EVERY MANAGER'S GUIDE TO OS/2

By Mike Edelhart and David Strom

First Draft 1988/02/10

License

Introduction Notes

Note (2012/04/17): This document is a draft document that was never released to the public until today. This draft document was wrote by Mike Edelhart and David Strom on 1988. It was released to the public under a Creative Commons on 2012 by request of Martín Itúrbide.

Table of Contents

Chapter 1. What is OS/2 and why should I care about it?

1.1 What is OS/2?

Operating System/2, known universally as OS/2, is a family of operating system products from Microsoft and IBM designed, for the first time, to take full advantage of remarkable computing powers in Intel's 80286 microprocessor. OS/2 is not a new version of DOS, the venerable operating system that has carried the IBM PC, XT, AT and a host of clones to extraordinary success. While maintaining overall compatibility with existing DOS applications and commands, OS/2 represents something quite new and different in microcomputing.

OS/2 is going to be undeniably important in the months ahead. Microsoft has made it the heart of the company's future developments, both in terms of systems software and applications. IBM has dubbed OS/2 a strategic operating system, the sole platform for Big Blue's future desktop computers and a component of Systems Application Architecture, IBM's plan to create a universal application environment across all of its varied machines. Virtually every significant microcomputer hardware vendor has also announced intentions of making OS/2 available on their 80286 boxes.

All of this support guarantees that OS/2 will become a major operating system and the focus of intense development by application program developers. It does not, however, mean that OS/2 will totally supplant DOS nor that the current generation of application programs will become worthless simply because a new possibility has emerged.

1.1.1 An era of two standards

For corporate managers, program developers, networkers and buyers this means that, like it or not, we have entered an era of two standards--the existing DOS standard and the emerging universe of OS/2. We will also be facing a time of transition in applications: a migration from the applications we have grown up with on our PCS--Lotus 1-2-3, dBASE, WordStar--to new and as yet unseen programs for OS/2 that promise breathtaking power and functionality.

Managing these two worlds, overseeing the transition from the old technology to the new on a desktop by desktop basis is likely to dominate micro managing throughout 1988 and well into 1989. Any manager who ignores OS/2, or who fails to become as expert with it as he now is with DOS, runs the risk of limiting the continued growth of PCs in his organization and even of losing his job. A manager, on the other hand, who falls too deeply in love with the new capabilities of this tantalizing environment and who doesn't slow down to recognize the numerous technical traps hidden in OS/2 or to ask essential questions about how or whether OS/2 can best help users in his firm, may be setting himself up for waste, frustration and embarrassment.

1.1.2 Managers must balance DOS and OS/2

The key for negotiating these operating system shoals is to find a stable middle ground in these turbulent times. This requires, first of all a full understanding of DOS and PCs, their many continuing virtues and their distressing limitations. It also demands a thorough grounding in the capabilities of OS/2 and the 80286 microprocessor, particularly in terms of what new powers they may bring to applications,

which in turn can produce genuine user and corporate benefits. Proper management of the OS/2 challenge also calls for extensive thought about the role micros have played in an organization to date and the role they should play for the rest of this decade and beyond. Finally, proper response to OS/2 must be based on an understanding of the PC users in an organization, their level of proficiency, the tools they need to improve their job performance, the training they require and the sacrifices they are willing to accept.

These issues, from the technical underpinnings of the operating systems through the human issues of the workers who will have to use them, form the basis of this book.

1.2 What are the different forms of OS/2 and how do they relate to one another?

1.2.1 OS/2 Version 1.0

Announced in April of 1987 and delivered in stages throughout 1987 and early 1988, version 1.0 of OS/2 is a transition product from DOS to the full flower of OS/2. Presented in virtually identical forms by both Microsoft and IBM, version 1.0 offers a first look at the OS/2 API for developers and a glimpse of OS/2 operations for managers and users.

This initial version however is missing some crucial parts of the OS/2 equation. Most glaring is the Presentation Manager, OS/2 graphical user interface. In its place, version 1.0 uses the Screen Manager, a character based interface that looks much like IBM's TopView.

