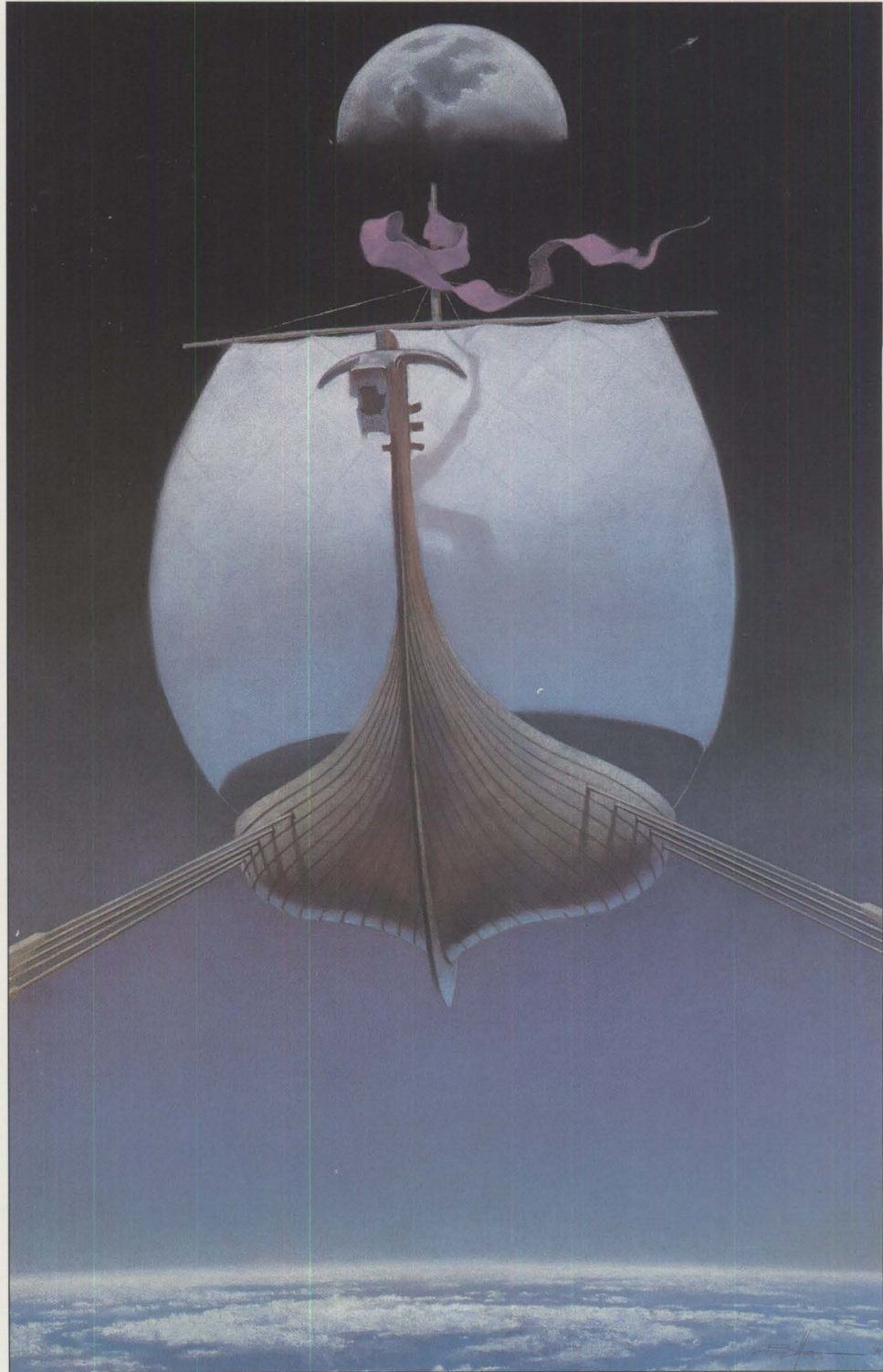


Issue 4, 1991

# Personal Systems



IBM Personal Systems Technical Solutions



\$10.00

*IBM Personal Systems Technical Solutions* is published quarterly by IBM United States technical support center, International Business Machines Corporation, Roanoke, Texas, U.S.A.

Editor	Libby Boyd
Consulting Editor	Ed Bamberger
Communications	Elisa Davis
Design	Corporate Graphics
Illustrator	Bill Carr
Manager	Bill Hawkins

To correspond with *IBM Personal Systems Technical Solutions*, please write the editor at:

IBM Corporation  
Internal Zip 40-A2-04  
One East Kirkwood Blvd.  
Roanoke, TX 76299-0015

To subscribe to this publication, request IBM System Library Subscription Service (SLSS) from an IBM branch, and specify form number GBOF-1229.

Copying or reprinting material from this magazine is strictly prohibited without the written permission of the editor. Titles and abstracts, but no other portions, of information in this publication may be copied and distributed by computer-based and other information-service systems.

IBM believes the statements contained herein are accurate as of the date of publication of this document. However, IBM hereby disclaims all warranties as to materials and workmanship, either expressed or implied, including without limitation any implied warranty of merchantability or fitness for a particular purpose. In no event will IBM be liable to you for any damages, including any lost profits, lost savings or other incidental or consequential damage arising out of the use or inability to use any information provided through this service even if IBM has been advised of the possibility of such damages, or for any claim by any other party.

Some states do not allow the limitation or exclusion of liability for incidental or consequential damages so the above limitation or exclusion may not apply to you.

This publication could contain technical inaccuracies or typographical errors. Also, illustrations contained herein may show prototype equipment. Your system configuration may differ slightly.

IBM has tested the programs contained in this publication. However, IBM does not guarantee that the programs contain no errors.

This information is not intended to be a statement of direction or an assertion of future

action. IBM expressly reserves the right to change or withdraw current products that may or may not have the same characteristics or codes listed in this publication. Should IBM modify its products in a way that may affect the information contained in this publication, IBM assumes no obligation whatever to inform any user of the modifications.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not imply giving license to these patents.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such products, programming or services in your country.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever.

All specifications are subject to change without notice.

© Copyright 1991 by International Business Machines Corporation

# Contents

*OUR COVER: Nine hundred years ago, the men and women of a small Viking ship, led by Leif Eriksson, pushed their knowledge and technology to the limits. Their courage and skill landed them on the shores of North America four hundred years before Christopher Columbus. Today, the men and women of IBM continue this tradition of exploration and leadership, keeping IBM at the frontier of computing technology.*

## Hardware

- 1** Power Factor: Non-Linear Loads and the Power Distribution System

## Software

- 6** Database Manager: Highlights and Direction
- 13** OS/2® Communications Manager™
- 23** Improving OS/2 Application Performance
- 33** Creating PM Windows with Dialog Templates
- 38** REXX Program for OS/2 LAN Server
- 49** Micro Focus® COBOL/2 and the DOS Database Requester
- 54** IBM DOS 5.0 Facts and Features
- 61** IBM DOS 5.0 Upgrade
- 65** DOS 5.0 Performance Improvements
- 70** DOS Memory Management Facilities
- 74** Disk Caching Under DOS
- 77** NetWare® Client-Server Interaction
- 89** LANACS Protocols

## Random Data

- 95** New Book: *Client-Server Programming with OS/2 Extended Edition*
- 96** Little Solutions
- 99** New Products

## **Trademarks**

IBM, AIX, Audio Visual Connection, Micro Channel, Operating System/2, OS/2, Personal Computer AT, AT, Personal System/2, PS/2, PROFS, and RT are registered trademarks of International Business Machines Corp.

AIXwindows, Application System/400, AS/400, AVC, C/2, COBOL/2, Communications Manager, DB2, ETHERAND, FORTRAN/2, OS/400, NetView, Pascal/2, Personal Computer XT, PC XT, PS/1, Presentation Manager, QMF, RISC System/6000, SQL, SQL/DS, Systems Application Architecture, SAA, SNA, System/370, System/390, VTAM, and 3090 are trademarks of International Business Machines Corp.

ARCserve is a registered trademark of Cheyenne Software, Inc.

PostScript is a registered trademark of Adobe Systems Inc.

Intel is a registered trademark, and i860, 386, i486, 486, 80286, 80386, 80386SX, and 80486 are trademarks of Intel Corp.

Microsoft and MS-DOS are registered trademarks, and Windows is a trademark of Microsoft Corp.

Micro Focus is a registered trademark of Micro Focus Limited.

Novell and NetWare are registered trademarks of Novell Inc.

OSF and Open Software Foundation are trademarks of Open Software Foundation Inc.

Sytron Plus is a registered trademark of Sytron Corp.

Xerox and Ethernet are registered trademarks of Xerox Inc.

X Window System is a trademark of Massachusetts Institute of Technology

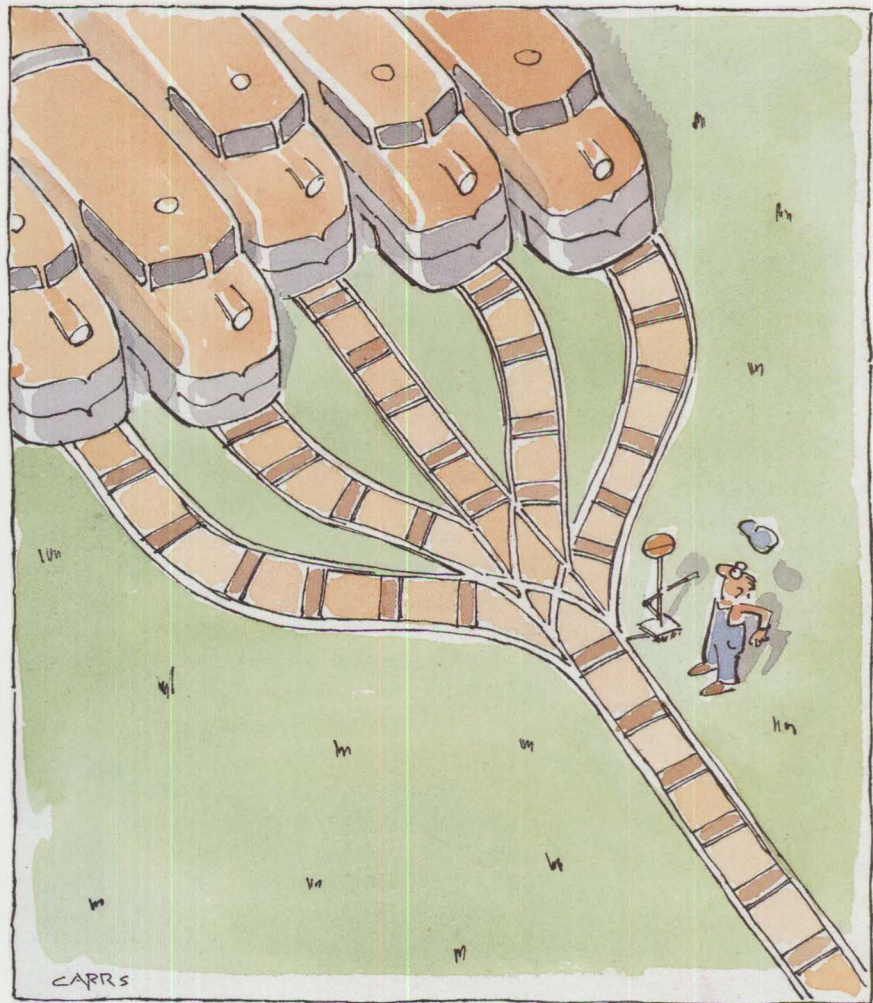
## Power Factor: Non-Linear Loads and the Power Distribution System

Roger Cox  
IBM Corporation  
Boca Raton, Florida

Power management is commonly a major concern to companies installing large numbers of personal computers. This article discusses problems that may be encountered after installation.

One of the major issues with power electronics today is how harmonic currents are generated by the non-linear load of Switch Mode Power Supplies (SMPS) used in PS/2® computers and other non-linear load products.

The effect of a non-linear load from a single SMPS or other non-linear load is usually of little consequence to a home or office installation or to the utility power distribution system. When non-linear loads become an appreciable portion of the load on a three-phase distribution transformer, problems can occur. When a business wants to install hundreds or thousands of PS/2s in its office building along with the other non-linear loads, various forms of overloading can occur. The following overview



discusses the problems and some possible solutions.

### What Are Non-Linear Loads?

Maybe the easiest way to explain a non-linear load is to describe what a linear load is. A linear load produces a current that is directly proportional

to the line voltage. For alternating current, the current increases or decreases proportionately as the voltage increases or decreases. This current can be in-phase or out-of-phase, which are discussed later, but in either event, the sine wave of voltage produces a sine wave of current of the same frequency.

A non-linear load produces a current that is not directly proportional to the line voltage and is not a sine wave of the same frequency. Often the current is not continuous and can be switched on for only part of the cycle.

A simple comparison helps illustrate the difference between the two. A non-linear load can be compared to a linear load like a fluorescent bulb to an incandescent light bulb. The major difference between the two is that the fluorescent bulb's current pulsates not directly proportional to the voltage, and the incandescent bulb's current varies directly proportional to line voltage.

### AC Power Terminology

There are a large number of single-phase, two-phase, and three-phase power distribution systems throughout the world. Single-phase stands by itself, even though it could be supplied from any of the three systems just mentioned; that is, it could be line-to-neutral (L-N) from a three-phase system, and so on. There are numerous grounding schemes used around the world; for example, grounded wye; corner grounded delta; tap-grounded delta; floating wye, delta, and tee; impedance grounded wye; and grounded open delta. This discussion is limited to the wye configuration for three-phase systems, and the content is independent of the grounding scheme. Although U.S. voltages may be discussed, the general principle applies to all countries and voltages.

**Single-Phase Systems:** Single-phase sinusoidal (sine wave) 50 or 60 Hz is the power supplied to all PC-type products. The voltage waveform provided by the utility company is a sine wave, which is usually of good quality, and it maintains its waveshape. The shape of

the current waveform is entirely a function of the load. A resistive-type load provides a sine wave of current that is in phase with the voltage sine wave. The term *in phase* means that corresponding points of the voltage and current waveshapes occur at the same time on the time scale: positive peaks, negative peaks, as well as positive and negative zero crossings. An inductive or capacitive load also provides a sine wave of current, but it is lagging or leading with respect to the voltage. They are not in phase and are called, of course, *out of phase*. Any combination of resistive-, inductive-, and capacitive-type loads always provides a sine wave of current and the loads are considered linear loads.

**Three-Phase Systems:** A three-phase system is just three single-phase systems that are displaced in time phase from one another. The time displacement between phases is one-third of a cycle or 120 degrees of a complete cycle of 360 degrees. The discussion of the sine wave current that could be lagging or leading or in phase applies equally to the three-phase system. In building distributions, the wye connection is usually used for the transformer secondary that uses four-wire distribution: line 1, line 2, line 3, and neutral. The single-phase voltage is measured from any one of the lines to neutral.

The neutral in the three-phase system is the common point and the return for the three phases. It conducts the total return current for the three phases. An interesting fact in three-phase theory is that the neutral current in a balanced linear load system is zero. Even though each phase has a return current, the addition of three equal currents, each 120 degrees apart, makes the return current zero. That is why building wiring used to

allow the neutrals to be wired at a 50 percent reduction in current carrying capacity. This is the root of the problem associated with building wiring, which is discussed later.

**Power and Volt-Amperes:** Power is measured by a wattmeter and represents the amount of work being done (or heat generated). In power distribution books, the terms *real power* and *reactive power* are used. Real power represents the amount of work being done, is generally referred to as simply *power* in the computer industry, and is measured by watts. Reactive power is really a misnomer because it is not power. It does not represent any work being done and does not contribute to the watts measured by a watt meter. This reactive power comes from the inductive and capacitive loads that produce the out-of-phase currents mentioned previously.

The term *volt-amperes* is the product of the voltage and the current being measured. It is a combination of the real power and the reactive power. It is called volt-amperes to distinguish it from the real or the reactive powers. Because it is a combination of the real and reactive power, it has a larger value than either. Present-day watt meters give readouts of the total current and the voltage as well as the watts. Multiplying the total current and the voltage gives you the volt-amperes.

**Power Factor:** Power factor is defined two different ways today, depending on the type of load. The traditional definition is concerned with the sine wave of voltage and the sine wave of current. When an inductive or capacitive load produces a load current that is displaced in time from the line voltage, the cosine of the angle of displacement is defined as the power factor. When

the equations are worked out, power factor is also the power (watts) divided by the volt-amperes (VA).

$$P.F. = \cos \text{ Angle} = \text{Power/Volt-Ampere}$$

For non-linear loads, which were defined in the section "What Are Non-Linear Loads?" there is no single "angle of displacement" that can be used to represent the power factor. The second half of the previous equation is what is used for non-linear load power factor, power/volt-amperes.

**Average Root-Mean-Square (RMS), Peak RMS, and True RMS Measurements:** Many measurement techniques assume that the voltage and current waveforms are approximately sinusoidal. When this is not true, as in the case of non-linear loads, errors result from the measurements. There are three common measurement techniques used for AC voltages and currents: average RMS, peak RMS, and true RMS.

The average RMS measures average voltage and current over one half cycle, then converts the measurement (1.11 X average) to be representative for a sine wave because that is what is normally encountered. Most inexpensive digital voltmeters use average conversion because it is the least expensive to implement. It is also the approach used in older-style analog meters.

The peak RMS measures the peak voltage and current over one cycle and then converts the measurement (0.707 X peak) to be representative for a sine wave because that is what is normally encountered. As the RMS indicates, the average RMS and peak RMS are calibrated to equal true RMS on sine waves.

Waveform Description	Average RMS	Peak RMS	True RMS
Sine Wave	100.0A	100.0A	100.0A
Square Wave	100.0A	63.6A	90.0A
Triangle Wave	100.0A	125.7A	103.3A
Non-Linear Load	100.0A	400.1A	199.4A

Figure 1. Voltage, Current, P.F., and Spectrum Measurements on Non-Sinusoidal AC Power Circuits (Reference 1)

The RMS value of an AC waveform (voltage or current) is equivalent to the DC (direct current) value that produces the same heating (or work) in a purely resistive load. Therefore, a true RMS circuit makes accurate measurements even if the waveform is non-sinusoidal (or non-linear).

So on sine waves, all three measurement techniques give the same reading. On non-sinusoidal waveforms, the readings of line voltages and currents can vary dramatically, as shown in Figure 1.

For non-linear load currents, measurement with the average RMS meter can give a reading of one-half the actual value, or the true value is twice the average RMS meter reading.

### Harmonics and Non-Linear Loads

Harmonics, as related to our subject, are multiples of the fundamental line frequency (60 Hz in the U.S.). The second harmonic is two times 60 Hz or 120 Hz. The third harmonic is three times 60 Hz or 180 Hz. Any non-sinusoidal waveform can be represented by an infinite series of sine waves at integers of the fundamental. With our non-linear load current, the waveshape of the current is usually composed of the fundamental and up to the fiftieth harmonic with the third, fifth, seventh, and ninth being the largest. The even harmonics

are very small and usually insignificant.

Earlier it was shown that a three-phase balanced linear load had zero current in the neutral. In a three-phase balanced non-linear load, the neutral current is not zero but consists of the third harmonic and odd multiples of the third. This is the third, ninth, fifteenth, twenty-first, and so on. What we find is that the odd multiples of the third do not cancel at all in the neutral but are additive.

In examining the typical input current of an SMPS, the current pulse waveform can be run through a program to do a harmonic analysis on it. The output of the analysis provides a breakdown of the fundamental and the individual harmonics up to the 25th or the 50th, depending on the program. The output can be listed as current in milliamps or amps or can be listed in percent of the fundamental. The fundamental is the single frequency component at the line frequency. A typical percentage listing is shown in Figure 2.

Since the third, ninth, and other odd multiples of the third harmonic add in the neutral wire, the neutral wire winds up carrying a current that is much greater than it had been designed to carry.

Types of loads that are sources of harmonics are all types of AC-to-DC

Harmonic	Percentage
Fundamental	100
3rd	70 to 90
5th	50 to 70
7th	20 to 40
9th	5 to 20

Figure 2. Typical Percentage Listing

converters such as motor speed controllers and rectifiers for DC loads, fluorescent lamps, motor-driven computer peripherals, and the fairly recent entry of the SMPS used in almost all computer and entertainment equipment. The conversion to SMPS has been accomplished with a reduction in size, weight, and cost, which has been the key to its universal acceptance.

## Problems Caused by Harmonics

This universal acceptance of SMPS and the attendant acceptance of personal and other computers has led to the generation of a large problem list. When companies connect hundreds and thousands of PCs in a building, the triplen (odd multiples of the third) harmonic currents add to those of the other non-linear loads, creating unexpected electrical problems. Power line harmonics have been well documented to be the cause of:

- Overheating of the neutral conductor wires and connections, leading to failure of neutral conductors that could cause overvoltage damage to computers and electronic office equipment. This is amplified by the wiring practice of reducing the wire size of the neutral conductor.
- Intermittent electrical noise from connections that have been loosened by thermal cycling. This

noise can be intense enough to corrupt digital circuit signals and cause malfunctions.

- Transformer overheating, insulation damage and failure.
- Distortion (peak voltage reduction) of line voltage waveshapes severe enough to impair the ride-through capability of computers and related equipment.
- Overheating of motors operating on a distorted voltage source.
- Nuisance tripping of circuit breakers.
- Burned wires in manufactured office partitions.
- Power factor correction capacitors overheated and damaged.

(See Reference 2.)

*When companies connect hundreds and thousands of PCs in a building, the triplen harmonic currents add to those of the other non-linear loads, creating unexpected electrical problems.*

## Detection of Problems

The traditional measurement technique for facility power distribution systems has been using average RMS meters. From the previous discussion, the average RMS current meters provide a reading of about one-half of the true RMS high harmonic non-linear load. Measurements with average RMS meters made on building wiring and trans-

former currents usually indicate currents within ratings.

Because of the harmonic problem, all measurements should be made with true RMS meters and all ancillary equipment (such as current transformers) should have frequency bandwidths to cover the probable harmonics. If peak current readings are also made, transformer deratings can be calculated using the formula provided by CBEMA in their 1988 information letter. This measurement data can then be compared to the facility ratings to detect problem areas.

Equipment is available that measures the true RMS currents and provides a printout of the harmonic content of the currents. This measurement can be made on individual SMPS and other harmonic loads or can be made on the building distribution wiring showing the total effects on wiring and transformers.

## Possible Solutions

The possible solutions fall into four main categories. They are:

- Passive filters
- Filter traps
- Active filters
- Other facility changes

Passive filters usually consist of a series inductor and a third harmonic tuned filter and are located between the wall outlet and the offending non-linear device. They operate at line frequency so they are relatively bulky. They must be fairly well matched to the size of the load so that they do not cause an increase in RMS line current, even though harmonic currents are reduced.

Filter traps are generally known as *zero sequence filters* or *zig-zag transformer filters*, not the passive filter



just described. This filter is placed as close as possible to the offending load and at a point where three-phase wiring is available. Careful consideration must be given to the placement and sizing of the filter for a given power distribution system because the filter attracts currents from other offending loads powered from that distribution net. This can be an effective method of correcting the problem with existing high harmonic loads. When distribution transformers have to be replaced or added, this type of filter can be built into the transformer.

IBM Canada Ltd. has recently used the zig-zag transformer to dramatically reduce the harmonic current in the distribution panel of the office area of their Toronto manufacturing plant. The neutral current was reduced 90 percent in their installation.

Active filters add current to the existing input current so that it gives a sine wave. They can work very well, but they are best for a high-power three-phase system, and they are very expensive, bulky, and complex.

Some of the other facility changes mentioned in the *CBEMA Information Letter* are:

- Derate the distribution transformers
- Use individual neutral wires for each phase
- Use dual neutral wires for shared neutral wiring
- Use low impedance delta-wye distribution transformers
- Avoid the use of power factor correction capacitors

As is usually the case, the lowest-cost solution is probably a combination of the solutions described. There should also be a periodic inspection of the electrical system, especially after additions of hardware or changes of system configurations. These inspections should include measurements of phase and neutral currents, temperatures of transformers, their connections and the connections in the total distribution system. Filter traps should also be looked at when system configurations change because the amount of current could possibly exceed its rating.

### System Power Supplies with Power Factor Correction

An obvious question one could ask is, "Why not reduce the harmonics at the source?" The answer, of course, is that one can. There are circuit approaches that force the input current to appear resistive and proportional to the line voltage and be in-phase. These approaches can be designed into new products for future sale and add some cost to the system. In Europe, an IEC 555-2 standard is being changed to add harmonic limits. This standard applies

to all household and other equipment (which includes the PC products). When this standard is approved and adopted, all PC products for Europe have to include power factor correction to meet the harmonic limits.

Reference 1: Figure by A. McEachern. Contributed by BMI, Foster City, California, (415) 570-5355, University of Wisconsin-Milwaukee Conference on "Effects of Non-Linear Loads on the Power Distribution System," April 5-6, 1989.

Reference 2: Computer Business Equipment Manufacturers Association (CBEMA) Information Letter, 1988, and "DP Experience with Switching Supplies" by John M. Roberts, IBM. University of Wisconsin-Milwaukee Conference on "Effects of Non-Linear Loads On The Power Distribution System," April 5-6, 1989.

### ABOUT THE AUTHOR

*Roger Cox is a senior engineer at IBM's Entry Systems Division. He has worked in the Systems Power Development department since he joined IBM in 1984. He has been involved in system power-related areas since the beginnings of PS/2 design, and is the power technical interface to IBM plants.*

### Further Reading

Any good college text on Alternating Current Circuits will cover three-phase systems.

*Magnetic Circuits and Transformers*, "Harmonic Phenomena in Three Phase Circuits," MIT Staff. MIT Press, 1943.

*Magnetic Circuits and Transformers*, "Zigzag Autotransformers," MIT Staff. MIT Press, 1943.

"Three Phase Power Source Overloading Caused by Small Computers and Electronic Office Equipment" CBEMA Information Letter, 1988.

# Database Manager: Highlights and Direction

*Philip J. Sullivan  
IBM Corporation  
Austin, Texas*

**Database Manager is IBM's relational database management system for Operating System/2® (OS/2). It offers a robust application development and run-time platform for a variety of OS/2 and DOS database applications. This article describes the capabilities and scope of Database Manager, and IBM's intentions for future versions of the product.**

IBM OS/2 Extended Edition Version 1.3 includes a Systems Application Architecture™ (SAA™) relational database management system, named Database Manager, that supports the SAA Structured Query Language (SQL™). Database Manager can operate as a multiple-user client-server distributed relational database management system attached to a local area network (LAN) or as a stand-alone single-user relational database.

Database Manager has benefited from IBM Research-developed database technologies and architectures, including the relational model of data and SQL. These IBM technologies and architectures, coupled with the new technologies and architectures developed expressly for Data-

base Manager, have positioned this product to be an industrial-strength relational database. It provides high performance, concurrent data access, robust data integrity and protection facilities, and consistency with the family of IBM SAA relational database management systems.

## Relational Model of Data

Database Manager is predicated on the relational model of data. The relational model of data was invented by E. F. Codd at the IBM San Jose Research Center in the late 1970s. Today, the relational model has been widely accepted as an industry standard by the marketplace. The relational model has been designed to be easy to understand and use. Information is presented to the user in an easy to recognize and easy to use table format. A table is a logical data structure consisting of rows (records) and columns (fields). The user defines and accesses data in terms of tables and performs operations on these tables.

The main advantage of the relational database model is the clear separation between the user perception of data, that is, tables consisting of columns and rows, and the internal implementation of data, which is hidden from the user. The simple table format, along with SQL and high-level application development tools, means that the application developer and the database administrator do not have to understand complex physical data structures and access methods, which are characteristics of hierarchical file manage-

ment systems. The benefits are ease of use and productivity.

## Structured Query Language

SQL is the common interface to Database Manager and all IBM SAA relational database management systems. SQL, originally developed at the IBM San Jose Research Center, has also become a standard in the industry. SQL is a high-level data definition and data manipulation language, used for defining, accessing, and changing data in tables.

SQL is considered simple to learn, yet powerful in expressing sophisticated queries. A single statement in SQL can perform the same function as many lines of application code developed with a conventional programming language, such as C, COBOL, FORTRAN, or Pascal.

All data accesses to Database Manager are performed with SQL statements through the SQL common interface. SQL supports arithmetic operations on retrieved values. The query functions support selective retrieval from single or multiple data tables and dynamic sorting of the set of resulting rows. Built-in functions include summation, grouping, ordering, and basic statistics (for example, calculating an average of the values in a column).

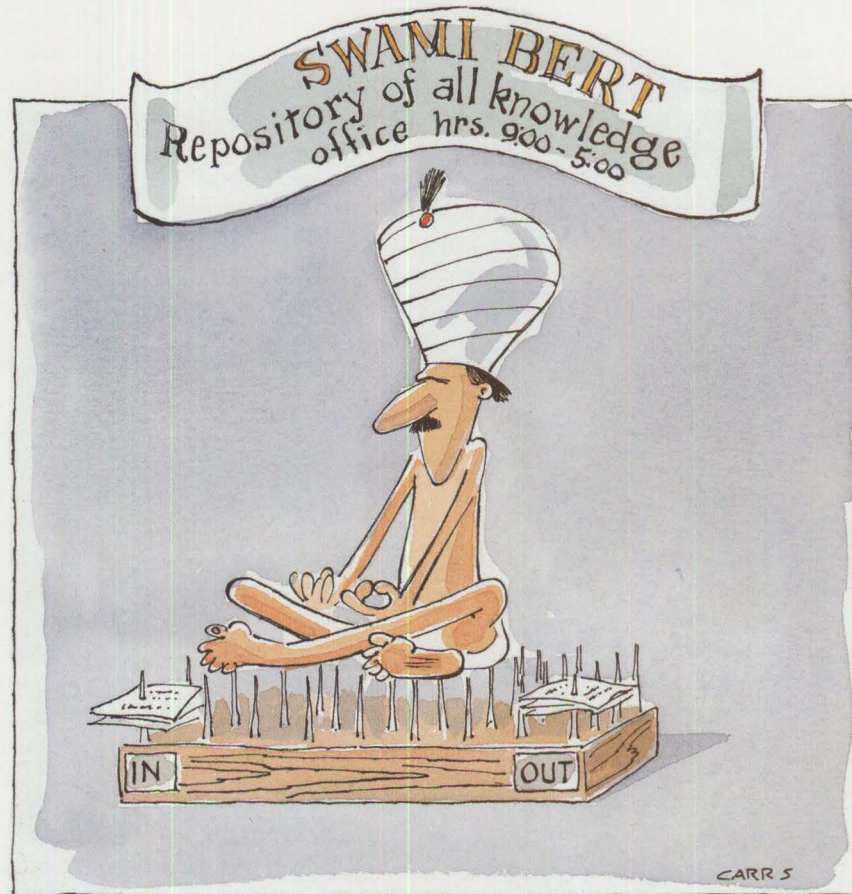
SQL statements can be entered interactively (through Query Manager, described later in the article), embedded in a precompiled application program written in a high-level programming language, or embedded in an application program writ-

ten with IBM Procedures Language 2/REXX. Database Manager provides language precompilers to prepare embedded SQL statements for subsequent application program compilation and execution by Database Manager. Precompiler support is provided for IBM C/2, IBM COBOL/2™, IBM FORTRAN/2™, and IBM Pascal/2™.

Database Manager supports both *Static SQL* and *Dynamic SQL* statements embedded in an application program. The difference between Static SQL and Dynamic SQL is the instance when compilation of the statements occurs at the database. For Static SQL statements, the compilation occurs during precompiling or binding of the application to Database Manager. The compilation is done only once, no matter how many times the statements are executed during the course of running the application. Dynamic SQL statements are compiled each time the statements are executed during the course of running the application. Static SQL is a more efficient process and will, in most instances, provide better performance than Dynamic SQL. However, if the application must issue arbitrary SQL queries, Dynamic SQL is required.

### Client-Server Distributed Database Architecture

Database Manager has been designed to be a multiple-user client-server distributed database management system for IBM OS/2 local area network environments. The architecture provides for separa-



tion of the application program from the database itself.

Database Manager remote-unit-of-work functions provide OS/2 and DOS client applications with the capability to transparently access data (read and update) in a single remote OS/2 database server, per unit-of-work. A client application can also serially access multiple databases.

Database Manager client-server distributed database architecture provides a cost-effective solution to a wide-range of applications, such as decision-support and personal productivity applications, which require multiple users to have fast, concur-

rent, and easy access to consistent data.

### Performance

Database Manager has demonstrated excellent performance. In October 1990, the National Software Testing Laboratory (NSTL) published results of benchmark tests of Database Manager. The purpose of the tests was to compare the performance of Database Manager with three competitive offerings, each one a participant in the OS/2 client-server database marketplace. The test results showed Database Manager to have comparable or superior performance capability to the competitive offerings in three of four client-server application sce-

narios. In the fourth scenario, Database Manager, while not faster than the competition, was competitive.

IBM database technologies have been applied to enhance performance; for example, query optimization techniques, join algorithms, and indexing and locking techniques. *Record Level Locking* allows maximum concurrent access to data and is especially important to database transaction processing performance. Granular levels of data isolation are provided to enhance performance. These include *Repeatable Read*, *Cursor Stability*, and *Uncommitted Read*.

*Database Application Remote Interface* may be used to enhance performance. It allows a developer to write an application program where the application processing can be split between the database client and the database server on a local area network. When the application is run, some of the application processing load can be transferred from the client to the server, resulting in a reduction of data traffic on the LAN and a significant improvement in database application performance.

### **Robust Data Integrity and Protection**

Database Manager has extensive data integrity and protection features to preserve the integrity of data, thereby ensuring users that information being accessed is consistent.

*Transaction Management* is a feature that allows multiple applications to run concurrently against common tables in Database Manager. Through the use of transaction management, Database Manager provides a high level of data control and protection against serialization problems and database transaction

failures in a multiple-tasking, multiple-user database environment. If an application accesses Database Manager and terminates normally, a COMMIT statement is issued, which allows the database updates to become permanent. If an application is interrupted in the midst of a transaction, the system can perform a ROLLBACK on all uncompleted work after an application failure, or on the next system restart after a system failure. These functions help ensure the integrity and consistency of the database information.

### *Database Manager has extensive data integrity and protection features.*

Database Manager provides transaction management through the use of locks, multiple levels of data isolation, and a Recovery Log. The lock function and the specified level of data isolation are used to prevent another application from updating a data record while a transaction is pending against that record. All changes to data tables and indexes have entries written to the Recovery Log that provide sufficient information to allow Database Manager to back out of an update before any changes are written to the data.

Through the use of the *Recovery Log*, updated during normal database processing using a write-ahead logging scheme, Database Manager can return the database to a consistent state after system failures, such as a power failure, a user-initiated system reset, or a software failure

(except in some instances where logical media failure has occurred). In the event of such a failure, Database Manager can undo and redo the necessary database operations to return the database to a consistent state by retrieving and reconstructing committed data and rolling back uncommitted data using the Recovery Log.

Database Manager is also capable of restoring a backup copy of a database, where data on the media, disk or diskette is destroyed by physical or logical damage. Physical media failure can occur due to a bad sector on the disk or diskette. Logical media failure can occur when data is over-written by another program and becomes unrecognizable to Database Manager. Logical media failure can also occur if a system fails during a data update, such that some sectors are left unchanged while some are updated. Database Manager attempts to recover from logical media failures using the Recovery Log. If this attempt fails, the backup copy of the database must be restored.

*Referential Integrity* is another important feature used to provide robust data integrity and protection. It ensures the consistency of data values among related columns of different tables. For example, a user may define an EMPLOYEE table that contains employee and department numbers and a DEPARTMENT table that contains department numbers. In addition, the user may want to ensure that for every department number in the EMPLOYEE table there must be an equal and unique department number in the DEPARTMENT table. Such a constraint defined on the EMPLOYEE table is called a referential constraint. The department number in the DEPARTMENT table is called a primary key, and the department number in the EMPLOYEE table is called the for-

eign key in this constraint. Enforcement of this constraint provides referential integrity. The Database Manager records and enforces this data relationship, and enforcement by application logic is not necessary.

In addition to ensuring the consistency of data, Referential Integrity can also improve application development productivity by allowing this function to be moved out of each application program and into the Database Manager.

### Data Security

In any database environment, there is a need to protect information from unauthorized access. Database Manager provides for protection from unauthorized access of data by a granular grant/revoke privileges scheme.

*SQL Grant/Revoke Security* prevents unauthorized access by coordinating security functions through an OS/2 Extended Edition component called User Profile Management and through SQL GRANT/REVOKE Authorization statements. User Profile Management establishes access levels used by Database Manager.

In order to access and use objects in the Database Manager, the user must be identified to User Profile Management and be validated by a password on the first use of the Database Manager. The user is then associated with a valid user ID. Access to a specific database and the objects within it (for example, tables, views, and access plans) is controlled by SQL GRANT/REVOKE statements. A creator or other specifically authorized user of a database object (such as a systems administrator or database administrator) may protect the object by only granting access rights to specific users and/or groups. An-

other user must be specifically authorized to access and update a database object. These rights can also be revoked as required. A creator also has the option to allow public access to all database objects.

### IBM SAA Relational Database Consistency

Database Manager is a member of the family of IBM SAA relational database management systems: IBM Database 2 (DB2) and IBM Structured Query Language/Data System (SQL/DS™), which run on IBM System/390™ and IBM System/370™ mainframe and mid-range systems, and IBM OS/400™ database manager, which runs on the IBM Application System/400™.

Syntax and Semantics of the Database Manager SQL interface functions are designed for consistency with the family of IBM SAA relational databases. For example, the consistent handling of host variables, precompile, bind, and query optimization simplifies the application developer's work in writing heterogeneous cross-system applications for an enterprise-wide distributed database environment.

### Import Data from DB2 and SQL/DS Databases

Database Manager provides a facility, named SQLQMF, which allows users to import table data stored in DB2 or SQL/DS into a Database Manager database. The SQLQMF facility takes host data exported by the IBM Query Management Facility (QMF™), stored in QMF format, downloads the data, and converts it into a file that can be imported into a Database Manager table.

## Database Manager

### Components

Database Manager consists of three major components: Database Services, Remote Data Services, and Query Manager.

**Database Services:** Accesses to Database Manager are performed by Database Services through the SQL common programming interface (CPI) and utility application programming interfaces (APIs). Database Services manages the data stored in the database, generates access plans to the data, and includes transaction-management, data integrity and protection, multiple-user concurrency, and security functions.

Database Services also provides a number of utility functions that assist the user in maintaining and manipulating the contents of a database. These utilities include functions to import and export data from a DOS or OS/2 file, save and restore individual data tables, perform database backup and restore functions, and optimize system performance.

**Remote Data Services:** Remote Data Services provides database connection services to allow Database Manager to function as a client-server distributed database management system on a LAN. This capability allows multiple OS/2 and DOS personal computer client applications to transparently access a remote OS/2 distributed database server attached to a LAN, or stand-alone OS/2 personal computer clients not attached to a LAN, to access a remote OS/2 distributed database server.

Remote Data Services supports the following LAN environments: IBM Token-Ring, IBM PC Network LAN, or ETHERAND™ and IEEE 802.2. OS/2 database clients and


servers utilize the APPC communication protocol, provided by Communications Manager, or the SQL LAN-Only Option (SQLLOO) to connect in a LAN environment. APPC is also used to connect remote OS/2 database clients to remote OS/2 database servers. DOS database clients attached to a LAN utilize the NetBIOS communication protocol to connect to a remote OS/2 database server.

**Query Manager:** Query Manager provides database tools and functions that allow users to access, manipulate, and administer data in Database Manager.

Query Manager provides an easy to use prompted user interface, which is based on the SAA Common User Access (CUA) guidelines and takes advantage of the OS/2 Presentation Manager™ windowing and graphic services. The prompted interface hides SQL from the application user, so the user does not have to learn SQL syntax. However, once a prompted query has been created, its SQL syntax can be viewed optionally. This allows users to learn SQL syntax so they can enter SQL statements in a non-prompted mode, if desired.

*Query and Report Writing Tools* provide functions to create a database query (inquiry), initiate the data access request, and display the results of the query via the report feature. The user may save the query and report objects and save or print the accessed information. A Business Graphics Interface allows the user to optionally install and use a third party business graphics program written to this interface. This permits graphic presentation (pie, line, bar, tower graphs, and so forth) of report data that was accessed by the query and report features.

*Custom Application Tools* provide functions that allow a user to create custom database applications. Panels may be used to create custom data entry and edit formats. *Procedures* may be used to create Query Manager commands or procedure language statements to execute previously defined query and form (report) objects, or to combine and automate several panel operations. *Menus* may be used to create a customized list of application selections. Items listed on the menu may be selected to initiate and automatically run queries, generate reports, run and call procedures, panels, and other menus.



### *Query Manager provides an easy to use prompted user interface*

*Database Administration Tools* provide functions to administer and manage a database system; for example, creating, altering, and dropping tables, views, indexes; entering and editing data; defining referential constraints; importing and exporting data; reorganizing tables; monitoring database activities. One of the Database Manager features that is supported through Query Manager is *Operational Status*, which provides a snapshot of information about current database activity. This feature provides information about where the databases are located, alias names, the time and date a database was last backed up, and how many applications are currently connected to a specific database.

A *command line* is also provided for users to issue Query Manager com-

mands directly. These commands include functions such as print, get, import, and display. These commands may be used to execute Query Manager objects, such as Procedures, directly from the command line.

## **Future Versions and Releases of Database Manager**

Future versions and releases of Database Manager will focus on providing industrial-strength distributed relational database solutions for stand-alone LAN environments and, in concert with other IBM SAA relational database products, enterprise-wide distributed relational database solutions for interconnected LANs, and host systems. Emphasis will be placed on data integrity and protection, concurrency, connectivity and interoperability, database administration, usability, security, performance, capacity, and availability.

## **1991 Direction**

**OS/2 Database Manager Access to IBM SAA Host Databases:** As IBM begins the roll-out of heterogeneous SAA distributed relational database functions, Database Manager will initially provide functions to support heterogeneous remote-unit-of-work (RUOW) database applications.

Support for RUOW and the IBM Distributed Relational Database Architecture (DRDA) will allow Database Manager client applications (OS/2, Windows 3.0, and DOS) to transparently access (read and update) a single remote IBM SAA host distributed relational database per unit-of-work. A Database Manager client application will also be able to serially access multiple SAA host relational databases.

*Distributed Database Application Gateways*, as single-user and multiple-user offerings, will provide the transparent connection between personal computer database clients and an IBM SAA host relational database server, such as DB2, SQL/DS, or the OS/400 database manager.

**Forward Recovery:** Data protection is essential to customers who have mission-critical database applications, such as an online order entry application. Database Manager capabilities will be enhanced through the addition of *Forward Recovery* functions.

Forward Recovery, also known as "Archival Log," will allow the user to recover lost online data due to media failure, such as a disk crash. During normal online database transactions, information necessary to maintain data integrity and consistency is preserved on log journals. Journals are normally kept on separate media from the database. Should media failure occur, a backup off-line copy of the database may be loaded on the system and restored as the online database. Forward Recovery provides the means to apply log journal information against the restored database. Log journals contain changes made to the online database since the last backup (off-line copy) of the online database was made. After forward recovery has been completed, the state of the online database is as it was prior to the media failure and subsequent loss of data.

The addition of Forward Recovery is consistent with IBM's direction to continue to focus on providing leadership in database functions. Forward Recovery enables Database Manager to be used for a variety of mission-critical applications, such as

online transaction processing (OLTP).

#### **Microsoft® Windows™ 3.0**

**Support:** Database Manager DOS Database Requester functions will be enhanced to provide support for applications developed for the Microsoft Windows 3.0 environment. With this capability, Windows 3.0 client applications written to the Database Manager SQL CPI will be able to transparently access a remote Database Manager server attached to a LAN.

#### **NetBIOS Support for OS/2 Clients:**

NetBIOS support for OS/2 clients will be provided as an optional LAN communication connectivity protocol to an OS/2 database server. NetBIOS is a popular personal computer protocol for connecting personal computers together. NetBIOS will require less random access memory (RAM) than APPC, which will result in a reduction in the amount of memory required for client workstations. This option will also simplify database installation and configuration.

#### **Database Tool Enhancements:**

Database Administration Tools will be provided as a separate installable option. The tools will include functions to configure Database Manager and a database, backup and restore a database, perform forward recovery, and configure remote client-server and gateway connections. The Database Administration Tools will have graphical user interfaces that provide an easy-to-use prompted interface based on SAA CUA guidelines and take advantage of the OS/2 Presentation Manager windowing and graphic services.

*Database Manager Command Interface* will allow users to create and run SQL statements, database envi-

ronment commands, and database utilities from the OS/2 command line interface and from OS/2 command files. Included with the Database Manager Command Interface is a facility (Re-Org Check) to inform the database administrator that it is beneficial to reorganize a table in the database, so that performance may be improved.

#### **Support for Non-IBM Personal Computer Platforms:**

Database Manager will be enabled to run on selected IBM-compatible personal computer hardware systems. Support for non-IBM systems will extend the versatility of Database Manager and provide additional flexibility in selecting hardware platforms: IBM PS/2 solutions, non-IBM hardware solutions, or mixed IBM and non-IBM hardware solutions.

#### **SAA Relational Database**

##### **Compatibility Enhancements:**

SAA enhancements will improve Database Manager compatibility with the IBM family of SAA relational database products.

