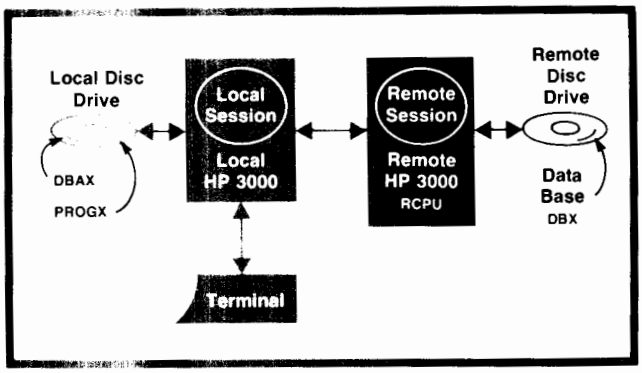


Fig. 6. *Remote data base access between HP 3000s can be implemented either by entering FILE equations or creating data base access files. Here a data base access file DBAX has been created and is used by PROGX, an IMAGE program, to access the remote data base DBX.*

The second method requires that a special file called the data base access file (DBA file) be created. This file provides IMAGE with the necessary information to establish a communications link and a remote session. It also specifies the remote data base file name so that the necessary IMAGE intrinsics can be executed on the remote computer. The following example will illustrate this (Fig. 6). The DBA file is built by the HP 3000 EDITOR. It is named DBAX and contains:

REC1 FILE DBX; DEV=RCPU #DISC	Specifies the location of the remote data base
REC2 DSLINE RCPU	Specifies the remote computer on which data base resides
REC3 USER ACCT, GROUP=HELLO RUSER ACCT	
.	Records 3 through n specify which local user, group, and account may access which user, group, and account in the remote computer.
.	
.	
RECN	

The local HP 3000 user then types the following to access the remote data base DBX:

: HELLO USER.ACCT	Local log on
: RUN PROGX	PROGX is an IMAGE program that opens DBAX, automatically establishes communication with the remote HP 3000, and accesses the remote data base.

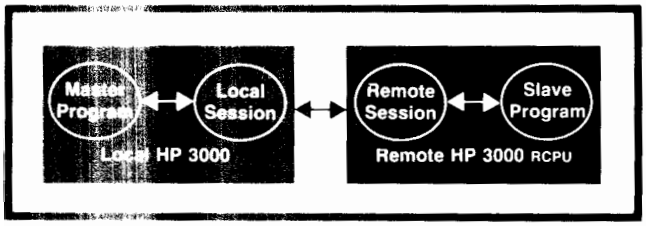


Fig. 7. *Program-to-program communication allows user programs in different systems to execute and exchange data in a coordinated manner. One of the programs is the master and is always in control. The slave program responds to the master's requests.*

Program-to-Program Communication (PTOPC)

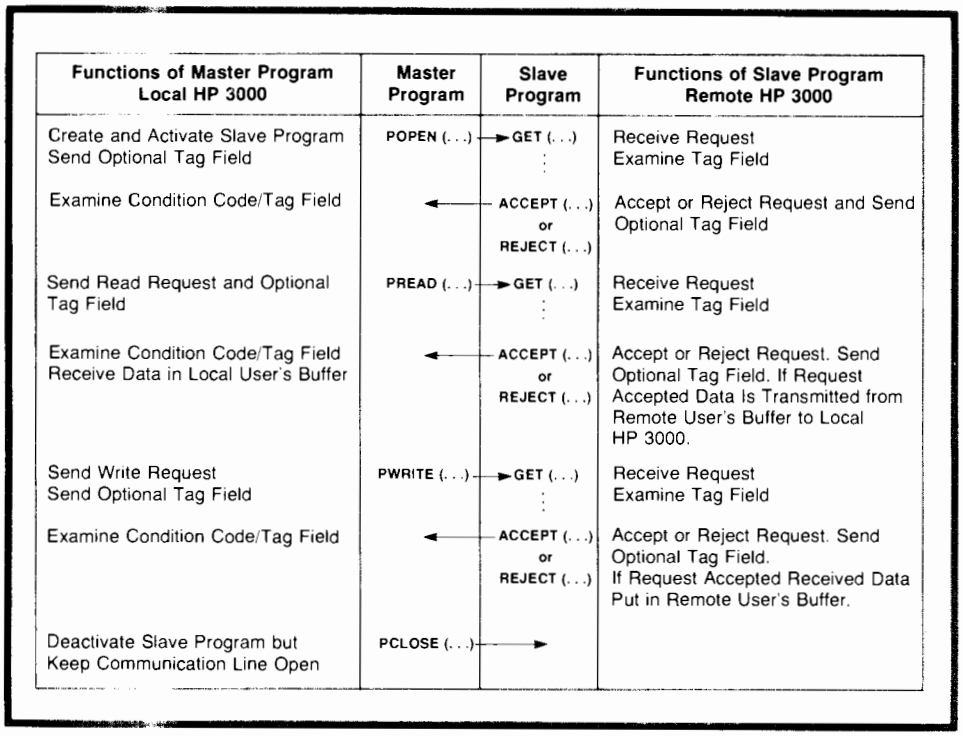
Program-to-program communication is made possible on the HP 3000 by a new set of intrinsics called PTOPC intrinsics. These are compatible with DS/1000 PTOPC (see article, page 15). They are directly callable by SPL, FORTRAN, BASIC, and COBOL programs.

Program-to-program communication allows two programs in different HP 3000 or HP 1000 systems to execute and exchange data in a coordinated manner. For data transfer to occur between the programs of remote computer systems, one of the programs must be the master and the other the slave (Fig. 7). The master program opens the data link, initiates the slave program, and is always in control. The slave program merely responds to requests received from the master program, either to accept or reject the master program requests. This master-slave relationship is dynamic in that it is determined solely by the design of the user's application program. Each computer may have master and slave programs active simultaneously, depending on the user's needs.

This capability is also the means used for HP 3000 to HP 2026 communication (see article, page 21).

The new PTOPC intrinsics for the HP 3000 are:

Master	Slave
POPEN	GET
PREAD	ACCEPT
PWRITE	REJECT
PCONTROL	
PCLOSE	
PCHECK (3000 only)	



Functions of Master Program Local HP 3000	Master Program	Slave Program	Functions of Slave Program Remote HP 3000
Create and Activate Slave Program Send Optional Tag Field	POPEN (. . .) →	GET (. . .) ⋮	Receive Request Examine Tag Field
Examine Condition Code/Tag Field	←	ACCEPT (. . .) or REJECT (. . .)	Accept or Reject Request and Send Optional Tag Field
Send Read Request and Optional Tag Field	PREAD (. . .) →	GET (. . .) ⋮	Receive Request Examine Tag Field
Examine Condition Code/Tag Field Receive Data in Local User's Buffer	←	ACCEPT (. . .) or REJECT (. . .)	Accept or Reject Request. Send Optional Tag Field. If Request Accepted Data Is Transmitted from Remote User's Buffer to Local HP 3000.
Send Write Request Send Optional Tag Field	PWRITE (. . .) →	GET (. . .) ⋮	Receive Request Examine Tag Field
Examine Condition Code/Tag Field	←	ACCEPT (. . .) or REJECT (. . .)	Accept or Reject Request. Send Optional Tag Field. If Request Accepted Received Data Put in Remote User's Buffer.
Deactivate Slave Program but Keep Communication Line Open	PCLOSE (. . .) →		

Fig. 8. *Functional flow in HP 3000 to HP 3000 program-to-program communication (PTOPC). The tag field, a user option, is a 20-word array that contains control information, usually directing the other program to take a specific action. The* GET *intrinsic receives the next request from the master program and allows the slave user program to* ACCEPT *or* REJECT *the master's request. If either the local computer system or the remote computer system is an HP 1000 or HP 2026 instead of an HP 3000, the functional flow on the HP 3000 side remains the same.*

Fig. 8 illustrates the functional flow involved in program-to-program communication.

Acknowledgments

The author would like to thank all of the people who contributed to the success of this project. The project engineer design team, which was responsible for the design and implementation, consisted of the following members: Bill Gard was an instrumental part of the concepts and overall design discussions. Bill's specific responsibility was the development of the network manager. Mike Philben and Hank Cureton were responsible for the user network services layer. Howard Morris was responsible for the message protocol controller and the communication line protocol drivers. Thanks should also be extended to Bob Gerstmyer for his efforts to get the necessary DS/3000 performance data to qualify the product, Alex Petruncola, who made the necessary changes to IMAGE for remote data base access, Art Benjamin for implementing data compression, and Todd Hirozawa for extending PTOPC intrinsics to be callable from BASIC and COBOL.

The major portion of the design and implementation of the HP 1000 to HP 3000 software in DS/1000 was part of the DS/3000 project. Jim Hartsell was the major contributor of the overall software design and was specifically responsible for the HP 1000's network services and network monitor software. Tom Keane was responsible for the HP 1000's message protocol controller and communication line protocol drivers.

Philip M. Sakakihara

DS/3000 project manager Phil Sakakihara joined HP in 1972 as a systems programmer on the IBM 370. One of his early projects was the solution of semiconductor diffusion problems using simulation. Later he served as project leader on several HP 3000 software products. A native of California's central coast area, Phil received his BA degree in mathematics from San Jose State University in 1966 and his MS in applied mathematics from the University of Santa Clara in 1973. Before coming to HP, he spent six years working on various aspects of a worldwide satellite network. Phil is married, has two sons, lives in San Jose, and enjoys fishing, camping, skiing of all kinds, jogging, and the martial arts.

Distributed Systems/1000

DS/1000 makes it possible to interconnect HP 1000 Computer Systems in virtually any configuration to integrate instrumentation, computation, and operations management tasks, and to link these systems with HP 3000 Series II Systems for distributed data processing.

by Robert R. Shatzer

DS/1000 IS A SOFTWARE and firmware package that provides an integrated set of high-level network facilities and procedures for communication between HP 1000 Computer Systems. DS/1000 also supports network communication between HP 1000s and HP 3000 Series II Computer Systems. HP 1000 Systems are based on Hewlett-Packard's multi-user, multiprogramming Real-Time Executive (RTE) operating system family. An HP 1000 network can use either the disc-based RTE-III or the memory-based RTE-M operating system.

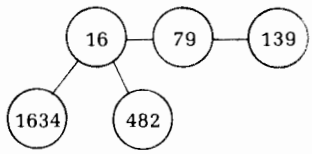
Distributed Systems/1000 represents a state-of-the-art advancement in small computer networks, building on Hewlett-Packard's experience, which began in 1973 with a central-satellite (star) network with fixed procedural relationships. In 1975, HP introduced the 91700A Distributed Systems,[1] which added flexible procedural relationships and a multitude of satellite types to the network.