The lack of the Presentation Manager affects more than just screen handling in OS/2. The Presentation Manager API also impacts heavily on OS/2's overall application environment.

However, version 1.0 of OS.2 does break the DOS 640K barrier and does provide full access to the 80286 processor's protected mode.

1.2.2 OS/2 Version 1.1

More recently, IBM and Microsoft have released OS.2 version 1.1. This update does contain the Presentation Manager, and represents the first true commercial personality for OS/2. IBM calls its flavor of OS/2 1.1 "The Standard Edition," and that is exactly what it is likely to become.

IBM's and Microsoft's versions of OS/2 remain essentially identical in 1.1. The differences are largely in the device drivers supplied with the product. IBM's version will have drivers for pure IBM hardware. Microsoft will create versions containing drivers for the hardware of anyone who contracts with them. OS/2's need for and sensitivity to device drivers is discussed in Chapter 2.

It is important to note that, as far as the marketplace is concerned, there is no such product as Microsoft OS/2. Microsoft will not sell OS/2 directly to users. The operating system will only be available through hardware licensees. So, there will be a Compaq OS/2 and a Tandy OS/2, both versions of the Microsoft product. IBM will sell its OS/2 through IBM retail channels, but this product, unlike PC-DOS, will run only on 100 per cent IBM hardware; it will not run on most AT clones. We discuss the implications of OS/2's flavors, and their ties to specific hardware in Chapter Nine.

All flavors of version 1.1 can be considered the predominant form that OS/2 will take for many months.

1.2.3 IBM OS/2 Extended Edition.

IBM will follow its standard edition--which is virtually the same as Microsoft's product--with a Big Blue only edition of OS/2 that goes far beyond MS OS/2 in any announced form.

IBM's Extended Edition adds a number of modules to OS/2 that extend its operation beyond traditional micro operating system tasks into the realm of distributed systems control and application generation. Basically, the extended edition transforms OS/2 from an operating system into a total systems development environment, a significant software component of corporate IBM computer arrays.

The Extended Edition doesn't change OS/2--the kernel will be the same as for version 1.1 and the Presentation Manager will remain as the screen interface. Rather, it adds new functions to it. Specifically, IBM intends to add these modules in the Extended Edition:

The Communication Manager. This is a set of communication protocols and programs designed to make it easy for OS/2 workstations to communicate with other IBM machines. It includes local area network services, micro-mainframe protocols, telephone access protocols and other communication formats. The Communication Manager should take over virtually all the functions currently provided by DOS communication, terminal emulation and networking software.

The Database Manager. This set of software tools allows OS/2 to build relational databases based upon Structured Query Language (SQL), the query facility used by IBM in its DB2 mainframe databases. With it, an OS/2 user can either create his a relational database in his own machine or pull information from SQL databases anywhere else in the system his PC is connected to. Then he can create reports and even customized applications to put the data into.

IBM has stated that it plans to release other modules for the Extended Edition.

While OS/2 is not a proprietary IBM product, and the kernel of the system will be the same in all flavors of the product, the Extended Edition is IBM's plan to create a major differentiation between its OS/2 and all others. The reality of this advantage remains to be tested by managers and users.

Although no Microsoft equivalent of the Extended Edition is planned, it is likely that leading software companies, working either as group of individually, will offer OS/2 extensions modules with capabilities similar to those IBM has in store. Such firms as Oracle and Ashton-Tate in databases and DCA and Hayes in communications ave figured prominently in these expectations.

1.2.4 OS/2 related programs

Apart from the various versions and flavors of OS/2 the situation is confused by the existence of programs that interact closely with OS/2, but are not actually part of it. The most prominent of these is the LAN Manager, a utility set that allows OS/2 application functions to be routed across local area networks.

The LAN Manager is not an integral part of OS/2, but when it is installed it functions very closely with the operating system itself. When the LAN Manager is installed at both the PC and LAN Server, OS/2 users can address networks drives, can share application tasks with other workstations and use network services all from within standard OS/2 just as they would services from their own PC.