*SQL Date and Time Arithmetic and Scalar* functions will allow applications to add and subtract the following data types: TIME, DATE, and TIMESTAMP. An example of the use and value of SQL

DATE/TIME/TIMESTAMP is an application designed to track and monitor the elapsed time of a commercial airline flight from the originating city and gate to the destination city and gate.

*SQL State* will provide applications with diagnostic information that is common across the family of IBM SAA relational database management systems. This capability makes possible the development of error handling routines that are consistent

with and portable across the SAA database family.

*User Defined Collating Sequence* will allow the database user to specify methods that will be used to sort and compare character data. This option may be used to ensure collating results that correspond to IBM host systems, such as System/370.

**Translate:** Translate will allow the user to transform character data in various ways; in particular, it allows characters to be converted to their uppercase or lowercase equivalents. This capability makes possible case-insensitive searches.

**Database Manager "Front-Ends" (Database Applications and Tools):** IBM will continue to provide Database Manager technical support to complementary independent application developers and to business partners. This support will be provided through developer assistance programs, whose purpose is to provide assistance in the design and development of a variety of OS/2 Presentation Manager, Windows 3.0, and DOS database applications and tools, such as general purpose query and report writers and customizing tools that work with Database Manager.

#### **Make Available Database Manager Code to Selected Software Developers:**

IBM's direction is to make components of Database Manager code available to complementary software developers who meet IBM's selection criteria. Selected software developers may integrate, package, and market Database Manager code with application programs they develop. General-purpose database query and report writing programs, and industry-specific custom application programs are ex-

amples of the types of programs that may use Database Manager code.

Database Manager code can provide the software developer with industrial-strength relational database management services, distributed database application connection services to OS/2 database servers attached to a LAN, and distributed database application connection services to IBM SAA host relational databases.

### **Future Direction**

#### **Distributed Database**

**Enhancements:** Over time, Database Manager will be enhanced to provide additional distributed database functions. Support will be provided for increasingly more complex data access, such as distributed unit-of-work, distributed (SQL) statements, and distributed tables.

#### **Compatible 32-Bit OS/2 and**

**AIX® Database Managers:** IBM's direction is to provide compatible 32-bit client-server SAA and AIX Database Managers that fully exploit the 32-bit OS/2 and AIX platforms. The 32-bit OS/2 Database Manager will be provided for Intel®-based PS/2 computers and IBM-compatible personal computers. The 32-bit AIX Database Manager will be provided for IBM RISC System/6000™ platforms.

The OS/2 Database Manager and the AIX Database Manager will be optimized for LAN performance and interoperability, conform to industry and defacto standards, and provide enhanced capabilities.

#### **Parallel Database Processing:**

IBM's direction is to enhance Database Manager to provide 32-bit parallel database processing. Parallelism will support multiple system processor nodes, such as multiple

PS/2 computers, or multiple RISC System/6000 computers, that are linked together to present a single-system image.

Database Manager parallel functions will provide customers with high throughput rates, fast response times, access to very large (giga-bytes) amounts of information, and fault-tolerant functions. Database parallelism will be especially beneficial to online transaction processing applications where fast response time and high availability are essential. Parallelism will also offer significant benefits to advanced complex-data applications, such as image and multimedia, where access to large amounts of data are application characteristics.

### *ABOUT THE AUTHOR*

*Phil Sullivan is a senior product planner in the IBM Personal Systems Programming Center in Austin, Texas. He joined IBM in 1966 as a marketing representative and has held various management and staff positions in marketing and product planning. For the past 10 years, Phil has been involved in the product planning of personal computer software products. He is currently responsible for IBM personal systems database product and market strategy. Phil holds a B.S. in economics from Mount St. Mary's College.*

Much of this information concerns future products, or future releases of products currently commercially available. The discussion on Windows 3.0 is based on information that Microsoft Corporation has made publicly available and is subject to change. The description of IBM's future products, performance, functions, and availability are based upon IBM's current intent and are subject to change.



## OS/2 Communications Manager

*Rich Harrison  
IBM Corporation  
Austin, Texas*

*This is an updated version of a  
Communications Manager document  
distributed by IBM April 16, 1991.*

**Communications Manager is IBM's strategic workstation-based communications offering. It provides many concurrent networking capabilities. This article describes the wide range of services offered by Communications Manager, the value of its features to the user and developer, and IBM's plans for enhancements to be included in the OS/2 2.0-based version.**

Communications Manager (CM) is one of the major components of Operating System/2 Extended Edition (OS/2 EE). CM provides a rich set of terminal emulation, LAN gateway, host connectivity, and workstation communication capabilities. CM exploits vital functions of OS/2 such as preemptive multitasking and I/O overlapping and is intended to facilitate the development of applications that require communications between two or more systems or workstations. By including most of the required communications functions within one component, application developers are relieved of understanding the low-level detail of communication tasks and may rely on CM to accomplish them quickly and efficiently. Additional services in the form of emulation functions are available to the end user, and facilities are included for use by the Net-

work administrator who may be dealing with the management and control of large numbers of end users.

OS/2 EE is part of the IBM Systems Application Architecture (SAA) family, which itself provides the platform for the development of portable applications for, and the interconnection of, cooperative SAA systems. As we move further into the 90s, customers will increasingly take advantage of the opportunities offered by "distributed processing"; that is, the ability to undertake processing on whichever system is most appropriate, with all systems being interconnected across a network. This environment will make even heavier demands on communications and increase the need for a versatile and robust communications subsystem such as CM.



CARR 5

What distinguishes CM from other communications offerings is its design and integration with the OS/2 environment and SAA. CM provides multiple concurrent connectivities, supports multiple concurrent protocols, allows concurrent emulations of different terminal types, and allows simultaneous file transfers. It also supports several APIs designed for the traditional and the contemporary program development environments, and has requisite interfaces for network management. All these aspects will be elaborated upon in the remainder of this article.

In short, CM provides the opportunity for productivity improvements for the program developer, the end user, and the network administrator. In particular, when the user is integrated into the network and has access to the organizational information processing assets, CM is extremely attractive.

IBM plans to make CM more open by allowing it to run on both IBM and other manufacturers' Intel-based hardware using the vendor's OS/2 Standard Edition (SE).

### **Systems Application Architecture (SAA)**

Communications Manager, in conjunction with OS/2 SE, participates in SAA, which provides the framework for consistency in the user interfaces, the programming interfaces, and the underlying communications support. The workstation is the window to the customer's enterprise-wide information system. It is vital that the user see a consistent and familiar means of access to and presentation of information regardless of the location of the system. Applications written to SAA specifications ensure that this occurs. This makes it easier for users to learn

new applications and to move from one application to another.

SAA can take these concepts one step further. It provides the capability for programs to be written across the computing environments of System/370, AS/400, and the Personal System/2®. The implication is not only that programs can work cooperatively across these hardware platforms, but also that programs written for one may be actually transferred to another. This is important because it allows workloads to be shifted as they grow or evolve over time.

Programming interfaces are covered by SAA, too. The strategic interface for distributed processing is Common Programming Interface - Communications (CPI-C). By writing to CPI-C, developers may confidently know that their applications will continue to be applicable within the SAA family into the future. CPI-C is currently available through a companion product - Networking Services/2. IBM's plan is to integrate this support into CM in the future.

Also available are interfaces to Presentation Manager. These interfaces enable developers to write high-performing, full-function applications that can be tailored to a specific user or environment. Calls may be used to control the program groups that a user sees and can access. This allows personalization of the system to a specific user or group of users. Presentation Manager has utilities for the printing/plotting, display, and interchange of picture files.

CM contains the Common Communications support that allows program interchanges among various systems. This takes place transparently to the end user and it is important because it ensures that such

interchanges take place efficiently and reliably. It is also very important to developers because it relieves them from having to write the routines that accomplish the communications functions. In complex environments with multiple transfers of data taking place simultaneously, all the power of CM and the preemptive multitasking capability of OS/2 are required to ensure that everything works smoothly. By using the facilities of CM, this process is taken care of in whichever communications environment the application is run. The details of this process are described in the sections that follow.

### **Connectivity Support**

Various methods have been used over the years to allow systems, controllers, terminals, and workstations to talk to each other. IBM's Systems Network Architecture (SNA) is a robust architecture that has become acknowledged as an industry standard. SNA supports both hierarchical and peer type networks. SNA networks are very prevalent today and are typically based on links using the Synchronous Data Link Control (SDLC) protocols between the host and other systems or controllers, and Distributed Function Terminal (DFT) protocols for terminals coax-attached to their controller. CM supports both these environments, the latter in the context of PCs being substituted for the fixed-function terminals (3270).

In recent years, there has been a strong trend toward the use of local area networks (LANs) to tie together the proliferating number of PCs in the workplace so they could share resources and access each other's data. Many different approaches were introduced before the leading industry LANs were identified. IBM initially introduced the PC Network LAN before adopting the Token-

Ring LAN as its strategic offering. Both the Ethernet™ and Token-Ring LAN have achieved very broad acceptance by the industry. Ethernet comes in several forms, the most popular of which are known as DIX Version 2.0 and IEEE 802.3. In order to allow its applications to run in the LAN environment, CM supports all the LANs mentioned in this paragraph.

Other ways for systems to communicate over a wide area network (WAN) include X.25 and ASYNC. The X.25 Packet Switched Data Networks (PSDNs) are available worldwide, generally through the official telephone and telegraph providers in Europe and elsewhere, and through private utilities and corporations in the U.S. They are an alternative and often cheaper way of transmitting data over a distance and use their own, standardized, link level protocols to interface with subscribers. CM supports the use of X.25 services both through the PSDNs and also using direct connection.

ASYNC is a simple, inexpensive way to access many of the financial, news, and informational bulletin board services that are springing up everywhere. Many systems, including several from IBM, use ASYNC either as a primary or secondary approach for communications. All PS/2s are equipped with ASYNC capability as part of the base product and CM provides essential ASYNC support. IBM plans to extend support to include the DMA ASYNC capability provided in the PS/2 Models 90 and 95.

From the preceding it should be evident that CM has a broad range of networking capabilities that allow communications across a range of IBM systems, as well as many heterogeneous non-IBM environments.

The applications can be used simultaneously with those developed by others, without having to take any specific action other than designing and developing the application with CM. This function, along with the rich networking capabilities described in the preceding section, represent a robust workstation communications offering.

### Terminal Emulation

Three distinct families of terminals may be emulated using the functions of CM on a PS/2 workstation. They are: 3270 to a System/370 host, 5250 to an AS/400 host, and ASCII to a variety of IBM, and selected IBM-compatible and non-IBM, hosts.

*Multiple 3270 sessions  
can be active at any one  
time.*

**3270 Emulation:** Interactive access to a System/370 host is supported through 3270 terminal emulation using the IBM LU2.0 protocol and can be defined to emulate the more commonly installed terminals. Connection may be via any of the previously described links, with the exception of ASYNC. In the case of LAN connections, some form of gateway is required (this is described in the section "LAN Gateways").

All 3270 base data stream functions are supported together with extended attributes and extended data stream functions (with seven colors and extended highlighting, plus online choice of fonts for the EGA, VGA, XGA, and IBM 8514/A displays).

The 3270 terminal emulation supports multiple interactive screens and keyboard remapping, as well as Emulator High Level Language API (EHLLAPI), which is described in more detail under the API section.

This means that 3270 terminals may be replaced by PS/2s, and the 3270 programs may be run with no loss of function. However, considerably more advantage can be taken of the power of the PS/2 compared to its fixed-function terminal counterpart.

For example, Presentation Manager windowing provides the ability to save and restore panel characteristics and clipboard editing functions. Windowing allows multiple simultaneous applications typical of OS/2, as well as applications in other windows that can be monitored at the same time that 3270 emulation is running.

The editing functions allow data to be transferred among applications running in different panels. Simple text, text with attributes, and bit images are supported for the mark, cut, copy, and undo operations, whereas simple text is supported for the paste operation. This means that information can be combined from different sources for purposes such as input to a report.

Multiple 3270 sessions can be active at any one time. Each of these sessions can support terminal emulation, host-directed print, or file transfer, and they can use an API (EHLLAPI or Server-Requester Programming Interface [SRPI]) at any time regardless of the method of connection. Each active 3270 session runs in its own window, and selecting between sessions is done with the mouse or keyboard. This means that a user may be servicing a payroll enquiry, transferring a file con-

taining a report, and printing the latest price list, all at the same time. This capability can enable substantial productivity gains.

An active 3270 session can be used to move data between a workstation and a System/370 host (in either direction). This file transfer capability provides appropriate translations (if required) when data is exchanged with the host. File transfer to and from an MVS/TSO, VM/CMS, CICS, or VSE/SP host can be performed through any active 3270 session; the only prerequisite is that the IBM 3270 PC File Transfer program must be installed and made accessible on the host computer. (*Note:* IBM plans to make long file names – up to 255 characters – already supported in OS/2 SE, part of 3270 file transfer support in the future.)

Multiple file transfers can be performed concurrently. Because 3270 file transfer may be going on in the background while the user is free to continue with other work, this is a powerful and efficient method of shipping files, reports, and documents around an organization.

Communications Manager also contains a keyboard remap function that allows the user to modify the default keyboard layout in a 3270 terminal emulation session. The remap facility supports key swapping, key disabling, and assignment of a string of keystrokes to a single key combination (accelerator keys). With these facilities, the keyboard layout can be arranged to be consistent with a previous application, keys rearranged to put those most frequently used at the most accessible positions, or frequently used combinations reduced to a single keystroke. These functions all enhance productivity.

3270 Emulation supports 3270 host-directed print, 3270 Graphics support enabling, and Presentation Space Print (3270 local copy). 3270 host-directed print allows (LU1, LU3, and non-SNA) printer data streams to be printed at the workstation printer. Multiple printer sessions are supported. 3270 Graphics support works with the GDDM-OS/2 Link program product which adds graphics support to the 3270 emulator. This allows the workstation to function as a GDDM mainframe graphics terminal. In addition, GDDM pictures may be printed and plotted or saved to a Presentation Manager metafile. Presentation Space Print may be either host- or user-initiated. The entire Presentation Space or a user-selected portion may be printed. All these facilities extend the scope of the workstation to participate in graphics- or print-oriented applications in addition to interactive ones.

In summary, 3270 Emulation support provides all the essential function of the 3270 terminals and also adds many more functions from the OS/2 platform.

**ASCII Terminal Emulation:** Two ASCII terminal emulators are provided. OS/2 workstations connected through an asynchronous link to a suitable host can emulate either of these terminals: IBM 3101 (Model 20) and DEC VT100.

These terminal emulators provide:

- Asynchronous access to a System/370 or System/390 host through a protocol converter
- Access to other IBM or non-IBM hosts that support either IBM 3101 or DEC VT100 terminals
- Access to a range of network data services, such as news, mail, and other online information services.

The asynchronous link can be switched, leased, or direct, and is compatible with the 1984 CCITT V24/V28 (RS232C) recommendations as implemented by IBM. Multiple asynchronous sessions can be configured, and up to three asynchronous ports may be used. However, only one ASCII terminal emulation session can be active at any one time. The active asynchronous session runs in its own OS/2 panel.

The ASCII terminal emulators support 7-bit (any parity) and 8-bit (no parity) asynchronous character data streams. The modem command strings provided for the explicitly supported modems, or their equivalents, may be edited by the user. This allows support for a variety of modems with different commands.

It is also possible through the use of a single key to have a “snapshot” copy of the display screen contents saved on a logfile. An “ABC” switch can be connected to any asynchronous communications port to provide user switching between a modem and other serial I/O devices (such as printers and plotters). The port is shared on a sequential use basis.

File transfer is supported and occurs through the active asynchronous terminal emulator session; therefore, only one asynchronous file transfer can be active at a time. However, the asynchronous file transfer can be concurrent with 3270 file transfers.

There are three ways that files can be transferred:

- Asynchronous file transfer to an IBM host

When an ASCII terminal emulator is running via a protocol converter to a System/370 host, files can be transferred using the IBM

3270 PC File Transfer Program. This program includes 4-byte CRC error detection. The host must be running MVS/TSO, VM/CMS, or VSE/SP and the file transfer program must be installed and accessible.

- Asynchronous transfer of files using the XMODEM protocol

Files can be transferred using XMODEM protocol to any other workstation that supports this protocol, whether IBM or non-IBM. The remote workstation must have a suitable file transfer program loaded when the file transfer is initiated, and the transfer is in 128-byte blocks with one byte checksum error detection.

- Asynchronous file transfer using "Send ASCII Text File"

ASYNCR terminal emulation can be used to send ASCII data from a file, in addition to placing received data into a file.

The default keyboard layouts for 3101 terminal emulation or VT100 terminal emulation can be changed through a remap procedure similar to that for 3270 terminal emulation. The procedure supports key swapping, key disabling, and assignment of a string of keystrokes to a single key combination. Also the clipboard functions of *mark* and *copy* are provided by Presentation Manager. These functions allow data to be transferred from the ASCII terminal emulation window to another application. Simple text, text with attributes, and bit images are supported for these editing commands.

In summary, the ASCII emulation support provides basic ASCII terminal and file transfer capability. It can also support ASCII applications and has an associated API, known as the

Asynchronous Communications Device Interface (ACDI). The ACDI characteristics are described in the API section.

Both the 3270 and ASCII terminal emulators use the Presentation Manager and its windowing facilities to allow user interaction with the system and to take advantage of the online choice of fonts for most displays. Each logical terminal appears in a separate window, which can be individually started, stopped, moved, and sized by the user. Window characteristics can be saved for future use. Because the operating environment can be tailored to suit individual user preferences, users achieve greater productivity.

#### **5250 Work Station Feature and Support for the AS/400 and System/36:**

The 5250 Work Station Feature (WSF), provided by Communications Manager, allows an OS/2 workstation to be used in place of a 5250 terminal or printer, or both. Support is provided for connections to an AS/400 or System/36 for 5250 WSF application programs. These applications may be used unaltered or they may be enhanced through the use of EHLLAPI in a manner similar to that used with 3270. EHLLAPI supports additional functions that can be called from an OS/2 program. (Refer to the API section for more details.)

The 5250 WSF program supports the functions of 5250 display terminals and printers. It supports sessions that make the workstation appear as an IBM 3196 or 3197 display terminal. A full-screen environment is used for the display. Printer sessions emulate the functions of a 5256, 5224, or 5219 host printer on many OS/2 workstation printers. A combination of up to five display or printer sessions is supported concu-

rently. All of these sessions do not have to be with the same host, but can be with any combination of AS/400s and System/36s in the network.

Communications to the AS/400 System allow IBM Token-Ring, X.25, Twinaxial, and remote connection via the IBM 5294 Remote Control Unit links, in addition to SDLC links. System/36 supports X.25, SDLC, and IBM Token-Ring. All these use LU6.2 protocols.

Unlike the other two terminal emulators under Communications Manager, 5250 WSF does not support file transfer between the workstation and the host computer. However, file transfer support is available using the AS/400 PC Support Program (release 2.0 or later) on an AS/400 or System/36 host. The AS/400 PC Support Program also provides shared folders, virtual printer, text assistance functions, and other useful facilities. Shared folder support allows workstation files and programs to be stored on the host disk and accessed as a normal workstation disk drive. Virtual printer allows output from workstation programs to be printed on host printers. Text assistance functions are required when using the AS/400 Office functions on a workstation. The AS/400 PC Support program should be installed if the user needs these functions.

The 5250 WSF program allows the AS/400 and System/36 terminals to be replaced with PS/2s. The use of PS/2s brings the power of the workstation directly to the operator. This support may be in operation simultaneously with other CM applications and emulators. Many organizations use IBM mid-range systems such as the AS/400 for their departmental host and System/370s for headquarters functions. The CM enables them

to interact with both hosts simultaneously in order to be responsive to business demands.


## LAN Gateways

In the LAN environment, individual workstations do not need their own communications adapters and modems. They can share those of a "gateway." A gateway is a separate box on the LAN that directs the data traffic passing through it from workstation to host (or another system) or vice versa. CM workstations can access gateways to System/370 architecture hosts, other hosts supporting LU6.2 protocols, and ASCII hosts. By using a gateway, line and modem costs can be reduced.

Traditionally, access to a System/370 host has been via a controller such as the 3174, which acts as a concentrator or a gateway. In the LAN environment, CM provides gateway functionality within the PS/2. The CM SNA Gateway support allows access to an IBM System/370 host by multiple users attached to the gateway via an IBM Token-Ring, IBM PC Network LAN, ETHERAND LAN, SDLC switched link, or an X.25 network. The link between the gateway and the host may be SDLC, X.25, or an IBM Token-Ring.

The gateway PS/2, which does not have to be dedicated to the gateway task (in contrast to the controllers), appears to the host as a single physical unit (PU2.0) with up to 254 logical unit (LU) sessions that may be shared among the workstations. Up to 256 workstations may be configured on the LAN, with 64 active at one time, each with multiple LUs. (Note that IBM plans to increase the number of active workstations.) The workstation appears to the user as if it were directly attached to the host.

LUs may be pre-assigned to workstations; or they may be "pooled" to allow greater efficiency in their allocation among workstations, and to reduce the configuration and start up requirements in the host. The protocols supported by the gateway between the workstation and the host are LU0 (often used in specialized industry applications), LUs 1, 2, and 3 (which represent 3270 operations), and LU6.2, which is the strategic distributed processing protocol.



### *CM workstations can access gateways to System/370 architecture hosts, other hosts supporting LU6.2 protocols, and ASCII hosts.*

In most environments, LAN workstations may use the same gateway when operating with:

- OS/2 EE
- IBM Personal Communications/3270
- 3270 Emulation Program Version 3.0
- 3270 Workstation Program Version 1.1
- APPC/PC Version 1.11

In summary, use of the CM gateway provides an efficient way to allow workstations to communicate with the System/370 host. In fact, it is so efficient internally that in many scenarios the workload associated with the gateway function will leave ample capacity for other services to be

accomplished on the same system. LAN Server or Database Server are likely candidates to be used there. The use of a gateway does not disrupt applications written for a direct-connect environment, provided the user system is properly configured.

Communications Manager does not provide a LAN APPN gateway to other IBM hosts; but a complementary product – the IBM SAA Networking Services/2 program (NS/2) – may be used in this capacity. By configuring the gateway as a PU2.1 node, links may be established to any IBM system supporting LU6.2 protocols. This allows LU6.2 sessions between the LAN workstation and the host. For example, the 5250 WSF could be used on the workstation through the gateway to an AS/400. In the future, IBM plans to make all the function of NS/2 available directly within CM.

For asynchronous connection to off-LAN hosts, databases, and services, a CM-based workstation can use the dial-out capabilities of the IBM LAN Asynchronous Connection Server (LANACS) Version 2.0. This provides line- and modem-pooling facilities for OS/2 EE and DOS workstations with the appropriate level of support. In this case the DOS-based LANACS gateway does have to be dedicated to the gateway function. Note also that the OS/2 workstation can use it only to dial out from the LAN; a remote OS/2 workstation cannot dial in.

## Application Programming Interfaces (API)

CM has several APIs used by the OS/2 communications facilities and available to the application programmer. These APIs support a wide variety of communication functions between workstations and hosts, as

well as between LAN workstations functioning as servers or requesters.

The EHLLAPI interface provides programming access to the workstation applications of 3270 terminal emulation or the 5250 Work Station Feature. EHLLAPI allows complex functions to be manipulated under program control, while the user sees the process as a single function. In this way, interactions with the host can be simplified and made easier to use. Another advantage of EHLLAPI is that data can be processed before being transmitted to the host application, thereby taking advantage of the power of the intelligent workstation.

EHLLAPI can also be used to perform an automatic logon, to intercept and respond to host messages, or to translate keystrokes into more complex requests. This interface can currently be called from IBM C/2, COBOL/2, PASCAL/2, BASIC/2, and Macro Assembler/2 languages. IBM plans to enhance EHLLAPI and increase flexibility by allowing more than one program to access it simultaneously, including calls from DOS programs running in one of the Multiple Virtual DOS machines under OS/2 SE 2.0. Also, the interface itself is planned to be extended to support structured fields that will facilitate such operations as file transfer.

APPC, together with its SAA counterpart, CPI-C, is IBM's strategic API to provide distributed transaction processing capabilities. An APPC application consists of two programs, usually at different SNA nodes, that cooperate to carry out a particular processing function. Both programs can exchange data and thus share each other's local resources such as processor cycles, databases, and work queues, as well as

each other's physical components such as keyboards and displays. APPC uses the LU6.2 interface architecture. The LU6.2 architecture has been developed to provide a peer-to-peer communication protocol between SNA applications (intelligent systems). LU6.2 allows sophisticated communication between workstations and other systems for distributed function processing.

LU6.2, as defined in the APPC architecture, is a particular type of SNA logical unit. Each LU provides a connection, or port, between its application programs and the network resources available to its programs. The resources may be physical, such as processor machine cycles and disk or tape files, or logical, such as sessions and queues. Some of these resources are attached to the same LU as the program. Other resources are remotely attached to other LUs.

APPC provides the following:

- A programming interface for basic and mapped conversations
- A confirm synchronization level
- Security support at session and conversation level
- Multiple LUs
- Parallel sessions with the ability to change the number of sessions with remote systems
- Concurrent multiple links

Currently, APPC applications under OS/2 EE cannot utilize a coaxial attachment to a 3x74 controller. IBM plans to enhance support for OS/2 workstations COAX-attached to a suitably configured 3174 to allow APPC applications to run over the COAX link. These applications will be able to communicate to other workstations similarly attached to the 3174, to whom the connection will appear as a direct LAN link.

The COAX-attached workstation can also utilize the 3174 as a gateway to a host or a bridge to a Token-Ring LAN attached via the Token-Ring adapter.

*Note:* SAA CPI-C is a variant of APPC that is consistent across the SAA family. Programs written to CPI-C may be easily migrated from one family member to another, for example, from an OS/2 platform to an OS/400 platform. CPI-C is not supported directly by OS/2 but is supported by the complementary product IBM SAA Networking Services/2. In the future, this product, which also has APPC performance improvements and an extension to the APPC interface, is planned to be incorporated into Communications Manager. This interface can currently be called from IBM C/2, COBOL/2, PASCAL/2, and Macro Assembler/2.

**SRPI:** This is the programming interface for the Enhanced Connectivity Facilities. It enables the writing of simple, communications-independent, requester programs that can call to host server programs, with synchronous returns. It is supported over links using LU2 protocols. Host server support is available under MVS/TSO and VM/CMS. Although SRPI is sufficient for a large number of tasks, APPC should be used if more extensive capabilities are needed. SRPI can currently be called from IBM C/2, COBOL/2, PASCAL/2, and Macro Assembler/2.

X.25 is an SAA common communications support protocol. Before describing the API associated with it, the scope of CM support for this important facility will be outlined. OS/2 EE X.25 Packet Switched Data Network (PSDN) support allows a PS/2 (Model 50 or higher) equipped with one or more IBM X.25 Inter-

face Co-Processor/2 adapters to attach to one or more X.25 PSDNs and communicate with other systems or hosts having appropriate X.25 support. Connection to public and private networks conforming to CCITT 1980 or 1984 X.25 recommendations is supported. Multiple IBM X.25 Interface Co-Processor/2 adapters are supported, depending on the available slots in the system unit. The software enables each adapter to offer either an X.21, X.21bis/V.24, or an X.21bis/V.35 interface, and support speeds up to 64K bps. The software can support a mixture of up to 128 Switched Virtual Circuits (SVC) and Permanent Virtual Circuits (PVC). SNA Communications is supported by the Qualified Logical Link Control (QLLC).

**X.25 Non-SNA API:** This interface at OSI level 3 provides facilities for appropriately programmed non-SNA Data Terminal Equipments (DTEs) to communicate with each other across one or more X.25 connections. Both SNA and non-SNA communications are possible over the same physical link, and up to 40 X.25 applications (SNA and non-SNA) can execute concurrently in a single workstation. The interface is currently supported by IBM C/2, PASCAL/2 and Macro Assembler/2. Thus the API allows non-SNA (in addition to SNA) applications to operate over the X.25 PSDNs.

**ACDI:** This interface is provided to allow the writing of applications (such as asynchronous emulators or file transfer programs) to exchange data over asynchronous links. The interface provides a high degree of independence from the asynchronous hardware used. Device-specific programming modules are required for each supported device type and are included in the product. They are transparent to user applications. Sup-

ported functions include the ability to manipulate the line characteristics and connection control (connect and disconnect) without having to deal with physical device-specific characteristics.

ACDI may be used to manipulate modem command strings to customize and automate dialing procedures, or to use an alternate modem. One ACDI application can share a V.24 (RS232C) ABC switch with serial devices such as printers or plotters. Recent enhancements allow the redirection of ACDI calls (across a LAN) to an ASYNC gateway, for example. This is currently done from the command line but IBM plans to make it accessible from programs for greater flexibility. ACDI is currently supported by IBM C/2, PASCAL/2, and Macro Assembler/2.

#### **Local Area Network (LAN)**

**Programming Interfaces:** Both IEEE 802.2 and IBM NetBIOS are supported for program-to-program communications. The IEEE 802.2 protocol is a lower-level communications interface. The APPC support provided by CM uses the IEEE 802.2 interface to access the LAN. Application programmers may access the IEEE 802.2 API for greater control where higher performance is critical. The NetBIOS programming interface provides an additional interface for communication across a LAN. NetBIOS is a name-oriented program-to-program interface that may be used for application restricted to the LAN environment. The NetBIOS protocol can be used concurrently with the IEEE 802.2 LAN access. Both of these APIs are currently supported by IBM C/2, PASCAL/2, and Macro Assembler/2.

IBM plans to restructure the OS/2 LAN Transport function to improve the performance and resource utiliza-

tion. The Network Device Interface Specification (NDIS) will be provided to support high performance media access control device drivers for Token-Ring, Ethernet, PC Network, and LAN Over Coax connectivity. In addition, new NetBIOS and 802.2 protocols will be included that support the NDIS interface and programming interfaces, and provide performance improvements.

#### **Conventional LU Application**

**(LUA) Interfaces:** These may be used for:

- Communications support for LU0 terminal functions
- A migration path for LU0-based applications
- Workable replacements for LU0-based control units
- Base communications for LU0, 1, 2, and 3 emulators (this is in addition to the LU1, 2, 3, and 6.2 support already available in OS/2 EE)

The LU0 protocol allows greater programming flexibility than the other supported LUs and has been widely used in the finance and retail industries. The recommended protocol for future applications is LU6.2. CM support for LU0 is to allow existing programs to continue operating during the migration period.

LU0 communication is enabled through the use of two programming interfaces. The Session Level Interface (SLI) is a higher level data stream interface that will normally be used by customer applications. The SNA Request Unit Interface (RUI) is at a lower level and is intended for system programming but is also available for customer use. Programming language support for both SLI and RUI is currently from the IBM C/2, COBOL/2, PASCAL/2



and Macro Assembler/2 compilers. Collectively, SLI and RUI are referred to as conventional LU application (LUA).

#### **Common Services Interface:**

CM provides functions for gathering and processing problem determination data. These functions include tracing of programming interfaces, data units, and/or system events; displaying and printing of all or selected error logs from file; system dumping; and displaying of all or selected message logs. These services can be invoked from the console or via a Common Services API, which allows all these Communications Manager RAS functions (trace, dump, errors, and messages) to be performed under program control and not require an operator. A user-written program can monitor errors and messages selectively, and take appropriate action. Message pop-ups on the screen can be suppressed. The same API also supports ASCII/EBCDIC conversions, code pages conversion, and the transfers of diagnostic data to a host. For example, applications can use this service to alert a NetView® operator of conditions requiring action.

#### **Subsystem Management Interface:**

CM allows a system administrator to control and obtain status information on the SNA communication resources maintained by CM. As a management tool, it displays the programs being used, the sessions being used by the programs, detailed information about the sessions, and resources that are active. It allows the activation or deactivation of sessions and data link controls. It also can be used to start and stop an attach manager that allows applications to be remotely started. These services are invoked in some cases from the console and in others via an API, as appropriate.

## **Installation and Configuration**

Configuring (tailoring the required components) and installing Communications Manager is accomplished by one of several options.

The Custom Install option enables the system or network administrator to design an installation diskette that installs only those components and features required at each workstation. A Custom Install diskette also provides the configuration file needed for the installed components and features. This option is particularly beneficial to novice users or when installing OS/2 Extended Edition on multiple workstations, because it allows the installation to be completed with minimal user interaction. Because each user does not need to have all the functions installed, the amount of memory and disk storage required at the workstation can be reduced.

The Basic Configuration Services option assists users who do not have the support of a system administrator and who are relatively inexperienced in communications. It helps them install the program and have it running for a number of communications environments. Basic Configuration Services may be used for one or more of the following:

- 3270 emulation, directly connected via SDLC, LAN, or DFT
- 5250 Work Station Feature connected via Twinax, SDLC, or LAN (Token-Ring only)
- ASCII emulation to single host
- Database Manager's Remote Data Services configuration between one OS/2 Database Requester and Server

Basic Configuration Services may be supplemented with Advanced

Configuration Services to cover other environments and additional Remote Data Services connections as required.

For those not using a Custom Install diskette or Basic Configuration Services, a process known as the Advanced Installation can be used. With Advanced Installation, the components and features are selected in a certain sequence. In addition to permitting the selective installation of various OS/2 components, the installation program allows the addition, removal, or reinstallation of OS/2 components and features at any time after the initial installation has been completed.

Finally, for system administrators with large numbers of similarly configured workstations, a Batch Configuration utility is provided. Basic or Advanced Configuration is used first to establish an individual representative prototype configuration file. Then, the system administrator creates an ASCII file that contains the essential differences between individual end users. This file and the prototype are processed by the utility to build a configuration file for each user. The support allows the creation, modification, and deletion of a subset of the parameters for the profiles in the following areas:

- Workstation Profile
- 3270
- SNA/APPC
- SDLC
- Twinax
- X.25
- SNA Network
- SNA Gateway
- LU0
- 802.2
- NetBIOS

Typically, the files would be generated under program control for use with the Custom Install option or one of the more automated forms of install offered by complementary program products such as NetView/DM and the SAA Delivery Manager.

IBM plans to open up the entire configuration file to programmed access, to further facilitate the configuration and installation of large numbers of workstations.

### Network Management

CM provides extensive network management support.

When errors occur at the workstation level, messages can be sent to the operator and are logged for subsequent analysis. Trace and dump facilities are provided as diagnostic tools. Online help is available to assist in local problems resolution, or to suggest contacting an administrator or service coordinator as appropriate.

At the LAN level, network management may be facilitated by the IBM LAN Manager Version 2.0. This program uses CM for sending and receiving information across the LAN and monitors the activity and status of the workstations and the links. An alternative approach is to use NetView PC when NetView PC is installed.

With wide area network links and a System/370 or System/390, the IBM NetView program at the host computer can be used. This provides a central point for control. CM sup-

ports a NetView-managed network by automatically forwarding alerts generated by the hardware or software associated with data links. Support for SDLC, asynchronous, Token-Ring, ETHERAND, PC Network, twinaxial, and X.25 connections is included. CM requires APPC to be installed on each workstation for the forwarding of alerts, and the links to the host must be SDLC or Token-Ring.

The NetView operator can also request vital product data pertaining to the workstations, such as machine type, serial and model number, and the name, version, release, and modification of the installed OS/2 EE software components.

Exchanges with the NetView host are accomplished by the Common Services Interface (CSI). This is a published programming interface that can be used by user applications to notify (through the alert mechanism) a central monitoring facility. The CSI may also be used to remotely invoke diagnostic tools, such as running traces and dumps, without involving the workstation user.

IBM plans to extend this support to allow the NetView operator (or a program at the NetView host using CLISTs or REXX execs) to issue most OS/2 commands remotely for execution at the PS/2. This will provide powerful manipulative and LAN management capabilities from the host.

In APPC application environments, subsystem management is provided to the system administrator to con-

trol and obtain status information on the resources maintained by CM. It allows the activation and deactivation of sessions, and supports the remote starting or stopping of the attach manager.

### Conclusion

Communications Manager is IBM's strategic workstation-based communications offering. It is accepted as the platform of choice by many IBM workstation-based products, including OSI/CS, TCP/IP, CICS, and Office Vision. CM is designed for use within the SAA framework. It provides many concurrent networking capabilities, which represents significant productivity savings for both developers and users. CM is the development and distributed processing workstation communications platform for the 1990s.

### ABOUT THE AUTHOR

*Rich Harrison is a senior product planner at IBM's Personal Systems Programming Center in Austin, Texas. He has been associated with Communications Manager since its inception. Rich joined IBM in 1956 and has held a variety of professional and management positions in systems engineering, application development and, more recently, communications software development. He holds a B.A. in mathematics from Cambridge University (United Kingdom).*

Much of this information concerns future products or future releases of products currently commercially available. The description of IBM's future products, performance, functions, and availability are based on IBM's current intent and are subject to change.

## Improving OS/2 Application Performance

Mark W. Brooks and  
Gary S. Kaufman  
IBM Corporation  
Markham, Ontario, Canada

**This is a project history. It describes the experiences of a team assembled to improve the performance of an OS/2 application under development.**

The development team of a medium-size (45 KLOC) OS/2 project identified performance as a problem that would inhibit satisfaction with the product.

Performance goals were set for the product using usability studies and other benchmarking information. At the highest level these goals were to:

- Reduce the elapsed time taken to perform the most important product functions
- Reduce the amount of system resources used by the application

A performance team was formed that achieved a significant performance improvement: an overall reduction in elapsed time of more than 300 percent across the product, and as much as 600 percent in several instances.

While the performance team worked in a specialized OS/2 environment, and the following is specific to that

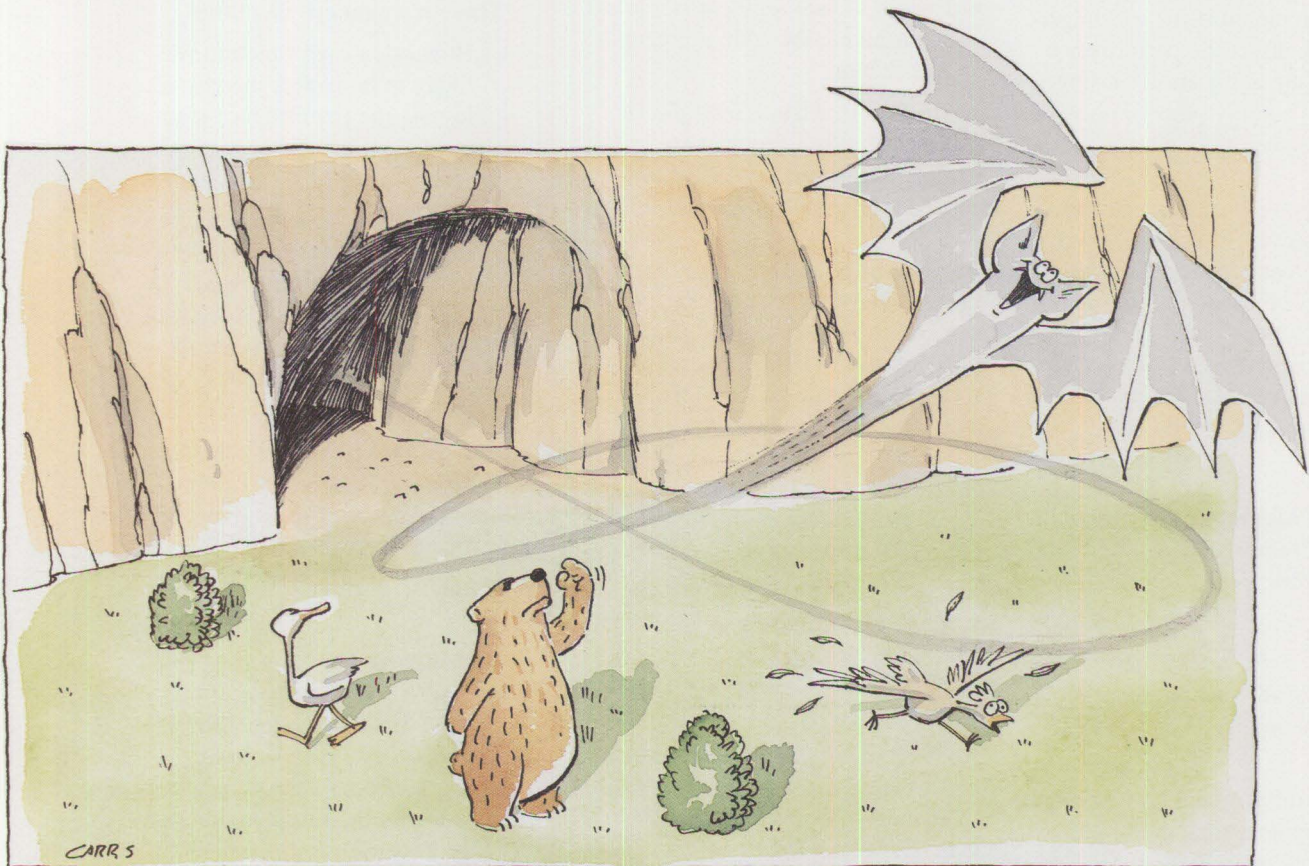
environment, the information presented in this discussion might prove valuable to any technical group concerned with performance issues.

### Objectives

To reduce performance problems, the team believed it was necessary to minimize memory working set and maximize speed.

**Memory Working Set:** *Memory working set* is the minimum amount of memory required to run a particular scenario without frequent swapping and still achieve acceptable performance.

This means a given working set measurement is *always* scenario-dependent. The size of the application's working set was one of the



key factors in its overall performance.

The working set of an application consists of a series of code and data segments that need to be loaded into memory while a function is executing. In OS/2, a code segment is normally loaded into memory when a function in the code segment needs to be executed by the process. Once the function in the code segment has finished executing, the operating system can discard the segment if it is short of memory. Therefore, the larger the working set of the application, the more likely it will need to discard and reload code segments into memory from the much slower fixed disk. When the operating system does not have enough memory to hold all the code segments, it will be forced to constantly load and discard code segments. This situation is called "overcommitment" (or thrashing) and can affect the performance of not only the application causing the problem but any other application that may be running.

A similar problem occurs with data segments; however, instead of being discarded, a data segment is written out to and, when required, reloaded from the OS/2 swapper file. Therefore, a swapped data segment is at least twice as costly, in terms of disk I/O, as a similarly sized code segment.

#### Measuring Memory Working Set:

The MAP file produced by the OS/2 linker provides:

- An approximate size of your code and data segments
- An idea of the maximum possible size of an application's working set
- A list of the size and type of application segments that could be loaded into memory

Using the sum of these application segments, as well as the size of any data segments allocated dynamically within the program, it is possible to calculate the maximum possible amount of memory required by an application to run without swapping or reloading. It is unlikely that this *maximum* memory requirement will be reached as only a fraction of the total code and data segments should ever be loaded at once in a well-structured application.

The working set calculation is more complicated if multiple processes are sharing code or data segments. In general, the following rules apply:

- If multiple copies of an application are loaded and running simultaneously, only one copy of each code segment is loaded and shared between the processes
- Even if a code segment is marked as unshared, the code segment is still shared
- On the other hand, multiple versions of a data segment are loaded unless it is marked as "read-only" or non-shareable

To get a more accurate working set measurement, the performance team used an IBM internal tool with functions similar to IBM System Performance Monitor/2 (SPM/2) Version 1.0.

In summary, the larger the working set of an application, the longer it takes to run. An application with a large working set also affects the speed of other executing applications.

### Strategies for Improving Performance

Because the opportunity to improve performance came so late in the development cycle, the performance team was restricted by time limita-

tions. Therefore, the team initially focused on problems with the most commonly used code.

**Identifying Problems:** The first step towards performance improvement is to identify the bottlenecks in the code. To do this, the performance team used several tools:

- Simple file dump

First, the team used a simple file dump system. In this system, a statement was inserted at the entry point of every function and at every exit point throughout the entire function. The entry dump wrote the name of the function and an identifying number and started a timer for that entry point. The exit point ended the timer and wrote the function name and elapsed time to the dump file.

These trace dumps identified:

- How often each function was likely to be called, thereby allowing the performance team to concentrate on the critical code.
- The elapsed time for each function call. While this was somewhat inaccurate, it helped identify heavy processing routines and helped indicate the success of the tuning techniques.

The data from this type of trace system provided a good enough base for many of the initial performance decisions.

- Complex Tracing

The team also tried a more sophisticated IBM internally used tracing system with features similar to IBM SPM/2 Version 1.0. This system gives a precise timing breakdown for a long sequence of actions and their related function calls. Because of the complexity

of this trace data we found it particularly useful when analyzing an already identified problem. Similar systems are also available from non-IBM sources.

Once the problems were identified, the team decided to meet its performance goals by "fine-tuning" existing code, and by redesigning part of the product.

### Reducing the Memory Working Set:

It is important to maintain the working set at the lowest possible level to control and improve the speed of the application. For example, for the first few releases of the product, the overall working set was extremely large. Therefore, even on a loaded 16 MB PS/2, the application could easily slip into a condition known as thrashing (constantly accessing the hard disk to load or swap code/data segments).

To reduce the memory working set, the performance team looked at restructuring code/data segments and tuning dynamic memory allocation and usage.

**Improving Speed:** To improve speed, the performance team examined:

- "Traditional" code tuning
- OS/2 tuning
- Database tuning

Later sections on tuning will discuss these areas in detail.

### Design Considerations

The design of an application is the most important influence on performance. During the low-level design phase of the application development cycle, performance must be considered when making design decisions.

If an application's design inherently dictates poor performance, no amount of code tuning will save it!

### Traditional Code Tuning

The goals of code tuning are to:

- Decrease the number of processor cycles required by the program
- Decrease the number of address calculations performed
- Decrease the number of memory fetches
- Decrease the size of the object code

Traditional optimization deals with improving the compiler translation of C code into the machine language end product.