DS/1000 supports a generalized network architecture of HP 1000 Computer System nodes. These nodes can be configured in an arbitrary topology to suit the application, instead of requiring the application to fit the network architecture. HP 1000 systems can be interconnected via hardwired or modem links in star (i.e., hierarchical), ring, or string architectures or in any combination of these (see Fig. 1). Each HP 1000 in the network is assigned a nodal address. By referencing this nodal address, applications programmers and interactive terminal operators can extend their reach to resources on remote HP 1000 systems using high-level requests that are straightforward extensions of equivalent local requests. DS/1000 provides the systems programming required to manage the communications, thus eliminating the need to duplicate this in each application.

DS/1000 provides a variety of high-level user interface features, including remote file access, remote command processing, and program-to-program communication. These facilitate network resource sharing, distributed data file management, communication between applications programs, and the distribution of processor workloads.

Store and Forward

DS/1000 software and firmware processes high-level user requests not only between directly connected HP 1000 nodes, but also between HP 1000 Systems that are not directly connected but are linked through intervening HP 1000 nodes. DS/1000 software and firmware manages this memory-buffered "store and forward" operation without the need for any user application programming at the intervening nodes. For example, in the network below,



a user application program executing in node 482 can access a file at node 139 without any application programming at node 16, node 79, or for that matter, even at node 139 itself. The communications link between nodes 16 and 79 can be shared by nodes

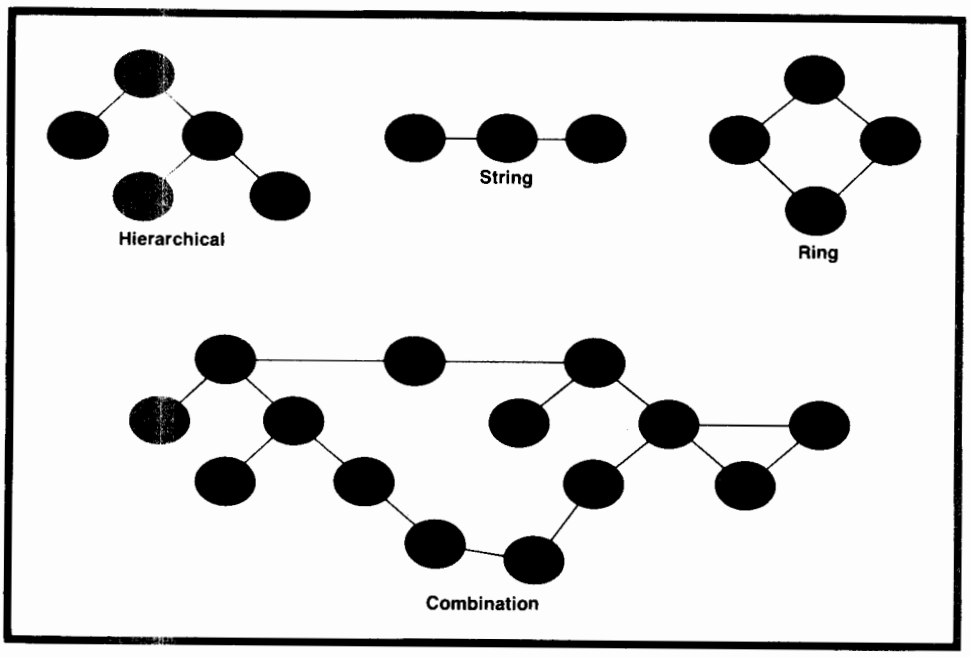


Fig. 1. *Distributed Systems/1000 allows the user to interconnect HP 1000 and HP 3000 computer systems in virtually any configuration.*

1634, 482, and 16. DS/1000's store and forward capability also makes multilevel networks practical and manageable, since an operator or program at node 79, for example, can control activities at all other nodes.

Modem or Hardwired Communications

DS/1000 uses the microcode capability of the HP 1000 to provide a high-speed multiline driver for modem or hardwired communications. Multiple lines can be active concurrently. Also provided is efficient error detection: vertically, longitudinally, and diagonally on each block received.

A DS/1000 network node can also communicate via a hardwired link to a directly connected HP 3000 Series II Computer System equipped with Distributed Systems/3000 software (see article, page 7). An interactive terminal on a directly connected HP 1000 system can be a virtual terminal to the HP 3000, executing local MPE commands remotely. Programs executing in the HP 1000 can access files and exchange data with programs on the HP 3000. Similarly, an HP 3000 terminal operator has remote RTE command processing capabilities, and HP 3000 applications programs can access files and peripherals and schedule and/or exchange data with programs on the HP 1000s.

User Interface

DS/1000 software has a layered architecture that makes only high-level system and file services visible to the user. These facilities include remote file access (RFA), distributed RTE EXEC remote command processing, and program-to-program communication capabilities. While these capabilities are available from both an HP 1000 and an HP 3000, only the HP 1000 calls are illustrated here for reasons of brevity.

Remote File Access (RFA)

Remote file access is the ability to access named files that exist on other nodes in the network. These files may be on moving-head or flexible disc media or on tape mini-cartridges. The program intrinsics necessary to implement RFA are nearly identical to those a programmer would use to access a file in the local system. For example, a set of intrinsics to open a file, read two records, and then close the file would be coded as follows:

```
DIMENSION IDCB(4), INAME(3), IBUFR(128), ICR(2)
DATA INAME/2HMY, 2HFI, 2HLE/
CALL DOPEN (IDCB, IERR, INAME, IOPT, ISECU, ICR)
CALL DREAD (IDCB, IERR, IBUFR, ILEN)
      .
      .
      .
CALL DREAD (IDCB, IERR, IBUFR, ILEN)
      .
      .
      .
CALL DCLOS (IDCB, IERR)
```

The ICR parameter is a two-word array. The first word specifies the cartridge that contains the file, just as is normally done for the RTE FMGR calls. The second parameter in the ICR array specifies the node in the network at which that file resides. In a network environment, the user can specify local files simply by using either the local node number for the second word of the ICR array or by specifying this parameter as a negative 1. In this manner, a user can redirect the ultimate execution destination of the RFA intrinsics

as the program executes.

In a similar manner, the programmer may access remote peripherals, such as line printers, magnetic tape units, plotters, and so forth, by using distributed executive (DEXEC) calls. The DEXEC calls may also be used to schedule or terminate programs, request system time, or inquire about the status of a program or I/O device. The format for a DEXEC call is essentially the same as that of a standard RTE EXEC call, except that the intrinsic name is changed from EXEC to DEXEC and there is one additional parameter in the calling sequence that allows the user to specify the node in the network at which the call is ultimately to be executed.

Remote Command Processing

A significant set of capabilities is provided in the area of operator commands. The user has two operator interface programs that allow virtually unlimited access to a remote system's resources. The operator uses a module called REMAT to access remote HP 1000 network nodes. REMAT provides the following capabilities:

CR	Create a file
DL	List a file directory
DU	Dump a file to a logical unit
PU	Purge a file
LI	List a file
LO	Load a program into an RTE-M node
RN	Rename a file
RW	Run a program
ST	Store a file or logical unit into a file
TE	Send a message to the remote operator
TR	Transfer to a command file
SW	Switch the destination of remote commands

In addition, several network resource management commands are available, as well as any RTE command that would normally be executable on the remote system. In the following example, a user at node 1 wishes to store a file from node 2 to node 3 and then tell the operators at both nodes that the file has been transferred:

```
*RU,REMAT
$SW,2,3,DS
#ST,FILEA,FILEB
#TE,FILEA HAS BEEN TRANSFERRED TO NODE 3
#SW,3,,DS
#TE,FILEB HAS BEEN TRANSFERRED FROM NODE 2
#EX
```

Note that the first SW command switches the effect of the subsequent store command (ST) from node 2 to node 3. The DS parameter is a user-specified security code to prevent unauthorized access to the remote systems. When either the source or destination node is redirected, the REMAT prompt changes from $ to # to indicate that at least one node is not local. When the switch command SW is used with a command involving a single node, only the first parameter is used for addressing. This can be seen in the next three commands. The first TE (tellop) command sends the specified character string to node 2, where it is displayed on the system console. The second switch command directs the second tellop command to node 3.

For operator interface to the HP 3000 system, the module RMOTE is used. With RMOTE, the user can execute any local RTE operator command or switch the destination of all subsequent commands to the HP 3000 system, as follows:

```
*RU,RMOTE
$TI                 Display local system time
1977  221  10  54  1
$SW                 Switch to remote HP 3000 system
#HELLO USER.ME      Sign on to HP 3000
#SHOWJOB
.
.
.
#EX
```

With RMOTE, the user has virtual-terminal access to the HP 3000. That is, the terminal or the local HP 1000 behaves as though it were connected directly to the HP 3000. Using this terminal, the user can then perform virtually any function that an HP 3000 terminal user can perform.

Program-to-Program Communication

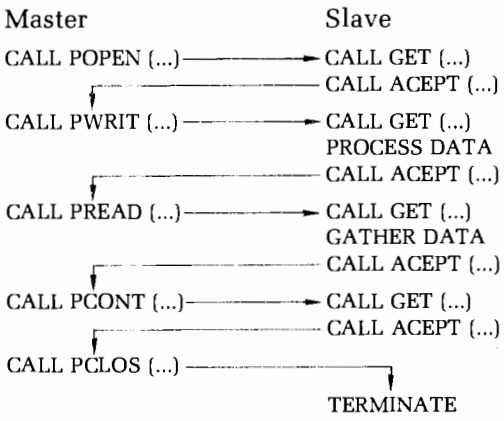
Another very powerful capability the user has available is program-to-program communication (PTOPC). This capability allows two or more user programs to send data or control information back and forth between them. This provides a more generalized solution to many problems than the RFA or DEXEC capabilities, because the receiving program can process data before storing it in a file, or perhaps the receiving program can print a report directly.

With program-to-program communication, the user designates that the program initiating or opening the PTOPC link is the master and the receiving program is the slave. A single slave may have several masters or a single master may talk to several slaves. Once the link is opened, the master initiates all communications with the slave and the slave responds by either accepting or rejecting the data or control information. When appropriate, the PTOPC link may be broken by either the master or the slave. Just as in RFA and DEXEC calls, the master and slave programs may be either in the same network node or in widely separated nodes.