## *Aligning data structures can make significant performance improvements to an application.*

Because development environments and compilers are now more sophisticated, some of these methods may be automatically implemented by the compiler.

If implemented correctly, the following suggestions should not affect the clarity of the code and should increase the speed of any application running on a PC AT or PS/2.

**Structural Alignment:** Aligning data structures can make significant performance improvements to an application. Some measurements show

that by simply aligning an application's structures, the speed of memory accesses can increase as much as 25 percent over that with unaligned structures.

Structural alignment is so effective because, in many cases, the number of memory accesses required for an operation is cut in half. This is because a memory fetch retrieves four bytes of data at a time, aligned on four-byte boundaries. Therefore, if a four-byte pointer is to be retrieved and the structure was aligned, the pointer could be retrieved in one memory fetch instead of two.

The requirements for aligning structures are as follows:

- The structure has a starting address on a four-byte boundary
- Every element in the structure that is four bytes or fewer in size does not cross a four-byte word boundary
- The structure elements that are greater than four bytes should be started on a four-byte word boundary

Figure 1 shows examples of unaligned and aligned structures.

**Data Conservation:** In most cases, it is advisable when creating data tables or retrieving data from a LAN server or flat file, to hold this data in memory for as long as possible because LAN communications (and disk I/O) are very expensive in terms of elapsed time.

For example, the given product involved a client-server architecture, and in the initial design two round trips to the server were made when in many cases, one would have been sufficient (with a minor design change).

To solve this problem, the team constructed a *refresh cache* that cached some of the server data in certain instances, thereby doubling the speed of many major functions.

**Register Variables:** Tagging a variable as a *register variable* essentially tells the compiler which variables are most commonly used. This allows the compiler to preload several CPU registers with commonly used data.

The variables most commonly tagged were the pointers used to access the major data structures throughout a module. Making this pointer a register variable eliminated the need to load the pointer from memory each time a structure element was referenced. Making this pointer a register variable increased speed by up to 20 percent in some routines.

Other heavily used variables, such as loop counters, were also tagged as register variables.

A small decrease in the size of the code produced by the compiler is another added benefit of the appropriate placement of register variables. This reduction averages about 1 percent for the entire application.

Tagging a variable as a register variable is completely portable to other machine architectures.

While some developers believe that the IBM C/2 compiler will properly place register variable tags automatically, the performance team felt it was necessary to place the tags manually so that no variables were missed. For example, any variable that has a pointer placed in front of it cannot be tagged by the compiler as a register variable, because if the compiler must take a pointer from a

Example of an unaligned structure:

```
struct nonal{
    int  one;      /* two bytes in size */
    long two;     /* four bytes in size */
    int  *three;  /* four bytes in size */
    long four;    /* four bytes in size */
}
```

Example of the above structure as an aligned structure:

```
struct al{
    long two;     /* four bytes in size */
    int  *three;  /* four bytes in size */
    long four;    /* four bytes in size */
    int  one;     /* two bytes in size */
    int  buf1;    /* two bytes in size */
}
```

or

```
struct al{
    int  one;     /* two bytes in size */
    int  buf1;    /* two bytes in size */
    long two;    /* four bytes in size */
    int  *three;  /* four bytes in size */
    long four;   /* four bytes in size */
}
```

Figure 1. Unaligned and Aligned Structures

variable, this variable must have a location in memory. To tag this variable as a register variable, the code should be modified to use a temporary pointer value.

#### Intrinsic Function Calls:

It is possible to force the compiler to place system (library) functions inline, rather than making a function call. This eliminates the overhead inherent in function calls.

In this product, the compiler was informed of what system functions to make intrinsic, using a set of pragma statements at the beginning of each source code file (Figure 2).

The declaration of the intrinsic functions marginally increases the size of the final application, but it is worth the space if the application is moving or copying small amounts of

```
#ifndef CODEVIEW
#pragma intrinsic(memcmp,memcpy,memset,strcmp,strcpy,strcmp, strlen)
#pragma message("Code now optimized for speed ")
#pragma loop_opt(on)
#endif
```

Figure 2. Pragma Statements

memory with each strcpy or memcpy call.

See the C compiler manuals for more detail about these statements.

**Loop Optimization:** In a “for a while” loop, it is possible to increase speed by restructuring the compare statement of the loop.

Comparing a variable against a number or another variable requires three separate machine language instructions:

#### Scenario 1

- Load the first variable from memory
- Load the second variable (or static number)
- Compare the variables

If, instead, you compare one variable against the value 0, the compiler will:

#### Scenario 2

- Load the first variable from memory
- Compare the variable to 0

An example of these is:

#### Scenario 1

```
for (i = 0; i <= 32; ++i)
```

For this comparison the C/2 compiler produces five assembler instructions and would be slower than:

#### Scenario 2

```
for (i = 32; -i >= 0;)
```

For this comparison, the C/2 compiler produces four assembler instructions.

This optimization works best for small, tight loops (loops that contain only a few statements). If the loop is large, the loop processing time far outweighs the termination condition processing.

In addition, all loops should be optimized such that any “loop invariants” are moved outside of the loop. A “loop invariant” is something that does not change throughout the life of the loop. For example, a statement such as “length = 10” should not appear inside a loop, because it is completely invariant and slows down the entire loop by repeating the identical action multiple times.

Some loop optimization was done in the code, but because the code did not contain many “tight” loops, it had no measurable impact on the performance of the application.

#### Array Index Optimization:

When using subscripts into arrays, for every index operation the compiler produces assembler code that:

- Loads the pointer to the array
- Loads the value of the subscript (if it is a variable)
- Adds the two to find the index variable

It is best to avoid array subscripts whenever possible. This can be done inside loops or other situations where pointers and pointer arithmetic can be used instead.

For example:

```
for(i = 0; i < 320; ++i)
array(i) = i;
```

is slower than:

```
cp = &array(0);
i = -1;
for (j = 320; -j >= 0; )
*cp++ = ++i;
```

**Code Path:** There are several issues regarding code paths to consider for performance improvements.

- Reduce path length

The path length of the code is an important performance factor. The more function calls (or PM messages) that a particular code path invokes, the slower the application will run. This is because of the inherent overhead in function calls/returns and message processing.

- Remove dead code

This is code that is in the application but is never executed. This code increases the working set (and therefore decreases the speed) of the application.

- Avoid identical logic

Duplicating a function or piece of logic that already exists can also contribute to performance problems. During the final code reviews for the product, several functions were discovered that duplicated, or almost duplicated, other existing functions.

**Return Codes:** Return codes from both system and application calls should always be checked for errors. If, however, there is a case where this does not apply, then do not code a statement such as:

```
rc = func();
```

if the return code will not be checked. The additional assignment to **rc** causes unnecessary statements to be executed and has a detrimental impact on performance.

**Levels of Indirection:** When using several levels of indirection (pointers), each pointer in the indirection chain must be loaded and examined to get to the final data item. Therefore, some CPU cycles can be saved

if a temporary variable is used to hold a data item that is repeatedly accessed.

For example, if the statement `Var1->var2->var3 = x;` is repeatedly executed (inside a "for" loop, for example), replace the statement with:

```
pVar = &Var1->var2->var3;
.
.
.
*pVar = x;
```

While this type of tuning can help the performance of the application, it can also make the code harder to understand.

#### Variable Initialization:

Unnecessary initialization should not be present in application code (for example, filling a structure with nulls and then assigning values to most or all of the structure elements). Instead of filling the whole structure with nulls, consider assigning values to each remaining field if you are already assigning other values to most of the structure elements.

Also remember that a character string only needs to have its first element initialized with a null.

For example, if you initialize a character string with `strcpy(String, "");`, you can replace it with the following:

```
String(0) = '\0';
```

– or –

```
*String = '\0';
```

Also, using "" as a zero-length string causes a pointer into the static data segment to be loaded, which is additional overhead (especially if the data segment is swapped out at the time).

**Other Techniques:** Because of the requirements for easy maintenance and the ongoing development of the product, several code optimization methods were not used by the performance team. This does not mean that such approaches would not be valid for other projects, but simply that in some cases the costs outweigh the benefits.

*When multiple small pieces of memory are required, it is far better to use DosAllocSeg to allocate a large segment of memory and then use DosSubAlloc to break this segment into usable chunks.*

Alternate methods include:

- Supercharging – Using in-line assembler for key functions
- Near pointers – Using near pointers whenever possible to save memory fetches

## OS/2 Tuning

### OS/2 Memory Management:

Minimizing the number of data segments that an application creates is the most important consideration when designing an application's memory management strategy. Multiple small segments can adversely affect both the application's performance and any other applications running in the system.

One of the reasons that multiple small segments have a negative effect is that the machine's memory becomes fragmented as these segments get swapped out. Therefore, the OS/2 memory compaction algorithm must run more often when new memory is required.

Second, if the segments are shared, space is reserved in every process's LDT (Local Descriptor Table) for that segment. This causes a large amount of system overhead for other applications and possible system shortages due to limited LDT space.

Here are some important points to remember when using OS/2 dynamic memory allocation:

- Use DosAllocSeg and DosSubAlloc.

When multiple small pieces of memory are required, it is far better to use DosAllocSeg to allocate a large segment of memory and then use DosSubAlloc to break this segment into usable chunks. This helps minimize the number of segments in the system.

Another advantage of using suballoc is that there is only one segment to be freed.

- Only use the OS/2 memory manager API calls to allocate memory. Never use the C calls (Malloc, Calloc, Strdup, and so forth). The reason is that Malloc, Calloc, and Strdup each allocate one segment for each call, and you end up with a lot of small segments, causing the OS/2 memory manager to thrash.

Also, do not allocate a heap, because Malloc and Calloc do not use it.

- Allocate only the amount required. A segment can always be "grown" (increased in size).



- DosAllocSeg should be a multiple of 4 KB up to 63.5 KB. *Note:* Suballocs can be any size.

Never allocate more than 63.5 KB, because a 63.5 KB segment is swapped (read or write) in one action, while a 64 KB segment takes two swap actions. It is expected that this problem will be resolved with OS/2 2.0.

- Do not use the OS/2 API call *DosMemAvail*. This call currently gives invalid results when OS/2 undertakes memory compaction, or swapping, or the system is overcommitted in its memory use.
- Memory leakage (allocating memory but not releasing it when finished) caused several performance impacts on the product. A careful review of dynamic memory allocation and its release is always a good idea. Functions with multiple exit points are usually the cause of memory leaks.

*Note:* Shared memory has an associated usage count. All processes accessing a shared segment must release it before the memory is freed.

**Segment Packing:** For a large application, it is unlikely that all code segments will be loaded into memory at one time. Therefore, the arrangement and distribution of functions within the code segment has a significant effect on performance.

The linker, by default, packs the code into multiple segments of 64 KB each. This causes problems when swapping (see comments on memory) and usually means that almost all of the code segments are loaded into memory at once (if there is enough memory), giving the application the largest working set requirements possible. This causes thrashing, large swap files, and a slow application.

Code segments should be managed as follows:

- One segment to process messages
- Frequently used routines grouped together
- Related routines grouped together
- Infrequently used routines grouped together
- Pack segments to 4 KB multiples

A first attempt at code segment packing is to split segments along source code module boundaries. This can be accomplished as follows:

- During the compile of each source module into its object file, specify the "/NM SEGMENTNAME" compiler option, where SEGMENTNAME is the name you want to assign to each code segment. If you do not use the /NM flag, the default segment name will be `MODULE_TEXT`, where `MODULE` is the name of the source code file.
- List the defined segment names in the program's .DEF file. For example:

```
SEGMENTS
    FILE1_TEXT
    INIT
    FILE2_TEXT
    FILE3_TEXT
    INITWINDOW
    BLUEMOON
```

Logically grouping functions within the given code segments can improve a segment arrangement. This logical grouping can be accomplished either by physically moving code to different source files or by using the pragma statements.

If you include a pragma statement before the body of each function but after the prototype statement, the C compiler automatically splits the code into the named segment.

In the example in Figure 3, the function "error\_mess" is moved from its default code segment (the name of its source code file) to the code segment called "BLUEMOON." The pragma statement is `ifdef`'ed with the `define` "CODEVIEW" so that if the program is compiled with the CodeView options and the /DCODEVIEW `define`, the code is not moved, and CodeView is able to work correctly.

The segment information at the top of an application's map file, which is produced by the linker (if specified), can be used to analyze the code segment packing. This information consists of a list of all the code and data segments and their sizes.

#### **DynaLink (DLL) Considerations:**

Here are some points to consider when using DLLs:

```
#ifndef CODEVIEW
#pragma alloc_text(BLUemoon, error_mess)
#endif
int error_mess(char *s)
{
    Body of function
}
```

Figure 3. Moving Code into a Named Segment

- Minimize the number of DLLs in the application. Too many separate DLLs can cause performance problems because each DLL (regardless of size) adds to:
  - The initialization time needed to start a process
  - The search time required to load a new code/data segment
  - The system resources/memory used as overhead

*Note:* Too many help files can cause similar problems.

- Do not allocate stack and heap segments from DLLs. A DLL always uses the stack and heap of the calling process that is launched from an EXE file. Although a DLL's stack and heap segments are never used, they are still packed into the DLL's data group along with its static and global variables. Therefore, simply removing the unused stacks from the DLL can reduce the working set considerably.
- Do not declare functions that do not access DLL data as "extern" or "\_loads". This unnecessarily loads the DLL's data segment along with the code.
- Set up the LIBPATH in CONFIG.SYS so that most used LIB directories are near the front of the search path.

Also, do not put non-DLL files in LIB directories because this increases the system search time for DLLs.

- Declare DLLs as LOADONCALL. This delays loading of a DLL from disk until it is actually required, thereby saving memory.
- Note:* This is not valid in all situations, but is a guideline.
- The caller should free DLLs when finished with them.

```

threadfunc()
{
    thread body
    .
    .
    DosEnterCritSec();
    DosSemClear(Waiting_parent_sem);
    DosExit();
}

```

Figure 4. Method for Freeing a Thread Stack

This is usually true, but in some cases it may be better to keep the DLL loaded throughout the life of the application.

#### Multitasking Considerations:

Multitasking is an extremely powerful feature of OS/2 that is all but ignored by many applications. In an application where concurrent processing is possible, spawning multiple threads (or processes, in some cases) can significantly increase the application's performance.

In Presentation Manager (PM) applications, multiple threads are necessary because the main thread (thread 1) should perform *only* message handling (no disk I/O, no semaphore handling, no lengthy processing). Allowing any of the preceding can not only affect performance, but can completely hang the system.

Here are some points to consider in multitasking applications:

- Do not create and destroy threads on demand. Thread and process spawning is expensive in terms of elapsed time. Wherever possible, create a "hot pool" of threads that wait for work and activate when required.
- Thread priority adjustment should be used sparingly. Any thread can raise its own priority, but onepmanship could occur, resulting

in all threads executing at the highest priority. This gives the operating system no opportunity for proper thread management.

- It is important to free the thread's stack space when it terminates. The proper method for freeing a thread stack is shown in Figure 4.

*Note:* The DosExit does an implicit DosExitCritSec. At that point, the parent thread that is waiting on the semaphore can free the thread's stack. This procedure is so that the parent thread does not free the stack until the child thread is completely finished.

#### Minimizing Message Event

**Initialization:** In the OS/2 PM environment, a lot of messages are often generated within an application.

In our application, the number of messages was not the problem; rather, the amount of unnecessary initialization taken to process each message needed improvement. For example, some parts of the code had initialization logic at the beginning of the window procedures. Although the initialization in question was required, it was being executed at the beginning of a WinProc statement before the type of message was determined (and, often, whether or not the WinProc should even process the

message). An example is shown in Figure 5.

In the preceding case, the initialization of the variable Pvar is taking place before the window procedure has determined if Pvar will be used. In this example, Pvar should be initialized inside the switch case MSG1. Even if Pvar is used in several or almost all of the switch's case statements, it should be initialized only when it is needed.

This type of approach to initialization in window procedures will increase the size of the code and may add to its complexity, but it is definitely worth the effort for performance.

**Other Techniques:** Because of the requirements for easy maintenance and the ongoing development of the product, several OS/2 optimization methods were not used by the performance team. This does not mean that such approaches would not be valid for other projects, but simply that in some cases the costs outweigh the benefits.

## Database Tuning

**Reorganizing:** Reorganizing a database is one of the simplest and most effective methods of improving database performance. The "Reorg" operation physically rearranges the rows in each table to optimize search time. When performing a Reorg it is important to choose the most heavily used index as the Reorg key. The Reorg can be done through the Query Manager interface.

After reorganizing the database, it is necessary to perform the database operation "RUNSTATS." This operation updates the system tables that the OS/2 Database Manager uses to

```
MRESULT EXPENTRY WinProc(HWND hwnd,USHORT msg,MPARAM
mp1,MPARAM mp2)
{
    Pvar = *mp1;
    switch( msg )
    {
        Case MSG1:
            <Pvar used here>.
        Case MSG2:
            .
        Default:
    }
}
```

Figure 5. Message Event Initialization

determine optimum search algorithms.

Also, after the first two steps, the application program should be rebound to the database. This updates the access plans stored in the system tables to take advantage of the Reorg.

**Database Design:** Table design has an extremely large impact on performance. An efficient design is the best approach to good database performance.

For example, this project had a unique situation where a database query that required matching of two columns was required. This caused a performance problem when the database exceeded about 3,000 entries. Creating an additional table that coalesced the two fields into one (for search purposes only) seemed to be the best solution. This improved performance by more than 10 seconds on a large database.

**Indexes:** Indexes can also greatly improve database performance, but there are associated costs. Each addi-

tional index on a table requires an update each time the table is changed. Therefore, in many cases, an index improves data retrieval, but negatively impacts table updates. It is important, therefore, to carefully consider the use of indexes. Also, if an index is used, consider the number of columns used in the index; the more columns, the greater the performance impact.

**Database Locking:** Database locking refers to the control of concurrent access to the database. Because this product did not gain much of a performance improvement by changing locking strategies, this technique will not be discussed in detail. For more information, refer to the *OS/2 Database Manager Programming Guide*.

## Conclusion

The performance team found that the most effective techniques for performance improvements were as follows:

- Partial redesign – The refresh cache eliminated many calls to the server

- Memory management
- Database tuning
- Structure alignment

Figure 6 shows the relative effectiveness of each method used to improve the performance of this product. Although this chart can be useful to any project, the effectiveness of each method used will vary from project to project. This variance will depend on the type and size of the product. In general, improving the performance of an OS/2 application depends not on the use of a set of techniques, but on the understanding and effective use of the resources being used by the application.

*The performance team consisted of project leader Yat-Fai Lee, Mark Brooks, Henry Galas, and Gary Kaufman.*

#### ABOUT THE AUTHORS

*Mark Brooks is currently the project leader of an application development project at IBM Canada Ltd. He holds a B.S.C. in computing science from Simon Fraser University in British Columbia, Canada.*

*Gary Kaufman works in OS/2 2.0 application development at IBM Canada Ltd. Gary holds a B.S.E.E. in electrical engineering from the University of British Columbia.*

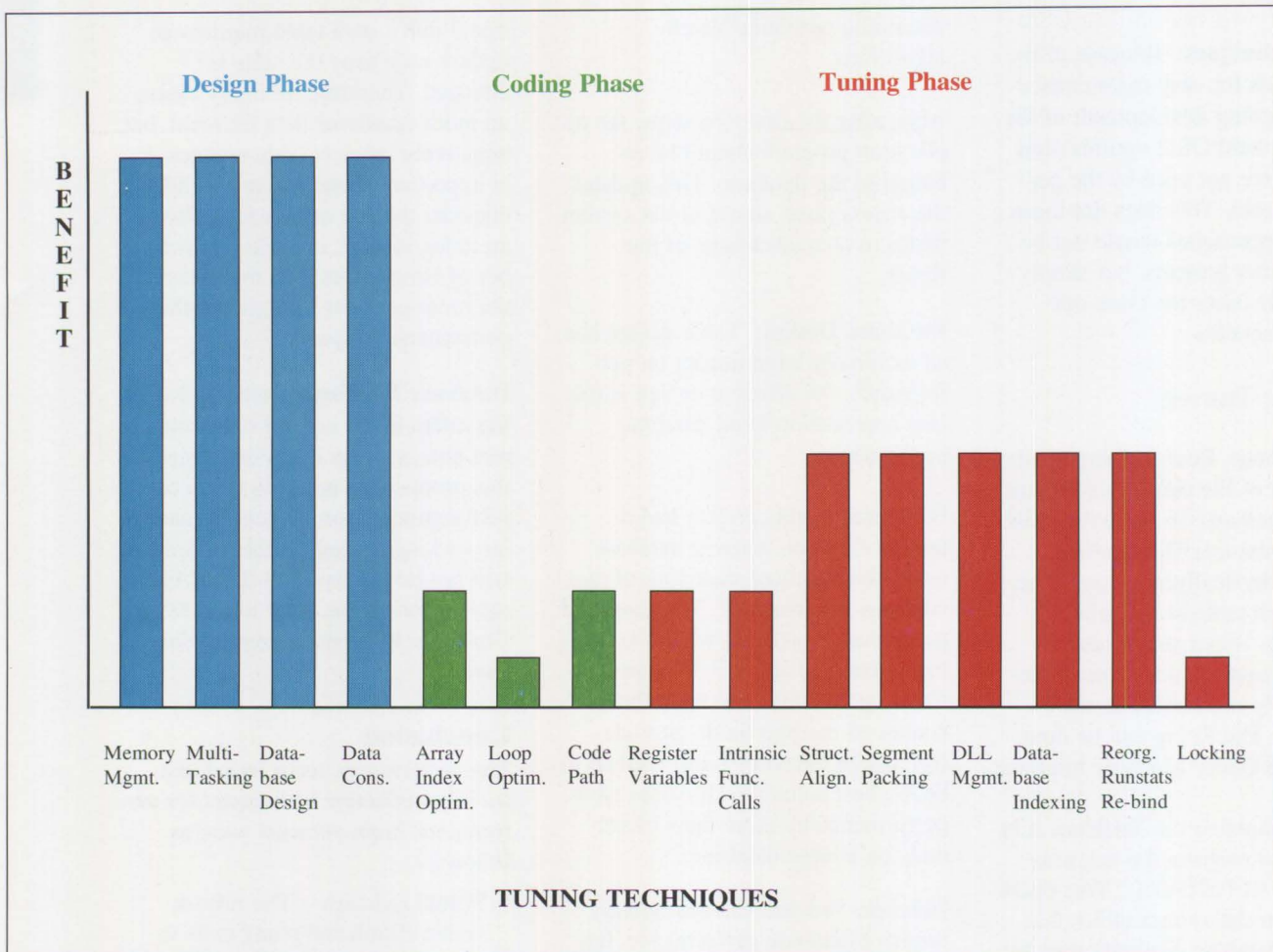


Figure 6. Relative Performance Improvements

## Creating PM Windows with Dialog Templates

Scott Sorensen  
Provo, Utah

There are advantages to using a template structure and the `WinLoadDlg()` function call to define Presentation Manager (PM) windows. This article explains how programmers can use this method of window creation.

Most programmers use the `WinCreateStdWindow()` and `WinCreateWindow()` functions to create window panels in their OS/2 PM programs. There are, however, alternative methods. One of these is to create dialog templates with the dialog box editor.

Window panels can be created efficiently with the dialog box editor or defined manually in a `DLGTEMPLATE` or `WINDOWTEMPLATE` structure in a resource file and then called with the `WinLoadDlg()` function. This method has some advantages over creating windows with the `WinCreateStdWindow()` and `WinCreateWindow()` functions. The two most obvious advantages are:

- The window panels will have the same relative dimensions on all display monitors, regardless of their resolution.

The dialog box editor allows you to design screen panels that will have the same relative dimensions on all display monitors, regardless of their resolution. This is because the screen panels are saved in dialog units, and dialog units are defined in terms of the system font. One dialog unit is equal to



one-fourth the width of the average system font character by one-eighth the height of a system font character. (Because system font characters are approximately one-half as wide as they are tall, the width and height of a dialog unit are also approximately equal.) On the other hand, the `WinCreateStdWindow()` and `WinCreateWindow()` functions define the window position and the window size in terms of pixels. This requires more complicated coding to give screen panels the same appearance on display monitors with different resolutions.

- Panel creation is faster and easier with the dialog box editor.

The dialog box editor has a WYSIWYG display that makes it easy to create window panels and position control windows. It automatically stores the size and position information in your window

template. This eliminates the need to create, size, and position the control windows in your program code.

### The `DLGTEMPLATE` or `WINDOWTEMPLATE` Structure

PM defines windows in window template structures. The dialog box editor saves the window panel in a type of window template called a `DLGTEMPLATE`. This file has the extension `.dlg`. The `DLGTEMPLATE` can then be included in the resource file and modified, if necessary, to add additional items that could not be created with the dialog box editor. The `DLGTEMPLATE` structure can also be created manually and inserted directly into the resource file. It is important to have a general understanding of this structure, regardless of which method you use to create the `DLGTEMPLATE`.

```

DLGTEMPLATE ID_MAINBOX LOADONCALL MOVEABLE DISCARDABLE
BEGIN   DIALOG "VM File Transfer", ID_MAINBOX, 21, 38, 336, 181, FS_NOBYTEALIGN |
        FS_DLGBOARDER | FS_DLGBOARDER | WS_VISIBLE | WS_CLIPSIBLINGS |
        WS_SAVEBITS, FCF_SYSMENU | FCF_TITLEBAR | FCF_MINBUTTON |
        FCF_MAXBUTTON
        BEGIN
            CONTROL "Help", ID_HelpPB, 160, 10, 27, 13, WC_BUTTON, BS_PUSHBUTTON |
                BS_DEFAULT | WS_TABSTOP | WS_VISIBLE
            CONTROL "", ID_StatusMLE, 22, 31, 199, 40, WC_MLE, MLS_BORDER |
                MLS_WORDWRAP | WS_GROUP | WS_TABSTOP | WS_VISIBLE
            CONTROL "Transfer file", ID_TransferGB, 18, 96, 298, 74, WC_STATIC,
                SS_GROUPBOX | WS_GROUP | WS_VISIBLE
            CONTROL "", ID_SessionEF, 131, 145, 21, 8, WC_ENTRYFIELD, ES_LEFT |
                ES_MARGIN | WS_TABSTOP | WS_VISIBLE
                PRESPARAMS PP_BACKGROUNDINDEX CLR_BLUE
            CONTROL "", ID_SourceEF, 131, 123, 164, 9, WC_ENTRYFIELD, ES_LEFT |
                ES_AUTOSCROLL | ES_MARGIN | WS_TABSTOP | WS_VISIBLE
                PRESPARAMS PP_BACKGROUNDINDEX CLR_RED
            CONTROL "", ID_DestinationEF, 130, 104, 165, 8, WC_ENTRYFIELD,
                ES_LEFT | ES_AUTOSCROLL | ES_MARGIN | WS_TABSTOP | WS_VISIBLE
                PRESPARAMS PP_BACKGROUNDINDEX CLR_GREEN

                /* Additional CONTROLS... */
        END
END

```

Figure 1. Dialog Template with Presentation Parameters

The keywords `DLGTEMPLATE` and `WINDOWTEMPLATE` are almost synonymous and can be interchanged in most cases. Every window in a PM program can be defined in a `DLGTEMPLATE` or `WINDOWTEMPLATE` structure. The template starts with a `WINDOWTEMPLATE` or a `DLGTEMPLATE` statement, which tells the resource compiler that a window definition is to follow. It also specifies the resource ID, load options, memory options, and code page to be associated with that window. The dialog box editor specifies the following options for this statement:

- The resource ID is defined by the user in the dialog box editor
- The load option is `LOADONCALL`
- The memory options are `MOVEABLE` and `DISCARDABLE`

These options will not need to be changed for most programs.

The `DLGTEMPLATE` or `WINDOWTEMPLATE` statement is followed by one or more `CONTROL`, `WINDOW`, or `DIALOG` statements. These statements can be nested. The `CONTROL` and `WINDOW` statements specify the text associated with a title bar or window text, a window ID, position, size, and class, window styles, and control data. The `DIALOG` statement is the same as the `CONTROL` and `WINDOW` statements with the exception that no window class is specified. With the `DIALOG` statement, the `WC_FRAME` class is implicit.

The dialog box editor uses the `CONTROL` statement to specify all control windows that are children of the dialog box. There is also a set of predefined control statements that can be used to specify certain con-

trol windows such as `CTEXT` for centered static text and `ENTRYFIELD` for an entry field control. A control window must be a child of either a `WINDOW` or a `DIALOG`.

The `WINDOW` statement or control statement can be used to specify user-defined windows if the window is registered with the `WinRegisterClass()` function before the `WinLoadDlg()` function call is made.

The template can also contain presentation parameters that define additional characteristics of a window. To do this, specify the keyword `PRESPARAMS` followed by the presentation parameter identifier and its value.

### Using the Template

The example `DLGTEMPLATE` structure (Figure 1) defines a dialog

```

void cdecl main()
{
    /* Declarations and Initializations */

    hab = WinInitialize((UHORT)NULL);
    hmq = WinCreateMsgQueue(hab, 0);
    hDlg = WinLoadDlg(HWND_DESKTOP,
        HWND_DESKTOP,
        (PFNWP)MainDlgBoxWndProc,
        (HMODULE)NULL,
        ID_MAINBOX, (PVOID)NULL);
    WinProcessDlg(hDlg);

    /* Clean up and terminate */
}

```

Figure 2. Loading and Displaying a Window

window that owns several control windows.

The dialog box editor was used to create the window panel and position the control windows. The structure was then copied into the resource file, and the statements for the presentation parameters were added. The presentation parameters simply change the color of each entry field to demonstrate how presentation parameters are set in a template structure. The keyword PREPARAMS was used, followed by the PP\_BACKGROUND\_COLOR\_INDEX identifier and a value such as CLR\_RED. In the main() function, the WinCreateStdWindow() function call is replaced with the WinLoadDlg() and WinProcessDlg() function calls. There is no longer any need for message queue processing, but a message queue must still exist or the application will not run. The main() function is very simple, as shown in Figure 2. This skeleton example demonstrates how the dialog box editor can be used to create a main window that will display the same on all display monitors.

The window will look like Figure 3.

### Adding Icons and Action Bars

An action bar can be added to a dialog window, and an icon can be associated with a dialog window in the WM\_INITDLG part of the dialog window procedure.

Once a resource ID has been defined in a header file, and the icon has been associated with this resource ID in the resource file, then the icon can be loaded with a call to the WinLoadPointer() function. This call must be followed by sending a message to the default dialog procedure with a WM\_SETICON message.

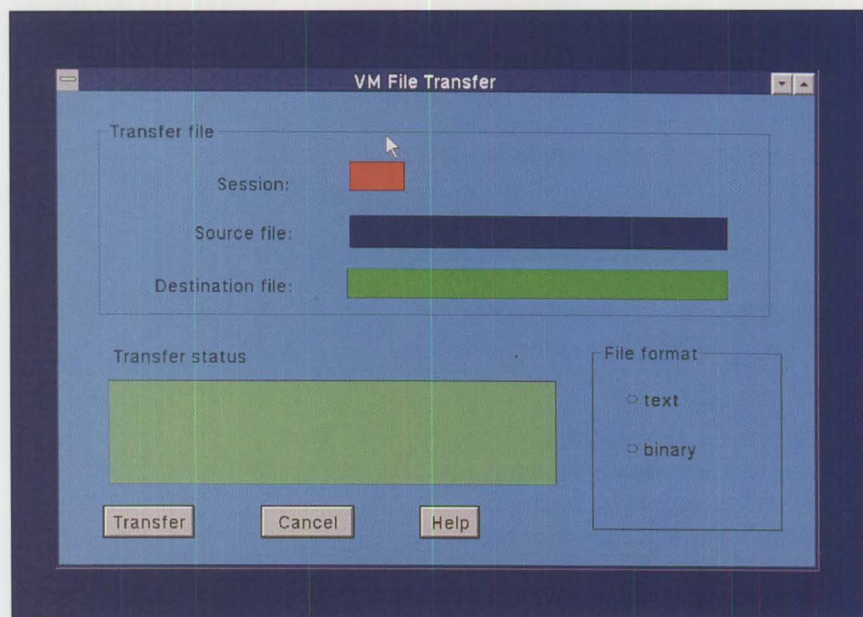


Figure 3. Window Defined in Figure 1 and Called in Figure 2

```

case WM_INITDLG:
    hDlgBoxIcon = WinLoadPointer(HWND_DESKTOP,
    NULL,
    ID_DLGBOX);
    WinDefDlgProc(hwnd, WM_SETICON,
    (MPARAM)hDlgBoxIcon,
    (MPARAM)0);

    hMenu = WinLoadMenu(hDlg, NULL, ID_MENU);
    WinSendMsg(hDlg, WM_UPDATEFRAME, (MPARAM)0,
    (MPARAM)0);
    ...
    ...
    ...
    break;

```

Figure 4. Associating an Icon with a Dialog Template Window

```

WINDOWTEMPLATE ID_MAINBOX
BEGIN
    FRAME NULL, 0, 50, 34, 285, 190, WS_VISIBLE,
        FCF_TITLEBAR | FCF_SYSMENU | FCF_SIZEBORDER | FCF_MINMAX |
        FCF_TASKLIST
    BEGIN
        WINDOW "", ID_CLIENT, 0, 0, 285, 180, WC_MYCLASS, WS_VISIBLE, FCF_TITLEBAR |
            FCF_SYSMENU
        BEGIN
            CONTROL "", ID_TASKLIST, 166, 9, 134, 78, WC_COMBOBOX, CBS_SIMPLE |
                WS_GROUP | WS_TABSTOP | WS_VISIBLE
            CONTROL "Remove/Restore Options", ID_TASKGROUP, 24, 62, 87, 56, WC_STATIC,
                SS_GROUPBOX | WS_GROUP | WS_VISIBLE
            CONTROL "Icon", ID_TASKICON, 33, 96, 74, 10, WC_BUTTON, BS_AUTORADIOBUTTON |
                WS_TABSTOP | WS_VISIBLE
            CONTROL "Program", ID_PROGRAM, 33, 81, 74, 10, WC_BUTTON, BS_AUTORADIOBUTTON |
                WS_TABSTOP | WS_VISIBLE
            CONTROL "Both", ID_BOTH, 33, 67, 74, 10, WC_BUTTON, BS_AUTORADIOBUTTON |
                WS_TABSTOP | WS_VISIBLE
            CONTROL "Program Name", 65535, 171, 100, 73, 8, WC_STATIC, SS_TEXT |
                DT_LEFT | DT_TOP | WS_GROUP | WS_VISIBLE
            CONTROL "Restore", ID_REMOVE, 25, 45, 86, 13, WC_BUTTON, BS_PUSHBUTTON |
                WS_TABSTOP | WS_VISIBLE
            CONTROL "Remove", ID_REMOVE, 25, 32, 86, 13, WC_BUTTON, BS_PUSHBUTTON |
                WS_TABSTOP | WS_VISIBLE
            CONTROL "Cancel", ID_CANCEL, 25, 10, 86, 13, WC_BUTTON, BS_PUSHBUTTON |
                BS_DEFAULT | WS_TABSTOP | WS_VISIBLE
        END
    END
END

```

Figure 5. Window Template with a Frame and Controls



When an action bar is needed, it can be added by putting a menu template into the resource file and then making a call to the `WinLoadMenu()` function. After the call to the `WinLoadMenu()` function, a `WM_UPDATEFRAME` message must be sent to the dialog window. The code required to add an action bar and an icon is shown in Figure 4.

The last bit of sample code demonstrates how to use a template to define a typical window with a frame, a client, and some control windows (Figure 5).

It is necessary to register `WC_MYCLASS` with a `WinRegisterClass()` function call before making the call to `WinLoadDlg()`. The menu is loaded with the `WinLoadMenu()` function in the `WM_INITDLG` case. This menu is defined in a menu template in the resource file.

The window will look like Figure 6.

### Improved Usability with OS/2 2.0

This method is not appropriate for all applications. Programs with only one screen panel and multiple control windows are particularly well

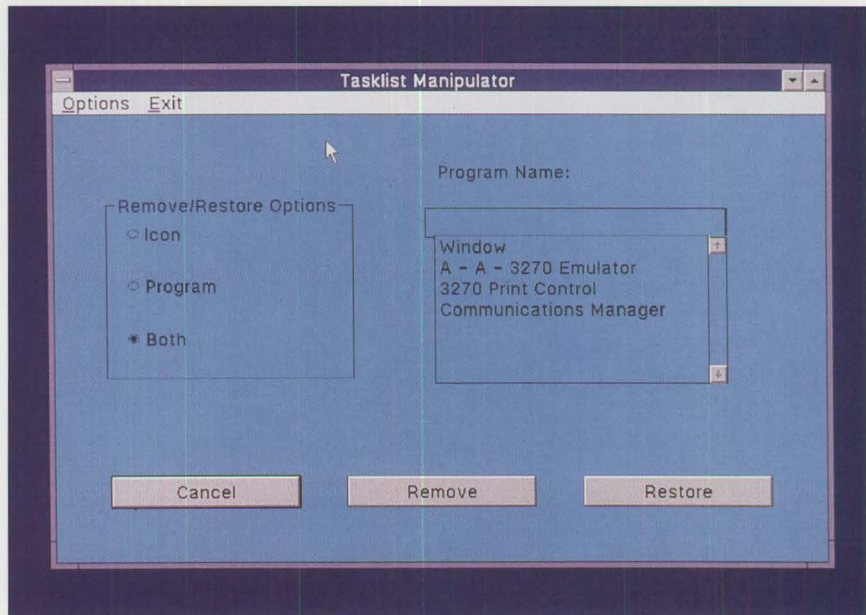


Figure 6. Window Defined in Figure 5 and Called in Figure 4

sued for this method. (The Presentation Manager control panel utility is a good example of this.)

The dialog box editor is a powerful tool. OS/2 version 2.0 will provide even better usability, making window panel creation easier and faster yet. Ease-of-use, along with the device independence offered by defining windows in window templates, are two of the advantages of using this method to create windows.

### ABOUT THE AUTHOR

*Scott Sorensen is part of IBM's OS/2 Application Assistance Center team, working through the cooperative education program. He specializes in OS/2 programming, with emphasis on Presentation Manager. Scott is a senior studying computer science at Brigham Young University.*

# REXX Program for OS/2 LAN Server

Carolyn Easter  
IBM Corporation  
Austin, Texas

**In OS/2 LAN Server environments, an administrator needs the ability to create user IDs that have identical logon resources. The program described here does just that. It uses REXX, OS/2, and LAN Server commands to create a new user ID modelled from another created user ID.**

REXX is a language included in OS/2 Extended Edition 1.30.1 and Standard Edition 1.30.1 and is ideal for those interested in creating sophisticated batch programs. LAN Server does not interact directly with REXX, but because it has a command line interface, the REXX programming language can be used to create useful tools.

An example of a REXX program follows this article (Figures 6a through 6g). The functions provided by this REXX program are:

- Create a user ID modeled after a defined user ID
- View user IDs defined to the named domain controller server
- View specific user ID details
- Delete a specific user ID

## Data Structures

This REXX program is interactive, relying on users to input information to the program. The program requires that the output from some of the commands be stored in a convenient place for processing. REXX provides a dynamic data construct called a *rxqueue*, where output from

command line commands or program data can be stored. Because OS/2 LAN Server command output is either printed to the screen or redirected to a file or printer, the *rxqueue* is an ideal place to store the output.

An example of a command where the output of the command is placed in the queue is shown in Figure 1. The command is enclosed in single quotes. Double quotes can also be used. The pipe character “|” is used in the command string to direct output to program files. The NET USERS command returns a list of the defined users on the server named IBMOS2S1. Now that the data is in a REXX queue structure, the REXX commands PULL and PUSH can be used to remove queue elements from the queue (PULL) or place elements in the queue (PUSH).

In this REXX program, a subroutine called PULLQUEUE removes the queue elements and places them in a data structure accessible to all portions of the REXX command file. The PULLQUEUE subroutine is shown in Figure 2. The first executable statement in the subroutine copies the number of the queued elements into the first variable of LI (the zero index). LI is called a *stem variable* and is similar to an array. One function not available using stem variables that is available when using arrays is the arithmetic operations on the array index. With stem variables, the index number (.I) cannot be incremented, decremented, or otherwise manipulated inside the stem variable declaration. The opera-

```
'NET ADMIN \\IBMOS2S1 /C NET USERS | rxqueue /FIFO'
```

Figure 1. Example of a Command Where Output Is in the Queue

tion **1 = 3\*I** can be used, then **I** used as an index (LI.I).

In the next line, the PULL function is used to pull the data from the queue into the variable LI.I where .I is a value **1** to the number of queue elements. By using the PARSE statement with the PULL function, the data from the queue is not uppercased, but remains in its original state. The PULL function by itself will uppercase the entire data string.

## Program Flow

Figure 3 shows the program flow. The program is started by typing in the name of the command file. There are no parameters. The file is named LANUSER2.CMD.

Before presenting the main menu for selection, it is important for the program to have two major pieces of information:

1. What is the name of the domain controller? When creating user IDs, the domain controller must be the server updated with the new user names because this server replicates the user names and group names to the other servers in the domain. The server name is validated as a domain controller using the NET VIEW

```
PULLQUEUE:
  LI.0 = queued();
  DO I = 1 TO LI.0
    PARSE PULL LI.I
  END
  RETURN
```

Figure 2. PULLQUEUE Subroutine

command and NET ACCOUNTS command.

2. Where is the administrator logged on? This is important when issuing LAN Server commands to the domain controller, because the NET ADMIN command must precede each command. It is also important for other OS/2 commands that may or may not support redirected or UNC file requests.

### Validation of Server Name

One of the first actions done by the program is to prompt the administrator for the name of the server. Granted, that if a utility like this program were going to be used on only one domain, the name of the server could be hard coded onto the variable name SERVER. For flexibility, the program was written to be used in any domain.

The NET VIEW command is used to list the server names available and to validate that the name of the server machine entered is one of the machines in this list. It is possible to administrate another domain while logged on to the default domain. This requires that the administrator has added the domain name to the IBMLAN.INI file, othdomains=xxxxxxx, of the requester being used as the administrator's workstation and that the administrator's user ID and password are known to the other domain. When the NET VIEW command lists the server names available, the other domain's servers will also be listed, and validation of the server name will succeed.

After the server name is validated, the server account information is checked using the NET ACCOUNTS command to ensure that it is a domain controller. If the

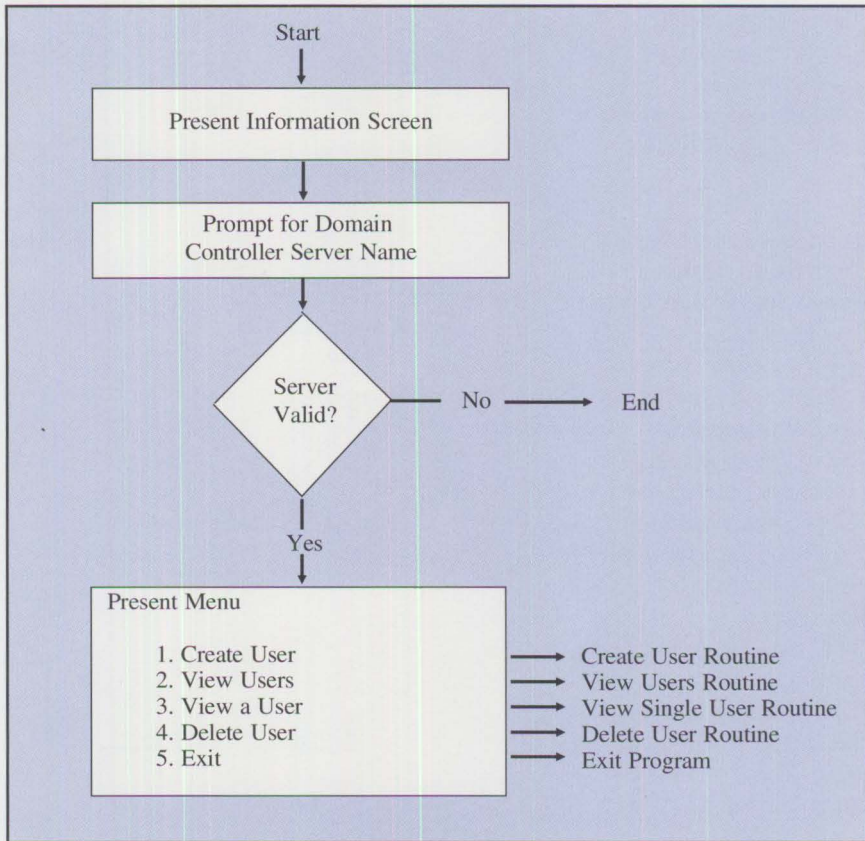


Figure 3. Initialization and Main Panel Program Flow

role is PRIMARY, the server is a domain controller and the program can continue. Otherwise, the program issues an error message and exits.

### Location Checking for Administrator

The only additional information needed before proceeding to the main menu is determining the location of the administrator executing this program. If the administrator is running the program at the domain controller, then the NET ADMIN prefix is not needed for the OS/2 LAN Server commands. If the user is remote, the NET ADMIN prefix must be issued in front of the commands. A variable LANFRONT is set up to reflect this situation. If LANFRONT is null, the administra-

tor is at the domain controller; otherwise, LANFRONT is the value 'NET ADMIN \\servername /c'. During the executable phase of the program, this variable is tested for NUL to indicate the location of the administrator.

The NET ACCOUNTS command is issued to determine if the administrator is located locally or remotely. If the role of the local workstation is PRIMARY, the administrator is at the domain controller. Otherwise, the administrator is remote.

### Create User

The program has several functions. All except the *create user* are trivial. The program flow for this function is shown in Figure 4.

This discussion focuses on the model user function and how it is accomplished using the NET USER command and knowledge of the directory structure of the domain control database.

The administrator selects "Create new user" from the menu and the program prompts for the user name to use as a model. This user ID is checked against a created list of users to make sure it exists in the domain. The program then prompts for the user name to create and is checked to make sure the ID does not exist. The program then loops each time a user is created and asks for another user ID to create. This allows many user IDs to be created from the model user. Each new user ID can be created as either an administrator or user, can have a unique password, and can have a unique comment string. These parameters are then used on the NET USER command.

After the user specifications are determined, the cloning of the model user ID environment takes place. This is accomplished by duplicating the user directory of the new model user ID. The user environment files are contained in the \BMLAN\DCDB\USERS\username where username is the name of the user. The list of files that might be present in this subdirectory are shown in Figure 5. All files in the subdirectories are copied to the new user's newly created subdirectory.

The program uses the MD command to create these subdirectories, using the variable ROOT to specify the path. The program uses the fact that an automatic share occurs at each server and is named \servername\BMLAN\$. Therefore, the value of ROOT is

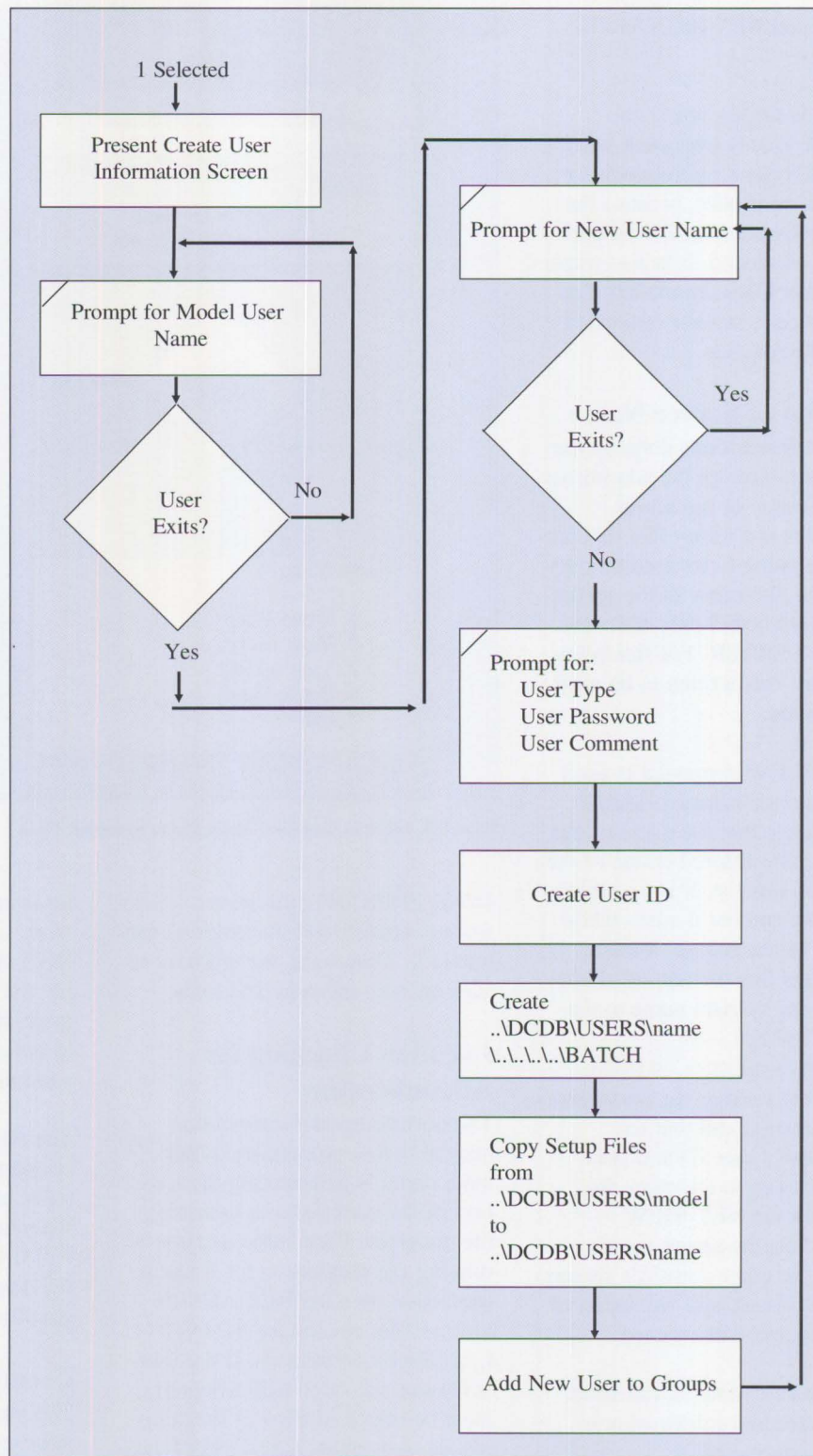


Figure 4. Program Flow for Create User Function

\\servername\IBMLAN\$. This share name is one of several that occur when the SERVER service starts. All drives are shared as \\servername\d\$ where **d** is the drive letter.

The next task is copying the files from the model ID subdirectory to the new user ID subdirectory. NET COPY is used because this command supports local and remote COPY commands.

The user is then added to the group IDs to which the model user has membership.

If the model user has a home directory, the program determines where the new user's home directory will be by following these guidelines:

1. If the model user's home directory name ends with the model

user's name, another like subdirectory is created using the new user's name as the subdirectory name.

2. Otherwise, the same subdirectory is used as the new user's home directory.

Home directories no longer have an ALIAS, but if desired, the administrator can create one.

The program returns for the user to enter another user ID to be created, using the same model user ID name entered for the previous user ID.

### Other Functions

Several other functions are available:

- View valid users for the domain
- View a user ID profile
- Delete a user

The delete user process removes the user ID, the user definition subdirectory on the domain controller, and associated access control profiles.

The User Profile Management delete user function does not delete the access control profile for the user definition subdirectory, only the permissions and leaves an empty access control profile in the NET.ACC file.

### Summary

The REXX programming language is flexible and can be used to create effective tools for administrating an OS/2 LAN Server domain. Imagine what could be done with the error logs, message logs, and audit trails provided with the LAN Server. Each of these logs gives valuable information to the administrator and can be used in an integrated way to provide better problem determination, statistical reporting, and general administration. Food for thought: this information could be formatted by a REXX program and imported to Database Manager.

### ABOUT THE AUTHOR

*Carolyn Easter is an advisory programmer in IBM's Entry Systems Division in Austin, Texas. She joined IBM in 1980 and was assigned to program the files and data base system on the 5520 Administrative System. She has a B.S. in mathematics from Southern Methodist University and a B.A. in computer science from the University of Texas at Austin.*

**LIST.A** – DOS Private Applications  
**LIST.S** – Served Applications panel entries  
**LIST.U** – Logon Assignments for DOS  
**USER.A** – OS/2 Private Applications  
**USER.L** – Logon Assignments for OS/2  
**USER.S** – Desktop Manager, Public Applications window entries  
**EXTERNAL.OFF** – present if user has ever attached to an external resource  
**EXTERNAL.ON** – present if user detaches from an external resource or logs off when attached to an external resource  
**PROFILE.CMD** – a file run when logon occurs on an OS/2 machine (not mandatory)  
**PROFILE.BAT** – a file run when logon occurs on a DOS machine (not mandatory)

Figure 5. List of Files in \IBMLAN\DCDB\USERS Subdirectory

```

/*****
/* 5748-XX8 (C) COPYRIGHT IBM CORP. 1990 */
/* ALL RIGHTS RESERVED */
/* LICENSED MATERIALS-PROPERTY OF IBM */
/* SEE COPYRIGHT INSTRUCTIONS, G120-2083 */
/*****
/*LANUSER2.CMD */
/* REXX exec for creating user IDs on LAN Server 1.30.1 domains */
/* Assumptions: User is logged on as an administrator */
/* VERSION 2.0 */
/* This version supports the enhanced architecture and */
/* command line support for the home directory. Messages */
/* have been separated for ease of translation. */
/*****
/* initialize */
MODEL=''
ROOT=''
R=''
SERVER=''
NAME=''
/* MESSAGES BLOCK - TRANSLATE THESE BLOCKS OF TEXT TO OTHER LANGUAGES */
GREET.0 = 8
GREET.1 = ' This module will allow you, the administrator, to: '
GREET.2 = ''
GREET.3 = ' - Create user IDs on a domain controller and model the'
GREET.4 = ' new user on a previously created user.'
GREET.5 = ' - See what users IDs are valid for the domain.'
GREET.6 = ' - See information about a specific user '
GREET.7 = ' - Delete a user'
GREET.8 = ''
CREATE.0 = 5
CREATE.1 = ''
CREATE.2 = ' This routine will create a user modeled after another user.'
CREATE.3 = ' This means all logon assignments and user attributes are copied'
CREATE.4 = ' to the new user as well as creating a home directory if the '
CREATE.5 = ' model user has a home directory. '
THANKS = ' Thank you!'
A = 'A'
US = 'U'
QUEST.1 = 'ENTER THE NAME OF THE USER TO DELETE'
QUEST.2 = 'ENTER THE NAME OF THE USER'
QUEST.3 = 'ENTER THE MODEL NAME. '
QUEST.4 = 'ENTER THE USER NAME. '
QUEST.5 = 'DO YOU WANT TO RETURN TO THE MENU (Y/N)?.'
QUEST.6 = 'DELETE ? (Y/N)'
QUEST.7 = 'TYPE IN THE USER COMMENT';
QUEST.8 = 'IF THE USER IS TO HAVE A PASSWORD, ENTER THE PASSWORD';
QUEST.10 = 'IS THE USER ADMIN OR USER? (A/U)';
QUEST.11 = 'WHAT IS THE NAME OF THE SERVER WHICH IS THE DOMAIN CONTROLLER?'
YES = 'Y'
NO = 'N'
INFO1 = 'THESE ARE THE VALID SERVER NAMES:'
INFO2 = 'PROGRAM WILL EXIT.'
INFO3 = 'Name being used as a model is: '
INFO4 = ' DELETING USER. '
INFO5 = ' CREATING USER.'
INFO6 = 'Last name used was: '
INFO7 = 'Wait 30 seconds if users home directory is on an additional server.'
ERROR1 = 'THE SERVER NAME YOU ENTERED IS NOT THE DOMAIN CONTROLLER'
ERROR2 = 'THE SERVER NAME IS NOT VALID'
ERROR3 = 'ENTER A VALID CHOICE, PLEASE'
ERROR5 = 'USER DOES NOT EXIST. RE-ENTER NAME PLEASE.'
ERROR6 = 'USER ALREADY EXISTS. RE-ENTER NEW USER NAME'

```

Figure 6a. Source Listing for LANUSER2.CMD

```

MENU.0 = 7
MENU.1=' SELECT THE FUNCTION YOU WISH TO PERFORM '
MENU.2=''
MENU.3=' 1. CREATE USER(S) USING A MODEL'
MENU.4=' 2. SEE WHAT USERS ARE DEFINED'
MENU.5=' 3. SEE INFORMATION ABOUT A SPECIFIC USER'
MENU.6=' 4. DELETE A USER'
MENU.7=' 5. EXIT THIS UTILITY'
/* BEGIN */
'@ECHO OFF'
'CLS';
DO I = 1 TO GREET.0
SAY GREET.I
END
'PAUSE';
SAY
SAY QUEST.11
SAY
PARSE UPPER PULL SERVER;
SERVER=STRIP(SERVER);
SAY THANKS
/*****
*/
/* VALIDATE SERVER NAME
*/
/*****
*/
' NET VIEW | rxqueue /FIFO'
CALL PULLQUEUE;
DO I = 1 TO LI.0
  PARSE VAR LI.I '\\ 'SERV COMMENT .
  IF SERV=SERVER THEN LEAVE
END
IF SERV SERVER THEN DO
  SAY ERROR2
  SAY INFO1
  ' NET VIEW ';
  'PAUSE';
  EXIT
END
ELSE DO
  'NET ADMIN \\ 'SERVER' /c NET ACCOUNTS | rxqueue /FIFO';
  CALL PULLQUEUE;
  PARSE VAR LI.6 FUNCTION ':' ROLE .
  ROLE = STRIP(ROLE);
  IF ROLE 'PRIMARY' THEN DO
    SAY ERROR1
    SAY INFO2
    EXIT;
  END
END
/*FIND OUT IF USER IS USING DOMAIN CONTROLLER */
LANFRONT='NET ADMIN \\ 'SERVER' /c';
ADMINQ='\"
ROOT='\\ 'SERVER'\IBMLAN$';
' NET ACCOUNTS | rxqueue /FIFO'
CALL PULLQUEUE;
PARSE VAR LI.6 FUNCTION ':' ROLE .
ROLE = STRIP(ROLE);
IF ROLE = 'PRIMARY' THEN DO /* USER IS AT DOMAIN CONTROLLER */
  LANFRONT='\"
  ADMINQ='\"
END
/* DETERMINE DRIVE LETTER OF DOMAIN CONTROLLER */
LANFRONT 'NET SHARE IBMLAN$ | rxqueue /FIFO'
CALL PULLQUEUE

```

Figure 6b. Source Listing for LANUSER2.CMD

```

PARSE VAR LI.2 P PATH .
PATH=STRIP(PATH)
PARSE VAR PATH DD ':\' IBM
/*****
/*SET UP A LIST OF USERS TO USE IN VALIDATION */
*****/
CALL SETUPUSERS;

/*****
/* BRING UP THE MAIN MENU */
*****/
DO FOREVER
  'CLS';
  DO I = 1 TO MENU.0
    SAY MENU.I
  END
  KEY=''
  DO WHILE KEY=''
    PARSE PULL KEY;
  END
  SELECT
    WHEN KEY=1 THEN CALL CREATM
    WHEN KEY=2 THEN CALL DUSERS
    WHEN KEY=3 THEN CALL USERINFO
    WHEN KEY=4 THEN CALL DELETE
    WHEN KEY=5 THEN EXIT
    OTHERWISE DO /* need a valid selection */
      SAY ERROR3
      'PAUSE';
  END
END
END
/*****SUBROUTINES *****/
/*****
/* CREATE LIST OF USERS */
*****/
SETUPUSERS:
  LANFRONT ' NET USERS | rxqueue /FIFO'
  CALL PULLQUEUE
  USERS.0 = 0
  C = 1
  DO I = 3 TO LI.0
    PARSE VAR LI.I NAMES.1 NAMES.2 NAMES.3 .
    NAMES.1 = STRIP(NAMES.1)
    NAMES.2 = STRIP(NAMES.2)
    NAMES.3 = STRIP(NAMES.3)
    IF NAMES.1 'THE' THEN DO
      DO U = 1 TO 3
        IF NAMES.U '' THEN DO
          USERS.C = NAMES.U
          C = C + 1
          USERS.0 = USERS.0 + 1
        END /*IF*/
      END /*DO*/
    END /*IF */
  END /*DO */
RETURN;
/*****
/* LOOK AT INFORMATION ABOUT A SPECIFIC USER */
*****/
USERINFO:
  SAY
  SAY QUEST.2; PARSE UPPER PULL INFO;
  LANFRONT 'NET USER' INFO '|MORE';

```

Figure 6c. Source Listing for LANUSER2.CMD



```

        'PAUSE';
RETURN
/*****
/* LOOK AT CREATED USERS
*****
DUSERS:
        LANFRONT ' NET USERS';
        'PAUSE';
RETURN

/*****
/* DELETE A USER
*****
DELETE:
DO FOREVER
        R=GETUSER(1,1);
        IF R=0 THEN DO
                SAY QUEST.6
                PARSE UPPER PULL ANS;
                IF ANS = YES THEN DO
                        SAY INFO4;
                        LANFRONT ' NET ACCESS 'DD':\IBMLAN\DCDB\USERS\'NAME' /D'
                        LANFRONT ' NET ACCESS 'DD':\IBMLAN\DCDB\USERS\'NAME'\BATCH /D'
                        LANFRONT ' NET USER ' NAME ' /D'
                        RET=MLIST(NAME,0)
                END
        END
        ELSE LEAVE
END /*DOFOREVER*/
RETURN
/*****
/* CREATE A USER - MODELS AFTER ALREADY CREATED USER
*****
CREATEM:
'CLS';
DO I = 1 TO CREATE.0
        SAY CREATE.I
END
'PAUSE';
'CLS';
R=GETUSER(3,1);
MODEL=NAME
NAME = ''
IF R=0 THEN DO
        LANFRONT 'NET USER 'MODEL' | rxqueue /FIFO '
        CALL PULLQUEUE;
        GROUP. = ""
        GROUP.0 = 0
        GR = 0
        LI.0 = LI.0 - 1
        PARSE VAR LI.27 GP MEMB LI.27
        DO I = 27 TO LI.0
                PARSE VAR LI.I G.1 G.2 .
                DO J = 1 TO 2
                        G.J = STRIP(G.J, '*')
                        IF G.J '' THEN DO
                                GR = GR + 1
                                GROUP.GR = G.J
                        END
                END
        END
        GROUP.0 = GR
DO FOREVER /* GET USER INFORMATION */
        'CLS';

```

Figure 6d. Source Listing for LANUSER2.CMD

```

SAY INFO3 MODEL
SAY INFO6 NAME
R=GETUSER(4,0);
IF R = 0 THEN DO
  AU= 0
  DO WHILE AU=0 /* GET USER TYPE */
    SAY QUEST.10;PARSE UPPER PULL AU;
    IF AU A & AU US THEN AU=0
  END
  PASSWORD=''
  SAY QUEST.8; /**GET PASSWORD */
  PARSE UPPER PULL PASSWORD
  COMMENT ='';
  SAY QUEST.7; /**GET COMMENT */
  PARSE PULL COMMENT
  SAY INFO5
  IF AU=A THEN DO
    AU='ADMIN';
  END
  ELSE DO
    AU='USER' ;
  END
  IF PASSWORD='' THEN DO
    PASSREQ='NO';
  END
  ELSE DO
    PASSREQ='YES';
  END
  LANFRONT ' NET USER ' NAME PASSWORD ' /ADD /PRIV:'AU '/PASSWORDREQ:'PASSREQ '
  /USERCOMMENT:'ADMINQ||COMMENT||ADMINQ; /* THIS LINE CONTINUED FROM PREVIOUS, ENTER AS ONE */
  R=MLIST(NAME,1)
  'MD ' ROOT'\DCDB\USERS\'NAME;
  'MD ' ROOT'\DCDB\USERS\'NAME'\BATCH';
  DIRY = DD':\IBMLAN\DCDB\USERS\'NAME;
  R=ACCESSPRO(SERVER,DIRY);
  IF R=1 THEN DO
    LANFRONT ' NET ACCESS 'DD':\IBMLAN\DCDB\USERS\'NAME '/GRANT ' NAME':RWCXDAP'
    LANFRONT ' NET ACCESS 'DD':\IBMLAN\DCDB\USERS\'NAME'\BATCH /GRANT ' NAME':RWCXDAP';
  END
  ELSE DO
    LANFRONT ' NET ACCESS 'DD':\IBMLAN\DCDB\USERS\'NAME '/ADD ' NAME':RWCXDAP'
    LANFRONT ' NET ACCESS 'DD':\IBMLAN\DCDB\USERS\'NAME'\BATCH /ADD ' NAME':RWCXDAP';
  END
  ' NET COPY ' ROOT'\DCDB\USERS\'MODEL' ' ROOT'\DCDB\USERS\'NAME;
  ' NET COPY ' ROOT'\DCDB\USERS\'MODEL'\BATCH' ROOT'\DCDB\USERS\'NAME'\BATCH';
DO I = 1 TO GROUP.0
  IF GROUP.I = 'USERS' THEN ITERATE
  IF GROUP.I = 'ADMINS' THEN ITERATE
  LANFRONT 'NET GROUP 'GROUP.I NAME ' /ADD '
END
R= GETHOME(MODEL) /* find out if user has home directory */
IF R = 1 THEN DO /* MODEL USER HAS HOME DIRECTORY */
  PARSE VAR HNAME N.1 '\ ' N.2 '\ ' N.3 '\ ' N.4 '\ ' N.5 '\ ' N.6 '\ ' N.7 '\ ' N.8
  DIRH=''
  DO I = 1 TO 8
    IF N.I = "" THEN LEAVE
    IF MODEL = N.I THEN LEAVE
    DIRH = DIRH'\ 'N.I
  END
  PARSE VAR DR DR '$'
  IF MODEL= N.I THEN DO
    DIRH = DIRH'\ 'NAME

```

Figure 6e. Source Listing for LANUSER2.CMD

```

        'MD \\SERV\DR'$DIRH
    END
    SAY INFO7
    'PAUSE'
    DIRY = DR':DIRH
    R=ACCESSPRO(SERV,DIRY);
    IF R=1 THEN 'NET ADMIN \\SERV ' /c NET ACCESS 'DIRY NAME':XRWCDAAP /GRANT'
    ELSE 'NET ADMIN \\SERV ' /c NET ACCESS 'DIRY NAME';XRWCDAAP /A'
    HDIR = HDIR||DIRH
    LANFRONT ' NET USER ' NAME ' /HOMEDIR:'HDIR
END
END
ELSE LEAVE
END /*DO FOREVER */
END
RETURN;
/*****
/* GET USER NAME ENTERED BY ADMINISTRATOR */
*****/
GETUSER:
ARG PROMPT,S
FLAG = 0
DO WHILE FLAG = 0
    SAY QUEST.PROMPT
    PARSE UPPER PULL NAME;
    L=length(NAME);
    IF L=0 THEN DO
        SAY QUEST.5
        PARSE UPPER PULL ANS;
        IF ANS=YES THEN DO
            FLAG=1;
            RU=1
        END;
        ELSE NOP
    END
    ELSE DO;
        RET = USEREXIST(NAME)
        IF RET= 0 & S = 1 THEN DO
            SAY ERROR5
            'PAUSE'
        END
        ELSE DO
            IF RET = 1 & S = 0 THEN DO
                SAY ERROR6
                'PAUSE'
            END
            ELSE DO
                FLAG = 1
                RU = 0
            END
        END
    END
END
END /*DO WHILE */
RETURN RU
/*****
/* CHECK EXISTENCE OF A DEFINED USER */
*****/
USEREXIST:
ARG U
RES=0
DO I = 1 TO USERS.0
    IF U = USERS.I THEN DO
        RES=1
        LEAVE
    END
END

```

Figure 6f. Source Listing for LANUSER2.CMD

```

    END
END
RETURN RES;
/*****
/* ADD OR DELETE USER FROM CREATED LIST OF USERS
*/
*****/
MLIST:
ARG U,ACT
IF ACT = 0 THEN DO
    DO I = 1 TO USERS.0
        IF U = USERS.I THEN DO
            USERS.I = ""
            LEAVE
        END
    END
END
ELSE DO
    USERS.0 = USERS.0 + 1
    URS=USERS.0
    USERS.URS = U
END
RETURN 0;
/*****
/* CHECK EXISTENCE OF AN ACCESS CONTROL PROFILE
*/
*****/
ACCESSPRO:
ARG S,DIR
'NET ADMIN \\S' /C NET ACCESS 'DIR' | rxqueue /FIFO'
CALL PULLQUEUE;
RES=0
PARSE VAR LI.1 NETMSG ':'
IF NETMSG 'NET222' THEN RES=1
RETURN RES;
/*****
/* GET HOME DIRECTORY INFORMATION
*/
*****/
GETHOME:
ARG NAMMOD
LANFRONT 'NET USER 'NAMMOD' | rxqueue /FIFO '
CALL PULLQUEUE;
PARSE VAR LI.22 HO DI HDR '\ ' SERV '\ ' DR '\ ' HNAME .
HDIR=HDR'\ 'SERV'\ 'DR
IF HDR = '' THEN DO
    PARSE VAR LI.22 HO DI '\ ' SERV '\ ' DR '\ ' HNAME .
    HDIR='\ 'SERV'\ 'DR
END
HDIR = STRIP(HDIR)
IF SERV = '' THEN RES= 0
    ELSE RES = 1
RETURN RES
/*****
/* PULL QUEUE ROUTINE
*/
*****/
PULLQUEUE:
LI.0 = queued();
DO I = 1 TO LI.0
    PARSE PULL LI.I
END
RETURN

```

Figure 6g. Source Listing for LANUSER2.CMD

## Micro Focus COBOL/2 and the DOS Database Requester

Andrew Morbitzer  
Dallas, Texas

Although IBM OS/2 EE Database Manager only supports IBM COBOL/2™, today it is not rare to find that a combination of different vendor packages and languages is used. The use of Micro Focus COBOL/2 is demonstrated here using a sample program.

Many customers are finding the DOS Database Requester a desirable step in the evolution from DOS to OS/2 systems. Some are choosing

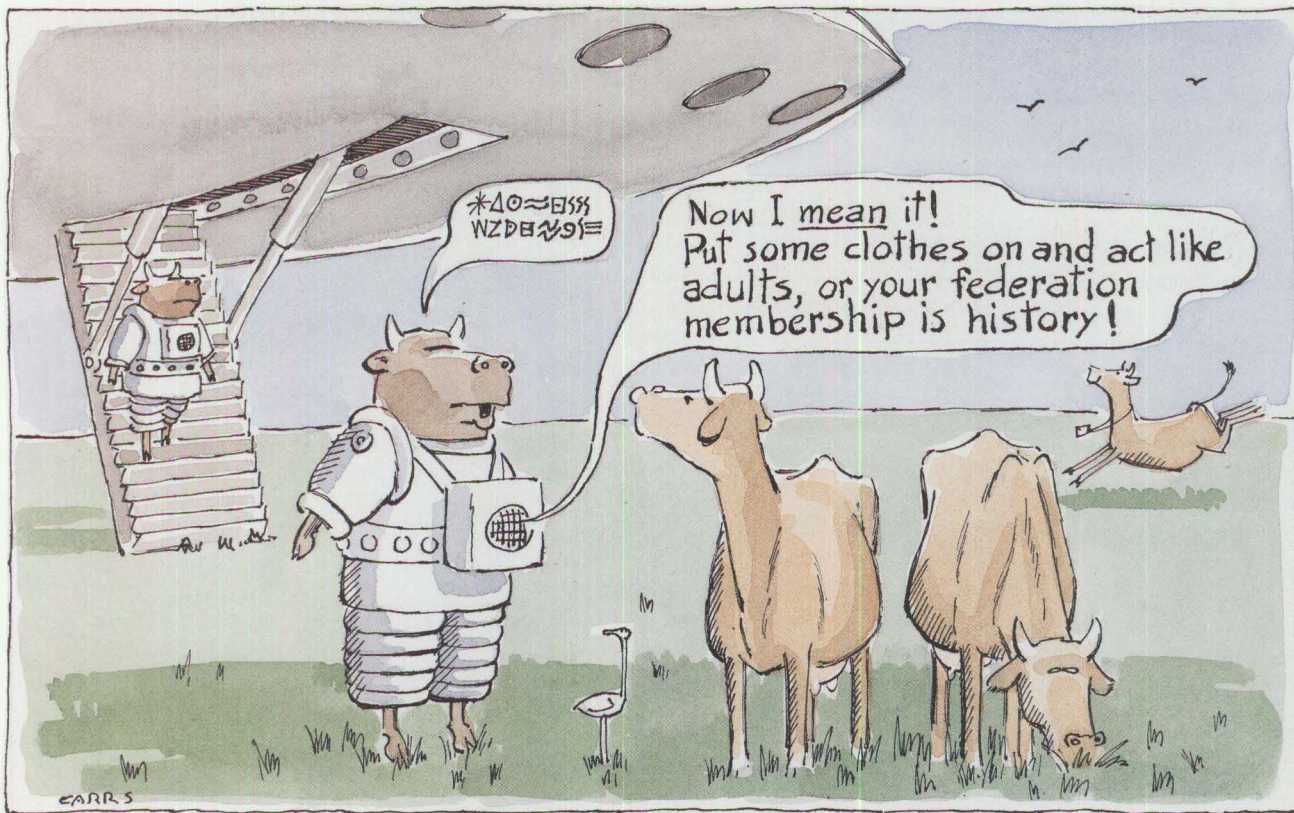
Micro Focus COBOL/2 to implement DOS Database Requester applications. Because Micro Focus does not develop the OS/2 Database Manager, some programmers may wonder if there are reasons not to use Micro Focus COBOL/2 to develop applications. The following demonstration should alleviate insecurities about using Micro Focus COBOL/2 to produce DOS Database Requester applications to run against a database server running IBM OS/2 EE 1.3 and Database Manager.

The program in Figures 1a and 1b was developed to illustrate the use of Micro Focus COBOL/2. It runs on a DOS Database Requester and catalogs the SAMPLE database, which is included with IBM OS/2 EE 1.3 and exists on a database server. The environment for this application is:

- A PS/2 Model 80-A31 running IBM OS/2 EE 1.3
- A PS/2 P70 running DOS 4.01 and IBM Local Area Network Support Program 1.2
- An 8228, which provides the token ring connection between the server and requester

*Note:* The environment is best set up and understood with the help of the article "Installing and Using the DOS Database Requester" in Issue 2, 1991, of *IBM Personal Systems Technical Solutions*, G325-5011. The procedure described here would follow the steps in that article.

The DOS machine used to run the program had Micro Focus COBOL/2 with updates to version 2.4.37. (When installing Micro Focus COBOL/2, choose to install for DOS only.) Also installed on the



DOS machine were the IBM C/2 libraries. No fixes were applied to the libraries, and they were set up for real mode. Installing the IBM C/2 libraries is required to give your program access to LLIBCA.LIB during the LINK step. The machine used for compiling these programs had Micro Focus COBOL/2 installed in the C:\COBOL directory. The following files were copied into this directory:

- LLIBCA.LIB
- PCDRSTAT.LIB
- SQLCA\_.CBL.

The SQLCA\_.CBL and PCDRSTAT.LIB files are located in the \DBDRQLIB. The SQLCA\_.CBL file is the only .CBL file needed for the program in Figure 1, however, other .CBL files from the \DBDRQLIB directory may be needed for your Micro Focus COBOL/2 programs. One such file is the SQLDA\_.CBL file used to process dynamic SQL statements. These files must be copied into the C:\COBOL directory to ensure that the compile and link steps have access to the proper external files. If you do not copy the external files into the directory where the compiling will be done, you must ensure that the environment gives access to these files. Adding the directories where the external files are located to the ' PATH = ' statement in the AUTOEXEC .BAT file on the DOS Database Requester will give the required access. This alleviates the need to copy all the required files into one directory.

Once the machines have been properly configured, the following steps are required to actually get the program running.

The first step to creating a successful DOS Database Requester pro-

gram is ensuring that the database calls and parameters the programmer wants to use are supported by DOS Database Requester. A few calls that run on an OS/2 database requester will either not be available to a DOS Database Requester, or need to be modified to run properly on a DOS machine. The program in Figure 1, which catalogs a database, is an example of this.

*After the program has been written, the second step is precompiling it with the SQLPREP instruction.*

On an OS/2 database requester, one of the parameters for the catalog function call lists whether the database is local or remote. If local is indicated, another parameter lists the drive on which the local database exists. The same call for DOS Database Requester ignores the local or remote parameter, because DOS machines cannot have local databases, and the call uses the drive parameter to list whether adapter 0 or adapter 1 is to be used for the communications.

After the program has been written, the second step is precompiling it with the SQLPREP instruction. The program must end with '.SQB'. The .SQB signals the precompiler that a COBOL program needs to be precompiled. The precompiling of the catalog program for this article used the IBM precompiler included with IBM OS/2 EE 1.3. The Micro Focus precompiler was not tested.

The SQLPREP does many things for the program. SQL inside the program is replaced with COBOL function calls to be used by the COBOL compiler. SQLPREPping the program can also allow the OS/2 Database Manager optimizer to build an access plan to the data according to the current statistics that Database Manager holds about the database. During SQLPREP, you have the option to defer binding, which means that your program will not build an access plan to the data in the database. Deferring the bind step means that you must go back and perform an SQLBIND later, but before running the program.

Many programmers and programming shops choose to defer binding to ensure the access plan for their program is up to date. Care should be taken that first-time programmers of OS/2 database requester applications be sure to bind their executables if they have previously chosen to defer binding. Whether binding is deferred or not, the output of the SQLPREP is a .CBL file that will be used when compiling.

The third step is compiling the application. Two versions of each copy file are needed to compile a COBOL database requester application. One version includes no underscore, such as SQLCA.CBL, and is intended for use on OS/2 applications. The other version includes an underscore, such as SQLCA\_.CBL, and is intended for use on a DOS application. Examination of the SQLDA\_.CBL and SQLDA.CBL files included with OS/2 EE 1.3 shows that these are identical files. SQLCA.CBL and SQL CA\_.CBL are also identical for the 1.3 release. The other .CBL files are different for DOS; OS/2 will not work as intended all of the time if the incorrect version is used. The programmer should inspect the .CBL

files in the \SQLLIB directory for differences before assuming that the files required for DOS Database Requester applications will work on OS/2 Database Requesters.

Performing the precompile, compile, and link steps on an OS/2 machine for testing may make program development simpler by removing some obstacles that may be present when moving back and forth between DOS and OS/2. Precompiling and binding can be done only on an OS/2 machine. Compiling the application for introduction into the production environment is best done on the DOS machine after Micro Focus COBOL/2 has been installed according to the instructions in the beginning of this article.

No special compile options were used for the test programs run for this demonstration. The testing done found that including many compiler directives can cause problems that are not easily traceable. A good recommendation is to begin with a simple program before adding complex procedures and directives. No changes should be made to the COBOL.DIR file after installation until a DOS database requester program is running with the default COBOL.DIR.

Once compiling is complete, link the program. You can use either the Micro Focus LINKer or the IBM OS/2 LINKer. The IBM LINKer should be used if you intend to use the IBM support path. The only link library specified is the PCDRSTAT.LIB. If you choose, you can explicitly specify the PCDRSTAT.LIB, LCOBOL.LIB, and LLIBCA.LIB when asked for libraries on the LINK step; however, the LINKer will automatically make use of the LCOBOL.LIB and the LLIBC

A.LIB. No other special link options were used for the program in Figure 1. Again, use only the basic libraries and wait until a simple application is running before adding more libraries.

*Compiling the application for introduction into the production environment is best done on the DOS machine after Micro Focus COBOL/2 has been installed.*

Before running the programs, you must issue SQLLOGON from the DOS Database Requester command prompt. This is so that when the 'start using database' is issued from your program, the database server can verify that the user has access to the database. The SQLLOGON.EXE is one of the files that is copied onto the DOS Database Requester from the \DBDRQLIB directory on the database server. Once the program is precompiled (SQLPREP), compiled, and linked, you can begin trying to run it. To look for the errors in the COBOL/2 DOS Database Requester programs, make generous use of display statements and the SQLCODE. (A few examples of error handling in COBOL programs can be referenced in the *IBM OS/2 Extended Edition Database Manager Version 1.3 Programming Guide and Reference*, Volume 2 (S01F-0292) in the section on programming with COBOL/2.)

A concern when compiling Micro Focus COBOL/2 is the size of the

executable module. Care must be taken to ensure that this module can be loaded and run in the memory left after any device drivers and IBM DOS have been loaded. Micro Focus has a memory management facility called XM that is available with the Toolset and the Workbench. It allows for programs requiring more space than DOS provides under the 640 KB barrier. However, you should be familiar with the use of relocatable .GNT or .INT code and the runtime environment provided by the Toolset and Workbench products before creating applications too large for the available memory in DOS. Micro Focus COBOL/2 DOS Database Requester applications can be developed with no more effort than that required to develop an OS/2 database requester application. You must ensure that the COBOL/2 environment is using DOS. Once Micro Focus COBOL/2 is set up, you should find its environment very robust for developing DOS Database Requester applications, and you will be able to use the Micro Focus tools available for COBOL application development.

#### ABOUT THE AUTHOR

*Andrew Morbitzer is part of IBM's OS/2 Application Assistance Center team, working through the cooperative education program. He spends much of his time on OS/2 Database Manager performance and problem management at customer locations. He holds a B.B.A. in management information systems from Texas A&M University and is currently studying for his M.B.A. in business computer information systems at the University of North Texas.*

```

IDENTIFICATION DIVISION.

PROGRAM-ID.    DOS-DB-REQUESTER.

*****
* This COBOL program is an example of how to catalog a remote      *
* database from a DOS Database Requester.                          *
*****

ENVIRONMENT DIVISION.

DATA DIVISION.

WORKING-STORAGE SECTION.

*****
* SQLCA                                                            *
*****

EXEC SQL INCLUDE SQLCA END-EXEC.

*****
* SLENV                                                            *
* The SLENV_ is special to the DOS DB Requester. The OS/2 SLENV   *
* does not include the underscore character. When the .CBL file   *
* is compiled on the DOS machine, this file will be copied from   *
* the \DBDRQLIB directory.                                        *
*****

COPY SLENV_.

*****
* VARIABLES TO PASS TO CALLS                                       *
*****

77 COMMENT-LENGTH PIC S9(4) VALUE 22 COMP-5.
77 NN-LENGTH      PIC S9(4) VALUE 8   COMP-5.
77 ALIAS-LENGTH  PIC S9(4) VALUE 6   COMP-5.
77 DB-LENGTH     PIC S9(4) VALUE 6   COMP-5.
77 CMT-CODE-PAGE PIC S9(4) VALUE 0   COMP-5.

*****
* I used DRIVE so that my example will use the same variable names as the *
* IBM OS/2 Extended Edition 1.3 Database Manager Programming Guide and *
* Reference Volume 2: Reference on page 3-11. However, the DRIVE should *
* contain the reference to either adapter 0 or 1 for a DOS DB Requester. *
*****

77 D          PIC X.
77 DRIVE      REDEFINES D
              PIC 9(4) COMP-5.

77 T          PIC X.
77 DTYPE      REDEFINES T
              PIC 9(4) COMP-5.

77 COMMENT    PIC X(23) VALUE 'REMOTE SAMPLE DATABASE'.
77 NODE-NAME  PIC X(9)  VALUE 'ANDREWCM'.
77 ALIAS      PIC X(7)  VALUE 'SAMPLE'.
77 DATABASE   PIC X(7)  VALUE 'SAMPLE'.

PROCEDURE DIVISION.
PERFORM 7000-CATALOG

```

Figure 1a. Sample Program



```

        THRU 7000-EXIT.

    PERFORM 8000-TERMINATE
        THRU 8000-EXIT.

    STOP RUN.

7000-CATALOG.

*****
* The following MOVE statements move 0 for the adapter number to D      *
* and move 1 to database type T. The database type of 1 indicates      *
* a remote database. The database type is for an OS/2 requester and    *
* is not used by DOS DB Requester.                                     *
*****

        MOVE 0 TO D.
        MOVE 1 TO T.

*****
* The COBOL form of the following call to catalog a remote database is  *
* shown on page 6-17 of the IBM OS/2 Extended Edition Version 1.3      *
* Database Manager Programming Guide and Reference Volume 2: Reference. *
*****

        CALL "__SQLGCATD" USING DATABASE
                                ALIAS
                                NODE-NAME
                                COMMENT
                                SQLCA
                                BY VALUE DTYPE
                                BY VALUE CMT-CODE-PAGE
                                BY VALUE DRIVE
                                BY VALUE DB-LENGTH
                                BY VALUE ALIAS-LENGTH
                                BY VALUE NN-LENGTH
                                BY VALUE COMMENT-LENGTH.

        DISPLAY 'This is the SQLCODE from Cataloging = ' SQLCODE.

7000-EXIT.
    EXIT.

8000-TERMINATE.

    IF SQLCODE = 0
        DISPLAY 'The program has ended normally'
    ELSE
        GO TO 9000-SQL-ERROR.

    MOVE 0 TO RETURN-CODE.

8000-EXIT.
    EXIT.

9000-SQL-ERROR.
    DISPLAY 'A fatal SQL error has occurred with SQLCODE:' SQLCODE.
    DISPLAY 'The program has ABNORMALLY ended'.
    GOBACK.

```

Figure 1b. Sample Program

# IBM DOS 5.0 Facts and Features

*Midge Portney  
IBM Corporation  
Boca Raton, Florida*

**If you are a DOS user and have experienced RAM CRAM (not enough room in conventional memory to run everything you want to run at the same time), then IBM DOS 5.0, the tenth and latest version of IBM's single tasking operating system, is for you.**

Memory, memory, memory. . . . Memory relief with DOS 5.0 can be quite dramatic. If your system is an 80286 or higher with at least 1 MB of memory, parts of DOS can be moved into an area outside of the conventional 640 KB memory area. This means your operating system is using less than 20 KB of conventional memory. For other hardware, DOS 5.0 is approximately 49 KB, using a little more memory than DOS 3.3, but less than DOS 4.0.

IBM DOS 5.0 replaces and is upwardly compatible with IBM DOS 3.3 and IBM DOS 4.0. The features of DOS 5.0 are improved memory and file management, and improved and expanded utilities.

DOS 5.0 is available in two separate packages:

- The IBM DOS 5.0 Upgrade (U.S. offering) for U.S. users who have a previous level of IBM DOS (between 2.1 and 4.0) installed on the fixed disks of their IBM systems. The upgrade is priced at about one-half of the base DOS product price. Once the upgrade is installed, the DOS 5.0 code is

the same as the base DOS 5.0 product.

- The IBM DOS 5.0 base product (worldwide offering) is for U.S. users who *do not* have a previous level of IBM DOS (between 2.1 and 4.0) already installed on the fixed disks of their IBM systems and all IBM DOS non-U.S. customers. European countries offer the IBM DOS 5.0 base product at full and upgrade prices.

## Installation

Installation of both DOS packages is simple: the user inserts diskette #1 in the A drive and reboots. The default configuration puts DOS into the high memory area and loads SETVER.EXE. (SETVER allows applications designed for a specific version of DOS to run on DOS 5.0.)

When the IBM DOS 5.0 Upgrade is used, the existing AUTOEXEC.BAT and CONFIG.SYS get merged with these statements. In addition, the default configuration makes DOS come up under the DOS Shell.

If you boot your system with the default configuration that is created when you first install the base package, you will be surprised and delighted to find 616 KB of available memory.

## Memory Management

DOS 5.0 includes two memory management device drivers that are based on older versions distributed with DOS LAN Requester and Microsoft® Windows™ 3.0. Because the device drivers have been improved for DOS 5.0, make sure you use these new versions instead of the ones supplied with Microsoft Windows 3.0 or DOS LAN Requester.

- HIMEM.SYS manages the use of extended memory according to the Extended Memory Specification (XMS). This includes the high memory area (HMA), which is where DOS will be loaded. (HMA is the first 64 KB of extended memory above 1 MB.) This driver must be loaded before loading any other device drivers that use extended memory (such as SMARTDRV.SYS, RAMDRIVE.SYS, and EMM386.EXE).
- EMM386.EXE uses extended memory to emulate expanded memory only on 80386 and higher systems. You must have HIMEM.SYS loaded before this driver can be used. (The default amount is 256 KB, but you will probably want to increase it.) EMM386.EXE replaces XMAEM.EXE and XMA2EMS.EXE.

**Upper Memory Block:** The upper memory block (UMB) area is the area between 640 KB and 1 MB. Although this entire area is reserved for hardware, expanded memory pages, and video buffers, once the system is fully configured there are some areas that are unused. DOS 5.0 can use these areas to load either device drivers or terminate-and-stay-resident (TSR) programs, thereby freeing up conventional memory (the first 640 KB of memory). You must have EMM386.EXE loaded to use this area, and it is used only on 80386 and higher systems.

The DOS device drivers that can be loaded in this area (using the DEVICEHIGH command) are:

- EGA.SYS
- DISPLAY.SYS
- ANSI.SYS
- RAMDRIVE.SYS

- DRIVER.SYS
- SMARTDRV.SYS
- PRINTER.SYS

Two of these drivers are new with DOS 5.0:

- RAMDRIVE.SYS creates a temporary disk in RAM, or conventional memory. Use /E to create the disk in extended memory and /A to create it in expanded memory.

This replaces VDISK.SYS.

- SMARTDRV.SYS creates a disk cache in extended or expanded memory. This replaces IBMCACHE.SYS.