The master program can send data to the slave program and receive data from the slave with the PWRIT and PREAD calls. Along with the data goes a 20-word user-defined tag field that may be used to specify status, disposition of the data, or other information. The tag field may be passed without any data

by executing a PCONT request.

A typical PTOPC exchange is as follows:



At each master call, the 20-word tag field is sent to the slave, and with each ACEPT call by the slave, the tag field is returned.

Distributed System Operations

DS/1000 provides a high level of access to the remote network node without the user's being concerned with message formats, communication line protocol, communications error handling, and the like. The user need only be concerned with the job at hand. The layered architecture of DS/1000 provides the transparency required to isolate the user from the tedious aspects of data communications.

DS/1000 is a transaction-oriented system, that is, each user request causes a request to be sent from the master node to the slave node. The slave, after servicing the request, sends a reply back to the master node. At this point, the transaction has been processed and control is returned to the user program.

Fig. 2 shows the flow of data and control in a DS/1000 node. In the following description, the numbers in parentheses refer to the control and data flow paths in Fig. 2. Assume that the user program, in the upper left of the diagram, executes a master subroutine call to a network service intrisic (1). For example, this call may be a remote file access file write command, as discussed earlier. The user program is then suspended until a reply is received from the remote system. The network service instrinsic (in this example, the DWRIT routine) parses the request, checks it for errors, and then builds a message block that will eventually be sent over the communications line. The message block, containing both the user's request and the data to be written into the file, is then sent to the DS/1000 communications management routines (2). Here, the source and destination nodal addresses, a sequence number, and other control information are added to the message block. At this point, a transaction control block (TCB) is built (3). The TCB contains destination information, a sequence number, the communications logical unit (LU), the ID segment address of the master program, and a countdown timer that is used to return control to the user program in case a failure on the slave side or in the communications link prevents the reply from getting back.

The LU number is derived by referencing the nodal

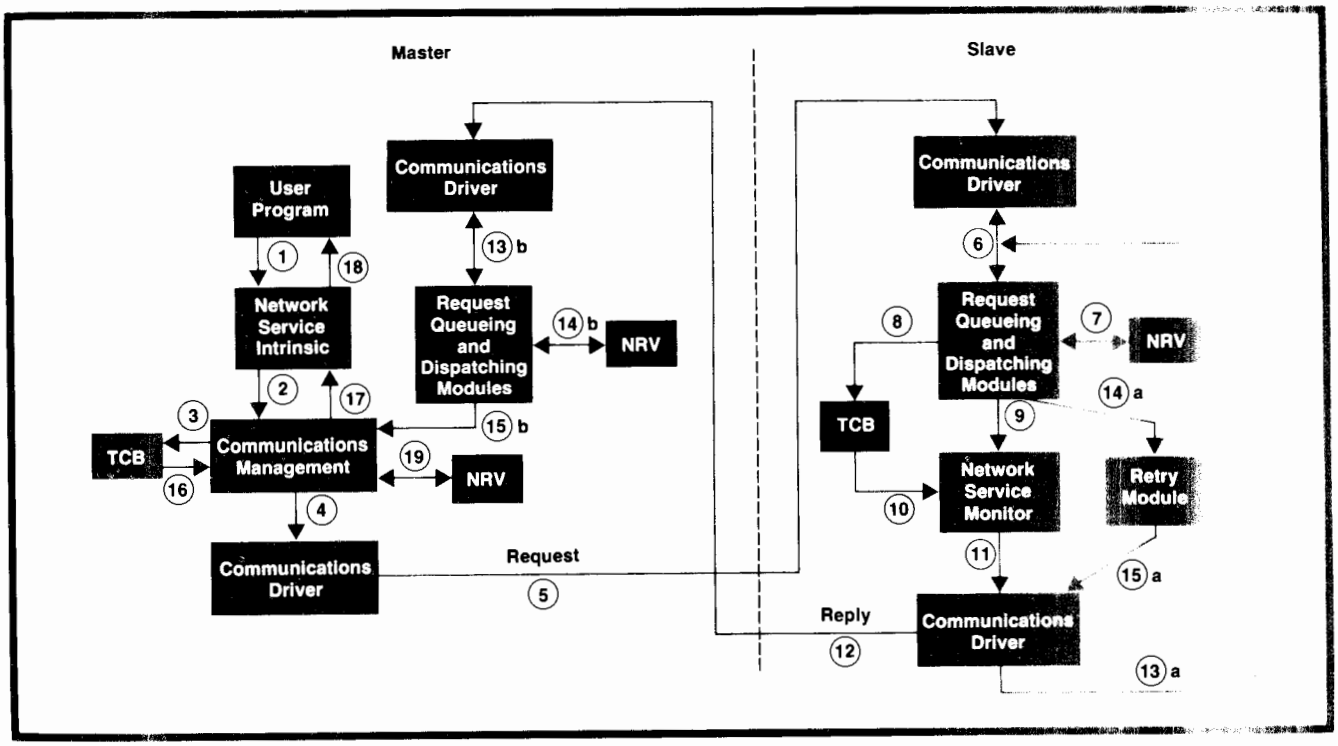


Fig. 2. *Data flow and control flow in master and slave DS/1000 nodes. See text for details.*

route vector (NRV) with the destination nodal address (19). The NRV is a table that specifies, for each remote node in the network, an LU number that the local node uses to access the remote node (see Fig. 3).

The TCB having been built, the communications driver with its supporting firmware is called (4). The driver microcode handles the actual transmission of the request and the data to the remote node (5). The transmission technique is bit-serial asynchronous, using 16-bit words, with a start bit, a stop bit, and one bit of horizontal parity. The entire transmission block, including request and data, is error checked using vertical, horizontal, and diagonal parity, yielding error control on the order of that provided by CRC-16. Retries in case of error are handled within the firmware, providing very low system overhead.

At this point, the request has been sent from the master node to the slave node. The first portion of the request is received by the slave node's communications driver as an unsolicited interrupt. The receipt of this interrupt causes the request queuing module to initiate a read request to the communications driver (6). The user's request and data are placed in system available memory and the request dispatching module examines the destination information contained in the message block, using the local NRV for this node (7). If the destination nodal address in the message block matches the local node number, a slave TCB is built (8). This TCB contains information similar to that contained in the master TCB. It also contains the identity of the network service monitor that will ultimately service the request. The TCB also contains a timer that is used to clean up and deallocate system resources in case the network service monitor cannot complete the servicing of the request.

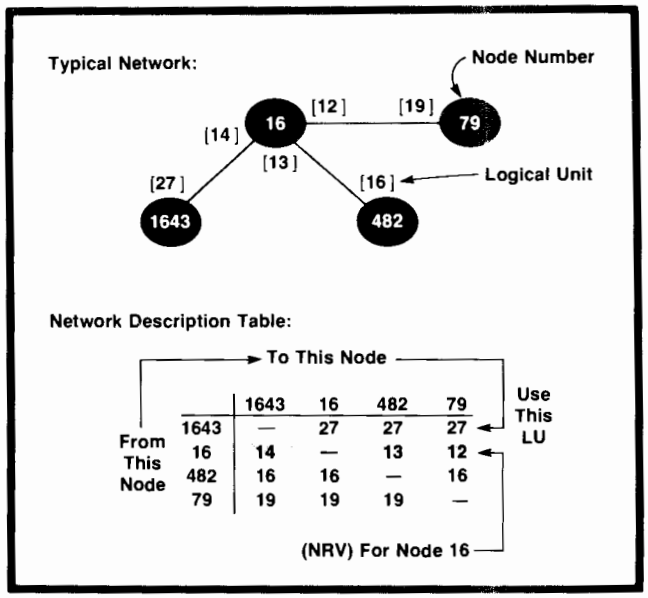


Fig. 3. *Store and forward function is implemented by means of the nodal route vector (NRV), resulting in very low system overhead. Requests and replies propagate from node to node to their destinations. The NRV, a row of the network description table (NDT), specifies the logical unit number that the local node uses to access each remote node in the network. The NDT is a generalized table structure that defines the topography of the entire network.*

Store and Forward Operation

If the destination nodal address in the message block does not match the local node number, the NRV is accessed to determine the next LU to which the message must be directed so that it can be forwarded to its destination. This "store and forward" function has been implemented before only in a few very large networks and systems; DS/1000 makes it available in small-computer networks for the first time. It is handled with very low system overhead, and is general in nature, in that the request will propagate from node to node until it reaches the destination node.

After the TCB is built, the request and data are passed to the network service monitor (9), which reconstructs the user call and then executes it. In this example, the DWRIT call issued by the master program is converted into a local RTE file manager WRITF call, which accesses a specified local file. It is here that the actual data control block for the file is maintained by the remote file access monitor. In general, most flags and control fields are maintained on the slave side so that a minimum of data is transmitted over the communications line.

When the user request has been serviced by the network service monitor and the local request completion information (and data, if appropriate to the type of request) has been gathered, a reply message block is built, containing the same basic information as the initial request message block, along with a flag to indicate that it is a reply instead of a request. This block also contains the request completion information and any error codes that may have been generated.

At this point, the TCB is no longer needed, so it is deallocated (10), and the reply message block is submitted to the communications driver (11) to be sent back to the master node (12). Once again, the driver firmware takes care of sending the message block and handling the retries in the event of a communications error. When the message block is given to the driver (11), the network service monitor is free to return to its request queue (9) and begin servicing another request that may have arrived while it was servicing the current one. This will happen even if there are I/O transactions currently pending on the communications line. In this way, the monitor has maximum availability to service user requests and is not tied up during I/O wait time. Since there is a monitor for each major network function that is supported (RFA, DEXEC, PTOPC, etc.), the system operates on the principle of

multiple single-server queues.

Error Handling

When the reply message block has been sent by the communications driver, the I/O completion status is returned to the dispatching and queueing modules (13a). If there were no communications errors that could not be handled by the firmware, the completion status is discarded. If there was an error, the reply message block is retransmitted a specified number of times by the retry module (14a). The retry module resubmits the message block to the communications driver for retransmission (15a). If the transmission is successful this time, the I/O completion information (13a) is discarded by the dispatching and queuing modules. If not, the cycle is repeated by the retry module until the retry count is exhausted, at which point a diagnostic message is logged on the system console at the local node.