The DOS TSR programs that can be loaded in the UMB area (using the LOADHIGH command) are:

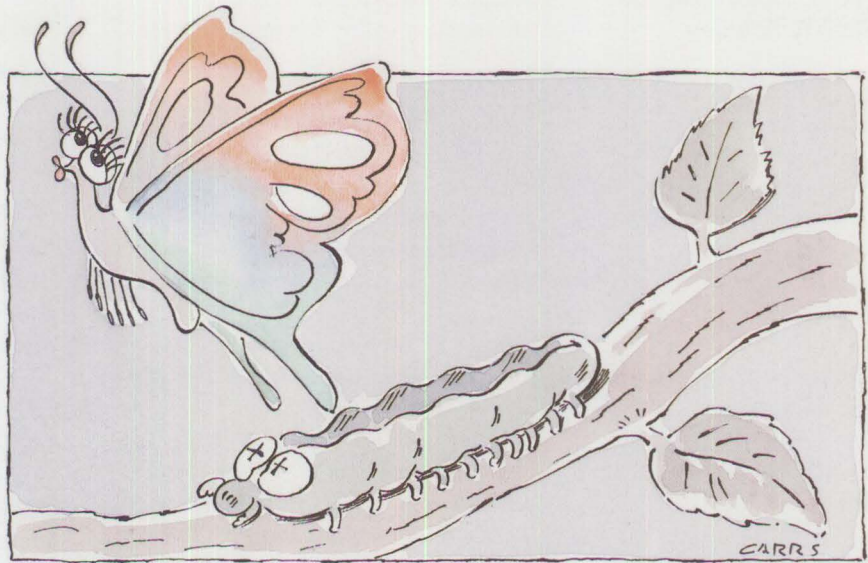
- DOSKEY.COM
- MOUSE.COM
- DOSSHELL.COM
- KEYB.COM
- GRAPHICS.COM
- NLSFUNC.EXE
- MODE.COM
- SHARE.EXE
- PRINT.EXE
- APPEND.EXE

The CONFIG.SYS file must have the following sequence before the UMB area can be used:

```
DEVICE=C:\DOS\HIMEM.SYS
DOS=HIGH,UMB
DEVICE=C:\DOS\EMM386.EXE
NOEMS (or RAM)
DEVICEHIGH=mydriver.sys
```

Place the LOADHIGH statements in the AUTOEXEC.BAT:

```
LOADHIGH myprog.exe
```



To fully utilize the UMB area, load the largest drivers and/or programs first, because DOS will use the largest space as the "HIGH" commands are processed sequentially.

To obtain a memory map, use MEM /C; to view the memory usage, use MEM/DEBUG; to print, use MEM /C > LPT1.

If you are using a VGA color display in graphics mode, you can use the area normally reserved for the monochrome display (B000-B7FF). If you don't run BASIC programs or use the DOS 5.0 full-screen editor, you can use the IBM BASIC ROM area (F600-FDFF) (except on the PS/2 Model 57). When using these with EMM386.EXE, include parameter (I=B000-B7FF I=F600-FDFF) to increase your UMB space by 64 KB.

In addition, use your system's reference diskette to consolidate the various adapter BIOS and data areas. For example, for XGA and Token-Ring (16/4), change the token ring

adapter addresses from D000 to C000.

### DOS Shell

The DOS Shell on IBM DOS 5.0 can perform application context switching in a suspend/resume mode. The applications must be loaded within the DOS Shell. These applications are swapped to disk or to extended memory.

Some DOS Shell highlights are:

- Active task list
- View of directories on the current disk
- View of several directories at one time
- Ability to rename a directory
- Look and feel of OS/2 Presentation Manager and Windows 3.0

### Differences between DOS 5.0 and DOS 4.0

Most of the DOS 5.0 files are compressed (as denoted by an underscore "\_" in the file extension) on the diskettes. In order to use them,

they must be expanded using the new EXPAND.EXE utility.

In DOS 4.0, ANSISYS has a switch /L, which keeps the number of rows set by the mode command as you switch between text and graphics applications. Because of its many incompatibilities with some applications, this support was eliminated from DOS 5.0.

VDISK.SYS has been replaced by RAMDRIVE.SYS, which has the same function. Any programs that used VDISK.SYS should be updated to use RAMDRIVE.SYS.

DOS 5.0 has two BASIC programs: QBasic and BASICA. In order for QBasic to run BASICA programs, the BASICA programs must be saved in ASCII mode, using the command:

```
(SAVE "filename", A).
```

In BASICA, the default is to save a file in binary mode.

BASICA in DOS 5.0 is larger than in DOS 4.0, because it will support PS/2 systems with less BASIC code in the BIOS, such as the PS/2 Model 57. Previous PS/2 models contained 32 KB of BASIC in BIOS. Newer machines contain only 5 KB, and that code now resides in the BASICA.COM file.

The expanded memory device drivers XMA2EMS.SYS and XMAEM.SYS have been replaced by the higher performance EMM386.EXE. As the name implies, EMM386.EXE only has support for memory on 80386 systems. If IBM XMA adapter support is needed for an 80286 system, the XMA2EMS.SYS file can be obtained through IBM DOS 5.0 service. If you have a non-IBM memory

adapter (EMS 3.2 or 4.0), use the driver supplied by the manufacturer.

If your mouse doesn't work properly with the DOS 5.0 Shell, you may need the new MOUSE.COM version 7.04, which is supplied with DOS 5.0 but doesn't get installed automatically. Installation information is found in the *Getting Started* manual.

The DOS 5.0 Shell doesn't recognize the .MEU extension of files created by the DOS 4.0 Shell. The utility MEUTOINI will convert these files for you. If you are installing the upgrade, it will be run automatically. Changes have been made to functions of the Shell, and certain installation switches are no longer supported. Further information is found in the *Getting Started* manual.

In DOS 4.0, SHARE was loaded automatically to support hardfile partitions of greater than 32 MB. DOS 5.0 has improved partition support, with a savings of 8 KB, and no longer requires SHARE.

## New and Enhanced Utilities

Here are the new utilities in DOS 5.0:

- **Doskey** provides command line history and macro capability.
- **Unformat** complements the new safe format.
- **Undelete** has the ability to recover deleted files.
- **Fc** is a second file compare utility that compares two files of different sizes.
- **Setver** allows applications to think there is a different level of DOS on the system. (For instance, some applications require DOS 3.3.)

Here are the enhanced utilities in DOS 5.0:

- **Dir** has sort options, attribute filter, and DIR/SUBDIR file matches
- **Format** has safe format as the default and saves control data. There is also a quick format option.
- **Attrib** allows the manipulation of system and hidden files
- **Restore** has a /D option that will allow users to view the backed-up file list.

## OS/2 Dual Boot

When installing DOS 5.0 on a dual boot system, before leaving the OS/2 command prompt, you must point to the DOS directories by typing **BOOT /DOS**. In addition, during the installation, if you are asked if you want to replace all occurrences of your files, say "NO" because some DOS and OS/2 files have the same name. A DOS FAT partition must be located before an OS/2 high performance file system (HPFS).

## OS/2 Upgrades

When OS/2 2.0 becomes available, you will be able to upgrade your DOS 5.0 to OS/2 2.0.

## DOS 5.0 and PS/1™

To install DOS 5.0 on a PS/1, the environment must point to the fixed disk instead of to ROM DOS, so that you can start your PS/1 from fixed disk instead of from ROM. There are certain things you need to know, so refer to the PS/1 documentation before installing DOS 5.0. Your ROM Shell will no longer work.

## DOS 5.0 Changing Hardfile Partition Size

If you have a fixed disk that is larger than 32 MB and is partitioned to 32 MB with your current version

of IBM DOS, you may repartition the fixed disk to any size. Install DOS 5.0 over your current version of IBM DOS. Although the upgrade was designed for this, you may also use the base DOS 5.0 program. If you are using the upgrade, back up your current level of DOS with the backup option of the install. Once DOS 5.0 is installed, try it out and make sure that it works to your satisfaction. Then do a complete backup of all of the disk partitions on your fixed disk using the BACKUP command.

Create a boot diskette in the A drive by formatting a blank diskette with /S. Also on the diskette, copy FDISK, FORMAT, and RESTORE from your DOS 5.0 subdirectory. Copy your favorite editor, too. (If you choose the editor on DOS 5.0, copy over QBASIC.EXE and EDIT.COM.) Remove the diskette from the A drive. Use FDISK to set up your new partition (you will be erasing your fixed disk, so make sure you backed it up). This will instruct you to restart DOS with the boot diskette in the A drive.

Once you have restarted the machine, run FORMAT /S from the boot diskette and format your hardfile. Then you can reinstall DOS 5.0 using your DOS 5.0 program. After it is installed, do a RESTORE of your backup diskettes. Use caution if you've had identical subdirectory names on different disk partitions.

### DOS 5.0 and 7552 Industrial Gearbox™

A special parameter with the HIMEM device driver is required in the CONFIG.SYS to load DOS HIGH on the IBM Industrial Computer Model 7552. When you have completed the initial DOS 5.0 installation, before you use Ctl-Alt-Del to

reboot, press F3 to get out of the installation procedure, put the first DOS 5.0 diskette in the A drive, and edit CONFIG.SYS in the root of C to include this parameter:

```
DEVICE = C:\DOS\HIMEM.SYS
        /MACHINE:15
```

If you have HIMEM.SYS loaded without /MACHINE:15, the 7552 Industrial Gearbox will hang when you try to boot your system.

All other industrial computers will work fine as installed.

*A new feature, command line help, is accessed by entering help or the command name followed by /?.*

### Online Documentation

To help you understand DOS 5.0 better, files with compatibility and how-to information are provided. In addition, there is an important file (on diskette #2 of the 3.5" package and diskette #3 of the 5.25" package) called the PACKING.LST file. This file tells you on which diskettes all of the DOS files are located. (If you are installing to diskette, it will also tell you which files are on the target diskettes.)

README.TXT has the latest online information. It contains additional information on memory, OS/2, Windows, mouse drivers, and different networks.

APPNOTES.TXT will help you run existing applications with DOS 5.0.

It gives you tips and techniques to get maximum productivity as quickly as possible with this latest version of DOS.

A new feature, command line help, is accessed by entering **help** or the command name followed by **/?**.

"Help" lists all the commands and their functions. Can't remember the syntax for the **del** command? Type

```
del /?
```

and not only will you get a brief explanation, but the permissible syntax functions, as well.

The installation feature has function keys that give you help or allow you to return to a previous screen or exit the program. The help is so complete you may never have to open the documentation. To start the installation, just put diskette #1 of the DOS 5.0 or the DOS 5.0 Upgrade in your computer and start your system (Ctl-Alt-Del).

You will enjoy losing yourself in the levels of DOS Shell help, and it is equally accessible with the mouse or keyboard. Put your book away and prepare for an online adventure!

For upward compatibility, BASICA (BASIC interpreter) and EDLIN (line editor) continue to be available. A second BASIC interpreter program, QBasic, with advanced features and a full-screen editor, EDIT, are exciting additions to the product. To find out some of the differences between QBasic and BASICA, and the conversion requirements to use QBasic to run your BASICA programs, look at the "Survival Guide," which is the first screen after invoking QBasic. Choose "Contents," and then "Converting BASICA Programs." (DOS 5.0 has a program, REMLINE.BAS, that helps in the

conversion.) Working through these screens gives you a lot of information that will make you comfortable with QBasic without ever opening a book! All documentation for QBasic is online.

Although the full-screen editor is part of QBasic, you can type "edit" to start it, without going into QBasic first. It is a text-mode editor with block and line marking, has mouse support, and full online documentation.

### Service

A good service and support structure is a key priority. Users want to be able to ask questions – quickly and easily. IBM has broadly expanded service and support to include both a toll-free number and access to an Electronic Bulletin Board. The toll-free number is 1 800 237-5511 and the customer number is DOS 5692. Corrective Service Diskettes will be available at your IBM Authorized Dealer, through the toll-free number, IBM branch offices, or the Electronic Bulletin Board (National Support Center, Atlanta, 404 835-6600). The DOS 5.0 service information card gives details about the enhanced service.

### National Language Support

Ever want to send a letter to someone in Turkey? Your U.S. version of IBM DOS 5.0 supports Turkish characters as well as a host of others. A small booklet titled *Keyboards and Code Pages* is included with DOS 5.0. This booklet has keyboard layouts (when you choose to write with Turkish characters, your keyboard layout will change), code page tables, and available accented characters.

## DOS 5.0 and Network Support

With each new version of DOS, your network must be modified accordingly. The network code needed to use DOS 5.0 with the IBM PC LAN program, IBM DOS LAN Requester, or Novell NetWare is included in the IBM DOS 5.0 Upgrade. This is the first time that DOS is supplying network support in its initial offering. If you have another type of network, call IBM Service or your network supplier.



*Although the full-screen editor is part of QBasic, you can type "edit" to start it, without going into QBasic first.*

### Hardware Support

The following new hardware features are supported in DOS 5.0:

- The new 2.88 MB 1-inch high diskette drives are high-density (2.88 MB, also known as 4 MB) 3.5-inch diskette drives. These all have media-sensing capability.
- Some models of 1.44 MB and 2.88 MB diskette drives have media-sensing capability. With this feature, when you format a diskette, the system recognizes the type of diskette in the drive and formats it accordingly. Previously, the FORMAT command was used to format a diskette according to the size of the drive containing the diskette. For example, a 1.44 MB diskette drive formatted a 720 KB diskette incorrectly as 1.44 MB. The de-

fault in the FORMAT statement was the size of the drive. This could cause data errors. With media-sensing, such errors are prevented.

- The 3.5-inch Read/Write Optical Disks are new IBM products that allow large amounts of data to be stored on an optical disk. New multimedia sound and motion technology, which requires vast amounts of data, is one of the many uses for these disks.
- Keyboards that now have DOS 5.0 support include the 122-key host-connected PS/2 keyboard and the 106-key Japanese keyboard.

### Extended DOS Support

ACCESS DOS is a supplement to IBM DOS 5.0 that provides extended keyboard, mouse, and sound access for IBM DOS users. It is very helpful for people with disabilities. Here are some features:

- Sticky keys – allow sequential rather than simultaneous processing of keystrokes (like Ctl-Alt-Del)
- Mouse keys – allow the numeric keypad to perform like a mouse
- Repeat keys – customize how fast held-down keys repeat
- Slow keys – customize the length of time a key is pressed before it is processed
- Bounce keys – prevent double characters from being processed
- Serial keys – allow control of the keyboard and mouse through an input device (not included)
- Toggle keys – use beeps to indicate when Caps, Num, and Scroll Lock keys are activated

This supplement is available free of charge through IBM. Call 1 800 IBM-2468.

## IBM DOS 5.0 and MS-DOS® 5.0 Upgrade

What are the differences between IBM DOS 5.0 and Microsoft's MS-DOS Upgrade? Both IBM DOS and MS-DOS, once installed, have the same commands and essentially the same documentation. But IBM has "customized" the generic MS-DOS.

MS-DOS 5.0 is offered only as an upgrade to a previous DOS system. IBM offers the choice of the DOS Upgrade, which is competitively priced, and the base DOS 5.0, for users who do not already have DOS installed. The base DOS will also work on systems that have a previous level of IBM DOS installed.

MS-DOS has been tested on clone hardware thoroughly; IBM has gone a step further in testing IBM DOS on IBM hardware.

**Install:** MS-DOS 5.0 and IBM DOS 5.0 both offer upgrades to an existing DOS system. IBM DOS Upgrade is self-booting, installs *only* over an existing IBM DOS from 2.1 through 4.0, and will not install to diskettes. In addition, IBM DOS 5.0 Upgrade requires you to have IBM hardware. If you are a user of IBM DOS prior to DOS 2.1 or an IBM DOS user with a diskette-only configuration, you can call 1 800 IBM-7699 to purchase a copy of the full package at the Upgrade price. MS-DOS does not check for any of these things except for the existence of a prior DOS. It does this by running the install as an application (your system must be running before you can start the MS-DOS install by typing A:SETUP). MS-DOS 5.0 Up-

grade will install to either fixed disk or diskette.

MS-DOS will install across a LAN.

**Hardware:** Although MS-DOS 5.0 Upgrade will install to all systems and IBM DOS, once installed, is only warranted on IBM hardware, the base IBM code is functionally the same (except for BASICA, QBasic, and EDIT). QBasic and EDIT may require an update from IBM. Please see additional information under "BASIC/Editor." If a problem is found using IBM DOS on clone hardware, the user can call IBM for service and support. If, after analysis, IBM determines that it is not a problem in the IBM DOS product, then IBM refers the user to the appropriate hardware vendor.

*Both IBM DOS and MS-DOS, once installed, have the same commands, and essentially the same documentation. But IBM has "customized" the generic MS-DOS.*

**Remote IPL:** There are some differences that may affect the user. The "hidden" files IBMBIO.COM and IBMDOS.COM are known as IO.SYS and MSDOS.SYS in MS-DOS 5.0. This difference affects the remote IPL feature (RIPL). IBM DOS may be installed by an OS/2 LAN Server for use by DOS RIPL machines without having to create separate boot images for each workstation. Because the MS-DOS system file names are different (IO.SYS and MSDOS.SYS), the boot images

can be created and RIPL'd with MS-DOS diskettes; however, OS/2 LAN Server's "parameter passing" feature (to allow workstations to use the same image) does not work.

**BASIC/Editor:** MS-DOS ships with only QBasic, and even though prior versions had GW-BASIC (similar to BASICA), it is no longer shipped with DOS 5.0. If IBM DOS 5.0 is installed on non-IBM hardware, an updated QBasic is required for QBasic and EDIT (the full-screen editor) to run properly. BASICA will not work on non-IBM hardware. If you do not have QBASIC.EXE dated 8/16/91 or later, please call IBM Service. You can also get this module from your dealer, your IBM branch office, or the Electronic Bulletin Board.

### Support Unique to IBM:

8514A – Using MS-DOS 5.0 with a color display and the IBM 8514A adapter, the install and DOS Shell come up in monochrome. IBM DOS works properly in these environments.

CD-ROM SETVER – Users should make sure the following statement is added to the SETVER table if they are using IBM DOS 5.0 (MS-DOS has it included).

```
SETVER MSCDEX.EXE 4.00
```

SHELL/MOUSE support – For users of DOS 4.0, there is a conversion utility that will update the MEU files to the IBM DOS Shell 5.0 format. In addition, the 5.0 Shell-compatible mouse is included in IBM DOS 5.0, although this is available through MS support.

4755 Cryptographic adapter with EMM386 – IBM DOS 5.0 supports the use of the 4755 Cryptographic

adapter if EMM386 is loaded. MS-DOS does not.

National Language Support – Country support for Greece, Turkey, and Iceland are available in IBM DOS 5.0, but not in MS-DOS 5.0.

#### **Corrective Service Diskettes**

**(CSD):** IBM has a policy of issuing CSDs periodically to better serve customer needs. In the past Microsoft has not provided this.

**Uninstall Option:** During installation, the IBM DOS 5.0 Upgrade gives the user the option of creating (or not creating) UNINSTALL diskettes. MS-DOS does not give the user this option; the user must copy some of the “old DOS” files onto diskettes. MS-DOS also copies many of the “old DOS” files into a separate directory of the fixed disk, limiting available fixed disk space.

**Bootable Diskettes:** MS-DOS 5.0 diskettes are not bootable, but rather, are run as an application. This means enough memory has to be available, with a stable environment, to ensure smooth installation. Because both IBM DOS 5.0 products are bootable, this is not a problem for the IBM products.

The IBM DOS self-booting feature is valuable for the repartitioning of hard disk drives and for recovering from computer viruses.

**Network Support:** IBM DOS 5.0 Upgrade contains full Novell NetWare drivers, including EMS and XMS support. MS-DOS upgrades only the base memory driver from Novell; therefore, the memory you save with MS-DOS 5.0 is used for the Novell drivers. With IBM DOS, you achieve memory savings and get

the memory management Novell drivers.

Drivers for 3COM networks and LAN MAN networks are included in MS-DOS. These drivers are also available on CompuServe in the MSNETWORKS forum, library one, supplemental disks. Try searching for “supp.” Call 3COM for additional information on the 3COM network products.

To use IBM DOS 5.0 with Banyan Vines 4.0X, on the command line type SETVER REDIR4.EXE 4.0. MS-DOS 5.0 already has this statement in the SETVER table. Banyan Vines 4.10 has DOS 5.0 support and is available.



### *The IBM DOS self-booting feature is valuable for the repartitioning of hard disk drives and for recovering from computer viruses.*

**Service:** The service philosophies of IBM and MS differ; each supports the customer a little differently. IBM offers free service for one year, while Microsoft offers it for three months. IBM Corrective Service Diskettes are cumulative fixes available to all customers. Microsoft corrective service is a single fix to a customer with a problem, and Microsoft uses CompuServe to make certain fixes available to customers.

The End User Bulletin Board for IBM is Electronic BBS and NSC/Atlanta, while Microsoft uses CompuServe. Both Microsoft and IBM have 24-hour/7-day phone numbers to report a problem. The IBM Support Center number is 1 800 237-5511; the MS Automated Support System number is 1 800 646-5103.

For technician support, IBM has programming services available Monday through Friday for 8 hours per day for 1 year (1 800 237-5511); Microsoft has Technician support Monday through Friday 8 hours per day for 90 days, with support after 90 days at \$2 per minute (1 900 896-9000) or \$20 per call (206 646-5108).

#### *ABOUT THE AUTHOR*

*Midge Portney is the external compatibility planner for entry operating systems and extensions. She has worked in DOS/FORTRAN development and system software compatibility testing for DOS at IBM's Entry Systems Division. Midge has a B.A. in mathematics from Antioch College, a B.A. in computer systems from Florida Atlantic University, and an M.A. in mathematics education from Harvard University.*

The information about MS-DOS 5.0 Upgrade, IBM DOS 5.0, and IBM DOS 5.0 Upgrade is based upon information and packages available on June 11, 1991, the time these products were announced. All references to MS-DOS 5.0 or MS-DOS refer to the MS-DOS 5.0 Upgrade. All references to IBM DOS 5.0 or IBM DOS refer to both base and upgrade products unless otherwise specified.



## IBM DOS 5.0 Upgrade

*Pylee Lennil and  
Clovis L. Tondo  
IBM Corporation  
Boca Raton, Florida*

**The IBM DOS 5.0 Upgrade allows DOS users to upgrade their current versions of IBM DOS to IBM DOS 5.0. The Upgrade is less expensive than the DOS 5.0 Base Installation program and is designed for users who already have IBM DOS installed.**

The DOS 5.0 Upgrade has several facilities to help the user upgrade from an earlier DOS version. The package:

- Allows users to back up the old version of DOS onto diskettes before installing DOS 5.0
- Allows users to restore the old DOS from backup diskettes
- Upgrades the DOS CONFIG.SYS and AUTOEXEC.BAT files to conform to the DOS 5.0 CONFIG.SYS and AUTOEXEC.BAT file standards
- Upgrades the DOS 4.0 shell menu files to DOS 5.0 shell menu file format
- Allows users to replace all occurrences of their DOS system files
- Provides network update files for PC LAN Program and Novell networks

### Backing Up the Old DOS

The DOS 5.0 Upgrade allows users to back up DOS versions 2.1 through 5.0 onto diskettes before installing DOS 5.0. When the upgrade program starts, a backup menu is displayed. If the user chooses to back up the old system, the program copies the old DOS files to the backup

diskettes. If necessary, the user can restore the old DOS version from the backup diskettes using the Uninstall feature.

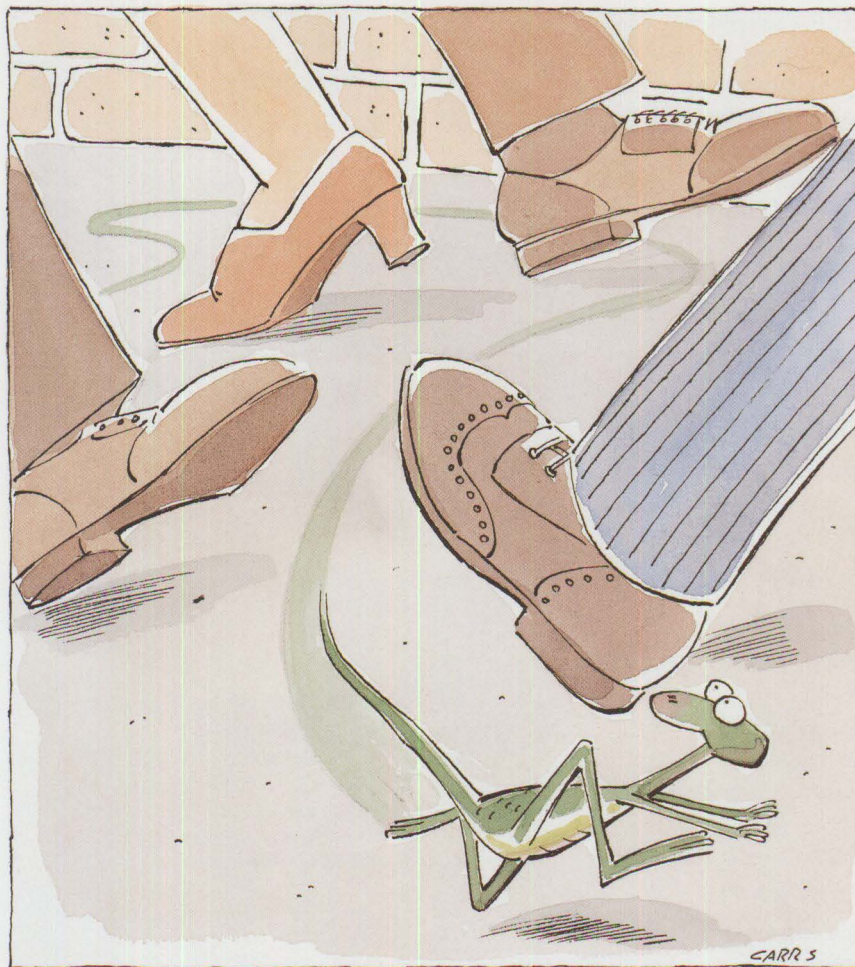
### Restoring the Old DOS

The Upgrade allows users to restore the old DOS, which was previously saved using the backup facility. The program displays the Uninstall menu. If the Uninstall option is selected, the program requests that the first backup diskette is inserted and DOS 5.0 is removed from the fixed disk. Then it reboots the system. An internal procedure restores the old DOS from the backup diskettes.

It is important to note that users must go through the Uninstall procedure to uninstall DOS 5.0. This is because the boot record for earlier versions of DOS is somewhat different, and it is necessary to use the Upgrade to restore the boot record and the files.

### Upgrading CONFIG.SYS and AUTOEXEC.BAT

The Upgrade allows users to upgrade the existing CONFIG.SYS and AUTOEXEC.BAT files after copying the DOS 5.0 system files to the disk. The upgrade involves making the necessary changes to these



### Default version for CONFIG.SYS

```
DEVICE=C:\DOS\SETVER.EXE
DEVICE=C:\DOS\HIMEM.SYS
DOS=HIGH
FILES=10
```

### Default version for AUTOEXEC.BAT

```
@ECHO OFF
PROMPT $p$g
PATH=C:\DOS
SET TEMP=C:\DOS
C:\DOS\DOSSHLL.EXE
```

Figure 1. Defaults

files to conform to the DOS 5.0 CONFIG.SYS and AUTOEXEC.BAT standards. The changes are necessary because DOS 5.0 has some new commands, and some existing commands were modified. The program copies the CONFIG.SYS and AUTOEXEC.BAT files to the CONFIG.DAT and AUTOEXEC.DAT files, respectively, before it upgrades those files to DOS 5.0 standards. The CONFIG.SYS and AUTOEXEC.BAT file modifications are necessary because of the following changes:

- These commands were removed from DOS 5.0:
  - INSTALL=IFSFUNC.EXE
  - INSTALL=FILESYS.EXE
- These commands were added in DOS 5.0:
  - DEVICE=SETVER.EXE
  - DEVICE=SMARTDRV.SYS
  - DOS=HIGH
  - Note that  
DEVICE=SETVER.EXE  
provides version changes to programs;

```
DEVICE=HIMEM.SYS
and
DOS=HIGH
load DOS into high memory
area.
```

- These commands were replaced by new commands:
  - DEVICE=XMAEM.SYS  
and  
DEVICE=XMA2EMS.SYS  
were replaced by  
DEVICE=EMM386.EXE  
(Note that  
DEVICE=XMA2EMS.SYS  
still is required for 80286 systems with expanded memory adapter hardware.)
  - DEVICE=IBMCACHE.SYS  
was replaced by  
DEVICE=SMARTDRV.SYS
  - DEVICE=VDISK.SYS

```
was replaced by
DEVICE=RAMDRIVE.SYS
```

If the CONFIG.SYS and AUTOEXEC.BAT do not exist under the current DOS system, the DOS 5.0 Upgrade creates default versions for those files. Figure 1 shows the defaults.

If the CONFIG.SYS file already exists on the root directory, the Upgrade program modifies this file to conform to the DOS 5.0 CONFIG.SYS standards.

Figure 2 shows a sample CONFIG.SYS file. "Old CONFIG.SYS" shows the file before DOS 5.0. "New CONFIG.SYS" shows CONFIG.SYS after the update. Note that the upgrade program may not be able to upgrade some lines in the file; therefore, those lines will be commented (the program begins the line with REM).

### Old CONFIG.SYS

```
break=on
files=10
device=xmaem.sys
device=xma2ems.sys frame=C000
device=ibmcache.sys
install=c:\dos\ifsfunc.exe
device=c:\net\pcnet.sys
device=c:\dos1\vdisk.sys
```

### New CONFIG.SYS

```
DEVICE=C:\DOS\SETVER.EXE
DEVICE=C:\DOS\HIMEM.SYS
DOS=HIGH
break=on
files=10
DEVICE=C:\DOS\EMM386.EXE frame=C000
REM device=ibmcache.sys
DEVICE=C:\DOS\SMARTDRV.SYS
REM install=c:\dos\ifsfunc.exe
device=c:\net\pcnet.sys
REM device=c:\dos1\vdisk.sys
DEVICE=C:\DOS\RAMDRIVE.SYS
```

Figure 2. Comparison of Old and New CONFIG.SYS

**Old AUTOEXEC.BAT**

```
PATH=C:\;C:\DOS1;
ifsfunc.exe
call autouser.bat
prompt=$p$g
```

**New AUTOEXEC.BAT**

```
PATH=C:\DOS;C:\;C:\DOS1
REM ifsfunc.exe
call autouser.bat
prompt=$p$g
SET TEMP=C:\DOS
```

**Note that the path command has been updated to include the new DOS path.**

**Figure 3. Comparison of Old and New AUTOEXEC.BAT**

The user should examine the new CONFIG.SYS and CONFIG.DAT (the old CONFIG.SYS) files and should make the necessary changes on the commented lines, based on the support available for the commands in those lines, to obtain similar support in DOS 5.0.

Similarly, if an AUTOEXEC.BAT exists in the root directory, the program modifies this file to conform to DOS 5.0 AUTOEXEC.BAT standards.

Figure 3 shows a sample AUTOEXEC.BAT file before and after upgrading to the DOS 5.0 standard. "Old AUTOEXEC.BAT" shows the file before DOS 5.0 installation. "New AUTOEXEC.BAT" shows the file after the update. Note that the program may not be able to upgrade some lines. In this case, the program will comment those lines (the program begins the line with REM). The user should examine the new and old AUTOEXEC.BAT files and

should make necessary changes to include these lines based on the support available for the commands specified in the lines under DOS 5.0.

### Upgrading the DOS 4.0 DOS Shell Menu Files

If the user has DOS 4.0 .MEU files, and if DOS 4.0 is the current DOS on the fixed disk, the DOS 5.0 Upgrade invokes a utility to convert the .MEU files to .INI format.

DOS 4.0 uses a collection of .MEU files supporting menus. These are binary files that form a dependency tree. The first file is SHELL.MEU,

which may contain one or more submenus. Each submenu is stored in a separate .MEU file. The utility understands the hierarchy for the shell menu files and converts those files into DOSSHELL.INI format.

The menu system in DOS 5.0 uses a file called DOSSHELL.INI. This is an ASCII file that contains the information about menus and submenus for DOS 5.0. (There is one .INI file as opposed to various .MEU files.) The update program copies a version of DOSSHELL.INI to the user's disk. If the user wants to upgrade the existing .MEU files, the program



uses a utility to transform the .MEU files, then append that information to the existing DOSSHELL.INI. The existing .MEU files are not deleted.

The DOS 5.0 shell has a few lower limits when compared to the DOS 4.0 shell. For example:

- The maximum number of prompts in the DOS 5.0 shell is 9; it was 250 in the DOS 4.0 shell.
- The maximum length of help text is 256 bytes; it was 500 bytes.
- The program item title is 23 characters long; it was 40.

Refer to *Getting Started* in the DOS 5.0 Upgrade for more details.

### Easy Migration

The DOS 5.0 Upgrade facilitates the transition from earlier DOS versions, starting with version 2.1, to the latest version of DOS: DOS 5.0.

The user may want to back up the existing system before installing DOS 5.0. The program provides a facility to back up the system files associated with an earlier version of DOS. If the user chooses to reinstall the earlier version of DOS at a later date, the program knows how to

copy the files from the backup diskettes and replace the existing boot record on the fixed disk.

The upgrade program updates existing CONFIG.SYS and AUTOEXEC.BAT files. If those files do not exist, it brings DOS 5.0 versions of those files with the rest of the system.

It then copies the DOS 5.0 system files to the user's machine.

There is one more step: if the user has .MEU files (for DOS 4.0 shell menus), those files will be converted and appended to the .INI file that DOS 5.0 uses.

The purpose of the upgrade program is to make it easier for a DOS user to migrate to DOS 5.0.

### ABOUT THE AUTHORS

*Pylee Lennil is a staff programmer at IBM Entry Systems Division. He is presently involved in the development of AIX. He previously spent several years in PC DOS development. Pylee joined IBM in 1983. He received his B.S. in physics from Kerala University in India, and an M.S. in computer engineering from the University of Lowell in Lowell, Massachusetts.*

*Clovis L. Tondo is an advisory programmer at IBM Entry Systems Division, with AIX system design. He has an M.S. from Southern Illinois University and a Doctorate in computer science from Nova University. Dr. Tondo is also a visiting professor of computer science at Nova University, Fort Lauderdale, Florida.*



# DOS 5.0 Performance Improvements

*Duc J. Vianney  
IBM Corporation  
Boca Raton, Florida*

**DOS 5.0 provides two major improvements over DOS 4.0 that will considerably affect the performance of your applications: reduction in memory requirements and enhancement of fundamental system utilities such as expanded memory manager (EMM386.SYS), disk caching (SMARTDRV.SYS), and virtual disk RAMDRIVE (RAMDRIVE.SYS). This article discusses those improvements including empirical data collected from certain industry-standard benchmarks.**

DOS 5.0 achieves its memory reduction goals by:

- Reducing the amount of low 640 KB RAM, that is, conventional memory, consumed by DOS.
- Providing the option of running parts of DOS from the High Memory Area (HMA) as defined in the Extended Memory Specification (XMS) 2.0. This is available on many 286-, 386-, and 486-based machines that provide at least 64 KB of extended memory
- Relocating buffers into the HMA. The HMA is the first 64 KB of memory above the 1 MB boundary that marks the end of conventional memory and the beginning of extended memory.

## Physical Size Reduction of

**Resident DOS** The actual physical size of the resident DOS module was reduced by 1,378 bytes compared to DOS 4.0. The reduction

was achieved through the following code changes:

- Replaced support for buffers in EMS and hash organization of buffers with DOS 3.3 LRU algorithm. This means a return to DOS 3.3 buffer management algorithm, in which buffers are organized as a circular linked list, maintained in LRU order, and provide a pointer to the last buffer used. The internal hash organization has also been removed, although the secondary caching scheme is still supported in DOS 5.0 (for example, BUFFERS=XY).
- Removed FASTOPEN extents caching. The removal of FASTSEEK support from the FASTOPEN.EXE program and the resident DOS results in a savings of 1 KB of resident code and data.
- Enabled access to large partitions without loading share. In DOS 5.0, you no longer need to have SHARE loaded for 32 MB and greater fixed disks.
- Removed IFS. This includes the removal of the undocumented IFS, which was added in DOS 4.0.
- Optimized message retriever and other COMMAND.COM code. The code optimization in COMMAND.COM code has reduced the amount of resident memory required by about 1,800 bytes.
- Added reusable COMMAND.COM code. The resident COMMAND.COM code has been made reusable. As a result, multiple invocations of COMMAND.COM share the same code, creating a savings of 2 KB for each invocation after the initial COMMAND.COM.
- Built Upper Memory Block (UMB) support into DOS. Pro-

grams and device drivers can be loaded high via the device line DEVICEHIGH= in CONFIG.SYS and LOADHIGH from command line or AUTOEXEC.BAT.

**DOS Execution in HMA:** The second approach used in reducing memory requirements is to run certain DOS functions in HMA, which include:

- Execution of COMMAND.COM resident code in the HMA. COMMAND.COM was made relocatable by separating the code and data segments, and by making instructions that are not relative – jump tables, for example – to become relocatable.
- Internal DOS interface for sharing HMA with other DOS device drivers and DOS utilities.

## Buffers Relocation in HMA:

The relocation of the DOS kernel and various buffers used by the kernel also contributes significantly to the reduction of conventional memory. DOS 5.0 moves the following modules into HMA:

- DOS Kernel and BIOS – When the DOS kernel and BIOS are loaded into memory, if the HMA is found, DOS and BIOS are relocated to the HMA and initialized there. Otherwise, they will be initialized and executed in the conventional memory area.
- DOS Buffers – In DOS 4.0, the number of buffers requested by the line BUFFERS= in the CONFIG.SYS file is allocated in conventional memory. In DOS 5.0, buffers automatically go into the HMA space, if available.
- Code Page Buffers – These are buffers used by DISPLAY.SYS and its relocation into the HMA can mean an additional 20 KB

Conventional Memory (in Kbytes)	DOS 4.0	DOS 5.0 Low	DOS 5.0 High
Interrupt Area	1.0 KB	1.0 KB	1.0 KB
BIOS Data Area	0.3 KB	0.3 KB	0.3 KB
System Data	0.5 KB	0.5 KB	0.5 KB
DOS	56.0 KB	54.0 KB	13.0 KB
Program Area	5.8 KB	4.6 KB	2.6 KB
Total	63.6 KB	60.4 KB	14.4 KB

Figure 1. Memory Used by DOS Functions

savings for international users with two code pages loaded.

- Part of the HIMEM.SYS – HIMEM is an extended memory manager whose main function is to control and manage the use of extended memory by such programs as disk caching, RAM disk, and other extended memory

programs like Windows 3.0. In DOS 5.0, HIMEM.SYS implements an internal interface with DOS that relocates parts of HIMEM.SYS to the HMA.

**Effect of Size Reduction:** When a system is booted and initialized under either DOS 5.0 or 4.0, the Extended BIOS area occupies 1 KB at

the high end of the 640 KB boundary, leaving only 639 KB available to DOS. Using a one line CONFIG.SYS with FILES=20 and no AUTOEXEC.BAT file, DOS 4.0 yields 589,152 bytes of conventional memory to applications, while DOS 5.0 low yields 596,536 bytes and DOS 5.0 high yields 636,400 bytes, respectively. “DOS 5.0 low” means DOS is configured to run in conventional memory (as specified by the statement DOS=LOW in CONFIG.SYS); “DOS high” means DOS is configured to run in the HMA (as specified by the statement DOS=HIGH in CONFIG.SYS).

In summary, Figure 1 shows the amount of conventional memory taken by various DOS functions under DOS 5.0 and DOS 4.0.

#### Performance Comparison Between DOS Low and DOS High:

Given the complexity in size reduction and relocation of the kernel to the HMA, a primary concern is whether the implementation of the memory reduction strategy impacts performance of the base operating system. To measure the performance impact of DOS relocation into HMA, a set of benchmarks was run on three PS/2 platforms: 8555-061, 8580-311, and 8595-OJ9. The benchmarks include the PC Labs version 5.6 (see Reference 1 at the end of the article), PC Week version 1.1 (see Reference 2), and a host of simulated application benchmarks from the NSTL (see Reference 3). The data illustrated in Figure 2 represents the percent of performance difference between DOS high and DOS low on each hardware platform and the average of all three platforms. A negative (positive) percent value means DOS high is slower (faster) than DOS low by that percentage. The result in Figure 2 suggests that there is no performance difference between DOS

System Under Test	8555-061	8580-311	8595-OJ9	Average
<b>CPU Performance Tests:</b>				
PC Labs				
Instruction Mix	NC	NC	NC	NC
Instruction Mix - 80286	NC	NC	NC	NC
Instruction Mix - 80386	NC	NC	NC	NC
Integer Add	NC	NC	NC	NC
PC Week				
Total: CPU/sort	NC	NC	NC	NC
<b>I/O Performance Tests:</b>				
PC Labs				
BIOS Disk Seek - Random	NC	NC	NC	NC
BIOS Disk Seek - Sequential	NC	NC	NC	NC
DOS File Access Total - Small and Large Record Size	NC	NC	NC	NC
PC Week				
Disk Speed - Database Test	-1%	+2%	-2%	NC
<b>Application Tests:</b>				
NSTL-C	NC	NC	NC	NC
NSTL-Foxpro	+4%	+1%	+8%	+4%
NSTL-Word	NC	NC	NC	NC

NC = No Change

Figure 2. DOS Low and DOS High Performance Comparison

low and high in most primitive processing and I/O operations. However, certain application benchmarks, such as the NSTL-Foxpro (see Reference 3), might run faster when DOS is loaded in the HMA. In the CPU performance tests, both PC Labs and PC Week benchmarks show no change (NC) in performance between DOS low and high. In the I/O performance tests, the PC Week benchmark shows its database test ran 2% faster on the 8580-311 in DOS 5.0 high, yet 1% and 2% slower on the 8555-061 and 8595-OJ9. On the average, these variations cancel each other out, which results in insignificant performance change. In the application benchmarks, the Foxpro benchmark ran consistently faster in DOS high, resulting an average of 4% performance gain overall.

**Performance Comparison Between DOS 5.0 and DOS 4.0:** Using the same measurement approach just described, the performance of DOS 5.0 low and DOS 4.0 is shown in the Figure 3. Again, "NC" means the performance change is not observable, while a negative (positive) percent value means DOS 5.0 is slower (faster) than DOS 4.0 by that percentage. While the processor subsystem shows no reaction to either DOS, DOS 5.0 certainly improves the performance of applications considerably through its I/O performance improvement as shown by the PC Labs benchmark. The PC Labs DOS File Access benchmark ran 25% faster on the 8555-061 and 8580-311, 9% faster on the 8595-OJ9, and on the average, 20% faster on DOS 5.0 compared to DOS 4.0. The Foxpro benchmark seems to benefit most from DOS 5.0, with the the average improvement of 9% over DOS 4.0.

System Under Test	8555-061	8580-311	8595-OJ9	Average
<b>CPU Performance Tests:</b>				
PC Labs				
Instruction Mix	NC	NC	NC	NC
Instruction Mix - 80286	NC	NC	NC	NC
Instruction Mix - 80386	NC	NC	NC	NC
Integer Add	NC	NC	NC	NC
PC Week				
Total: CPU/sort	NC	NC	NC	NC
<b>I/O Performance Tests:</b>				
PC Labs				
BIOS Disk Seek - Random	NC	NC	NC	NC
BIOS Disk Seek - Sequential	NC	NC	NC	NC
DOS File Access Total - Small and Large Record Size	NC	NC	NC	NC
PC Week				
Disk Speed - Database Test	-1%	+2%	-2%	NC
<b>Application Tests:</b>				
NSTL-C	NC	NC	NC	NC
NSTL-Foxpro	+4%	+1%	+8%	+4%
NSTL-Word	NC	NC	NC	NC

NC = No Change

Figure 3. DOS 4.0 and DOS 5.0 Performance Comparison

### EMM386.EXE ENHANCEMENT

EMM386.EXE is an expanded memory emulator that uses extended memory to simulate expanded memory to meet the requirements of applications that use expanded memory on 386- and 486-based systems. DOS 5.0 EMM386 provides a full LIM 4.0 EMS standard, in compliance with VCPI, fully supports the Virtual DMA Services specification, and searches for and provides upper memory blocks (UMBs) via the XMS 2.0 Allocate Upper Memory Block call. EMM386 must be loaded from a device line in CONFIG.SYS.

Figure 4 shows the performance gain of various Excel (see Reference 3) functions performed under DOS 5.0 and EMM386.EXE compared to

DOS 4.0 and XMAEM.SYS. While no performance change was observed in transcendental functions, charting, and array operations, considerable gain was seen in the last four groups of functions: defined functions (21%), scaled insert/delete (45%), scaled cut/paste (29%), and large link and recalc functions (43%).

### RAMDRIVE.SYS

The virtual disk, known as VDISK.SYS in DOS 4.0, is now called RAMDRIVE.SYS. It is a utility program that uses system memory to emulate a temporary disk drive. As a result, because the information is read from memory (known as VDISK or RAMDRIVE) rather than from a physical disk, the response of your application is very fast. However, because the informa-

EXCEL 2.1 Functions:	Percent of Performance Gain
Transcendental Functions	NC
Cut/Paste	+3%
Charting (line, bar, area, pie, column, etc.)	NC
Insert/Delete (rows, columns)	+7%
Financial Functions (ddb, dv, ipmt, nper, npv, rate, etc.)	+3%
Array Operations (constant and parameterized arrays)	NC
Database Functions (dmax, dmin, dcounta, dproduct, dstdev, etc.)	+14%
File Operations (open, close, delete, save)	+9%
Defined Functions (solving system of equations)	+21%
Scaled Insert/Delete	+45%
Scaled Cut/Paste	+29%

Figure 4. Performance of EXCEL Functions

tion is stored temporarily in memory, it will be lost when the system is turned off. In DOS 5.0, RAMDRIVE uses XMS calls to allocate and manipulate extended memory. If RAMDRIVE is instructed to use extended memory and an XMS manager is not present, RAMDRIVE won't install. This makes RAMDRIVE compatible with the latest standard for using extended memory, and removes machine-dependent code from RAMDRIVE. RAMDRIVE now runs with DOS versions up to 5.x.

In general, it has been observed that I/O operations perform better if they are running from the ram drive under DOS 5.0 as opposed to running

from a virtual disk under DOS 4.0. Figure 5 shows how the DOS file access test in the PC Labs performs under both DOS 5.0 and 4.0. In DOS 5.0, the test was run from the RAMDRIVE created by the device line RAMDRIVE.SYS, while in DOS 4.0, the test was run from the virtual disk created by the device line VDISK.SYS in the CONFIG.SYS file. The value in the table represents the percent of performance gain between DOS 5.0 and DOS 4.0. For example, on system 8555-061, based on the total time of tests involving small record size (Total - Small Record Size), the test ran 21% faster on DOS 5.0 compared to DOS 4.0. In the average, across three different

System Under Test	8555-061	8580-311	8595-OJ9	Average
<b>PC Labs-DOS File Access</b>				
Total-Small Record Size	+21%	+29%	+29%	+26%
Total-Large Record Size	+30%	+46%	+42%	+40%
<b>DOS File Access</b>				
Total-Small / Large Record Size	+22%	+32%	+31%	+39%

Figure 5. Ram Drive Performance Comparison

platforms, the same test ran 26% faster on DOS 5.0.

## SMARTDRV.SYS

SMARTDRV is a disk caching program that can be used to significantly reduce the time required to read and write data from your hard disk. SMARTDRV can use either extended memory (default) or expanded memory (optional) for its disk cache.

In DOS 5.0, SMARTDRV uses XMS calls to allocate and manipulate extended memory. As a result, if you put the cache in expanded memory provided by EMM386.EXE, your system might not run as fast as expected because of the overhead required by EMM386.EXE to emulate expanded memory. There are a few enhancements added to the DOS 5.0 SMARTDRV:

- It can now buffer all disk I/O via its track buffer in conventional memory to safeguard against bus master DMA hardware that may not translate virtual addresses correctly. This double-buffering mechanism may be enabled or disabled on the installation command line (/B+ or /B-) or, by default, the DMA Services Active bit in the BIOS data area is checked and double-buffering done when DMA Services are active.
- A quick cache-locking mechanism has been introduced to allow Windows to demand page without flushing other data from the SMARTDRV cache. There are now two cache-locking mechanisms. The older mechanism locks and unlocks the cache through IOCTL Writes. In this case, SMARTDRV locks tracks currently in the cache. If any cache elements are free, they remain unlocked and may change their contents while the lock is in



effect. In the new mechanism, whenever the cache is locked, SMARTDRV will not replace any track currently cached. If any elements of the cache are free when the cache is locked, they too became locked as soon as they are used.

- Caching strategy has been modified to improve SMARTDRV performance with disk writes. Disk writes may update cache contents but never cause new tracks to be added to the cache.
- SMARTDRV's resident track buffer is no longer repositioned when it falls across a DMA boundary. DOS detects this condition and breaks the DMA into pieces without significantly affecting SMARTDRV performance.

Figure 6 shows the performance gain of tests that were running with and without SMARTDRV under DOS 5.0, on an 8555-061 and an 8580-311 system. The PC Labs benchmark shows the average gain of 30% for the DOS File Access Test and 51% for the DOS Variable Size File Access test. In the Disktester (see Reference 4) benchmark, typical average gain with SMARTDRV over non-SMARTDRV ranges from 31% to 53%. The Disktester benchmark consists of a series of file operations such as file creation, sequential read and write, and random read and write performed on various file size, record size, and number of records. The data presented in Figure 6 is the average result of all tests.

## Conclusion

DOS 5.0 provides sharp improvements over DOS 4.0. It requires not

System Under Test	8555-061	8580-311	Average
PC Labs Benchmark			
<b>DOS File Access Test:</b>			
Total-Small Record Size	+33%	+29%	+31%
Total-Large Record Size	+10%	+32%	+21%
Total-Small / Large Record Size	+31%	+29%	+30%
<b>DOS Variable Size File Access</b>			
Total-Small Record Size	+49%	+56%	+53%
Total-Large Record Size	+40%	+56%	+48%
Total-Small / Large Record Size	+46%	+56%	+51%
<b>Disktester Benchmark</b>			
FS=256,RS=512,#RCDS=512	+36%	+26%	+31%
FS=256,RS=4096,#RCDS=64	+34%	+33%	+33%
FS=512,RS=1024,#RCDS=512	+44%	+47%	+46%
FS=512,RS=2048,#RCDS=256	+44%	+46%	+45%
FS=1024,RS=2048,#RCDS=512	+53%	+53%	+53%
FS=1024,RS=8192,#RCDS=128	+45%	+50%	+48%

Note: FS = File Size RS = Record Size #RCDS = Number of Records

Figure 6. Average Result of All Tests

only less memory but offers more functions at better performance. Applications can now take advantage of HIMEM.SYS for a more robust environment, relocation of the DOS kernel and various buffers to gain more conventional memory, and high performance utilities for a faster time to solution.

## References:

- [1] PC Labs Hardware Benchmark, Version 5.6, PC Magazine Labs Benchmark, 11/1990.
- [2] PC Week Labs System Benchmark, Ziff Communications Co., 1990.
- [3] NSTL Benchmarks. A series of application-oriented benchmarks available from the National Software Testing Laboratories. NSTL-C Benchmark: A compilation and link of various C programs for used in a development environment. NSTL-Foxpro Benchmark: A collection of tests based on the Foxpro 1.01 (Fox Software) involving database creation, record and file manipulation, and report generation. NSTL-Word Benchmark: Contains tests involving Microsoft Word 5.0 such as

loading a document, block copy, spelling check, text scrolling, text saving, and so forth. NSTL-Excel Benchmark: Microsoft Excel 2.1-based tests that include most basic spreadsheet functions.

[4] Disktester, Version 1.04, Intelligent Devices Corporation, 1989.

## ABOUT THE AUTHOR

*Duc Vianney is an advisory programmer at IBM's Personal Systems Programming Center in Boca Raton. He is currently involved with the performance evaluation of DOS 5.0, Windows and OS/2. He received a B.S. in mathematics from the University of Texas at El Paso and a Ph.D. in computer science from the University of Southwestern Louisiana.*

# DOS Memory Management Facilities

Pylee Lennil  
IBM Corporation  
Boca Raton, Florida

This article describes the different kinds of memory and how DOS manages each one. It includes a description of the DOS 5.0 memory management enhancements.

DOS memory management facilities allow applications to access system memory on a variety of hardware in a well-behaved manner. The basic DOS memory management functions are:

- Loading applications into memory
- Allocating memory blocks to applications during execution
- Providing an environment that allows applications to access memory above 640 KB

The type of memory management support provided by DOS depends on the system processor and the DOS version used. Intel® 8086/8088 processors support only real mode and have a limited address space of 1 MB. The 80286™ and higher processors support both real and protected modes and can address up to 16 or 32 MB. In real mode, processors can address only up to 1 MB, and in protected mode, the processors can address up to 16 or 32 MB. Protected mode provides large memory space to applications and is used by operating systems like OS/2. Because DOS and its applications run under real mode, their address space is restricted to 1 MB. The actual memory space available is even less because address space between 640 KB and 1 MB is re-

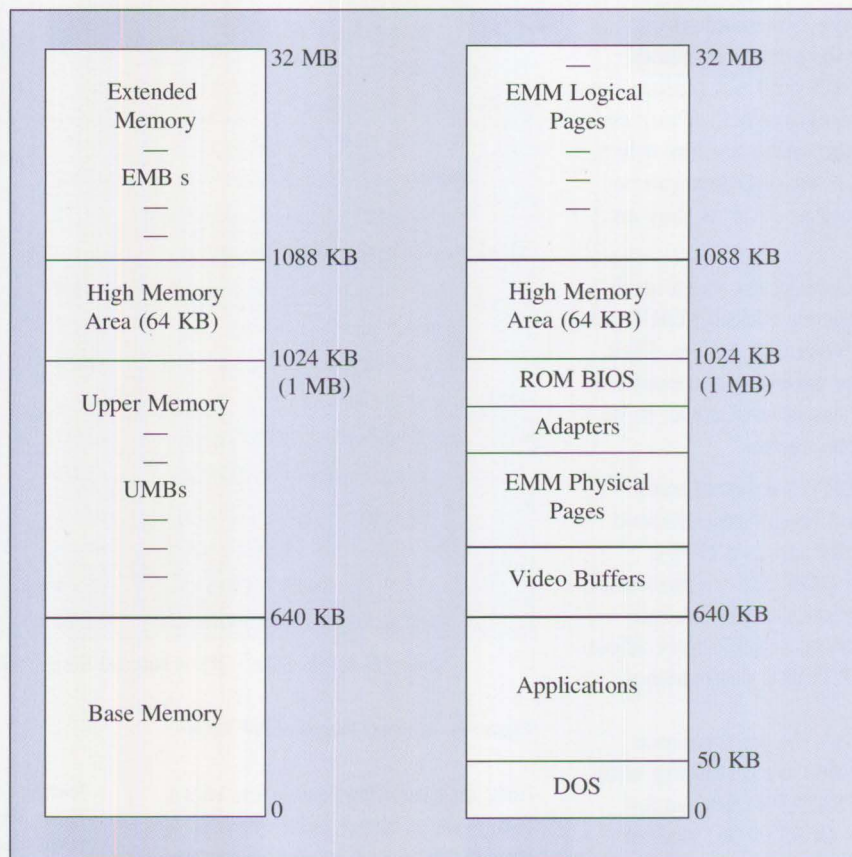


Figure 1. System Memory Areas

served for video buffers, adapters, and ROM BIOS.

System memory is divided into base memory, upper memory, high memory, and extended memory (Figure 1). Normally, DOS and its applications run in base memory. Accessing memory above 640 KB requires systems with 80286 or higher processors, special memory management drivers, and DOS 4.0 or 5.0.

## Base Memory Management

Base memory is located between 0 and 640 KB. DOS occupies approximately 50 KB of base memory. The remaining base memory is available for applications. When an application is loaded, it receives all the available memory up to 640 KB.

The application must release the unused memory. This way, it can subsequently allocate or free memory blocks as needed during execution by calling the DOS memory management function calls. The caller must specify memory size, and then DOS returns the segment of the allocated memory block. Following are the three DOS memory management functions. A detailed description of these functions is given in the *Disk Operating System User's Guide and Reference* (84F9682).

- Function 48H – allocate memory blocks
- Function 49H – free memory blocks
- Function 4AH – resize memory blocks

## Upper Memory Management

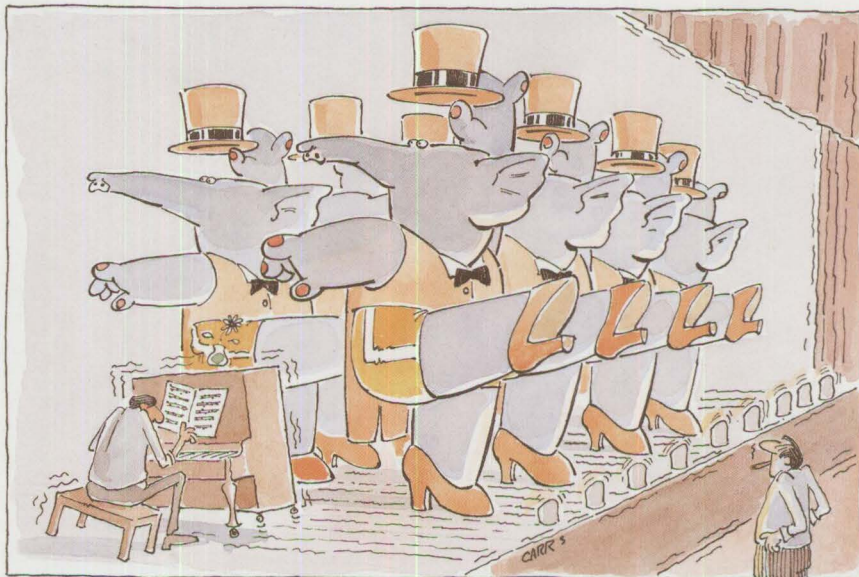
Upper memory is located between 640 KB and 1 MB. Normally this area is reserved for video buffers, adapters, and ROM BIOS. If adapters are not present, the address spaces they would occupy are free. Upper memory blocks (UMB) are these unused address spaces. Because no physical memory exists at these locations, neither DOS nor applications can access these blocks directly. To make use of these blocks, you can use an emulator program.

How do emulators work? The emulator simulates extended memory using protected mode page tables. The emulator gains control every time DOS or applications access the upper memory block, and it maps the address into extended memory.

UMBs can be accessed by loading Microsoft Extended Memory Specification (XMS) driver (HIMEM.SYS). It provides functions that request and release upper memory blocks. The UMBs can be used as data buffers, or they can be loaded with programs. A detailed description of the HIMEM.SYS can be found in the Microsoft Extended Memory Specification (XMS) Version 2.0.

## High Memory Management

High memory is a 64 KB block of memory located between 1024 KB and 1088 KB. This memory is unique because it can be accessed by programs running in real mode, even though it exists over 1 MB. The high memory area has a unique segment number FFFFh. Using this segment, a program can address from FFFF:10h to FFFF:FFFFh (64 KB minus 16 bytes). Note that high memory does not start at FFFF:0000h, because the first 16 bytes are reserved for system use. High memory can be used as data



buffers or it can be loaded with programs.

Unlike other memory areas, high memory has several restrictions. Systems based on 80286 or higher processors provide high memory support. The entire area must be used as a single segment with a unique segment number FFFFh. The segment cannot be shared with other programs. Finally, it can be accessed only if address line A20 is enabled. High memory can be accessed using the Microsoft XMS driver HIMEM.SYS. HIMEM.SYS allows programs to access high memory in a well-behaved and machine-independent manner. It provides functions to enable the line A20, as well as to allocate or free the high memory area. A detailed description of the HIMEM.SYS driver can be found in the Microsoft XMS Version 2.0.

## Extended Memory Management

Extended memory starts at 1088 KB and occupies all available memory

above 1 MB (Figure 1). This memory area cannot be accessed by programs running in real mode. Accessing extended memory requires extended or expanded memory drivers running in protect or virtual 86 mode supported by 80286 and higher processors. DOS 5.0 contains both expanded memory manager EMM386.EXE and the extended memory manager HIMEM.SYS.

### Extended Memory Driver:

Microsoft XMS driver (HIMEM.SYS) provides functions to allocate, lock, free and move extended memory blocks. The allocated extended memory blocks can be used only for data storage. A detailed description of HIMEM.SYS can be found in the Microsoft XMS Version 2.0.

**Expanded Memory Driver:** The expanded memory driver, EMM386.EXE, allows DOS and applications to access extended memory blocks (EMB). The expanded memory driver creates one or more 16 KB physical pages in upper mem-

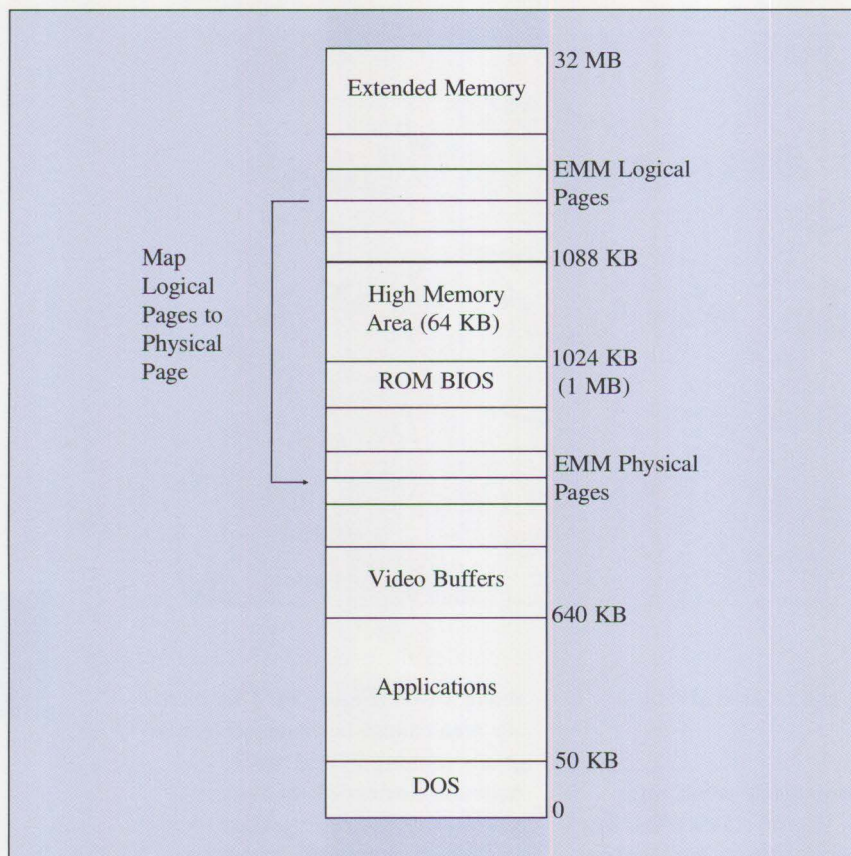


Figure 2. Expanded Memory Logical and Physical Pages

ory or in base memory (Figure 2). It also divides extended memory into 16 KB logical pages. To access extended memory logical pages, the program transfers data into one of the physical pages. The expanded memory driver maps the extended memory logical page into this physical page. This is like using a physical page as a window to access extended memory logical pages.

The actual location of the physical pages may vary depending on the configuration of the hardware. To use expanded memory, you must load the expanded memory driver using the **device=** command in CONFIG.SYS. The driver provides a set of functions to allocate physical pages and map logical pages into physical pages. Extended memory

blocks can be used as data buffers or they can be loaded with programs. A detailed description of expanded memory manager functions can be found in Lotus/Intel/Microsoft Expanded Memory Specification Version 4.0.

### DOS 5.0 Memory Management Enhancements

The major problem facing applications under DOS is the limited memory space. In base memory, DOS occupies approximately 50 KB. If other stay-resident programs or drivers are already loaded, the application may have little memory to run. DOS 5.0 has implemented the following memory management enhancements that provide additional memory to applications:

- DOS 5.0 kernel can be relocated to high memory on 80286 or higher processor-based systems, leaving only a small kernel stub in the base memory (Figure 3). This releases approximately 30 KB of memory to applications.
- DOS 5.0 allows loading device drivers and stay-resident programs into upper memory on 80386 or higher processor-based systems.
- DOS 5.0 does not require SHARE.EXE to be loaded for greater than 32 MB hardfile partition support.

Relocating DOS kernel, stay-resident programs, or device drivers requires DOS 5.0 memory management drivers HIMEM.SYS and EMM386.EXE. HIMEM.SYS allocates extended memory, high memory, and upper memory area. Therefore, HIMEM.SYS must be loaded before relocation can be done. Relocating DOS kernel needs only HIMEM.SYS, whereas relocating device drivers and stay-resident programs requires both HIMEM.SYS and EMM386.EXE. Also, DOS 5.0 provides the configuration commands shown in Figure 4 to relocate DOS drivers and stay-resident programs.

The following command loads DOS into high memory area. If **dos=high** is not specified, DOS will be kept in base memory.

```
device = himem.sys
dos=high
```

The EMM386.EXE simulates expanded memory and also allocates upper memory blocks. Specifying the **ram** option allows emulation of expanded memory, as well as allocation of upper memory blocks. If no emulation is needed, then the **noems** option should be used instead of

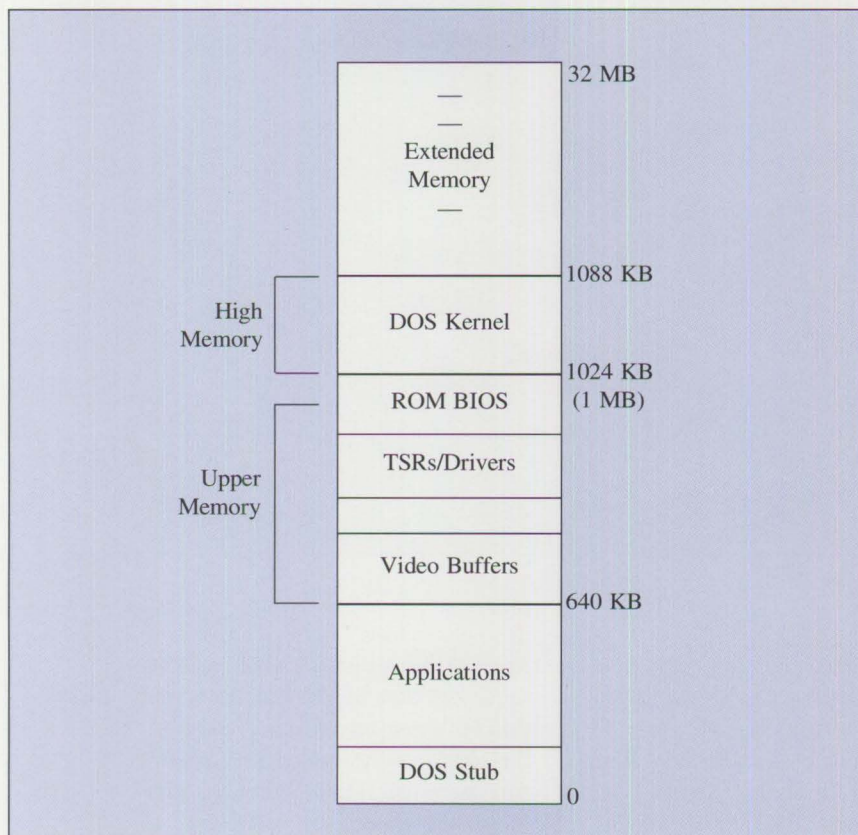


Figure 3. DOS 5.0 Memory Structure

**ram** option. Because EMM386.EXE calls HIMEM.SYS to allocate extended and upper memory blocks, HIMEM.SYS must be loaded prior to EMM386.EXE.

The following command emulates expanded memory and loads the

driver **mydriver.sys** into upper memory block while keeping DOS in base memory:

```
device = himem.sys
device = EMM386.exe ram
dos=umb
devicehigh=mydriver.sys
```

dos=high	Load DOS into high memory area
dos=umb	Load drivers and stay-resident programs into upper memory area
loadhigh=xxxx.com	Load stay-resident program xxxx.com into upper memory
devicehigh=xxxx.sys	Load driver xxxx.sys into upper memory

Figure 4. Configuration Commands

The following command loads DOS into high memory area, the driver **mydriver.sys**, and the stay-resident program **myprog.com** into upper memory area. The **noems** option prevents EMM386.EXE from emulating expanded memory.

```
device = himem.sys
device = emm386.exe noems
dos=high,umb
devicehigh=mydriver.sys
loadhigh=myprog.com
```

To summarize, the DOS memory management facilities are available based on the type of processors and the DOS version used. On 80286 or higher processor-based systems, an application can access memory above 640 KB by using extended or expanded memory drivers. DOS 5.0 provides additional memory to the base memory applications by loading DOS into high memory or drivers and stay-resident programs into upper memory.

#### ABOUT THE AUTHOR

*Pylee Lennil is a staff programmer at IBM Entry Systems Division in Boca Raton, Florida. He is presently involved in the development of AIX. Previously, he spent several years in PC DOS development. Pylee joined IBM in 1983. He received a B.S. in physics from Kerala University in India, and an M.S. in computer engineering from the University of Lowell in Lowell, Massachusetts.*

## Disk Caching Under DOS

Pylee Lennil  
IBM Corporation  
Boca Raton, Florida

**Disk caching is a software scheme that improves system performance by saving the recently accessed disk data blocks for later use. This article presents a general overview of disk caching and how various disk caching schemes are implemented under DOS.**

Applications running under DOS waste most of their time waiting for data from disk, especially if the applications perform frequent disk access. This is because accessing data from a disk is much slower than accessing the same amount of data from memory. To access a specific disk sector involves physically moving the read/write head to the right track and then waiting until the specific sector appears under the head. These mechanical movements take time, affecting the performance of the applications.

The most common technique used for solving this hardware problem through software is by creating a disk caching mechanism. What is disk caching? Disk caching involves saving blocks of data to and from the disk in a memory buffer to reduce the number of disk accesses. The fewer the disk accesses, the better the performance.

### Disk Caching

Disk caching can be divided into the following categories:

*Write-through* caching immediately writes data blocks to disk. The main advantage of this method is that it



maintains file system integrity. Since the blocks are not kept in memory for delayed writing, the chances of losing the data by accidental system shut-off are eliminated. On the other hand, this method requires frequent disk access, which impacts system performance. DOS uses write-through caching only if specifically requested through DOS Extended Open function call.

*Write-back* disk caching is based on a delayed disk write. A cache buffer keeps data blocks to be written to a specific track on the disk. Several disk writes to the same track are performed at one time, avoiding repeated writes to the same track. This is nothing more than flushing the cache buffers periodically to the appropriate track on the disk. The delayed write has an inherent problem. If the system is turned off before the data is written to the tracks, the system fails to update the file system information, causing file system corruption. DOS normally writes the data blocks to disk only when the file is closed.

*Read-through* disk caching reads the requested data blocks into an application buffer. It also keeps a copy of the blocks in system cache buffers for future use. The assumption here is that the application will most likely request the same sectors in subsequent reads. DOS utilizes this method during all disk reads by keeping a copy of the most recently read disk sectors in a group of system buffers. Read-through caching is also used by disk caching programs like IBMCACHE and SMARTDRV.

*Read-ahead* disk caching involves reading a few additional sectors immediately following the requested sector into a cache buffer. These sectors may be beneficial in the case of a sequential disk access. In DOS, this method is implemented as look-ahead buffers supported by the `BUFFERS=m,n` command in `CONFIG.SYS`.

These disk caching methods are used in DOS either internal or external to DOS. The internal disk caching schemes are built into DOS as system buffers, look-ahead buffers,

and cluster and disk directory information cache buffers. The external caching is enabled by loading caching programs like IBMCACHE, SMARTDRV, or any other similar caching program. These programs are loaded as DOS drivers or stay resident programs.

### DOS Internal Disk Caching

DOS internal disk caching is designed based on the workings of the DOS file allocation table, disk directory, and the general operation of the disk. The internal disk caching methods are:

**System Buffers:** DOS system buffers provide a basic disk caching mechanism that uses the read-through caching method. System buffers are a group of circular buffers maintained by DOS for caching the data flowing between the disk and the application. As shown in Figure 1, each buffer consists of a buffer header and a 512-byte buffer data area. Each buffer can hold one sector. The buffer header contains information such as the link address to the next buffer, the sector number associated with the buffer, and a flag indicating if the buffer is free.

How do system buffers work? To understand buffers, consider the following steps required for accessing a file:

1. Read the file's path information from the disk directory
2. Read the file's sector location information from the file allocation table
3. Read the file data sectors from the disk

During these steps, the sectors read are kept in the system buffers. During subsequent disk read requests,

DOS checks the system buffers to see if the required sectors are available in the cache. If they are available, further disk reads are avoided, thereby improving performance. The number of system buffers is specified by the "m" parameter in the BUFFERS=m,n command. Once the buffers are full, buffers holding the least recently used sectors are freed to make room for newly read disk sectors.

System buffers provide better performance during random disk reads. Specifying a large number of system buffers may further enhance random disk reads, but it may degrade the performance of sequential disk reads. This is because large buffers require an extensive search through the buffers for sectors that are unlikely to be found under sequential disk reads. Also, a large number of buffers will occupy large portions of base memory, leaving less memory for applications.

**Look-ahead Buffers:** In DOS, caching the adjacent sectors is done using a group of look-ahead buffers. These buffers hold a few sectors ahead of the requested sectors, with the assumption that these sectors will be requested during a sequential file access. The look-ahead buffers are enabled using the BUFFERS=m,n command in the CONFIG.SYS file. Here "n" specifies the number of look-ahead buffers. For instance if n=2, then when the application requests DOS to read sector 50, DOS reads sectors 51 and 52 in addition to sector 50, and keeps these additional sectors in the look-ahead buffers. Look-ahead buffers provide best performance during sequential disk reads and provide little or no improvements under random disk reads.

### File Allocation and Path

**Information Caching:** DOS disk caching is also designed based on the workings of the DOS disk directory and file allocation table. Performance will be improved if portions of the file allocation table or disk directory are kept in main memory for subsequent use. This caching method is implemented in FASTOPEN under DOS. How does FASTOPEN work? FASTOPEN is a stay-resident program that caches the file sector location information and the file path information of the most recently accessed files. To see how FASTOPEN works, consider the following steps DOS performs during a file access:

1. Reads the file's path information from the disk directory
2. Reads the file's sector location information from the file allocation table

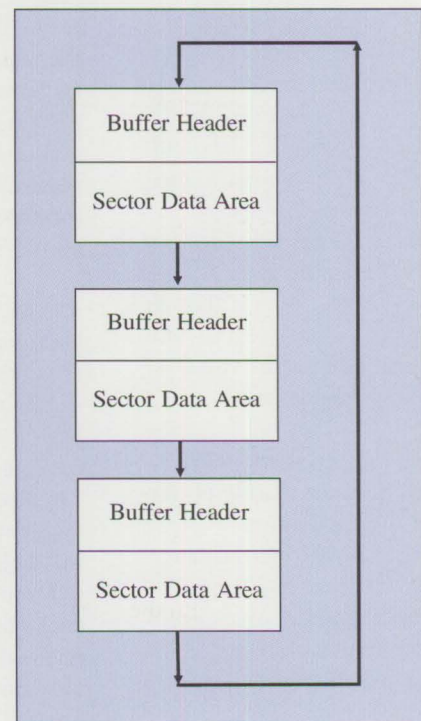


Figure 1. DOS System Buffers

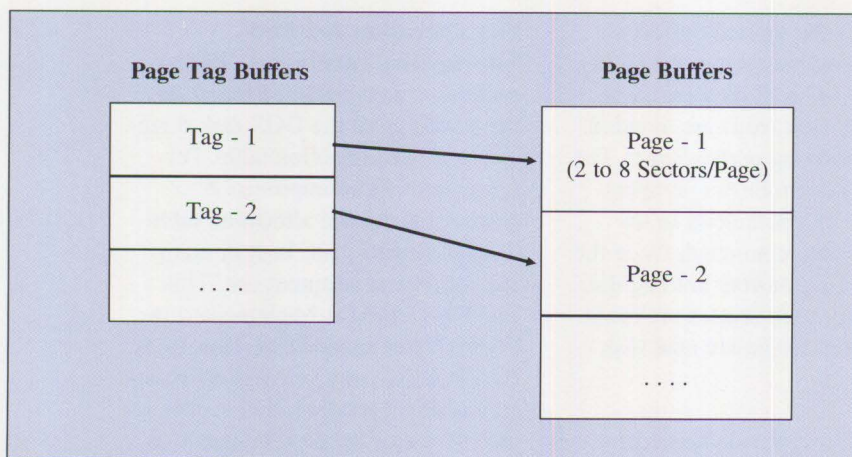


Figure 2. Page and Page Tag Buffers

During file access, FASTOPEN keeps both the path and sector locations of files in cache buffers. This information can be reused for subsequent access of the same files. This effectively reduces steps 1 and 2, which, in turn, improves performance. FASTOPEN is enabled using the command FASTOPEN C: (n,m). Here "n" is the number of file path buffers, and "m" is the number of sector information buffers. If the buffers are full, then space is created by deleting the least recently used file information from the cache buffers. The "m" and "n" values in the FASTOPEN command must be specified based on the way files and directories exist on the disk. A large "m" value should be used if the file path consists of many levels of sub-directories and a large "n" value if files are highly fragmented.

Performance improvement under FASTOPEN depends mainly on the behavior of the application. FASTOPEN improves performance for applications accessing fragmented files. It also improves performance for applications that require repeated file and directory access.

### DOS External Disk Caching

DOS external disk caching is provided by cache programs like IBMCACHE and SMARTDRV. These programs can be loaded as DOS device drivers. They maintain their own cache buffers either in base memory, extended memory, or expanded memory. During disk access, the program takes control and reads the sectors transferred between the disk and the application into cache buffers for future use. Unlike system buffers, which provide optimal performance under random disk access, these programs improve performance for both sequential and random disk-access environments.

The cache buffer consists of two areas as shown in Figure 2. The large area contains pages and the small areas contain page tags. Each page has a fixed size that holds two to eight disk sectors. Each tag identifies a page in the page buffer. The tag contains information on the sectors held in each page. During disk I/O, multiple pages are read into the cache buffer, depending on the size of the data block requested. Even if only one sector in a page is requested, the entire page is read into

the buffer. This works like the look-ahead buffer method. Additional sectors read are the most likely to be used by sequential reads.

The major advantage of DOS external disk caching programs is that they support both read-through and read-ahead caching methods. This improves the performance of sequential as well as random disk reads. External caching allows users to specify a large number of buffers capable of loading them into extended or expanded memory. Because they are device drivers or resident programs, they can be loaded only if needed.

### Summary

System buffers improve performance for random disk I/O. Look-ahead buffers improve performance under sequential disk I/O. FASTOPEN improves performance for accessing fragmented files, and in the case of repeated access of files and directory path names. External disk caching programs improve performance for both sequential and random disk I/O.

### ABOUT THE AUTHOR

*Pylee Lennil is a staff programmer at IBM Entry Systems Division in Boca Raton, Florida. He is presently involved in AIX development. Before joining the AIX team, he spent several years in PC DOS development. Pylee began his career with IBM in 1983. He received a B.S. in physics from Kerala University in India and an M.S. in computer engineering from the University of Lowell in Lowell, Massachusetts.*



## NetWare Client-Server Interaction

*Paul Turner  
Novell Inc.  
Provo, Utah*

On a NetWare LAN, several software components work together to enable communication between the workstation client and the file server. The NetWare workstation shell (NETx.COM) manages the connection with the server, while IPX.COM forms the communications link with the network hardware. This article presents a behind-the-scenes look at how these pieces use NetWare's communications and routing protocols to establish and maintain the vital client-server connection.

The NetWare shell facilitates client-server communications for DOS-based workstations. In a typical client-server interaction, one station (the client) requests services from another station (the server). Through the shell, DOS-based applications can request file services (such as writing to and reading from files) from NetWare file servers. At the workstation, the shell, the user application, and the user together act as the client requesting file services; the NetWare file server acts as the server providing file services.

The shell, then, is the liaison between the client (the user application) and the server. The shell performs the tasks necessary to request file services from a NetWare file server: for example, establishing a connection with the file server,

maintaining the connection, and terminating the connection.

The NetWare shell is a terminate-and-stay-resident (TSR) program called NETx.COM (where x depends on the version of DOS being run). NETx.COM is loaded into a NetWare workstation's memory when the workstation is booted. Before you load the shell, however, you must load another TSR called IPX.COM.

### IPX.COM

Novell's Internetwork Packet Exchange (IPX) protocol serves as the communications link with the network interface card (NIC) installed in the workstation. At installation, a customized version of IPX.COM is generated for each workstation by linking in a driver written specifi-



cally for the NIC that resides in that workstation. Once IPX.COM is loaded, any workstation program, including the shell, can communicate on the network through NetWare's IPX protocol.

Recently, as part of its Open Data-Link Interface (ODI) specification, Novell began distributing a new set of modules that provide greater flexibility than IPX.COM. Under ODI, an intermediate link support layer (LSL) separates the NIC driver from IPX and other protocol stacks. This separation allows NIC drivers to be independent of IPX, thereby eliminating the need to generate a custom IPX.COM file for each workstation. The LSL also allows multiple protocol stacks to communicate over the same NIC simultaneously.

To bring up a workstation using ODI, the link support layer (LSL.COM) must be loaded first, followed by a NIC driver (such as TOKEN.COM or NE2000.COM). Finally, the protocol stacks (IPX, TCP/IP, and so on) are loaded and bound to a NIC driver through the LSL.

Together, the ODI NIC driver, the LSL, and the IPX protocol stack provide the same functionality as IPX.COM. All references to "IPX.COM" in this article apply also to the ODI implementation of IPX.

In addition to interfacing with the NIC, IPX.COM performs several communication-related functions. For example, it manages the IPX sockets used with the workstation. The shell and other applications access IPX.COM to open and close IPX sockets. When the workstation receives an IPX packet, IPX.COM checks which socket the packet is ad-

ressed to and passes the packet to the program having that socket open.

IPX.COM is also responsible for determining the address of the network segment to which the workstation is physically connected. The workstation's network number, along with its node address, make up the workstation's full internetwork address.

IPX can determine the workstation's network number in one of two ways. In the first method, IPX.COM watches for any routing information protocol (RIP) broadcasts sent on the network. Because RIP packets are not forwarded to other network segments, IPX knows that this type of broadcast originated on the segment to which the workstation is directly connected. To determine the workstation's network number, IPX simply reads the source network address contained in the IPX header of a RIP broadcast.



*The shell (NETx.COM)  
acts as the interface  
between user applications  
and NetWare file servers.*

IPX uses an alternate method if the shell requests a route to a network number before IPX can determine the workstation's network number from a RIP broadcast. In this case, IPX broadcasts a Get Local Target packet, which requests the fastest route to the destination network number requested by the shell. Upon receiving a response, IPX.COM checks the source network number in the IPX header of the response packet. This source network number

(the network number of the router that responded to the Get Local Target request) is the workstation's network number.

### **The NetWare Shell**

The shell (NETx.COM) acts as the interface between user applications and NetWare file servers. As user applications make requests, the shell determines whether the requests should be handled locally by DOS or sent to a server on the network. If the shell determines that the request should be sent to a network server, the shell formulates a request packet, hands it to IPX.COM for transmission, and waits for a response.

Prior to submitting any requests to a server, the shell must establish a connection with that server. The shell can establish a connection to a server in two ways. When the shell is first loaded at the workstation, it logically attaches to the first server that responds (usually the server nearest to the workstation). The LOGIN and ATTACH command line utilities, when executed, also establish a server connection.

To establish a connection, the workstation and the server must exchange several packets. One of these packets requests that a connection number be assigned to the shell. Another proposes the maximum packet size that will be allowed in the interaction between the file server and the shell. Before sending these initial packets, the shell needs the server's address and a route to that address.

### **Getting a Server's Address**

To acquire a server's address, the shell can use the service advertising protocol (SAP) to broadcast a request for the address of the nearest server – a Get Nearest Server re-

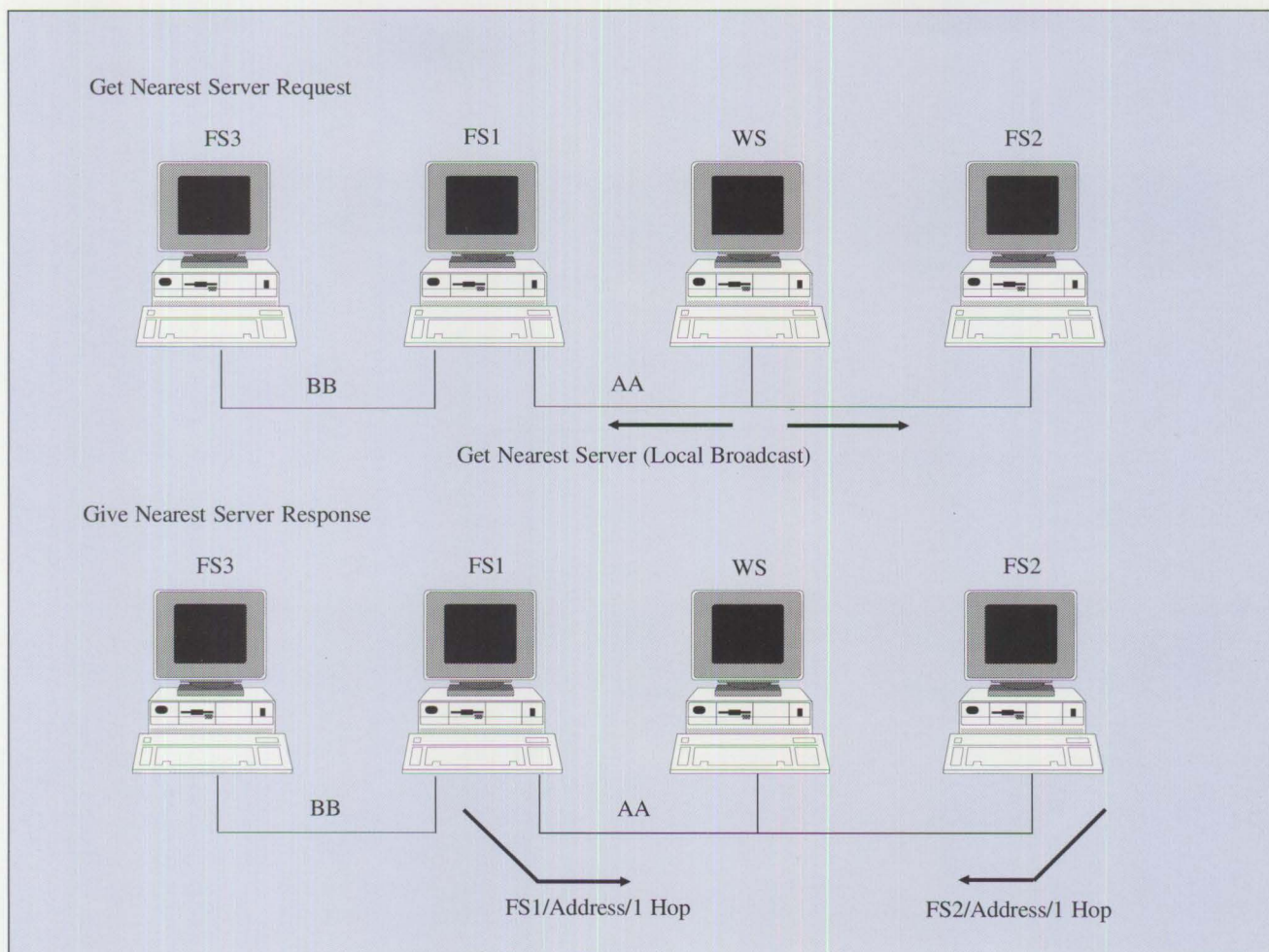


Figure 1. Getting the Address of the Nearest File Server

quest. All routers on the workstation's network segment having information about the nearest server respond to the Get Nearest Server request. Each response contains the nearest server's name, its full inter-network address, and the number of hops required to reach the server (Figure 1).

When first loaded at a workstation, the shell issues a Get Nearest Server request to establish an initial connection to a file server. If the shell loses its connections with all file servers, it resorts to the Get Nearest Server request method to re-establish a server connection.

A second method the shell uses to get a server's address is to use the NetWare core protocol (NCP) to access a file server's bindery. The bindery is a database within NetWare file servers that contains information about many network entities, including users, groups, and servers.

Because the shell must already have a server connection before it can access the server's bindery, the shell can use this method only after it has established an initial connection to a file server. The LOGIN and ATTACH utilities use this method, as does the new "preferred server"

shell (version 3.01 or later). These programs allow the user to specify a specific file server name, then use that name to scan the bindery for the server's address.

### Getting a Route to a Server

Once the shell has the server's address, it needs a route to get to that address. The shell uses this route for all subsequent communications with the server for the duration of the connection.

The shell submits a Get Local Target request to IPX.COM to obtain a route. IPX first compares the net-

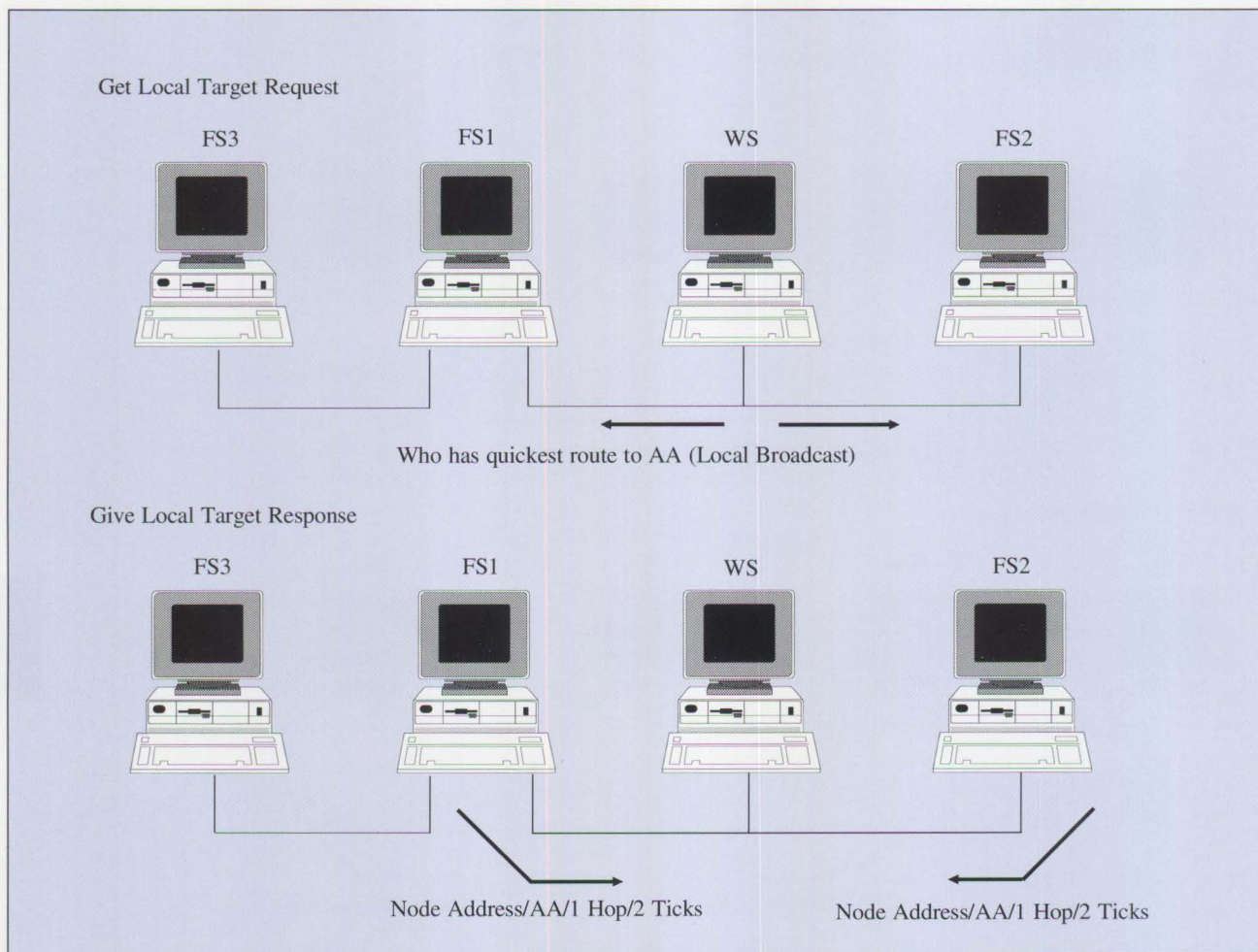


Figure 2. Requesting the Fastest Route to an Address

work number of the desired server to the workstation's network number. If these two numbers are the same, IPX tells the shell to send requests directly to the server (without going through an intermediate router).