Now when the reply message block is received back at the master node, it is again seen as an unsolicited interrupt to the communications driver. Upon receiving this interrupt, the driver schedules the request queuing module, which in turn initiates a read request to the communications driver to read the reply message block (13b). Once again, the reply message block is placed into system available memory and the request dispatching module examines the destination information contained in the block and accesses the NRV to determine whether this is the original source node (14b). If this is not the originating node, the request is forwarded to the appropriate LU as described above. Note here that the sense of the source and destination nodes are reversed because the reply flag is set in the message block. "Source" is always the master node and "destination" is always the slave node.

The reply message block is now returned to the master node communications management routines (15b), where the communications header information is stripped off and the TCB is deallocated (16). Control is then passed to the network service instrinsic (17), where the reply information is reformatted so that it matches the information that would be returned if the corresponding call had been executed locally. Finally, control is returned to the user program and execution of that program is resumed (18).

If the user specifies that the destination node is local, the same general flow is followed, except that the communications driver, and hence the retry module, is bypassed. That is, line 4 feeds into line 6 and line 11 returns via line 15b and all the modules are contained in the same node.

In addition to the modules and data flow described above, there is a routine to handle diagnostic error reporting that may be necessitated by conditions detected within the request queuing and dispatching modules. This is done so that these critical-response-time modules are not slowed down or impeded by the overhead involved in reporting these messages. More important, there is a module that performs the watchdog timer function on all master and slave requests and, in general, deallocates system resources when they are left locked up by an abortive communications error.

Acknowledgments

A great many people contributed their expertise and support to the development of DS/1000. I want to acknowledge the extensive contributions of project team members Dan Gibbons, Craig Hamilton, Jim Hartsell, Dave Tribby, and Chuck Whelan in the areas of project definition, implementation and testing. Bill Stevens provided a great deal of marketing direction and guidance, while Paul McGillicuddy lent his technical marketing expertise. Lyle Weiman did the product support evaluation and provided a great deal of friendly criticism and guidance. Dennis Leitterman provided for the hardware support function on the microcode. Dave Hancock and Dixie Condo guided the product through manufacturing. Ray Spear and Doug Tsui did a large amount of pre-release testing and quality certification. In addition, many people, too numerous to mention, provided the support, enthusiasm, encouragement and guidance that is so necessary to a project of this magnitude.

Reference

1. S. Dickey, "Distributed Computer Systems," Hewlett-Packard Journal, November 1974.

Robert R. Shatzer

Bob Shatzer is project manager for data communications and computer network products at HP's Data Systems Division. Himself a product of Detroit, Michigan, Bob received his BSEE degree in 1968 from Wayne State University. During and after his studies, he gained several years' experience in digital logic design, systems development, software development, and engineering management. With HP since 1973, he's provided product support for RTE-based systems and developed data communications software. He has two patents to his credit. Bob is a member of the IEEE Computer Society, the International Solar Energy Society, and the Northern California Solar Energy Association. A solar energy enthusiast, he's currently installing a solar system to provide all of his home heating needs. He's married, has two daughters, lives in San Jose, California, and enjoys photography, gardening, and camping.

Data Entry and Communications Systems Have Network Capabilities

HP 2026 Systems are designed for high-performance data entry, local file inquiry, and data communications with each other and the HP 3000.

by John R. Nielsen and David S. Kaplan

THE HP 2026 SYSTEM is a 21MX-Computer-based system designed for high-performance data entry, local file inquiry, and data communications. It combines the 21MX E-series CPU hardware and a special high-performance operating system with flexible applications and utility packages to provide essentially a turnkey solution for distributed data capture and communications applications.

Originally developed for Hewlett-Packard's internal needs, the HP 2026 currently has an installed base of over 100 systems worldwide. The HP 2026 is well suited for medium to large manufacturing companies with dispersed factory and sales locations. These companies can use the system to capture all types of data and to communicate to other sites efficiently. Orders, acknowledgments, accounting data, and simple messages are typical types of data that can be entered and communicated using HP 2026 Systems.

Basic Capabilities

The HP 2026 System has five basic capabilities:

- Data entry
- Local file inquiry
- Data communication with HP computers
- Remote job entry to host computers
- Light local processing.

Data entry and local file inquiry operate under control of the foreground data entry program, which supports up to 16 HP 2645A CRT Terminals running multidrop operation at speeds up to 9600 bps. Operators can call up any data entry or file inquiry job from any terminal and enter data into formatted screens, request local file information by key values, or do a combination of the two. Data captured can then be placed in transaction files on disc for later transmission to remote sites or used to build or update local permanent files.

The system supports two memory partitions. Data entry runs in the foreground partition concurrently with any background program such as data transmission, remote job entry, report writer, and so on. Twenty-three background programs are currently supplied with the system (see Fig. 1).

All data entry or inquiry applications are designed using a special interactive program called DEAL (data entry applications language). DEAL allows creation of screen formats (including video enhancements, protected and unprotected fields, soft key loading, and so on), edit specifications for the various input fields, screen sequencing and repeat definitions, and record output formatting definitions if data is to be stored in transaction files. More than 40 preprogrammed edit instructions can be specified, including traditional edits (alpha, numeric, range check, check digit), arithmetic edits (add, subtract, multiply, divide), disc file edits (search, retrieve, update, add), data manipulation edits (data move, literal insert, formatting), and logical control edits (branch, branch and link, return, CRT display operations).

Through the use of these edits, essentially any type of data entry application can be developed. File search, retrieve, and update applications can be de-

DATEN	Multistation Data Entry/File Inquiry Routine
DEAL	Data Entry Applications Design Utility
TELCM	HP 2026 to HP 2026 Transmission Routine
DS	HP 2026 to HP 3000 Transmission Routine
MLRJE	IBM 360/370 Multileaving Workstation (HASP)
R2780	IBM 2780 Remote Job Entry Emulator
REPOR	Report Writer
REFUT	Data Selection and Reformatting Utility
SORT	Multikey Sort Utility
CSORT	Message Sort/Routing Utility
UTIL	Data Copy, Dump, Manipulation Utility
CFILE	Index Sequential File Build Utility
KFILE	Index Sequential Alternate Key Build Utility
UFILE	Index Sequential File Update Utility
FLMNG	Library File Management Utility
PRINT	File (Including IBM Standard Label) Print Utility
SELCT	Message Select, Print, and Punch Utility
EDITR	TWX/Telex Message Edit Utility
CD2MT	Card To Mag Tape/Disc Utility
CSAVE	System Backup/Recovery Utility
UPDAT	Software Update Utility
TIMES	Program Timing Statistics Print Utility
SYNCT	Phone Line Diagnostic Utility

Fig. 1. *The HP 2026 system is designed for high-performance data entry, local file inquiry, and data communications. Its standard software includes a special operating system and flexible applications and utility packages. Listed here are the major HP 2026 programs and utilities.*

veloped in a similar fashion. Using the binary search edit, any record can be found by its key value in any size file in less than 1/3 second. This file access capability provides powerful input field editing as well as fast search and retrieve functions.

Data communication between HP 2026s or to HP 3000s is the third basic capability of the system. Essentially, any data in any record format can be sent to any other HP 2026 or HP 3000 system either directly or via central concentrators.

The fourth basic capability is remote job entry (RJE) to host computers. IBM 2780 or IBM 360/370 multileaving (HASP) workstation emulators allow jobs to be submitted to IBM (or other compatible) host systems for processing, with output from these jobs routed back to the HP 2026.

The HP 2026 is also capable of light local processing using system utilities and the edit instructions. Reports from local disc files or transmitted data files can be produced by the HP 2026's report writer. Input record selection, row and column calculations, output formatting, sorting, and control breaks can be specified to produce desired reports. Specifications are produced using data entry formats and are permanently saved on disc for future use. Sequential disc file management utilities are provided for building and maintaining local disc files. Record sorting, reformatting, and printing can also be accomplished with standard utilities. Software updates of either data entry screen applications or actual program updates can be done without the need for systems personnel to link programs or generate operating systems. Updates are usually produced at central sites and then transmitted to remote sites where a utility program takes them and updates the local system as appropriate.

HP 2026 to HP 2026 Data Communications

Any HP 2026 can transmit data at speeds of up to 9600 bps to any other HP 2026 by means of the TELCM program. Either of two line protocols can be chosen, depending on line and modem requirements. Leased lines or voice-grade dial-up lines can be used without any software modifications or considerations.

There are two configurations of the TELCM program. The first employs an error signaling technique using the reverse channel band of a voice-grade circuit to signal transmission errors. This reverse channel band is typically in the range of 350 to 450 Hz, with a baud rate in the neighborhood of 150 bps. Data can be sent in one direction over this channel at the same time as data is sent in the other direction over the main channel. Thus on a half-duplex circuit, which is defined as allowing data to be sent in only one direction at a time, data can actually flow in both directions simultaneously by making use of this narrow reverse channel band.

This feature makes it possible to eliminate line turnarounds for error signaling and thus increase main channel data throughput. A full line turnaround typically takes 100 to 300 ms on land lines (depending on modems) and approximately 750 ms on satellite circuits. Elimination of these line turnarounds can result in a 20-40% throughput increase, depending on block sizes and circuit routing.

As Fig. 2 indicates, when using the reverse channel protocol, the next block of data is started immediately as soon as the current block is complete. As this next block is being sent, the reverse channel line is monitored for an error acknowledgment on the previous block of data. If a positive acknowledgment (change of state of the line from mark hold to space hold or vice versa) is detected during transmission of this block, the block continues to be sent. If a negative acknowledgment occurs, then sending of the current block is aborted and retransmission of the previous block (in error) begins immediately. Using this technique, data is continuously sent on the main channel without ever stopping to receive acknowledgments. Since most blocks are sent without errors, the absolute line throughput rate very closely approaches the modem signaling rate.

In addition to the throughput benefits of this protocol, a less obvious benefit results. Since the main channel is continuously sending data, equalization of modems and character synchronization is needed only once, at the start of the transmission. Resynchronization is needed only when block errors occur. Studies have indicated that most line errors occur just after modem equalization. Therefore, a protocol that turns the line around after each block for acknowledgments and then reequalizes the line is susceptible to this increase of errors while the reverse channel protocol is not.

The reverse channel protocol does have some specific requirements other protocols may not have. First, the modem must support the reverse channel feature. About half of the modems now available offer reverse channel as either a standard or an optional feature. Second, disabling of line echo suppressors is an absolute must for successful use of the protocol. Modems are supposed to disable all echo suppressors upon establishment of the line, but occasionally fail to do so. If all echo suppressors are not disabled, the reverse channel signal cannot be received by the sending station, and therefore detection of block acknowledgments is impossible and the transmission aborts.

Since all modems do not have the reverse channel feature and since in some countries none are available, an alternate configuration of TELCM is provided. This version employs a protocol similar to the binary

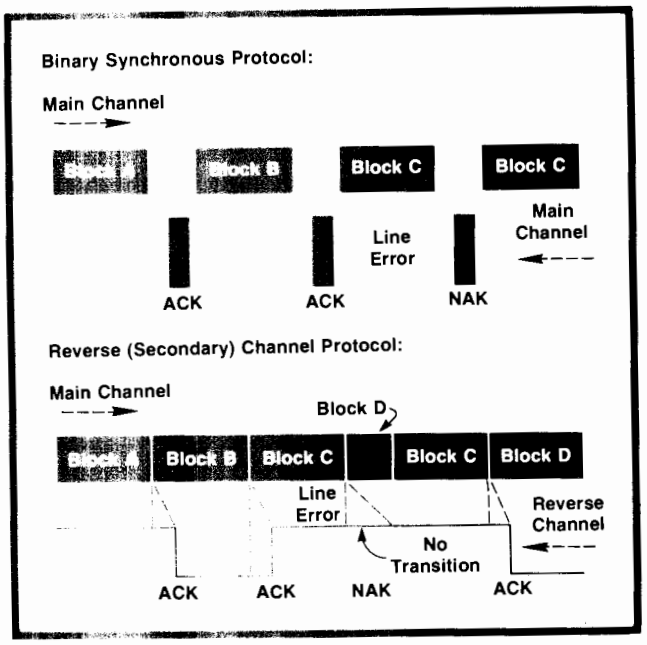


Fig. 2. *For HP 2026 to HP 2026 data communications, either of two line protocols can be chosen. The reverse channel protocol can be used if the modems support the reverse channel band of voice-grade circuits. By eliminating line turnarounds for acknowledgments, this protocol can increase throughput 20-40% over the binary synchronous protocol, which is used on degraded circuits or if no reverse channel is available.*

synchronous protocol shown in Fig. 2. Special timeout and modem equalization delays are employed, and extra block check characters are added. This makes this protocol especially effective on degraded circuits, those prone to high error rates, dropouts, and so on. This protocol is typically selected when no reverse channel modem is available or when line quality is poor and maximizing the chance of success is more important than throughput performance.

Data Compression/Expansion

Both configurations of TELCM use a special data compression/expansion algorithm that employs five different techniques. If the data for transmission is either ASCII or EBCDIC "text", then the following four techniques are used:

Blanks. Strings of blanks are replaced by a blank identifier followed by a binary count of the blanks replaced.

Numerics. Strings of numerics and associated special characters ($% −+.) are packed two per byte along with a numeric identifier and a binary count of the string length. Only four bits are necessary to represent the ten numeric digits and six extra special characters often found with numeric strings. Thus 2-to-1 (+2) compression is achieved on numeric strings.

Alphas. Strings of alpha text and associated special characters (". ?,-) are packed one character per five bits or three characters per sixteen-bit word along with an alpha identifier and a binary count of the string length. Thus 3-to-2 (+2) compression is achieved on alpha strings.

Headers. Identical start-of-record strings between sequentially transmitted records are replaced by a header identifier and a binary count of the number of matching characters. The start of one record often matches the start of the previous record, especially if routing information is used to begin each record.

If the data for transmission is binary (program object code, screen formats and edits, and so on), it probably contains a large number of repeated characters (especially binary zeros). Repeating-character compression gives the best results for this type of data. Therefore, the following compression technique is used for transmission of binary data files:

Repeating Characters. Strings of repeated characters are replaced by a repeat identifier, the repeated character, and the binary count of the repeated character string length.

The amount of data compression achieved on any file of data is highly dependent on the content of the data. Blocks with many blanks will compress most, while pure text will usually compress least. In general, a mean compression of about fifty percent with a variance of about fifteen percent can be expected for normal mixtures of data. When compression is combined with the reverse channel protocol, the effective throughput rate over the line is typically twice the absolute modem speed. In other words, a 4800-bps link will normally achieve a 9600-bps data throughput rate. This can result in considerable line cost savings, especially when long-distance dial-up lines are employed.

Distributed Systems/2026

DS/2026 is designed to provide two basic communications capabilities to users of HP 2026 and HP 3000 Computer Systems. The first is easy file transfer between the two systems. In addition, the HP 2026 system console can be used as a virtual HP 3000 terminal for remote command processing. Communication is via synchronous modems at speeds up to 9600 bps.

In designing the DS/2026 link to the HP 3000, four basic objectives were identified, two relating to operational costs and two to system use. As with the HP 2026 link using TELCM, minimization of direct communications costs, that is, the amount paid to common carriers for the data links, was desired. Also, to minimize the cost of system operation, maximum use of resources was required, both in the use of the data link and in the independent use of the two systems for other tasks. It was felt that the design of applications creating data to be transmitted should not be more

difficult on either system than the design of applications creating similar data files for local use. Finally, because the HP 2026 was designed for operation by technically unsophisticated personnel, the use of DS/2026 had to be straightforward.

To meet the communications cost goal, data transfers are in batch mode. In general, the data link is only up for file transfer when there is actually data to be sent. (One function, QSEND, allows the link to be idled while waiting for files to be queued on the HP 2026. It is anticipated that this function will be used for in-house communications only.)

Batch data transfer not only optimizes the use of the telephone link, but also considerably increases system use over interactive transfer. The CPU overhead of data communications on the HP 3000 is present only when useful work is being performed. More remote HP 2026 systems can be handled by one HP 3000 in this mode, and with lower cost for interfaces, modems, and telephones. On the HP 2026 side more time is available for other operations that need to be run concurrently with data entry, such as report printing, communications to other HP 2026s, or use of any of the other utilities provided on the system.

Because data is always transferred as files, applications programs creating data on either system can be scheduled to run without knowledge of the current data link status. HP 3000 jobs can create data for several remote systems in one run, regardless of which systems, if any, might actually be on line at the time. Probably more significant from the standpoint of efficiency, HP 2026 data entry destined for a central HP 3000 can be done at top speed and information retrieval can continue even if telephone problems temporarily halt the transfer of the data to or from the central system.

Because the data link is generally not active when files are being created, a means for applications programs to specify what files are to be sent, and where, was needed. Therefore, each system has a queue file containing the names of files to be transmitted. On the HP 3000, there is such a file in each remote HP 2026's group. An HP 3000 program can add a record to the particular remote system's queue file for each file created. Likewise, a data entry application on the HP 2026 can add a record to the HP 2026's queue file after a file is completed, or a data entry operator can run a standard HP 2026 program for general file queuing. If transmission is in progress when the queue entry is added, the file is sent on the same call. Otherwise, it is sent automatically the next time the link is established for transmission.

For the programmer, this queuing scheme provides a very easy means of converting an application program that creates a tape for the remote system to one that transmits the file directly from disc. On the HP 3000, a step is added to the job stream to run a program to write the file's name to the appropriate queue file. On the HP 2026, a few instructions are added to the DEAL program to call a system subroutine that adds the desired file name to the queue.

Queue files are particularly helpful to the HP 2026 system operator, who doesn't need to know anything about the files to be transmitted. Aside from program startup, sending queued files requires just one console command and receiving queued files requires one command plus designation of the I/O device to be used. After the DS/2026 program is initiated, all that remains is for the link to be established by one operator's dialing the other's telephone number. Under some circumstances the HP 2026 operator may want to specify specific files to be sent or received, and functions are provided to do so. A complete list of DS/2026 functions appears in Fig. 3.

One function combines the operation of sending queued and/or operator-specified files with that of receiving queued files. Before leaving for the night, the HP 2026 system operator can use this function to set up the system to send a day's work and then receive any HP 3000 files that are ready. The HP 3000 operator can then establish the link when convenient. After sending and receiving, the HP 2026 program drops the data link, then sets up to receive another call so the HP 3000 operator can have data sent throughout the night. As with all file transfer functions, file information is displayed on the HP 2026 console. Also like the other functions, abortive transfers caused by telephone line problems are recovered from automatically, with reestablishment of the link the only manual operation.

To make the operation of DS/2026 as easy as possible for the HP 2026 operator, the program commands are similar to those for TELCM. An operator who is familiar with data communication to either an HP 2026 or an HP 3000 can quickly learn to do communications with the other. So the operator does not have to type the generally constant log-on information repeatedly, the normally used HP 3000 HELLO command can be stored in a disc file to be used automatically. This default can be overridden on any use of the system.

Use of DS/2026 to provide remote HP 3000 terminal capabilities (interactive mode), of course, requires more expertise than the file transfer mode. The 2026 operator must be familiar with HP 3000 commands, subsystems, and programs. Once the DS/2026 command specifying interactive mode has been entered and the data link has been established, operation is essentially the same as a local HP 3000 session, except that the prompt character is the DS/3000 remote session prompt # instead of :. All HP 3000 terminal capabilities available to a DS/3000 remote session are

RECV	Receive an Operator-Specified File from an HP 3000
ARECV	Receive Files Queued on the HP 3000, Drop Link When Done, Then Set Up for Another Call
QSEND	Send Queued Files to an HP 3000, Waiting for More to Be Queued When Done
MSEND	Send Operator Specified and/or Queued Files to HP 3000, Drop Link When Done
MS2AR	Send Operator-Specified and/or Queued Files to the HP 3000, Receive Queued Files from the HP 3000, Drop Link When Done, Then Revert to ARECV
INTER	Use HP 2026 Console as a Remote Virtual HP 3000 Terminal

All of the commands that initiate file transfer may specify that interactive mode is to be entered when done instead of dropping the link.

Fig. 3. *Distributed Systems/2026 provides for file transfer between HP 2026 and HP 3000 systems, and for the HP 2026 to act as a virtual HP 3000 terminal for remote command processing. Listed here are DS/2026 functions.*

available on the HP 2026 as well.

DS/3000 provides two basic means of transferring files from one computer to another: remote file access (RFA) and program-to-program communications (PTOPC). PTOPC is used for the DS/2026/3000 link for both communications efficiency and design consistency. With the use of PTOPC, the programs determine the format of data. Therefore, records can be put into fixed-length blocks for transmission, regardless of actual record lengths and formats. With records that are short compared to the line buffer size, significant savings can be realized by such blocking, because of the reduction of line turnarounds and their associated delays. Overlapping data link I/O with file I/O on each system is easier with PTOPC than with RFA, yielding additional communications savings. Also program logic is more straightforword with a program running on each machine, neither having to be concerned with the details of the file system on the other computer.