If the network number submitted by the shell and the workstation's network number are not the same, IPX broadcasts a RIP request for the fastest route to the submitted network number. Whichever router on the workstation's network segment has the shortest route to the network responds to the request. More than one router might respond if several rout-

ers have a route equal to the shortest route. IPX accepts only the first router's response and discards all others.

IPX then returns to the shell the node address of the first router that responded. The shell places the node address of this router in the MAC header of a Create Connection request packet. It addresses the IPX header of the request packet to the file server to which it wants to connect. With the packet addressed in this fashion, the router receives the packet first, checks the IPX destination address, and forwards the packet to the network number on

which the file server resides (Figure 2).

### Establishing an Initial Connection

To establish its connection to a file server, the shell uses a combination of the SAP, the RIP, and the NCP. The sequence followed is slightly different for the preferred server shell than it is for previous shell versions.

Figure 3 shows the steps taken by shells prior to version 3.01 to connect with a file server. The first column represents the call or packet

sent. The second column lists the source, or sender, of the packet. The third column lists the addressee of the packet. The final column indicates the protocol used for handling the packet.

We have already seen how the first four steps work. In steps 1 and 2, the shell obtains the address of the nearest server. Step 3 is IPX.COM's request for the fastest route to the address that the shell received in step 2. Step 4 is the response by all routers with the shortest route to that segment.

Steps 5 through 8 show the packets exchanged between the shell and the server to establish the initial connection. Once this connection is made, the shell moves to the background (terminates-and-stays-resident) and returns the DOS prompt to the user. The user can then execute LOGIN.EXE to log in to the connected server or to another server.

### The Preferred Server Shell

The preferred server shell (version 3.01 and above) features several additional functions not offered by older versions of the shell. As its name implies, the preferred server shell allows users to specify, either at the command line, or in the SHELL.CFG or NET.CFG file, the server to which they would like to connect. Whether or not a preferred server is specified, the preferred server shell goes through the same initial eight steps as the old shells.

During the initial eight steps, the shell might connect to a server other than the preferred server. If so, the shell performs additional steps to establish connection with the preferred server (Figure 4).

Call	Source	Destination	Protocol
1. Get Nearest Server	Client	Broadcast	SAP
2. Give Nearest Server	Router	Client	SAP
3. Get Local Target	Client	Broadcast	RIP
4. Give Local Target	Router	Client	RIP
5. Create Connection	Client	File Server	NCP
6. Request Processed; Connection # Assigned	File Server	Client	NCP
7. Propose Packet Size	Client	File Server	NCP
8. Return Maximum Packet Size	File Server	Client	NCP

Figure 3. Initial Connection Sequence for the NetWare Shell (Prior to Version 3.01)

For example, if the workstation in Figure 1 initially connects to FS1 and the user specified FS3 as the preferred server, the shell will follow a sequence similar to that shown in Figure 4. In step 9, the preferred server shell uses the bindery method of acquiring the server's address.

Steps 11 and 12 of this preferred server sequence are not always re-

quired. If the preferred server resides on the same network segment as the workstation, the shell skips these two steps and sends the Create Connection request directly to the server. The shell destroys the connection with the initial server once it has successfully connected to the preferred server.

Call	Source	Destination	Protocol
1. Get Nearest Server	Client	Broadcast	SAP
2. Give Nearest Server	Router	Client	SAP
3. Get Local Target	Client	Broadcast	RIP
4. Give Local Target	Router	Client	RIP
5. Create Connection	Client	FS1	NCP
6. Request Processed; Connection # Assigned	FS1	Client	NCP
7. Propose Packet Size	Client	FS1	NCP
8. Return Maximum Packet Size	FS1	Client	NCP
9. Query Bindery for Preferred Server	Client	FS1	NCP
10. Address of Preferred Server	FS1	Client	NCP
11. Get Local Target	Client	Broadcast	RIP
12. Give Local Target	FS1's Router	Client	RIP
13. Create Connection	Client	FS3	NCP
14. Request Processed; Connection # Assigned	FS3	Client	NCP
15. Propose Packet Size	Client	FS3	NCP
16. Return Maximum Packet Size	FS3	Client	NCP
17. Destroy Service Connection	Client	FS1	NCP

Figure 4. Connection Sequence for the Preferred Server Shell

Call	Source	Destination	Protocol
1. Query Bindery for Preferred Server	Client	FS1	NCP
2. Address of Preferred Server	FS1	Client	NCP
3. Get Local Target	Client	Broadcast	RIP
4. Give Local Target	FS1's Router	Client	RIP
5. Create Connection	Client	FS3	NCP
6. Request Processed; Connection # Assigned	FS3	Client	NCP
7. Propose Packet Size	Client	FS3	NCP
8. Return Maximum Packet Size	FS3	Client	NCP
9. Destroy Service Connection	Client	FS1	NCP

Figure 5. Additional Steps Performed by LOGIN.EXE

Another major difference between the old shell and the preferred server shell involves the receipt of Give Nearest Server responses. Older shells accept the first Give Nearest Server response they receive and ignore all subsequent responses. Preferred server shells also accept the first response, but save the next four Give Nearest Server responses in case a connection cannot be made to the first server.

Servers respond to Get Nearest Server requests even if they have no free connections. Consequently, older shells fail to establish a connection (steps 5 and 6 of Figure 3) if the first Give Nearest Server response received is from a server with no free connections. The preferred server shells, on the other hand, can refer to the next saved Give Nearest Serv-

er response if the current attempt to establish a connection fails.

### LOGIN.EXE

Users can run LOGIN.EXE any time after the shell has connected to a NetWare file server. LOGIN submits the user's name and password to the file server for verification. It also establishes a new server connection if the user specifies a file server name in the LOGIN command.

If the server specified at the command line is not the one the shell is already connected to, LOGIN follows the steps outlined in Figure 5. Once these steps are completed, LOGIN verifies the username and password. If the server specified at the command line is located on the same segment as the workstation, steps 3 and 4 are not necessary.

Server's Name	Full Address	Intermediate Router's Node Address	Packet Sequence Number	Connection Number	Receive Time Out	Maxmum Time Out

Figure 6. Portion of the Shell's Connection Table

ATTACH.EXE uses the same sequence as that described for LOGIN.EXE when establishing connections to a file server.

### Connection Management

Communication between any two workstations requires a certain amount of responsibility on both sides to ensure that no information is lost. In the NetWare environment, NICs maintain error checking at the bit level. File servers and workstation shells handle at the packet and session level.

It is a common misconception that SPX is used for packet-level error checking between workstations and servers; however, SPX is used only for peer-to-peer interaction. It is actually the NCP that governs the way connection control information is exchanged. Every NCP packet a client submits to a NetWare file server must contain a connection number and a sequence number. The connection number assigned to the client when the file server connection is established. The sequence number identifies each packet so that both the server and the shell can tell if a packet gets lost.

### The Shell's Connection Table

NETx.COM (the shell) can support up to eight server connections concurrently. NETx.COM maintains a connection table to track these connections (Figure 6).

Within each entry in this table, the shell stores the name and full inter-network address of the server to which it is connected. If the shell is forwarding packets through an intermediate router to the server, the node address of that router will be stored in the Router's Node Address field. The shell's connection number and packet sequence number are

Connection #	Network Address	Node Address	Socket Number	Sequence Number	Watchdog Count	Watchdog Timer	NIC Number	Intermediate Router's Address

Figure 7. A Portion of the NetWare File Server's Connection Table

also kept in the table. The sequence number is set to zero when the connection is first established and incremented with each new request sent.

The shell's connection table also maintains two timeouts. Max Time Out is the maximum time the shell will wait for a response from the server before resending a request packet. This timeout is based on an estimate of the time (in ticks) needed to deliver a packet to the server. The router provides this time estimate in its Give Immediate Address response. (If the workstation and the server are on the same segment, this value is set to one tick.) To set its maximum timeout for communications with a server, the shell multiplies this estimate by 16 and adds 10 ticks to the result.

The Receive Timeout is a dynamic timeout originally set to four times the time estimate received in the Give Local Target response plus 10 ticks.

$$\text{Receive Timeout} = 4t + 10$$

Once initially set, the receive timeout adjusts up or down to adapt to changing network conditions. If the shell issues a request to a server and does not receive a response within its current receive timeout, the receive timeout is increased by one and a half times its original value.

$$\text{New Receive Timeout} = 1.5 (\text{current Receive Timeout})$$

It remains at this new value for all subsequent retries of the current request and on first try of the next request. If the next request requires a retry, the receive timeout is increased again. The shell continues to increase the receive timeout until it reaches the maximum timeout.

Each time the shell does not have to issue a retry to a request, it decreases the receive timeout. To determine how much to decrease the receive timeout, the shell takes the time to receive a response to the last request (that didn't require a retry), multiplies that value by two, and adds 10 ticks. The shell sets the new receive timeout to the average of this calculated value and the current receive timeout.

$$\text{New Receive Timeout} = [2(\text{Time to receive response}) + 10 + \text{Current Receive Timeout}] / 2$$

The number of times the shell will resend a request to a server is determined by the IPX Retry Count. If this count is exceeded, the shell gives up and presents the user with a "Network error on server xxxxx. Error xxxxx from network. Abort, Retry?" message. All drivers have a default for this retry count. This default differs from driver to driver but is generally between 20 and 40. The SHELL.CFG file's "IPX RETRY COUNT = xx" option allows the default IPX retry count to be modified; however, some drivers ignore this en-

try in the configuration file and leave the retry count at their default.

### The File Server Connection Table

The file server connection table shown in part in Figure 7 allows the server to keep information about each of the clients it is servicing. The server uses the address fields for addressing response packets and for security purposes. When a packet arrives with a service request, it contains the connection number assigned to the sender. The server matches the packet's IPX source address (network, node, socket) with the address registered for that connection number. If the addresses don't match, the server regards the request as a security breach.

The NIC Number and Intermediate Router's Address fields are used for sending responses to clients. As a request packet is received, the server places the number of the NIC the request came in on in the NIC Number field. This number would be A, B, C, or D for NetWare version 2.15c and earlier, or the NIC's network number for NetWare versions 3.0 and higher.

If the packet was forwarded through one or more routers, the server stores the node address of the last router in the Intermediate Router's Address field. This saves the server from having to find a route to the client when it is ready to send a response. The server gets the node

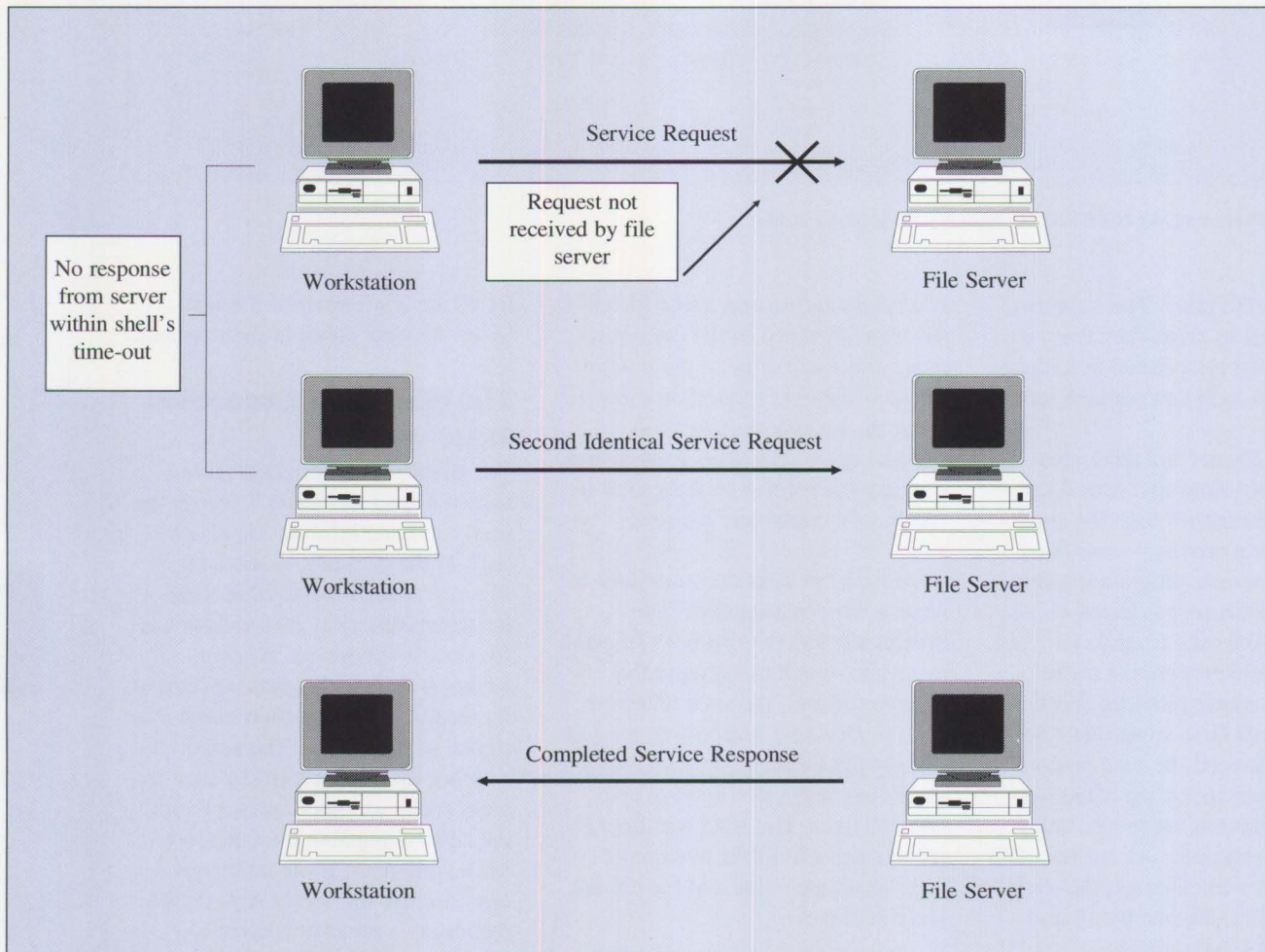


Figure 8. Request Lost in Transit to File Server

address of the first router from the Intermediate Router's Address field and places it in the MAC header destination address field. It then sends the packet through the NIC specified in the NIC Number field. Of course, it first places both the client's and its own full network address in the destination and source address fields of the IPX header, respectively.

The Sequence Number field is used for packet-level error checking. The Watchdog Count and Timer fields are used by the watchdog process, which is discussed later.

File servers also maintain a 100-byte reply buffer for each of their connections. If a response to a client is less than 100 bytes, the server places a copy of the response in the buffer corresponding to that connection. That way, if the client does not receive the response and resends the request, the server will not have to reprocess it.

### Packet-Level Error Checking

The bit-level error checking provided by network adapters detects the corruption of individual bits within a packet. When an adapter finds that part of a received packet

is corrupted, it discards the entire packet. Because of the driver's simple design, it contains no mechanism to request that the packet be resent or to inform the upper-layer processes and applications that an error occurred. Therefore, the upper-level sending process (the shell or file server) must determine when a packet has not reached its intended destination.

In the NetWare environment, this packet-level error checking is the responsibility of the shell. The NCP permits a workstation shell to submit only one server request at a time. Furthermore, the server's response



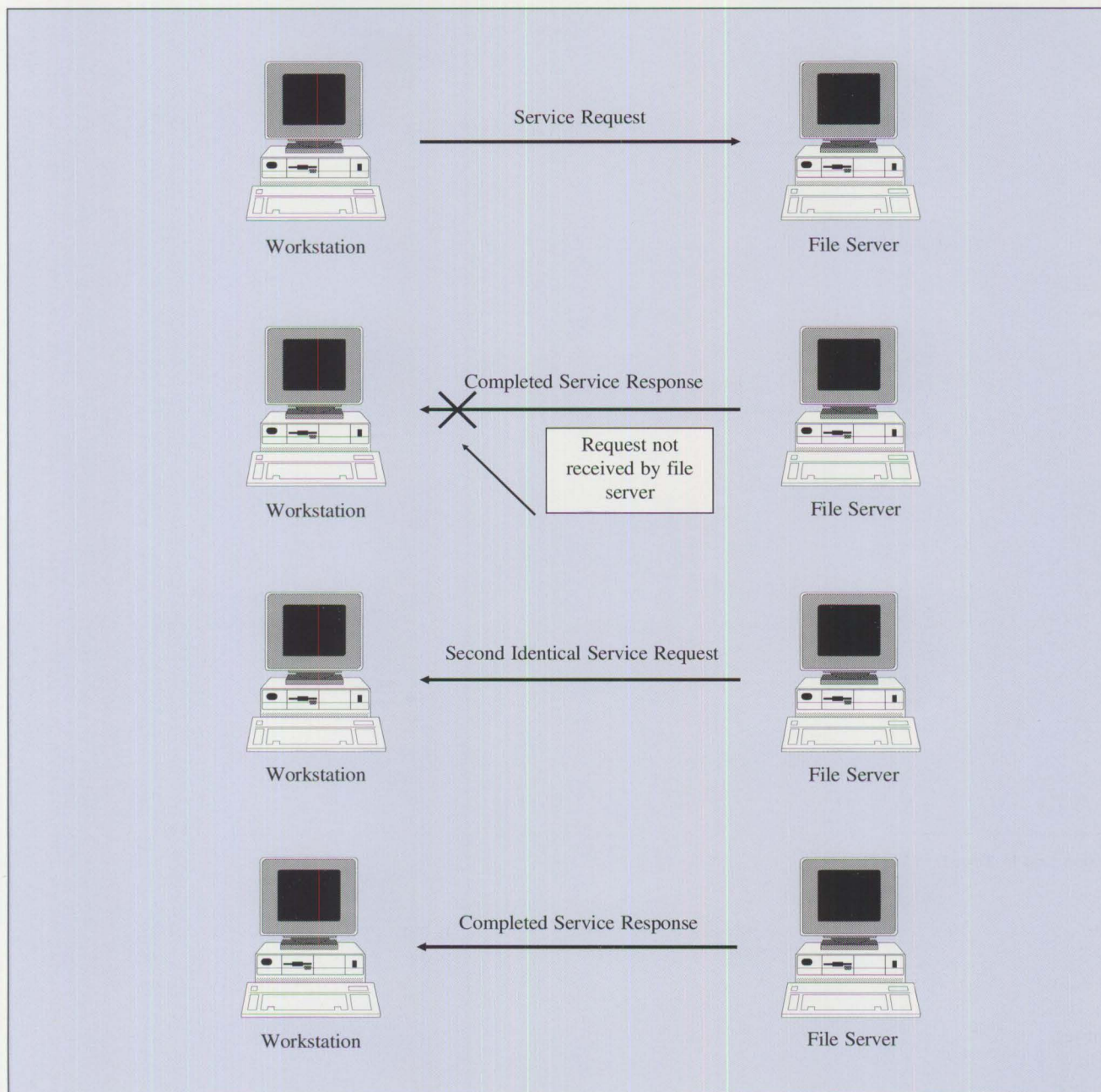


Figure 9. Response Lost in Transit to Shell

must fit in a single packet – the shell should never request more than a packet's worth of information. To guarantee that no packets have been lost, the shell has only to make sure it receives a completed response to each of its requests.

Each workstation request contains a sequence number within the NCP header. The server's response is labeled with the same sequence number. Ultimately, the shell is responsible for getting a completed response for each service requests it submits. If the shell does not receive a re-

sponse to its most recent request within the specified receive timeout. The shell continues to resubmit the request until it either receives a response or exceeds its IPX Retry count.

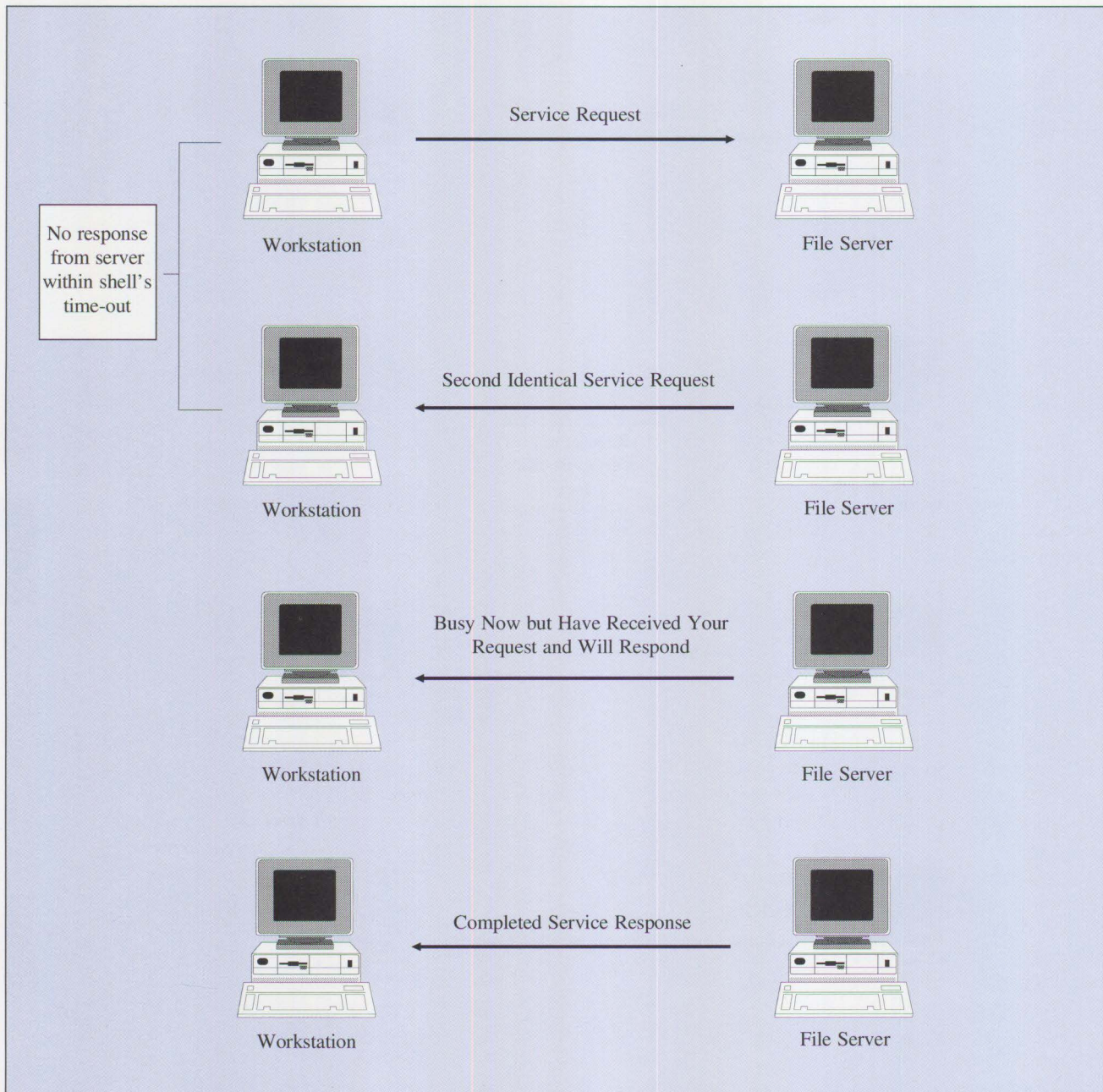


Figure 10. File Server is Busy

Three conditions can cause the shell to timeout while waiting for a response from a server. Figure 8 illustrates a case in which the request is lost in transit to the server. The shell's timeout eventually expires, upon which the shell resends the same request. The server receives

the duplicate request, processes it, and sends back the response.

In Figure 9, the request is received by the server, but the response is lost in transit to the workstation. Once the workstation's timeout reaches its limit, the shell sends a

second, identical request to the server.

When a server receives any request, it checks the request's sequence number to see that it is one greater than the sequence number registered in the server's connection table. If it

is, the server increments that number and processes the request as usual. However, if the two sequence numbers are the same, the server determines that the client, for whatever reason, is resubmitting its last request.

In some cases, the server might have a copy of the last response in its 100-byte response buffer for that connection. When a server sends a response less than 100 bytes long, it places a copy in that client's buffer – that is, the buffer corresponding to that client's connection number. Since a large percentage of responses are smaller than 100 bytes, there is a good chance that a server will have a copy of the response when requests are resubmitted by clients. (This type of response increments the Duplicate Replies Sent counter on the the FCONSOLE Statistics-LAN I/O Statistics screen.)

For requests larger than 100 bytes, the server must reprocess the request before sending the response. (This type of response increments the Re-executed Requests counter in FCONSOLE.)

The response might still be in transit to the workstation when the shell times out and resubmits the request (that is, the shell receives the completed response after resending the request). In this case, the server will send another response, but the shell will ignore it.

Sometimes the server is too busy to respond within the shell's timeout period (Figure 10). As usual, the shell resends the request. When the server receives this second request, it sends a reply stating that the initial request was received successfully, but the processing is not yet completed. (This intermediate response increments the Positive Ac-

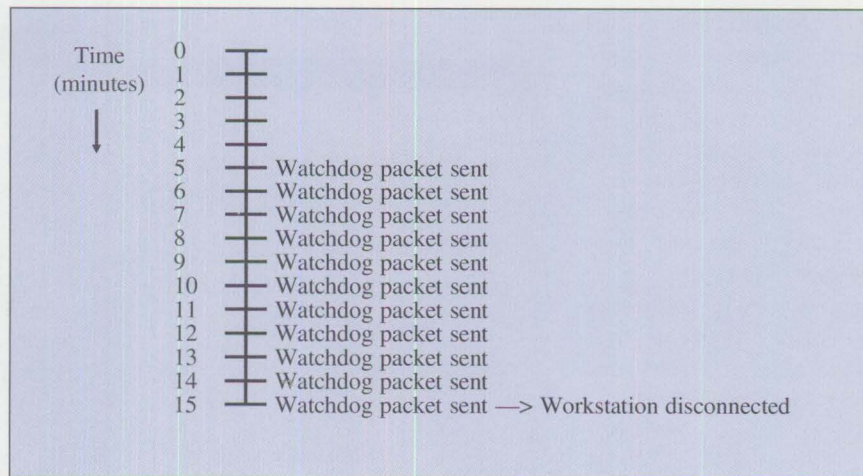


Figure 11. Watchdog Timetable for a Connection that Does Not Respond

knowledgments Sent counter within the FCONSOLE Statistics-LAN I/O Statistics screen.)

When the shell receives this reply from the server, it resets its timeout to zero and waits for the request. If the shell's timeout expires again, it sends a third copy of the request just in case the response was lost in transit from the server. This process continues until the shell finally receives a completed response.

### Connection-Level Error Checking

The connection between a workstation and server can be lost due to a power failure or a communications problem. Both the file server and the workstation shell is equipped to handle this eventuality. At the workstation, the connection is checked each time a request is made. If the shell does not receive a response to a request after it retries a certain number of times (as dictated by the IPX Retry Count), the shell assumes a connection problem and displays a message for the user. At this point, the user can either order the shell to resubmit the request or abort the operation completely.

If the operation is aborted, the shell removes that connection from its Connection table. If it has no other server connections, the shell attempts to create a new connection with a server (using the initial connection sequence outlined in Figure 3). If this attempt fails, the shell informs the user with the following message: *You are not connected to any file servers. The shell will try to connect to a file server whenever the current default drive is changed to an invalid drive. <Current drive is no longer valid>.*


At a NetWare file server, connection-level error checking occurs via address verification and periodic watchdog polling. When a file server receives a request packet for a certain connection, it verifies that the IPX source address within the request packet matches the address recorded for that connection in its connection table. If the addresses do not match, the server sends a response indicating that the connection number and address do not match.

### The Watchdog Process

When a workstation is turned off, regardless of whether the user issued

the LOGOUT command first, the station's connection remains occupied at the server. To clear these unused connections, the server uses the watchdog process to poll any clients the server hasn't heard from in the last five minutes. The server tracks the time for each connection in the Watchdog Timer field of the server's connection table. If the shell being polled is still operational, it responds to the watchdog query, and the server resets its timer for that connection.

However, if the workstation has been turned off, or if there is a communications problem on the network, the server will receive no response from the shell. The watchdog process then resets the connection's Watchdog Timer field to zero, but increments the Watchdog Counter field by one. The watchdog process sends another query to the workstation a minute later. As long as the watchdog continues to hear nothing from the workstation, it sends a packet every minute until it has sent a total of 11 polling packets to that workstation. The server clears the workstation's connection if it receives no response to the last watchdog packet.



*NetWare version 3.x-based servers provide a setable parameter that allows administrators to monitor when workstations are logged out by the watchdog process.*

Figure 11 illustrates the timetable for a connection that does not answer a server's queries. NetWare version 3.x-based servers provide a setable parameter that allows administrators to monitor when workstations are logged out by the watchdog process. This option is set with the following command: SET CONSOLE DISPLAY WATCHDOG LOGOUTS = ON.

## Conclusions

A series of protocols governs NetWare's client-server communications. These protocols can be broken up by functionality: protocols used for all communications (the medium access protocols and IPX), administrative protocols (the RIP and SAP), protocols concerned with connection control (the NCP and Watchdog), and, finally, the protocol that defines the coding of service requests (the NCP).

This article has explained the operation and interoperation of these protocols; however, it makes no attempt to apply this information to all possible network configurations and environments. It is up to network administrators to apply this information to their specific networks.

## ABOUT THE AUTHOR

*Paul Turner is a consultant with the systems research department of Novell's Systems Engineering Division.*

*Novell Inc.  
P.O. Box 5900  
Provo, Utah 84606*

# LANACS Protocols

*Mike Colucci  
IBM Corporation  
Dallas, Texas*

**Local Area Network Asynchronous Connection Server (LANACS) is a communications gateway. It makes available to LAN users a modem pool of up to 32 asynchronous lines. This is a description of the LANACS enveloping scheme in the token ring environment. Understanding the frame formats and their enveloping scheme facilitates problem determination when using traces.**

In explaining the reasons why LANACS cannot communicate with other NetBIOS software, I normally use the following analogy:

I have a PS/2 in Dallas on a LAN with a LANACS gateway. The LANACS gateway can dial a mainframe in Atlanta. If the mainframe is in a soft drink plant, and I have a spigot on my PS/2, why can't I get the soft drink to flow from the plant to my spigot using LANACS as a gateway?

The answer is that LANACS does not understand the interface called Soft Drink Transfer Interface (SDTI). It does understand ACSI and EBI. With the addition of OS/2 EE ACDI redirection, it understands ACDI because ACDI is essentially the equivalent of EBI. Also, LANACS Version 2 includes TCP/IP support, so it understands the TELNET protocol of TCP/IP. So, LANACS is capable of speaking EBI, ACDI, ACSI, TCP/IP, and Async – but not SDTI.

Let's call the languages high-level frame formats. These high-level frame formats are enveloped in the lower-level frame formats, which are:

- Media Access Control (MAC)
- Logical Link Control (LLC)
- NetBIOS

## Frame Formats from MAC Layer to LANACS

The MAC-level frame is the lowest-level frame. It is what you see if you trace a token-ring. You might see a MAC-level abort frame, a MAC level frame used for LAN management, or a data frame that carries user data in an LLC frame. A MAC-level frame might also be a free-token frame, but you won't see it in a Trace and Performance (TAP) tool trace.

The LLC frame is the next higher level frame. In the data portion of the LLC frame, you might have NetBIOS, SNA, TCP/IP, and so forth.

So from bottom to top, the MAC-level frame encapsulates the LLC frame. The LLC frame encapsulates the NetBIOS frame. The NetBIOS frame encapsulates the data. The data comes from LANACS and can consist of actual ASCII data. But it can also consist of non-ASCII data that LANACS can interpret. These are the interfaces mentioned in my soft drink analogy.

## MAC Frame

As can be seen at the the bottom of Figure 1, there are several types of MAC frames. They all start with a bit pattern called a starting delimiter so the hardware can recognize the beginning of legitimate data. They all have an ending delimiter which is another particular bit pattern. That's all there is to the abort frame. The free-token frame has an access control

byte preceding the ending delimiter. And the MAC and data frames have a frame status byte following the ending delimiter.

The MAC and data frames are similar. Between the delimiters they have bytes for control, addressing, and data checking. The difference is that the MAC frame has a field for MAC-level commands and information and the data frame does not. Instead it has two fields. One is routing information and the other is called the LLC Protocol Data Unit (LPDU).

The important point here is that all the other fields surrounding the LPDU are the envelope provided by MAC-level software. That software, in this case, is on a token ring card. The protocol is named IEEE 802.5 by the IEEE standards organization. The LPDU itself is the LLC frame encapsulated in MAC information.

## LLC Frame

This frame contains the Service Access Point (SAP).

When an application starts, it opens an SAP, which might be considered a type of address. When an application connects to another system it provides both its own SAP (source SAP) and the SAP of its partner (destination SAP). The communicating systems must use SAPs known to each other to communicate. For example, the NetBIOS SAP is F0. An SNA SAP is 04. Either one can ride in an LLC frame. A destination has to have SAP 04 open or the source can't send it any 04 SAP information. It can't talk to NetBIOS SAP.

The control field in the LLC frame contains the LLC protocol, which consists of the commands and responses. The rest of the frame is the

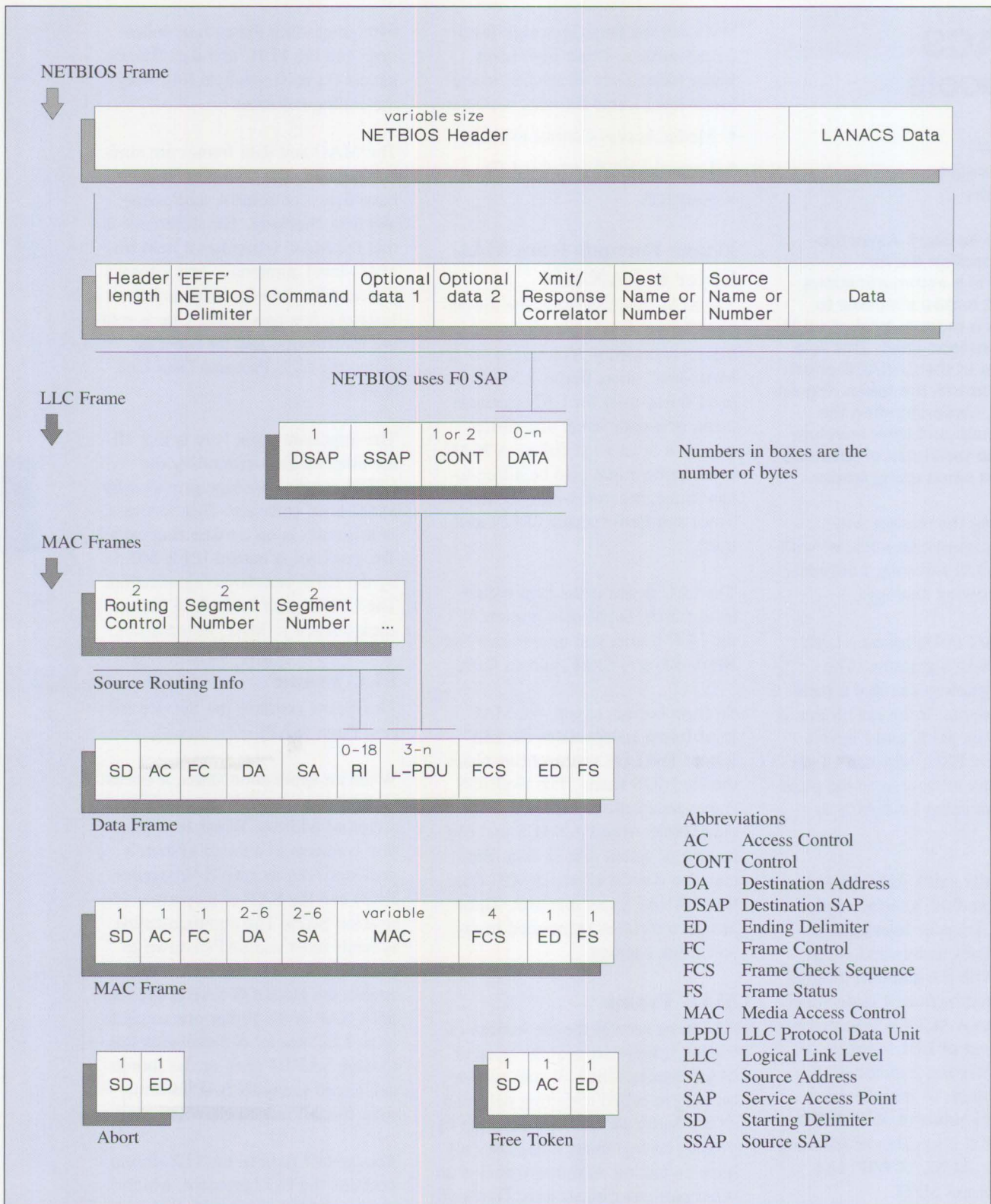


Figure 1. IBM Token-Ring Protocols Used with LANACS (1 of 2)

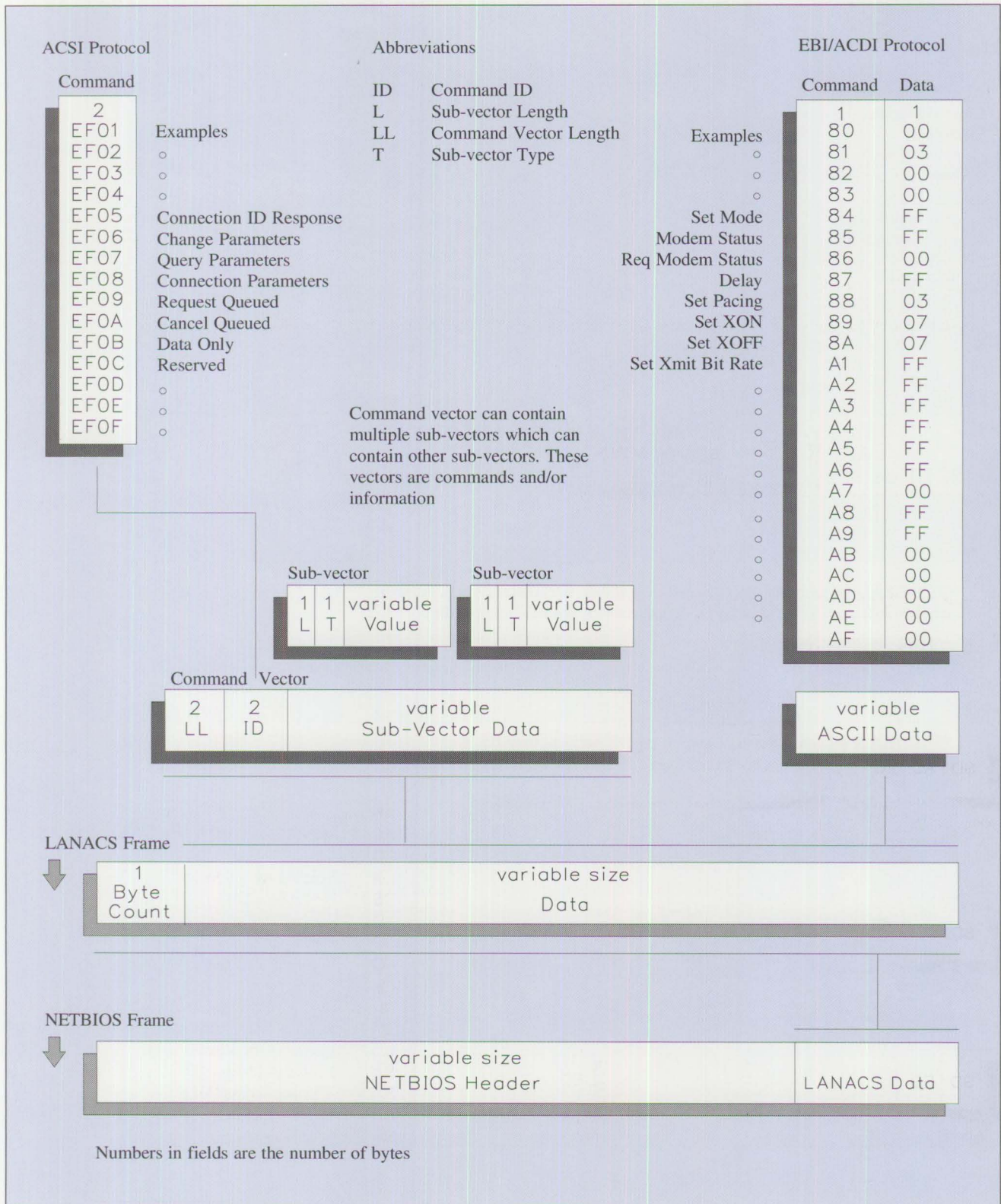


Figure 2. IBM Token-Ring Protocols Used with LANACS (2 of 2)

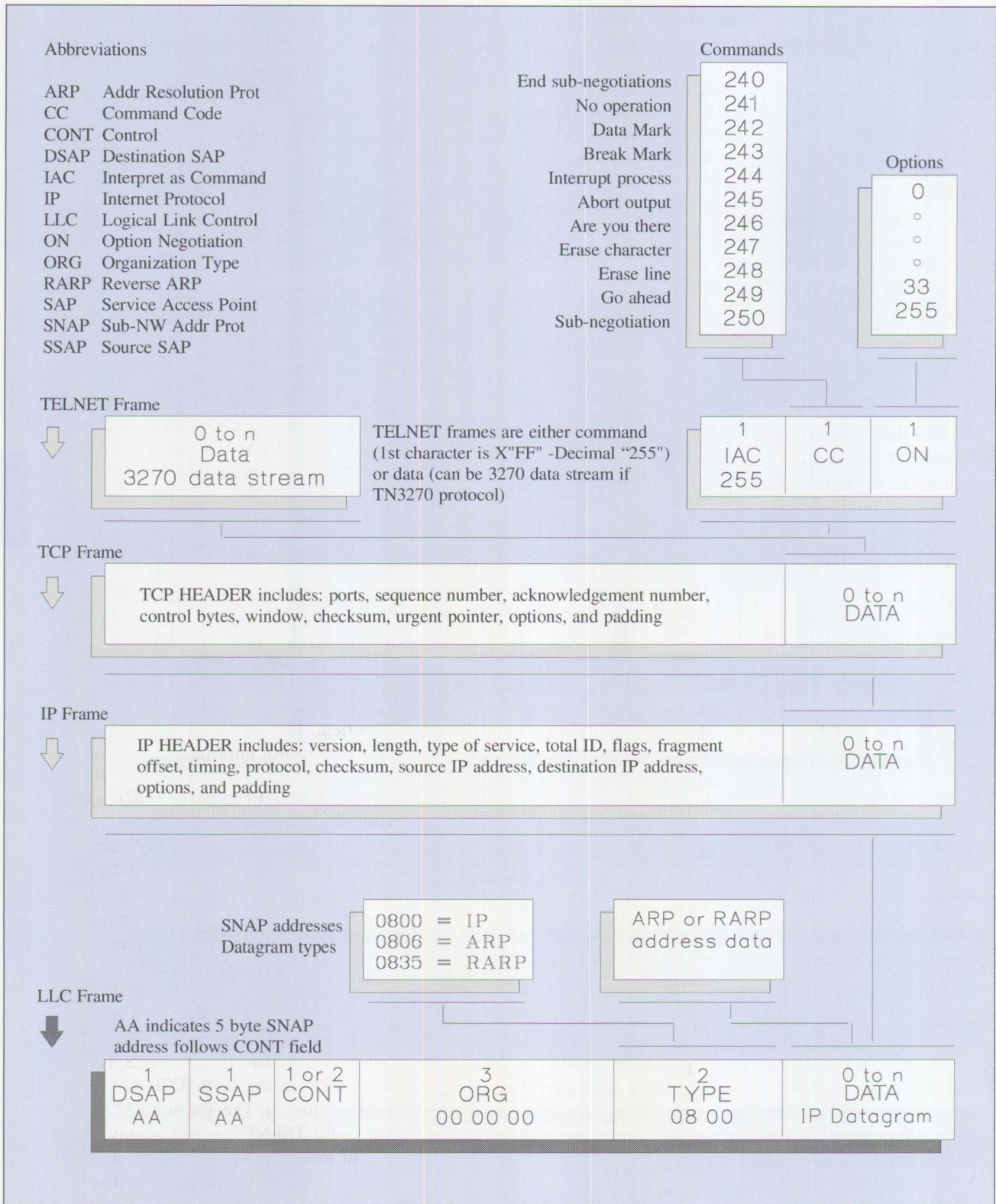


Figure 3. TCP/IP TELNET Protocol Used with LANACS



data. This data field is the NetBIOS frame encapsulated in LLC information. The LLC protocol is named IEEE 802.2 and is also software that is found on the token ring card.