Remote Job Entry

Remote job entry (RJE) is the use of a terminal or computer to send jobs to a host computer for batch processing and to subsequently receive any program output. Some RJE implementations also provide for sending commands to the host system and for receiving the host's replies.

The HP 2026 supplies two RJE emulation programs, MLRJE, an IBM multileaving RJE workstation emulator (sometimes referred to as HASP), and R2780, an IBM 2780 terminal emulator. Both operate with synchronous modems at up to 9600 bps. MLRJE, the more powerful of the two, can concurrently transmit jobs, receive job output, and send and receive console commands and messages. Jobs may be sent from a card reader, magnetic tape, disc files, or any combination of the three. As with DS/2026, disc files can be queued for transmission at a later time. Job output may be received on a line printer, on magnetic tape, or both simultaneously. To speed transmission, repeat characters are compressed and records are blocked. Because its capabilities give the HP 2026 operator most of the capabilities of a host system operator, and because of the operational convenience and speed attained with concurrent input and output, MLRJE is used whenever the host system supports remote IBM 360/370 multileaving workstations. Such hosts include the IBM 360 and 370 running under the MFT/OS, MVT/OS, OS/VS1, or OS/VS2 operating systems using HASP, ASP, JES1, JES2, or JES3.

Because not all host systems support multileaving RJE, the IBM 2780 emulator is supplied. 2780 terminals are supported by most IBM mainframes that support communications, and by some other companies' mainframes as well. The emulator also allows general data transfer with other 2780 emulators, including the HP 3000 and HP 1000 systems. The basic RJE functions of a multileaving workstation are also available on a 2780, but only one function can be performed at a time and data compression is not performed. R2780 supports input from the card reader or

David S. Kaplan
David Kaplan joined HP in 1973, just after receiving his SB degree in mathematics from the University of Chicago. He's been involved with software development for the 2026 and its forerunner, HP's in-house COMSYS. He's a member of the Telecommunications Association. David was born in Chicago. He's single and now lives in Santa Clara, California. He's on the board of directors at his synagogue and enjoys gourmet cooking.

John Richard Nielsen
Rich Nielsen is software manager for the 2026 and HP's in-house COMSYS. He had been a programmer analyst for COMSYS since joining HP in 1970, just after receiving his BS degree in statistics from Stanford University. He's a member of the Telecommunications Association. Rich is a California product, born in Berkeley and now living in Los Altos. He's married and has two daughters. A former professional bowler, he still enjoys bowling, but also plays golf, tennis, and bridge.

magnetic tape, and output to the line printer or magnetic tape, or both simultaneously. Unlike a 2780, it also allows command input from the system console and not just from the card reader.

Acknowledgments

Many people have contributed to the HP 2026's data communications development. Most of the software development has been done by the authors of this article. Some additional software development has been provided by Ron Niedrauer. Terry Eastham, Bill Taylor, and John Southworth have made major contributions to network analysis, while Gene Doucette has contributed to modem testing and interfacing.

SPECIFICATIONS
Distributed Systems/3000

MINIMUM SYSTEM: An HP 3000 Series II operating under MPE-II with 192K bytes of memory.
SOFTWARE: HP 32190A. HP 3000 software necessary for data communications with another HP 3000, HP 1000, and/or HP 2026.
HARDWARE: Serial Data Communication Hardware for Hardwired and Modem Connections.
HP 300055A Synchronous Single Line Controller for modem links up to 9600 bits per second.
HP 30360A Hardwired Serial Interface (for coaxial links).

HP 30220A/Coaxial Cable

OPTIONS	LENGTH (feet)
Standard	25
001	100
002	250
003	500
004	1000
005	2000

CUSTOMER TRAINING: HP 36900E. DS/3000 On-Site Training
PRICES IN U.S.A.: 32190A, $3,000; 30055A, $2,000; 30360A, $2,300; 30220A Standard, $175; Option 001, $200; Option 002, $225; Option 003, $430; Option 004, $825; Option 005, $1,600; 36900E, $3,000.
MANUFACTURING DIVISION: GENERAL SYSTEMS DIVISION
5303 Stevens Creek Boulevard
Santa Clara, California 95050 U.S.A.

Summary of User Network Services Available

DS/3000 Network Services Initiated from a Local HP 3000 to:	HP 3000	HP 2026	HP 1000
Virtual Terminal	•		
Remote Command Processing	•		•
Remote File Access	•		•
Remote Data Base Access	•		
Program-to-Program Communication	•	*	•
Peripheral Sharing	•		•
Bidirectional Interleaving	•		•
Data Compression	•		

DS/3000 Network Services Initiated to a Remote HP 3000 from:	HP 3000	HP 2028	HP 1000
Virtual Terminal	•	•	•
Remote Command Processing	•	•	•
Remote File Access	•		•
Remote Data Base Access	•		
Program-to-Program Communication	•	*	•
Peripheral Sharing	•		•
Bidirectional Interleaving	•		•
Data Compression	•		

*Batch File Transfer initiation available from HP 2026 only using PTOPC.

SPECIFICATIONS
Distributed Systems/1000

BASE SYSTEMS: HP 1000 Model 20/21 System
HP 1000 Model 30/31 System
HP 1000 Model 80/81 System
HP 21MX M-Series Computer system components
HP 21MX E-Series Computer system components
OPERATING SYSTEMS: Minimum RTE-M. 64K-byte memory based system (RTE-MII)
Mapped RTE-M. 128K-byte memory based system (RTE-MIII)
RTE-III. 128K-byte disc based system.
DS/1000 SOFTWARE/FIRMWARE:
91740A/B DS/1000 Network Software/Firmware. 91740A is for 21MX M-Series Computers. 91740B is for 21MX E-Series Computers and HP 1000 Computer Systems.
91741A DS/1000 Software Enhancement for HP 1000 to HP 3000 Communications.
DS/1000 HARDWARE:
12620A Privileged Interrupt Fence. One per system required with 12773A interface(s). Optional with 12771A interface(s).
12771A Computer Serial Interface Kit
91720A/21A Communications Cable (optional with 12771A).
12773A Computer Modem Interface
12889A Hardwired Serial Interface
30220A Coaxial Cable Kit (required with 12889A).
PRICES IN U.S.A.: 91740A/B DS/1000 Network Software/Firmware, $2500.
91741A DS/1000 Software Enhancement for HP 1000 to HP 3000 Communication, $500.
MANUFACTURING DIVISION: DATA SYSTEMS DIVISION
11000 Wolfe Road
Cupertino, California 95014 U.S.A.

SPECIFICATIONS
HP 2026 Data Entry/Communications System

BASE SYSTEM: 2113B 21MX E-Series Computer with 65,536 bytes of semiconductor memory, 128 instructions, 14 I/O channels.
12618A Synchronous Communications Interface
12962D Cartridge Disc Subsystem
14.7M bytes, HP 7905A Disc Drive
2645A Display Station as system console
SYSTEM OPTIONS: Replace 14.7M-byte cartridge disc subsystem with 50M-byte Disc Drive (HP 7920A).
12962C Cartridge Disc Subsystem in upright cabinet with space for optional magnetic tape unit.
MAXIMUM CONFIGURATION: Up to 16 HP 2645A data entry stations multidropped on up to four lines. Buffered asynchronous communication with system at 9600 bps (local), or 1200 bps with modem links. Two 12970A (800 bpi) or 12972A (1600 bpi) Magnetic Tape Subsystems. Eight 14.7M-byte Cartridge Disc Subsystems (HP 7905A) or six 50M-byte Disc Drives (HP 7920A). One Line Printer (200, 300, 600 or 1250 LPM). One Card Reader (600 CPM). One HP 2748B Punched Tape Reader Subsystem (500 cps). One HP 2895B Punched Tape Subsystem (75 cps).
PRICE IN U.S.A.: 2026 Base System, $38,500.
MANUFACTURING DIVISION: GENERAL SYSTEMS DIVISION
5303 Stevens Creek Boulevard
Santa Clara, California 95050 U.S.A.

Experimenting with Satellite-Linked Computer Networks

Project Prelude is an advanced computer communications experiment involving several companies, a satellite, and HP 3000 Series II Computer Systems.

by Rita W. Williams

IN RECENT MONTHS Distributed Systems/3000 has been an integral part of a series of experiments designed to stimulate discussion as to how to meet intracompany communication needs in the next decade. The project has been using the world's most powerful communications satellite in orbit, the Communications Technology Satellite (CTS). The CTS is a joint effort of the United States National Aeronautics and Space Administration (NASA) and the Canadian government's Department of Communications. It is in geosynchronous orbit at an altitude of 22,300 miles.

Various experiments have been sponsored to explore possible uses of the technology offered by satellite communications. Experiment #26 on the CTS has been conducted by Satellite Business Systems, an offspring of three companies: IBM, Aetna Life and Casualty, and COMSAT General. The objective of this experiment, called Project Prelude, is to test and evaluate satellite communications and manufacturers' advanced equipment by assembling these elements for the first time in a realistic business environment. The major feature of this experiment is the variety of high-speed communications applications that make use of satellite communications—full-motion and freeze-frame video, facsimile, and data processing.

Satellite Business Systems (SBS) has assembled a communications mini-network with the cooperation and participation of other companies: Hewlett-Packard, Advent, NEC America, Ikegami Electronics, Harris Communications, Arvin/Echo, Dacom, Rapifax Corporation, and Comsat Laboratories. The participating companies have provided equipment, training, and/or personnel to Project Prelude.

The Project Prelude network has been demonstrated between Seal Beach, California and Pittsburgh, Pennsylvania, between Houston, Texas and Harrison, New York, and between Chicago, Illinois and Catonsville, Maryland. Three companies, Rockwell International, Texaco, and Montgomery Ward, have each hosted the Project Prelude presentations for approximately a month. In that time, employees of the host companies viewed formal presentations and explored applications of their own design to use the satellite communication link. Many of the impromptu uses of the satellite link involved video-conferencing to discuss matters of immediate concern to managements of the host companies. Another impromptu use of the satellite link was to transmit seismic data. In that instance, data was written to magnetic tape on a CDC computer system, carried to an HP 3000 in Texas, transmitted via satellite to an HP 3000 in New York, and transmitted back again. Comparison of the data transmitted and received in Texas verified the integrity of the data.

Project Prelude Communication Link Hardware

Hewlett-Packard provided the following equipment to each of the two sites in the Project Prelude mini-network: an HP 3000 Series II Model 6 System, three HP 2645A Terminals, an HP 2648A Graphics Terminal, and an HP 2617A Line Printer. Equipment involved in the data link between each HP 3000 and the satellite included a modified hardwired serial interface communications controller and adapter board in the HP 3000, a master control unit to accomplish switching between data processing and video and facsimile, a Harris high-speed digital modem, two converters, an orthomode coupler or feed, and an eight-foot dish antenna (see Fig. 1). The Harris modems are multi-data-rate units with QPSK/BPSK modulation. The double up-conversion on transmission consisted of one RF conversion to 1.5 GHz and a second conversion to 14 GHz. To facilitate transportation from site to site, the antennas and other earth station equipment were mounted on trailers and stationed just outside the buildings where demonstrations were held (Fig. 2). The earth stations, which included the master control unit (MCU), the converters, the feed, and the antenna, were developed by the laboratories of COMSAT (Communications Satellite Corporation).

The first step in HP's participation in this experiment was to form an engineering team to determine the most appropriate interface to the communication

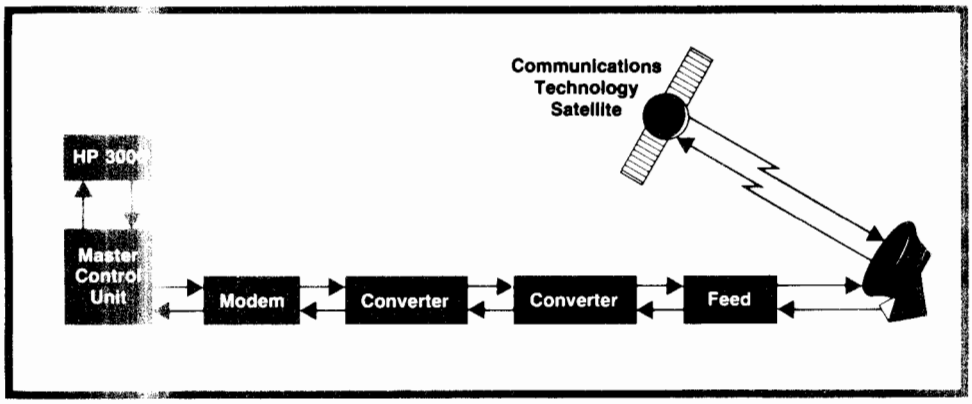


Fig. 1. *Project Prelude is an experiment in computer communications via satellite. The computers are two HP 3000 Series II Systems, one at each end of the communications link. Between each HP 3000 and the satellite is the equipment shown here. Applications include video and audio transmission, facsimile, and data processing.*

equipment involved in the data link. DS/3000 can use either of two communications controllers—hardwired serial interface (HSI) or synchronous single-line controller (SSLC). The SSLC is used in a modem link, but its highest supported speed is 9.6 kilobits per second. The HSI is used when HP 3000s are linked by coaxial cable and can run at 2.5 megabits per second when cable length is limited to less than 1000 feet. Because of its higher speed, the hardwired serial interface communications controller was selected.

The output drivers and receivers on one channel of an HSI were replaced with sockets with jumper plugs to connect input and output pins. An adapter board was designed to convert the differential signals received from the modem to TTL levels for input to the modified HSI and to convert the outgoing TTL clock and data signals to differential signals to drive the modem. Normally, the HSI provides its own clock, but in this application the clock was provided by the modem. During testing at COMSAT Laboratories in Clarksburg, Maryland, it was verified that the HP equipment operated reliably at the 2.5-megabit rate.

System and Communications Software

While hardware development was going on, software tests were run, simulating the timing expected in satellite communications. Because the satellite is in orbit at an altitude of 22,300 miles, the data transmitted from one HP 3000 to another must travel approximately 44,600 miles. Even at the speed of light, this transmission takes approximately 0.24 second. But the transmission is not really complete until the receiving system acknowledges to the sending system that the transmission was received correctly. Because the acknowledgment must also travel 44,600 miles, a completed data transmission requires a minimum of 0.48 second, possibly longer if an error condition requires a retransmission of the original data. These tests showed that the standard HP 3000 Series II communication and operating system software continued to function reliably in spite of the delay.

An optimization in data transfer between systems was accomplished by eliminating a series of handshakes meaningful only in communication between an HP 3000 and an HP 1000. These were unnecessary for this application.

Software Design Considerations

The influence of the link propagation delay on visible performance was a major concern in the design and implementation of demonstrations for Project Prelude. The most efficient use of the communication link was by means of DS/3000 program-to-program communication (PTOPC). At this level, the user has more control over the size of the buffer used in the transfer of data from one system to another. Program-to-program communication also allows the programmer to control blocking and unblocking of information transferred. Thus it is possible for a communications buffer to contain multiple records of a file, or even multiple files. Given a file of 100 records of 80 bytes each, the standard HP 3000 utility program FCOPY might require 100 transmissions and acknowledgments to copy the file from one system to another,

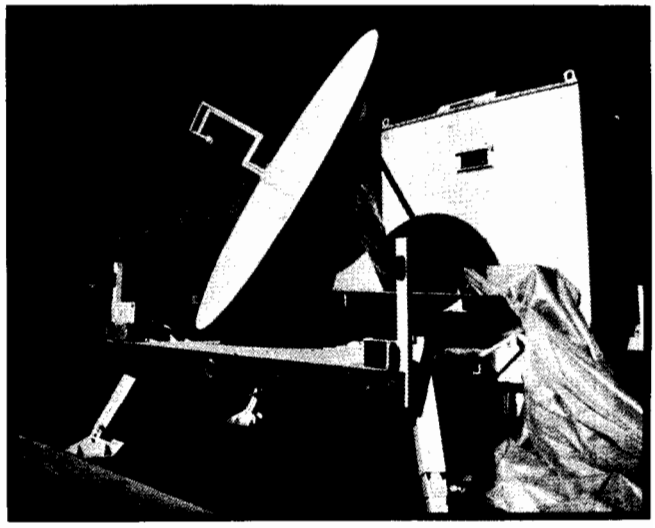


Fig. 2. *Project Prelude antennas and earth station equipment were mounted on trailers to facilitate movement between sites.*

while program-to-program communication can copy the file with only one transmission and acknowledgment, a significant reduction in communication overhead.

For maintenance purposes, each of the systems involved in the Project Prelude demonstrations was equipped with two sets of HSI and adapter boards. Although each set of communications boards would typically be identified by a different class name in the system configuration, this is not required. To enable the PTOPC master program to reference either device, both devices were given the same class name, and during the demonstrations, communication was enabled on only one device.

The need to access local and remote data bases was a fundamental requirement of the Project Prelude programs. The new DS/3000 extensions to IMAGE, HP's data base management software, simplify access to remote data bases and undoubtedly would have simplified the Project Prelude programming tasks. However these extensions (see article, page 7) were in the design phase during development of the Project Prelude demonstrations. Nevertheless, the then-current software was adequate.

Part of the demonstrations consisted of the creation of a data base of registration information about the persons in attendance. Each site in the two-system network had a local data base of such information. In some of the demonstrations it was desirable to interchange data bases so that each site had its own data base and the sister site's data base. By taking advantage of the accounting structure in MPE, the HP 3000 operating system, one identifying file name, AREG, could designate any of the four data bases in use, thus minimizing the number of programs needed to carry out the demonstrations. For example, the data base in Texas describing local attendees was AREG.LOCAL.DEMO1, while the Texas data base that described attendees at the sister site was AREG.REMOTE.DEMO1. The contents of Texas' AREG.LOCAL.DEMO1 were identical to New York's AREG.REMOTE.DEMO1, and Texas' AREG.REMOTE.DEMO1 was identical to New York's AREG.LOCAL.DEMO1. When accessing a particular data base, one merely logged on to the local and/or remote systems in the appropriate group and account and let MPE find the intended data base.

Project Prelude Demonstrations

Three different formal presentations were designed, each with a certain audience in mind. One was geared to an audience of data processing personnel, the second to an audience of administrative services personnel, and the third to a more general audience. Each of the three presentations was designed to demonstrate audio, video, facsimile, and data processing via the satellite communication link and was scheduled to last approximately one and one-half hours. A data base of attendees was common to all three demonstrations. Ultimately this data base was used to generate a personally addressed letter to each attendee in the data base.

The data processing demonstration required by far the most HP participation. The following paragraphs describe the software developed for this demonstration.

The demonstration begins with a six-step data base access operation (see Fig. 3). The first step is to enter the names of attendees into the data base via a data entry library screen. A COBOL program reads the unprotected fields of the screen and enters the information into the data base. This data base initialization occurs at both sites before the formal presentation begins.

The second step illustrates resource sharing, the resource in this case being the newly created data bases. This is done by entering a limited number of attendees into the sister site's local data base. These individuals can be late arrivals who write registration information on a sheet of paper. This sheet can then be sent by facsimile to the sister site. For example, the registration data sheet for late attendees in Texas can be transmitted to New York. New York then adds this data to the data base residing on the Texas system. To accomplish this step, the original data entry program was modified to place information from the unprotected fields into a buffer and write the buffer across the line to a slave program that adds to the data base.

The third step illustrates high-speed data transfer by copying the data base from the sister site. This is accomplished by master and slave programs passing information via PTOPC intrinsics such as PREAD. After this step, each site has two data bases, AREG.LOCAL and AREG.REMOTE, the sister site back-up.

The fourth and fifth steps illustrate this back-up feature. Each site deliberately purges its own local data base. Normally, this would be cause for great concern, but in this case each site has a readily obtainable back-up data base elsewhere in the mini-network. The back-up data bases are accessed to restore the "lost" information, and each site again has both data bases.

The sixth step is to add another attendee to a local data base, and add the necessary information to the back-up data base as well. The program used earlier to add to the sister site's data base, UPDATEP, is run again. This program asks the operator whether the information should be added to the local data base and whether the back-up data base should be updated. In this step, the operator answers yes to both questions. Appropriate messages are returned to the terminal to indicate error conditions or completion of the slave and master programs.