### NetBIOS Frame

This frame starts with a field that defines the length of the header. Next is a 2-byte specific bit pattern that is the delimiter for the frame. The NetBIOS command follows. The command gives and takes all the information it needs from a Network Control Block (NCB). The NCB varies depending on the command within it. The NetBIOS frame also contains source and destination information, just as the LLC and MAC frames. As each level is stripped off, the remainder has to have an address to go to until it reaches the application. Finally, following all this information in the NetBIOS header, is the data field. This data field is the LANACS data.

Just as a reminder, NetBIOS stands for Network Basic Input/Output System. The NetBIOS frame encapsulates the LANACS data frame. NetBIOS software is found in LAN Support Program. It is also part of OS/2 EE. It doesn't have a name given by IEEE or any other organization because it is an industry defacto standard.

### LANACS Frame

As can be seen from Figure 2, LANACS frames come in different forms, commonly called protocols. Actually, some are programming interfaces. (You'll find, although not accurate, the terms protocol and interface are often used interchangeably. Put simply, interfaces are commands. Protocols are the sequence in which the commands are used.) Figure 2 depicts both the

ACSI protocol and the EBI protocol. ACDI uses the EBI protocol.

**ACSI:** ACSI is a complicated language. It's called a protocol in its original documentation in an issue of *IBM Personal Computer Seminar Proceedings*, Volume 3, #4, October 1985 (G320-0323). Each command generates a series of NetBIOS commands. Sometimes, ACSI information or data is imbedded in a command, such as the data command **EFOB**.

**EBI:** EBI is a language written to PC interrupt X'14', which is the vector that points to the async adapter device driver. When EBI is installed, the vector is changed to point to a LAN device driver so COM ports can be redirected to the LAN. Extra commands are added to the device driver. EBI commands generate actions within the workstation or they generate NetBIOS packets that contain a command byte along with a data byte. Or it can generate a NetBIOS packet that contains ASCII data. This is unlike the ACSI data packet, in which a command vector precedes the ASCII data. The EBI NetBIOS packets (commands) are shown in Figure 2.

In-depth information on EBI is found in the *LANACS Installation and Configuration Guide*.

### TCP/IP Frames

Again, the protocol layers are encapsulated in each other, as shown in Figure 3.

**The LLC Frame:** The TCP/IP LLC frame is different than the NetBIOS LLC frame. The DSAP and SSAP fields contain hex 'AA' (decimal '170'). That indicates that following the control field is a five-byte SNAP address. The control field is normally followed by data. The 'AA'

also indicates that this is TCP/IP. The first three bytes of the SNAP address (the organization code field) have been assigned 0s (zeros) by the standards committee. The next two bytes (type field) indicate which protocol – IP, ARP, or RARP – is being used. If it is IP, the type field is hex '0800' (decimal '2048'). If it is ARP, the type field is hex '0806' (decimal '2054'). And RARP has a type field of hex '8035' (decimal '32821').

**The ARP Frame:** This frame also has a SNAP indicator of hex 'AAAA'. But the end of the frame contains actual TCP/IP addresses. In this case, the trace indicated a TCP/IP address of hex '09 13 8D 05' (9.19.141.05, the host address).

**The IP Frame:** This frame is encapsulated within the LLC frame as the data field. The IP header contains various information including addressing.

**The TCP Frame:** This frame is encapsulated within the IP frame as the data field. The TCP header contains port and control information.

**The TELNET Frame:** TELNET commands are identified by hex 'FF' (decimal '255') in the first data byte. They are used to negotiate the bind parameters between the two communicating entities. That is one type of TELNET frame. The other type of TELNET frame is data. It does not contain the hex 'FF' in the first byte. It can be straight data or 3270 datastream data. If TN3270, which is supported by LANACS, is used, the data will be in 3270 datastream format. The TN in TN3270 stands for TELNET, which is only one of many of a suite of protocols supported by TCP/IP. But remember that it is the only one supported by LANACS.

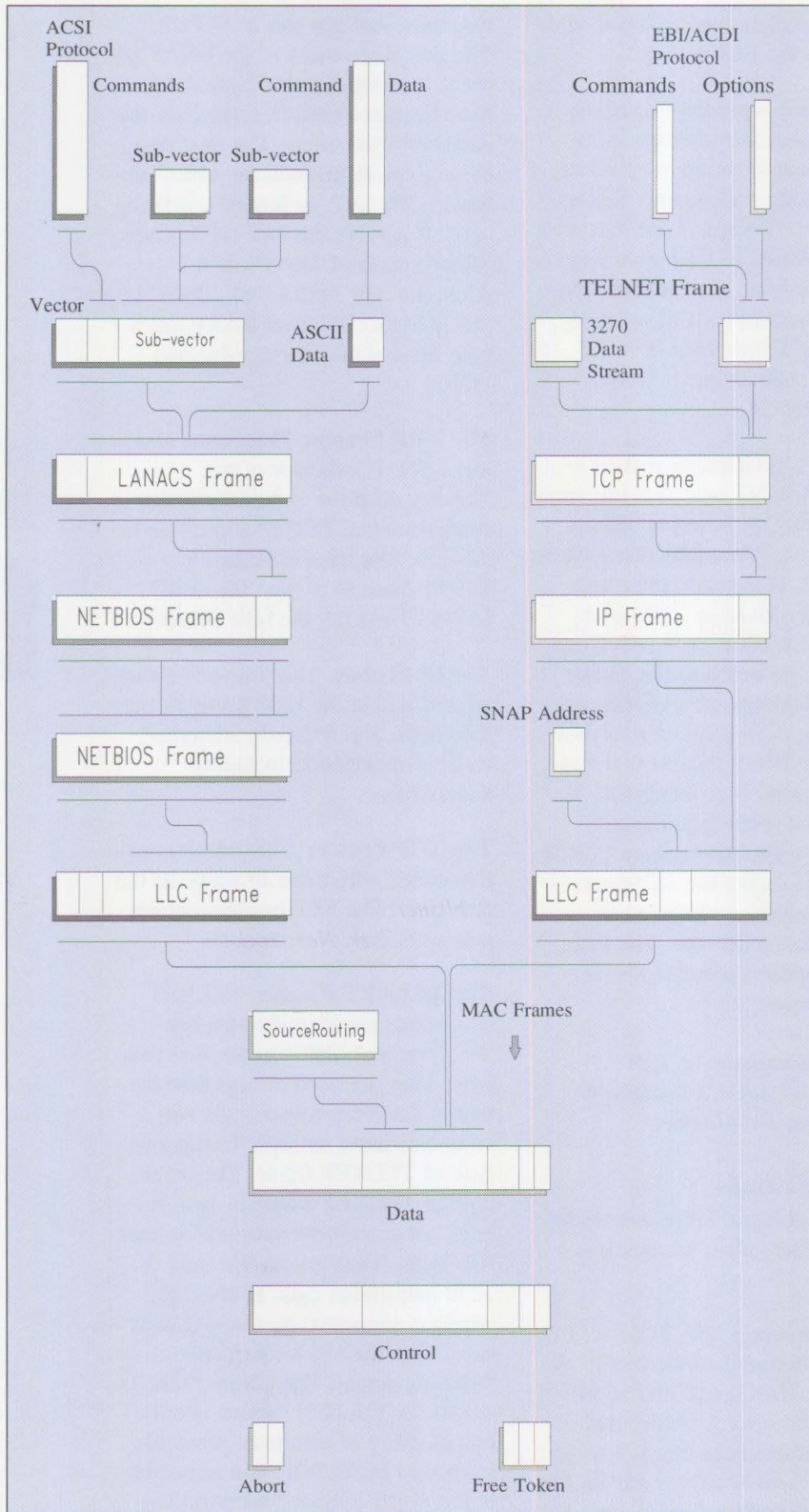


Figure 4. Local Area Network Protocols Used with LANACS

### Composite Frames

Finally, let's put them all together in Figure 4 to see how it flows.

(You will note that the SDTI is conspicuously absent.)

### ABOUT THE AUTHOR

Mike Colucci is an Advisory Market Support Representative in National Technical Support. He has been the focal point for LANACS support since its inception. He has also supported other asynchronous products such as Asynchronous Communications Server (ACS) and Remote NetBIOS Access Facility (RNAF). He has recently added support for the async ASCII portion of OS/2 EE Communications Manager to his responsibilities.

### Reading

For more information on LANACS, including traces, refer to *Local Area Network Connection Server (LANACS) Version 2 Simplifying the Complex* (GG66-3202).

For help in understanding frame formats, protocols, and NetBIOS commands, the following documentation is available:

*Token-Ring Network Architecture Reference* (SC30-3374)

*Local Area Network Technical Reference* (SC30-3383)

*TCP/IP Tutorial and Technical Overview* (GG24-3376)

*IBM Personal Computer Seminar Proceedings*, Vol 3, # 4, October 1985, "IBM Asynchronous Communications Server Protocol" (G320-0323)

## New Book: *Client-Server Programming with OS/2 Extended Edition*

This book on client-server computing was written by Robert Orfali and Dan Harkey of IBM. Through easy-to-read, in-depth tutorials and working code, the book shows how to integrate databases, LANs, and graphical front-end clients to build working client-server applications using OS/2 EE. A Club Med reservation system serves as a grand finale that brings together GUI clients, database servers, and LAN communications.

In addition to the tutorials, the book contains more than 300 pages of working code and benchmarks that demonstrate the many trade-offs in client-server design, such as:

- Processes versus threads
- APPC, NetBIOS, named pipes, or the LAN server
- Static SQL versus dynamic SQL
- Remote SQL versus stored procedures
- Transaction servers versus database servers
- GUI tools versus Presentation Manager programming

The software in this book provides a jumpstart for creating OS/2 client-server applications in record time. To create a robust client-server platform, the authors use OS/2 EE's advanced features, including:

- Referential integrity
- Remote Data Services
- Advanced peer-to-peer communications
- Threads
- Network and database administration utilities

The friendly style of this book appeals to a wide audience, ranging from novices to PC programmers and MIS professionals. It should also be of great interest to LAN and communications specialists, database administrators, and anyone interested in SNA, SAA, or OS/2 EE.

The book can also be used as a supplementary text for courses on networks, operating systems, OLTP, LANs, SQL, database theory, servers, and distributed systems. Finally, this book explains the power of OS/2 and how it fits with SAA and CUA.

*Authors: Robert Orfali and Dan Harkey*

*IBM Mechanicsburg order number: G325-0650*

*Description: 1000 pages, \$39.95*

*Publisher: Van Nostrand Reinhold, New York, New York*

### *IBM Personal Systems Developer*

The *IBM Personal Systems Developer* is a quarterly publication written primarily for OS/2 application programmers. It features a variety of technical articles, such as programming tips and techniques, product reviews of new software tools, application development case studies, and interviews with OS/2 industry leaders. IBM employees, customers, and software vendors write articles for the *Developer*. The magazine is published as part of IBM's Developer Assistance Program, which offers a variety of support services for companies that are writing OS/2 applications for resale.

Subscriptions can be ordered by calling 1 800 READ-OS2. IBM employees can subscribe through Mechanicsburg's Systems Library Subscription Service (SLSS), order number G362-0001.

## Little Solutions



*Here are hints and tips from our readers. If you have "little solutions" to share, send them to us in care of the editor.*

### Modifying Your LIBPATH Statement on the Fly

A real problem for some OS/2 users is the inability to dynamically modify the OS/2 library search path. While the PATH and DPATH statements can be modified easily, the LIBPATH statement is not part of the environment space and as such cannot be modified while OS/2 is executing.

While this remains a hard-and-fast rule, here's a trick you can use to gain a small measure of flexibility in your library path during execution.

```
LIBPATH=.;c:\library1;c:\library2;d:\library3;
```

Figure 1. Example LIBPATH Statement

Code your LIBPATH statement with .; as the first two characters on the parameter line. This causes OS/2 to search the current subdirectory first before searching the libraries defined in the rest of the library path. By manipulating the current directory, you can have some effect upon the library search path.

Figure 1 is an example of the LIBPATH statement with the current directory defined as the first directory to search. — *Dave Dill, IBM, Dallas*

### Keystroke Savers

Here are a few tips that will save many keystrokes when you're working in DOS.

The first tip is a little-known feature of DOS that allows you to eliminate the use of asterisks as wildcard parameters in conjunction with a DOS command (COPY, DEL, and so forth). DOS allows you to omit the asterisks so you can type:

```
DEL .
```

in place of

```
DEL *.*
```

For example, to copy all the files on your C drive root directory to a floppy disk you can type the command:

```
COPY C:\. A:
```

as opposed to

```
COPY C:\*.* A:
```

The next tip is a way to change from a subdirectory to a directory one level higher in your directory tree structure without specifying the directory name. For example, to change from the directory \BATCH\COMMANDS\UTILS to \BATCH\COMMANDS, you would normally type:

```
CD \BATCH\COMMANDS
```

To achieve the same result, you can simply type:

```
CD..
```

The last tip is a way to use batch files that will save you from typing commands that have replaceable parameters. For example, the program PKZIP (a widely used file compression utility), along with its companion program PKUNZIP allows you to compress many files into one

smaller file for easier portability between DOS machines. After the file is transferred to the destination machine, it can be decompressed back into its original form of many separate files. Let's assume you want to back up all the files with the extension .EXE in the directory \UTILS into one file called UTILS.ZIP. The command to do this (after making sure you are in the UTILS directory) is:

```
PKZIP -A UTILS.ZIP *.EXE
```

where **-A** tells it to add files to an archive file, UTILS.ZIP is the output archive filename, and **\*.EXE** is the range of files to include. You may then transfer the file to the destination machine for decompressing. For the purposes of this example, let's assume you put the file into the destination machine's C drive root directory and that you want the original decompressed files in a directory on the same drive called \TOOLS. The command to accomplish this is:

```
PKUNZIP -X UTIL.ZIP \TOOLS
```

where **-X** tells it to extract files from the archive file, UTIL.ZIP is the input archive filename, and \TOOLS is the output directory name.

Both of these commands are rather lengthy and cumbersome. Instead, simply create two simple batch files:

The first batch file is called ZIP.BAT and compresses the files into an archive file.

```
PKZIP -A %1 %2
```

```
ex. ZIP UTILS.ZIP *.EXE
```

The **%1** is a replaceable parameter whose value is determined by the first parameter after the name of the batch file ZIP; in the preceding example it is UTILS.ZIP. The **%2** is

also a replaceable parameter whose value is determined by the second parameter of the command. In our example it is **\*.EXE**.

The second batch file is called UNZIP.BAT and decompresses the archive file back into separate files with the option of specifying which directory to put the files into.

```
PKUNZIP -X %1 %2
```

```
ex. UNZIP UTILS.ZIP \TOOLS
```

The **%1** represents the input archive file name and **%2** represents the output directory name.

Make sure that these two batch files and the programs PKZIP.EXE and PKUNZIP.EXE are in directories that can be found in your DOS path (specified in your AUTOEXEC.BAT file).

The files shown here are examples of the use of the **%** parameter. You may use the **%** parameter in your own batch files that start programs that have parameters that need to be input.

These shortcuts can be real time-savers when used regularly.

— *Forrest York, Dallas*

## LAN Solutions

**Out of Resources?** Lan Server 1.3 will fail to start if resources requested in the IBMLAN.INI file are larger than those defined in the Communications Manager configuration file. Lan Server 1.30.1 will allow the server to start, although the server will have fewer resources than requested. An error message, NET3198, will be written to the error log to let you know all requested resources were not available when started. The error message does not

say there was an error, it is just an informational message.

### Changes to Home Directories:

Starting with LAN Server version 1.30.1, home directories will no longer be defined as an alias. Also the home directories can be located on any server at any location on the server.

Until version 1.30.1 of LAN Server (WR05015), when a home directory was created it was automatically put under the IBMLAN directory on the domain controller. Beginning with version 1.30.1, when the administrator is defining a home directory for a user, the system asks which server the home directory will be on and the path to the directory. If the path does not exist, the directory and path are created. This can also be done from the command line. Documentation for this is on the README file for CSD 5015.

When a home directory was created, an alias was automatically created and shared at user request with an access profile giving only that user access. With version 1.30.1, an alias is not created. The share will be a dynamic share just as you can do from the command line. There is a non-alias access profile created with only the user having access.

### Updating DLR from 1.2 to 1.3

**across the Network:** There is no predefined function (as with the CSD download) that allows you to download DLR 1.3 to DLR 1.2 machines. The options are to install the code on each diskette or to install the code on the server (because it is packed on the diskettes) and then copy it to an existing DLR workstation. You could implement this function by performing the following steps:

1. Install DLR 1.3 on one requester.
2. Access a shared driver at the OS/2 1.3 Server from the 1 DLR 1.3 requester.
3. Copy the DOSLAN subdirectory tree structure to the server from the DLR 1.3 machine.
4. Because the DLR 1.2 at 4098 machines can log on to the OS/2 1.3 domain, you can easily provide access to the files copied to the server in steps 2 and 3 to the DLR 1.2 at 4098 machines.
5. Copy the DOSLAN files from the server (that is, from steps 2 and 3) to a new subdirectory at the DLR 1.2 machines (that is, C:\DOSLAN13). Do not delete the original DLR 1.2 subdirectory until DLR 1.3 is working.
6. Edit the DLR 1.2 AUTOEXEC.BAT and change the PATH to C:\DOSLAN13 (or to wherever you copied the files).
7. Edit the C:\DOSLAN13\DOSLAN.INI file (after you have copied it) to reflect a unique machine name.
8. Reboot the DLR 1.2 machine. DLR 1.3 should then be available.

**DOS 5.00 and DLR:** If you want to run DOS 5.00 and DOS LAN Requester, there are different procedures depending on what you currently have installed.

DOS 3.3 and 4.01 users who have DLR installed can install the DOS 5.00 Upgrade and then install the fixes provided with the Upgrade on

the IBM Network diskette that is included. DLR must be installed *before* the upgrade. If you try to install DLR 1.3 after the DOS 5.00 Upgrade, it will fail, displaying a message that you are out of storage.

New DOS 5.00 users can install DOS 5.00, then install the DLR 1.30.1 diskettes that are now being shipped with OS/2 LAN Server 1.30.1.

New DOS 5.00 users who have DLR 1.30 and CSD 5015 can use the following procedure if the server is at CSD WR05015:

1. Install DOS 5.00.
2. Edit CONFIG.SYS and make FILES=30.
3. Install LAN Support Program
4. Reboot the machine.
5. On the first DLR 1.30 diskette, remove the "read only" from the file CIP\_MAIN.EXE (in the root directory).
6. Copy CIP\_MAIN.EXE from server \BMLAN\DOSLAN\NET to this diskette.
7. Make the file CIP\_MAIN.EXE "read only" on the diskette.
8. Install DLR 1.30 on the workstation.
9. Update base DLR files to WR05015 in the DOSLAN directory. This may be done with the IBM DOS 5.00 Upgrade, or the user can copy the following files from the server \BMLAN\DOSLAN\NET

to a diskette:

```
REDIR40.EXE
NET.COM
NETWORK.MSG
NETWORK1.CMD
NETWORK2.CMD
NETWORK3.CMD
```

Copy those same files to the DOSLAN directory on the workstation using the following commands:

```
attrib *.* -r
copy a:\*.* c:\doslan
attrib *.* +r
attrib doslan.ini -r
```

10. Start DLR, log on, and run INSTDLR.BAT (see CSD WR05015 READ.ME) to update the rest of the workstation DLR files.

**NET WHO:** If DLR users are dropped from NET WHO within one hour of logging on to the DLR, they need to add the /NMS:n (where **n** is one or more) in their DOSLAN.INI file. This uses a minimum of 2.7 K (4.4 K if HIMEM isn't specified). They must also install fix IC02772, which is available from the software support center.

**Sharing CDROM:** A CDROM can be successfully shared across the LAN using OS/2 LAN Server 1.2 or higher. The administrator must issue the following at the Server:

```
NET SHARE cdrom=d:\
```

Issue at the OS/2 requester:

```
NET USE D: \\servername\cdrom
```

The administrator needs to create a non-alias access control profile for the path to the CDROM drive and give **R** access.

— Monte Hall, IBM, Austin

# New Products

## Hardware

### IBM PS/2 Model 70 386 (081, 161, A81 and A16)

These new PS/2 system units address a wide range of customer needs and enhance the PS/2 family of systems with additional standard memory and enhanced fixed disks. All of these models have 4 MB of standard memory and a 32-bit microprocessor with the capability of supporting up to 16 MB of high-speed real memory.

The 80386 processor models 081 (80 million bytes/20MHz), 161 (160 million bytes/20MHz), A81 (80 million bytes/25MHz) and A16 (160 million bytes/25MHz) enhance ESDI files and support the addition of an 80387 optional Math Co-Processor, which offers significantly improved system performance for a desktop environment. The Models PS/2 70 386 8570-A81 and A16 can be upgraded to the 25MHz i486 processor by the installation of the 486/25 Power Platform.

These models are appropriate for advanced users who perform numeric-intensive applications requiring floating-point operations, large processing intensive applications or applications requiring high-performance fixed disks. These new models of the PS/2 70 386 family maintain compatibility with most existing software products for IBM personal computers and systems.

#### Highlights:

- 32-bit 80386 (20MHz-25MHz) and large capacity 80 MB or 160 MB fixed disks with integrated controller
- 4 MB memory on the system board and capacity of up to 8 MB high-speed memory on system board
- Micro Channel® architecture with one 16-bit and two 32-bit slot and optional 80387 Math Co-Processor

Letter # 191-079, June 11, 1991

### IBM PS/2 Model 55 SX and Memory Upgrade for IBM PS/2 Model 55 LS

New models of the PS/2 Model 55 SX (8555-041, 081) have additional standard memory, higher capacity and enhanced performance fixed disks.

The PS/2 55 SX new models have 4 MB standard memory, a 32-bit microprocessor with the capability of supporting up to 16 MB of system memory and a 40-million-byte or 80-million-byte fixed disk drive.

Additional functions of the PS/2 Model 55 SX include: 1.44 MB, 3.5-inch diskette drive; ports (keyboard, pointing device, serial/asynchronous, parallel, VGA); three I/O slots and keyboard.

Standard memory for the 32-bit microprocessor PS/2 Model 55 LS (8555-LE0 and LT0) is increased to 4 MB with the capability of supporting up to 16 MB of memory. The 8555-LE0 and 8555-LT0 can also be upgraded with a 40 MB or 80 MB fixed-disk drive.

#### Highlights:

- New PS/2 Models 8555-041 and 081 with fixed disks larger and faster than previous Model 55s; 4 MB standard memory; 80386SX 16MHz processor
- Upgrade of PS/2 Models 8555-LE0 and 8555-LT0 to 4 MB standard memory; 80386SX 16MHz processor

Letter # 191-077, June 11, 1991

### IBM PS/2 Model 40 SX (8540-040, 043, 045)

The PS/2 Model 40 SX system unit utilizes the 80386SX microprocessor operating at 20MHz with zero to two wait states to system memory. The Model 40 SX ships with 2 MB (million bytes) system board memory (expandable to 16 MB on planar), new 3.5-inch DASD options, five slots and four bays, and 16-bit video graphics array (VGA). The 80386SX microprocessor PS/2 Model 40 SX has the following integrated functions: parallel port, serial port, pointing device port, VGA port, keyboard port,

1.44MB diskette drive support, Math Co-Processor socket, and three single in-line memory module (SIMM) sockets (two available for memory expansion). The PS/2 Model 40 SX is offered in a diskette-only model (8540-040) and 40MB and 80MB fixed-disk models (8540-043 and 8540-045).

The Model 40 SX supports the IBM Enhanced (101-key) Keyboard, Space Saving (84-key) Keyboard, or the IBM Host Connected (122-key) Keyboard. The keyboard selection can only be specified in new equipment orders and cannot be ordered separately for on-order or installed equipment.

Also announced are two new 8 MB memory module options designed to expand system board memory. The IBM PS/2 8 MB Memory Module Kit-80ns (#1401, 6450129) contains 8 MB of 80 nanosecond memory on a single SIMM. The IBM PS/2 8 MB Memory Module Kit-70ns (#5116, 6450130) contains 8 MB of high speed, 70 nanosecond memory on a single SIMM. These memory module kits are supported on PS/2 Models 35 SX, 35 LS, 40 SX, 57 SX, 90 XP and 95 XP. The PS/2 Models 90 XP and 95 XP can now support up to 64 MB of memory on the system board.

Letter # 191-076, June 11, 1991

### IBM PS/2 Model 35 SX (8535-040, 043) IBM PS/2 Model 35 LS (8535-24X)

The PS/2 Model 35 SX utilizes the 80386SX microprocessor operating at 20MHz with zero to two wait states and has the following integrated functions: parallel port, serial port, pointing device port, video graphics array (VGA 16-bit) port, keyboard port, 1.44 MB (million bytes) diskette drive support, Math Co-Processor socket, and three single in-line memory module (SIMM) sockets (two available for memory expansion). The PS/2 Model 35 SX is a 3-slot 2-bay system and is offered in diskette-only (040) and a 40MB fixed-disk model (043). All models have 2 MB of memory standard on the system board (expandable to 16 MB). The PS/2 Model 35 LS provides a

small footprint and low priced, general business solution.

The Model 35 LS provides a local area network (LAN) workstation solution and is available with all the standard features of the Model 35 SX, with the following exception: no DASD devices are installed in this "medialess" system, but a 16/4 Token-Ring adapter with RIPL occupies one of the three adapter slots. The PS/2 Model 35 SX is fully upgradeable to the Model 35 SX configurations.

Both the PS/2 Model 35 SX and 35 LS support the IBM Enhanced Keyboard (101/102 keys), Space Saving Keyboard (84/85 keys) or the IBM Host Connected Keyboard (122 keys). The keyboard selection can only be specified in new equipment orders and cannot be ordered separately for on-order or installed equipment.

The PS/2 Floor Stand is a new option available to customers who wish to install the system unit in a vertical orientation.

Letter # 191-075, June 11, 1991

### **IBM Select-a-Keyboard and IBM PS/2 Host Connected Keyboard**

Select-a-Keyboard allows a customer to specify one of the PS/2 keyboards in this announcement for the initial orders of either the PS/2 Models 35 SX and LS, Model 40 SX or Model 57 SX. Those available are the Space Saving Keyboard, the Enhanced Keyboard and the new Host Connected Keyboard. They are available in a variety of languages.

The new Host Connected Keyboard will allow users to operate in both PC and Host modes from a PS/2, when used with either the IBM Personal Communications/3270 Version 2.0 or the IBM PC 3270 Emulation Program, Entry Level Version 2.0 products. This keyboard provides a key arrangement very similar to that found on the most popular 3270 style keyboard. Key caps are clearly marked for both 3270 emulation and PS/2 application usage.

Letter # 191-086, June 11, 1991

### **IBM PS/2 3.5-Inch Rewritable Optical Drive, 128 MB Rewritable Optical Cartridge, Optical Drive Kit A**

The PS/2 3.5-Inch Rewritable Optical Drive (#0162, 6450162) is a new technologically advanced SCSI storage device. This attractively priced optical drive features high performance (4.3 MB per second data transfer rate, 66ms average seek time) and high capacity (127 million bytes) in a compact 3.5-inch half-high package. The drive accepts either a single-sided removable 3.5-inch magneto-optical (MO) cartridge that can be written and read many times, or a 3.5-inch optical read only memory (O-ROM). The drive adheres to the ANSI industry standard Small Computer System Interface (SCSI) X3.131-1986 and can be installed in PS/2 Models 57 SX, 60, 65 SX, 80, 90 XP, 95 XP, 3510 and 3511. When combined with either of the PS/2 SCSI enclosures (3510-0V0, 3511-003), this versatile drive offers optical function to any PS/2 Micro Channel desktop or floor-standing system with a PS/2 SCSI adapter installed.

The IBM PS/2 128 MB Rewritable Optical Cartridge (38F8645 single pack, 38F8646 five pack) is a 3.5-inch MO media, similar in appearance to 3.5-inch diskette media. This media has a formatted capacity of up to 127 million bytes and can be written and read many times. These attributes allow this media to be used for online/off-line data storage, backup/restore, software distribution, or a wide variety of data transfer applications.

O-ROM is 3.5-inch optical read-only memory with up to 122 million bytes typical capacity, similar in appearance to 3.5-inch diskette media. This media can be used by software or application developers to distribute CD-ROM like applications, such as data bases, encyclopedias and image books.

The IBM PS/2 Optical Drive Kit A (#1121, 6451126) is specifically designed and required to support the installation of a PS/2 3.5-Inch Rewritable

Optical Drive (#0162, 6450162) in PS/2 Models 60 and 80-041, 071, 111 and 311 with either PS/2 Micro Channel SCSI adapter installed.

Letter # 191-082, June 11, 1991

### **IBM PS/2 2.3 GB External SCSI Tape Drive**

The family of PS/2 Small Computer System Interface (SCSI) storage options is enhanced through the introduction of the IBM PS/2 2.3GB External SCSI Tape Drive (3532-023). This self-contained external SCSI tape drive has the flexibility to be attached to any PS/2 Micro Channel system equipped with either a PS/2 Micro Channel SCSI Adapter or a PS/2 Micro Channel SCSI Adapter with Cache. The 3532 is a high performance, large capacity, tape drive that has an instantaneous data transfer rate of 245 KB per second and can store up to 2.3 GB of data on one 8mm tape cartridge. Device reliability is enhanced through the use of a powerful Error Correction Code (ECC) function.

Also being announced with the IBM PS/2 2.3GB External SCSI Tape Drive are two new SCSI cables that are unique to this device. The IBM SCSI Controller Cable (#2830, 31F4187) is required to attach the 3532 to the PS/2 system unit (via the external port of the SCSI adapter). The SCSI Device-to-Device Cable (#3131, 31F4186) is required to attach the 3532 to another 3532, or to other SCSI devices already present. If other SCSI devices are to be added in a "daisy-chain" fashion, consult the appropriate device information for cabling requirements. Either the Sytos Plus® OS/2 (15F7193) or Sytos Plus/IBM DOS (15F7194) File Backup Utility is required for operation. These utilities, when used with the 3532-023, provide a high-performance, large capacity, easy-to-use tape backup/restore system. Support in the Novell environment is provided by Cheyenne Software, Inc. with their ARCserve® program.

Letter # 191-085, June 11, 1991



## **IBM PS/2 2.88 MB DISKETTE DRIVE**

The PS/2 2.88 MB Diskette Drive is a 3.5-inch one-inch high diskette drive that features media sense capability, and up to 2.88 MB of formatted capacity when a 3.5-inch 4 MB diskette is installed. In addition, this drive maintains read/write compatibility with 720 KB and 1.44 MB diskette drives. The drive is designed to be installed in either PS/2 Models 35 SX and LS, 40 SX, or 57 SX.

Letter # 191-083, June 11, 1991

## **IBM PS/2 1.44 MB Diskette Drive, 5.25-Inch External Diskette Drive Adapter Cable, and Selected Upgrade Kits**

The PS/2 1.44 MB Diskette Drive (#0030, 6451130) is a new option for the PS/2 Models 35 SX, LS, 40 SX and 57 SX. This 3.5-inch one-inch high form factor diskette drive features up to 1.44 MB of formatted capacity, read/write compatibility with 720 MB formatted diskettes and media sense capability.

The IBM PS/2 5.25-Inch External Diskette Drive Adapter Cable (#1124, 6451124) is required to attach a PS/2 5.25-Inch External Diskette Drive to a PS/2 Model 35 SX, LS or 40 SX. This cable is installed internally and attaches the 5.25-inch adapter to the system unit diskette drive cable.

The IBM PS/2 Upgrade Kit for Diskette Drives (#5106, 6451127) provides the capability to upgrade a PS/2 Model 35 LS to accept either a 3.5-inch or 5.25-inch diskette drive.

The IBM PS/2 Model 35 LS Upgrade Kit for Hardfiles (#5107, 6451128) provides the capability to upgrade a PS/2 Model 35 LS to accept either a PS/2 40 MB Fixed Disk Drive (#1073, 6451073) or 80 MB Fixed Disk Drive (#1074, 6451074).

Letter # 191-084, June 11, 1991

## **IBM PS/2 Internal Tape Backup Program Version 2.0 (DOS compatible) and IBM PS/2 Internal Tape Backup Program Version 1.01 (OS/2 compatible)**

These backup programs enhance the PS/2 8540, 8550, 8560, 8565, 8570, 8580, 8590 and 8595 by allowing the user to transfer up to 120 MB of formatted data from a disk storage device to a removable mini-tape cartridge for later recall.

For the customer's convenience, the IBM PS/2 Internal Tape Backup Program Version 2.0 (DOS compatible) and the IBM PS/2 Internal Tape Backup Program Version 1.01 (OS/2 compatible) are also offered in separate convenience kits that include the appropriate program, the IBM PS/2 Internal Tape Backup Unit (#5279) and a formatted mini-tape cartridge in one package. These backup programs, when used with the IBM PS/2 Internal Tape Backup Unit, provide the customer with an easy-to-use, easy-to-install and technologically advanced internal tape backup system.

Upgrades from the earlier versions of the DOS compatible program to the IBM PS/2 Internal Tape Backup Program Version 2.0 (DOS compatible) or to the IBM PS/2 Internal Tape Backup Program Version 1.01 (OS/2 compatible) can be done by ordering the appropriate upgrade. Customers can upgrade from earlier versions of the OS/2 compatible program by ordering the IBM PS/2 Internal Tape Backup Program Version 1.01 (OS/2 compatible) upgrade.

Letter # 291-278, June 11, 1991

## **IBM ISDN Interface Coprocessor**

The IBM ISDN Interface Coprocessor provides attachment to the Integrated Services Digital Network Basic Rate Interface (ISDN BRI) and, when operating with supporting IBM software, provides the capability of full duplex data transmission by utilizing the two 64 kilobits

per second (Kbps) information (B) channels controlled by the 16 Kbps control (D) channel.

The IBM ISDN Interface Coprocessor and supporting software, when installed in an AT® BUS PS/2 Model 30 (286), Model 35, or Model 40, enable ISDN connectivity not only to other similarly equipped PS/2 workstations, but also to Micro Channel Architecture PS/2 workstations with the IBM ISDN Interface Coprocessor/2 Model 2 installed, IBM 7820 ISDN Terminal Adapters, IBM 3174 ISDN Basic Rate Interface Adapters, and an AS/400 system with the BRI Adapter feature.

Letter # 191-115, July 9, 1991

## **IBM PS/2 486/50 Processor Upgrade Option**

The PS/2 486/50 Processor Upgrade Option enhances the PS/2 Model 90 XP 486 and 95 XP 486 family of systems. The processor upgrade option features the new Intel 80486 50MHz microprocessor and provides the capability of processor performance upgrade on the PS/2 Model 90 XP 486 and 95 XP 486 systems. This new microprocessor includes an internal memory cache controller, 8 KB memory cache and an integrated floating-point processor unit. Additionally, an external 256 KB level two cache is provided as standard on the processor complex. The 486/50 Processor Upgrade Option significantly increases system processor performance in compute-intensive applications while providing investment protection.

Letter # 191-096, June 11, 1991

## **IBM PS/2 8504 Monochrome Display**

The PS/2 8504 is a 12-inch multimode analog display. It uses a Flatter Squarer Tube (FST) for optimal front of screen performance. Low emissions, front controls, 300-degree swivel with tilt and reduced footprint are leadership design characteristics. The PS/2 8504 Monochrome Display is suitable for the low-end, low-cost market, where color is not

exploited, but high clarity in text, graphics and images is required.

The PS/2 8504 Monochrome Display is engineered to meet the recommendations for lower field emissions (VLMF).

Letter # 191-080, June 11, 1991

### **IBM PS/2 8516 Touch Display Model 001**

The PS/2 8516 Touch Display brings a new dimension of interactive operation to the PS/2 product line, and is intended for new application areas where touch is used for input and interaction. No overlay is required for touch operation and, therefore, front-of-screen quality is maintained. Touch input resolution is equivalent to the display resolution. The display supports XGA mode and all VGA modes and can be operated as a Personal System/2 8515 Color Display. When attaching to PS/2 systems with XGA graphics capability, 256 colors at a time may be selected.

Device drivers are supplied with the display that support the touch operation with the IBM Disk Operating System (DOS) or OS/2 Standard Edition or OS/2 Extended Edition operating system.

In response to emerging customer requests, the PS/2 8516 Touch Display meets VLMF (Very Low Magnetic Field), field emission recommendations for Scandinavia.

Letter # 191-081, June 11, 1991

### **IBM Personal Printer Series II Impact Printers (2380 and 2381)**

The IBM Personal Printer Series II (PPS II) 2380 Printer is a narrow-carriage 9-wire impact printer. The IBM PPS II 2381 Printer is a wide-carriage 9-wire impact printer. Both printers are designed for attachment to IBM Personal Computers, IBM personal systems, and selected IBM displays and processors. Both are medium-usage printers, offering low-cost ownership and versatile paper handling with a speed of 320 characters per second (320 CPS) in FastDraft mode.

The IBM PPS II 2380 and 2381 Printers are excellent choices for customers needing versatile forms-handling impact printers. The printers also offer a new user-friendly operator panel that can be easily operated by the printer user.

Letter # 191-110, July 9, 1991

### **IBM Personal Printer Series II 2390 and 2391 Impact Printers**

The IBM Personal Printer Series II (PPS II) 2390 is a narrow-carriage 24-wire impact printer; the IBM PPS II 2391 is a wide-carriage 24-wire impact printer. Both printers are designed for attachment to IBM Personal Computers, IBM personal systems, and selected IBM displays and processors. Both are medium-usage printers, offering low-cost ownership and versatile paper-handling with speeds of up to 200 characters per second (CPS) in FastDraft mode and up to 60 CPS in Letter Quality mode.

The IBM Personal Printer Series II 2390 and 2391 are excellent choices for customers needing versatile paper-handling impact printers or entry-level, high-resolution desktop graphics printers. The printers also offer a new user-friendly operator panel that can be easily operated by the printer user.

Letter # 191-129, August 20, 1991

### **IBM PCradio Models 001, 002, 003**

The IBM PCradio is a hand-held rugged unit that extends host services to the mobile work force via a variety of communication alternatives in a DOS-based unit. All models provide:

- A 5-10MHz 80C186 processor with power management system
- Integrated Circuit (IC) memory card slot
- CGA liquid crystal display
- QWERTY keyboard, featuring imbedded numeric keypad
- 2400 bps internal modem
- Battery charger

- Serial and parallel (via the breakout box feature) ports for attaching external devices.

IBM will offer these models of the PCradio, depending on communications requirements (radio, cellular or telephone line). It will be available on a special-bid basis, with general availability in the fourth quarter 1991. The PCradio has not been approved by the Federal Communications Commission (FCC). It is not, and may not be, offered for sale or lease, or sold or leased until the approval of the FCC has been obtained.

Letter # 191-126, August 13, 1991

## **Software**

### **IBM System Performance Monitor/2 Version 1.0**

IBM System Performance Monitor/2 (SPM/2) Version 1.0 consists of an integrated package of performance monitoring and analyzing facilities executable in OS/2 SE or EE Version 1.2 or 1.3 environments. SPM/2 enables system administrators to monitor system performance, analyze performance problems and use SPM/2 as an aid for performance tuning, load balancing and network growth managing efforts. In addition, SPM/2 enables application developers to verify performance objectives and fine-tune applications.

#### **Highlights:**

- A Data Collection Facility that collects critical OS/2 CPU, disk and memory utilization data, enabling local or remote systems management
- A Monitor Facility that enables real-time performance monitoring
- Log and Report Facilities that unveil details of resource utilization on a dispatched process basis
- Memory and Directory Analyzers that enable in-depth analysis of OS/2 memory management and disk capacity information.

Letter # 291-273, June 11, 1991

## Index to Past Issues of IBM Personal Systems Technical solutions

### Issue 3, 1991 (G325-5012)

PS/2 Model L40 SX Laptop Portable Computer  
OS/2 2.0 Considerations  
Comparing PC-DOS, OS/2, and AIX PS2 — Part 2  
Using Dual Displays with OS/2  
Local Area Networks: The New Utility  
And You Thought LANs Were Just for the Office!  
Remote LAN Management Tools  
New Horizons for IBM's Shielded Twisted Pair Cabling  
Tuning and Self-Tuning Features of OS/2 LAN Server  
NetWare Communications and Routing Protocols  
Little Solutions for LANs

### Issue 2, 1991 (G325-5011)

IBM PS/2 Model 90 XP 486 and Model 95 XP 486  
Choosing an I/O Bus Architecture  
The Network Is the Message  
Invoking Printer Job Properties  
Comparing PC-DOS, OS/2, and AIX PS/2  
Programming PM Using the COBOL/2 Bindings  
Installing and Using the DOS Database Requester  
OS/2 LAN Server 1.3 Overview  
IBM Windows Connection 2.0  
SNA Definitions for 3270 Emulators – Part II  
IBM THINKable

### Issue 1, 1991 (G325-5010)

XGA – Raising Video Expectations  
Choosing betw. Shielded and Unshielded Wiring for Data Transmission  
Compatibility of LAN Servers and Requesters  
Running DOS LAN Requester and Novell Netware Concurrently  
Breaking the 640 KB DOS Memory Barrier  
Understanding an OS/2 CONFIG.SYS File  
OS/2 EE 1.2 Database Manager Performance  
OS/2 EE 1.2 Competitive Performance  
An Intelligent Front-End EASEL Application  
Enabling Software for National Language Support  
SNA Definitions for 3270 Emulators  
Diskette Failures Caused by Contamination

### Issue 4, 1990 (G325-5009)

First Look at the New IBM PS/1 Computer  
Using the IBM 4019 LaserPrinter Effectively  
Micro Channel Interface Chip Sets  
Token Ring Bus Master LAN Adapters  
Extension of Wiring Rules for 4-Mbit/s Token Ring Using UTP Lobes  
SCSI and DISK.386.SYS  
Operating System Platforms: A Business Perspective  
Minimum OS/2 1.2 DASD Requirements  
User Profile Management  
Understanding OS/2 1.2 LAN Server Performance  
PM: An Object-Oriented Approach  
DOS 4.00 SHARE  
A "C" Programming Model for DOS Device Drivers  
An Electronic Bulletin Board for PC Users

### Issue 3, 1990 (G325-5007)

DOS – A Look under the Hood to See How It Spins  
Memory Management in a DOS Environment  
FASTOPEN – The DOS Performance Enhancer  
DOS 4.00 Compatibility Issues  
'Out of Environment Space' Errors  
A New LAN Requester for DOS Systems  
Creating a Dialog Box Dynamically Using WinCreateDlg  
An Alternative for the OS/2 START Command  
CUA: A Consistent Interface

### Issue 2, 1990 (G325-5006)

OS/2 End User Advantages  
What's New in OS/2 Standard Edition Version 1.2?  
An Application Developer's View of OS/2  
Object-Oriented Programming with C and OS/2 PM – Is It Possible?  
Design Goals and Implementation of the New HPFS  
OS/2 EE 1.2 Database Manager – Remote Data Services  
OS/2 EE Database Manager Precompiler API  
UNION, INTERSECT, EXCEPT  
Writing a Database Manager COBOL/2 Program  
Database Manager Programming with Procedures Language 2/REXX  
APPC Performance Tips for OS/2 EE  
EASEL OS/2 EE PROFS: Host Code Interface  
PS/2 RPG II Application Platform and Toolkit  
The IBM Independence Series Products

### Issue 1, 1990 (G325-5005)

Introduction to Local Area Networks  
IEEE 802.3 LAN Considerations  
The IBM Token-Ring Network: A New Generation  
How to Design and Build a 4-Mbit/s Token-Ring LAN  
How to Design and Build a 16-Mbit/s Token-Ring LAN  
Making the Cabling Decision  
IBM Cabling System Highlights  
Communication Strategy for Growth

### Issue 4, 1989 (G325-5004)

Requirements for Advanced Bus Architecture  
Micro Channel System Configuration Considerations  
Features and Benefits  
Bus Masters and Applications  
Overview of Extended Micro Channel Functions  
New Micro Channel Features  
SCB – An Architecture for Micro Channel Bus Masters  
Design Alternatives with Micro Channel Systems  
The IBM PS/2 Micro Channel SCSI Adapters  
PS/2 Wizard Adapter  
Experience in Bus Master Design  
Bus Master Adapters from Independent Option Vendors  
Bus Masters and OS/2  
Micro Channel Issues in AIX PS/2  
Information for Developers  
Book Report: "The Winn Rosch Hardware Bible"

“When companies connect hundreds and thousands of PCs in a building, the triplen harmonic currents add to those of the other non-linear loads creating unexpected electrical problems. (page 4)

“Database Manager has extensive data integrity and protection features. (page 8)

“CM workstations can access gateways to System/370 architecture hosts, other hosts supporting LU6.2 protocols, and ASCII hosts. (page 18)

“Aligning data structure can make significant performance improvements to an application. (page 25)

“Window panels can be created efficiently with the dialog box editor. (page 33)

“Compiling the application for introduction into the production environment is best done on the DOS machine after Micro Focus COBOL/2 has been installed. (page 51)

“Memory relief with DOS 5.0 can be quite dramatic. (page 54)

“The shell (NETX.COM) acts as the interface between user applications and NetWare file servers. (page 78)

“LANACS frames come in different forms, commonly called protocols. (page 93)

“Here's a trick you can use to gain a small measure of flexibility in your library path during execution. (page 96)

6325-5013-00