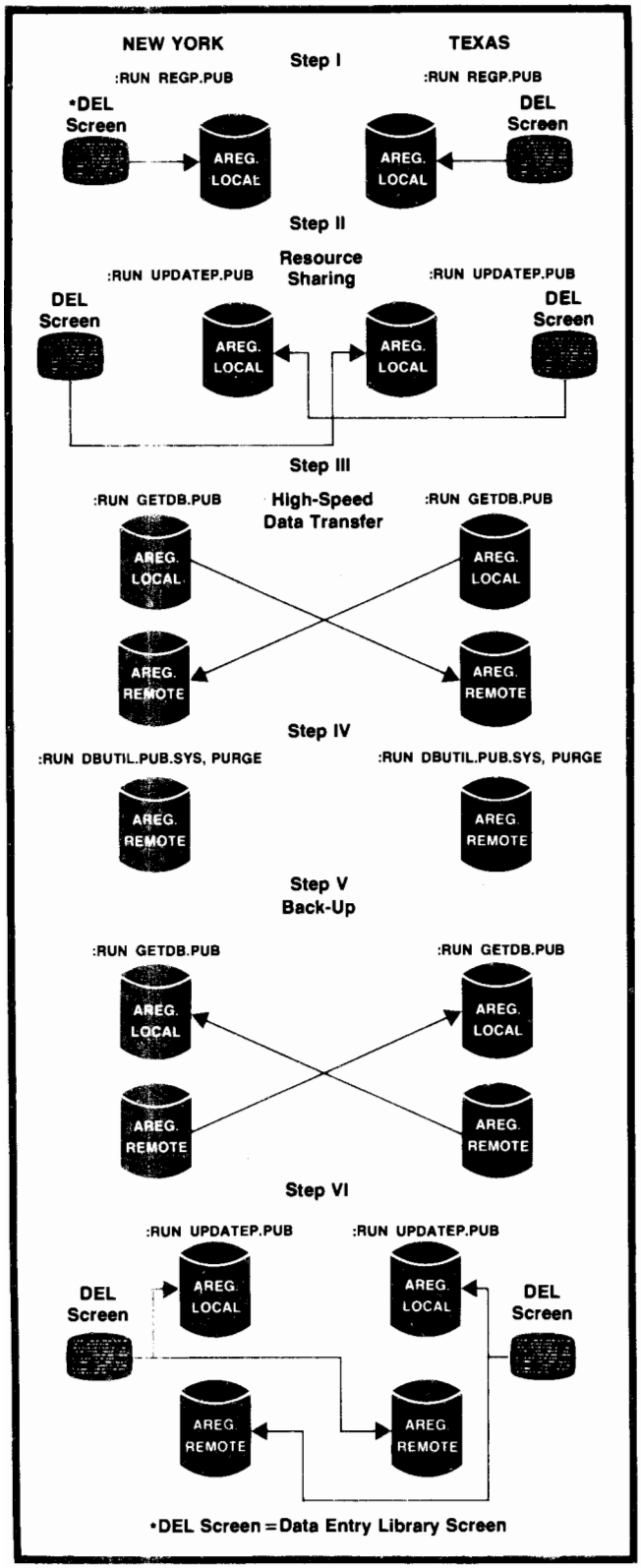


Fig. 3. *Data flow in the data processing demonstration, one of the Project Prelude experiments. The six-step demonstration involves creating local and remote data bases, updating and copying remote data bases, and replacing a purged local data base by copying a back-up data base from the remote site.*

The next major sequence in the data processing demonstration is an information retrieval operation that highlights distributed processing capabilities. A data entry library screen (Fig. 4) permits selection of any or all of ten items of interest. Because this application is based on an information network, some of these items are stored in a local computer and some in a remote computer. Some articles are not computer-based, but must be delivered in hard-copy form, either locally or by facsimile.

The method of displaying or delivering each item is specified in a compound statement within the framework of a CASE statement. The selection of items supplies indexes into the CASE statement.

The contents of the unprotected fields of the screen are read into a ten-word array by the program DOCDISTP. Each word is examined in turn to determine the items selected. If the contents of the word are alphanumeric, the position of the word relative to the beginning of the ten-word array is used as an index into the CASE statement. The structure of the statement is:

```
CASE X OF
    BEGIN          <<CASE 0          >>
      .            <<ITEM 1 ON SCREEN>>
      .
    END;

    BEGIN          <<CASE 1          >>
      .            <<ITEM 2 ON SCREEN>>
      .
    END;
      .
      .
```

When X, the index into the CASE statement, is less than four, execution of the CASE statement will display cumulative sales, expenses, and profits in tabular form. After a carriage return is entered from the terminal, the same data is displayed in graphic form. In two cases, the financial data is stored in the remote system and must be requested by the local program.

Cases four through seven place requests for hard-copy items. This request is displayed at the console and a confirmation is delivered to the operator at the originating terminal. If the item is not available locally, the request must be delivered to the console of the remote system. The slave program, running on the remote system, confirms the request to both the information analyst at the master console and the originating terminal.

Cases eight and nine display articles stored on disc. The operator may choose whether to display these on the originating terminal or on the line printer. If the operator chooses to display the article on the terminal, the article remains on the screen for study until a

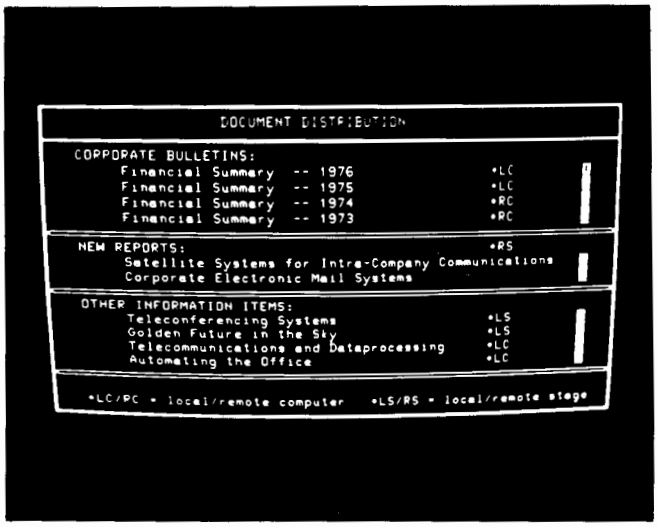


Fig. 4. *Data entry library screen permits selection of any or all of ten items in an information retrieval portion of the Project Prelude data processing demonstration. Some of the items are stored in the local computer system, some are in the remote system, and some are delivered in document form and transmitted by facsimile.*

carriage return is entered.

The last step in the demonstration features a limited version of the broadcast capability of satellite communications: a message from one site can be broadcast to multiple sites within the same network. Since the Project Prelude network is limited to two sites, broadcasting is simulated by generating an individually addressed letter to each attendee registered in the data bases. A skeleton of the letter is as follows:

I

January 5, 1978

Dear Leslie Jones
Chicago, Illinois
Sister Site: Catonsville, Maryland

Thank you for attending and participating in the Project Prelude experiment.

.
.
.

You may, therefore, find it appropriate to discuss this presentation with associates of your company also attending this session. The following list is provided for this purpose, and includes the individual name and company location.

II

Leslie Jones — Francis Smith
Dave White

III

Catonsville, Maryland:
Mary Moore — Matt Brown

The program that creates this letter is called DPLETP. It can supply text from any interactively designated text file and can be run in one of two modes: to generate letters to all local attendees in the data base or to five attendees designated interactively.

In Part I of the letter, the date and sites are read from the specified text file; the name is supplied via a sequential READ on the detail set of the data base or a calculated READ for a specific name. In Part II are the names of all attendees who work for the same company and are present at the same site. Part III contains the names of fellow employees in attendance at the sister site. The output procedure continues to loop until all desired letters have been output.

Generating letters to five designated attendees may involve outputting a letter on the line printer on the remote system. If five names are supplied as addressees and one of the names is not in the master's local data base, that name is passed to a slave program. If the name is found in the slave's data base, the slave program generates parts I and II of the letter in much the same way as the master program.

However, the third data base access must be handled differently, because now the slave program must request data and essentially control communications, in contrast to normal program-to-program communications. The request for data is more direct from a master program than from a slave program. The master's request proceeds as follows.

Master		Slave
LENGTH:=PREAD(...);	[Request Data]	GET(...);
	[Receive Data]	ACCEPT(...);

For a slave to request data, two master PTOPC operations are required. The ITAG, or tag field, construct in PTOPC is especially useful here. Word 0 of the 20-word ITAG can be used as a signal to the master to send data to the slave. The other nineteen words in ITAG can specify the information needed. The slave's request might proceed as follows.

Master	Slave	
PREAD,PWRITE, or PCONTROL	GET(...);	
	ACCEPT(...);	[Request Data]
IF ITAG=1, THEN PWRITE(...);	GET(...);	
	ACCEPT(...);	[Receive Data]

An alternate method for the slave to request data from the master is for the slave to respond with a REJECT instead of an ACCEPT. The logic in the master program must be able to interpret this response and produce the necessary data for transfer to the slave.

Summary

The data processing demonstration consists of a series of programs that spotlight data communications. Resource sharing, high-speed data transfer, back-up, broadcast, and distributed processing represent networking applications that can be implemented effectively using satellite technology. Using an HP 3000 Series II and off-the-shelf software products such as IMAGE and DS/3000, any user can develop similar applications to span a wide range of communication environments and data processing requirements.

Acknowledgments

Larry Kelly, Lloyd Summers, and Elio Toschi were responsible for the hardwired serial interface and adapter for the HP 3000s used in Project Prelude. The initial design of the HSI modification was by Kent Simcoe. Howard Morris did the software simulations of satellite timing. Jutta Kernke designed two data entry library screens for the demonstrations. Nadine Halsted wrote the program to read the information from the screens and enter it into the data base. Don Van Pernis wrote CFCOPY, a PTOPC program that copies files from one system to another.

HP's participation in this project was not limited to factory personnel. Field system engineers conducted the demonstrations and contributed to software development. Customer engineers and operating system specialists aided in the installation and maintenance of these systems as they toured the country. The contributions of field personnel were crucial to our successful role in the project.

Rita W. Williams

Rita Williams was responsible for technical liaison with SBS and COMSAT General during HP's participation in Project Prelude. Her duties included software design, training of field personnel, and assisting with network integration. She's been with HP since 1976, serving as a product-support software specialist. Born in Huntingburg, Indiana, Rita received her BS degree in psychology from Indiana University in 1968. After a few years in retail sales and contract and grant administration, she was bitten by the computer bug, and received her MS degree in computer science from San Jose State University in 1977. Her other interests include playing piano, birdwatching, reading, and real estate. She has two sons and lives in San Jose, California.

Hewlett-Packard Company, 1501 Page Mill Road, Palo Alto, California 94304

HEWLETT-PACKARD JOURNAL

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

CHANGE OF ADDRESS. To change your address or delete your name from our mailing list please send us your old address label (it peels off). Send changes to Hewlett-Packard Journal, 1501 Page Mill Road, Palo Alto, California 94304 U.S.A. Allow 60 days.