IBM and Microsoft both have versions of the LAN Manager. In the Microsoft scheme, the LAN Manager works with OS/2 on both the LAN server and the PCs. DOS based PCs can interact with OS/2 servers using old MS-Net or PC-Network programs. In the IBM version, the LAN Manager becomes the IBM OS/2 Local Area Network Server program. This server software will work with either the OS/2 Extended Edition or IBM's PC Local Area network Program ver. 1.3 on the workstations. In other words, IBM, unlike Microsoft, has no LAN Manager for OS/2 ver 1.1, and bundles the workstation component of the LAN Manager into its Extended Edition.

The two programs will have substantial compatibility, but some performance differences, which we will discuss in more detail in Chapter Eight.

In addition to the LAN Manager, other tools for OS/2 have or will soon be released. Lotus Development Corporation, for instance, has a toolkit of Lotus DBMS and Lotus 123G that tightly integrate with OS/2 so that PC users can interact with Lotus databases running on LAN servers. As with the LAN Manager, these tools appear to be part of OS/w once they are installed, but actually are a separate program.

OS/2 has been designed to be extensible, which means vendors and corporate users, can customize the system's operation to their circumstances. This is good news, in terms of making the operating system your own. But it can be bad news in terms of keeping track of and supporting a myriad of installable modules. The situation with OS/2 add-ons will probably much more closely resemble the enormous Lotus 1-2-3 aftermarket than any DOS trend.

1.3. How are 80286 processors different from 8088/86 chips, and what does this mean for OS/2?

For better or worse, OS/2 exists in the form it has today because of the Intel 80286 microprocessor. This is good news because the 286 offers application developers and users capabilities far beyond those of the 8088 chip that powers PCs. It is bad news, however, because technical complexities of dealing with the 286 and thorny compatibility issues have resulted in OS/2 lagging the chip it is designed for by almost 4 years and laboring under an unwiedly--some have even called it klugey--two-faced structure.

To understand why OS/2 is what it is, you have to understand where it came from.

1.3.1 The 8088 and the birth of a limited DOS

Shortly after the IBM PC exploded onto the corporate computing scene, it became clear that the machine was somewhat limited for the jobs it was being given. When the PC was first designed, no one at IBM thought it would be used for huge, critical corporate spreadsheets or databases. Nor did the PC's designers think that it would become a day-in, day-out functional tool for workers with a need to use many programs side-by-side.

The PC's processor, Intel's 16-bit 8088, was designed to handle just one operating system and just one application at a time. It is a single-tasking chip. The 8088 was also created to support just one megabyte of random access memory. In the PC, IBM and Microsoft reserved 260 kilobytes of this memory for housekeeping chores, leaving 640 kilobytes free for applications, an amount that sounded huge at the time the PC was released, but soon was eaten up by ever more complex PC programs.

The operating system IBM and Microsoft cooked up for this machine assumed a single user pursuing a single task on his PC. Far simpler than other corporate computing operating systems of its day, DOS, primarily concerned itself with providing peripheral support to the user--access to disk drives, printers and screens. It was straightforward, forgiving and comprehensible, but not particularly sophisticated.

The nuclear impact of PCs on business computing, however, caused both users and developers to push against the PC's constraints almost immediately. Developers wrote programs that squeezed the last ounce of performance from the machine with all manner of clever technical tricks. Others offered integrated software--numerous applications bound up in a single package--or screen environments that let users have apparent access to several programs of their choosing at one time. As the 640K memory limit became more of a problem, Lotus-Intel-and Microsoft, and AST independently, created extended memory systems that made more memory available, but in a rather limiting way. Try as developers might, the PC was simply being outgrown by the jobs being asked of it.

1.3.2 The fenced-in appearance of the 80286

Meanwhile, IBM had released the PC-AT using Intel's new, advanced 80286 microprocessor. On the surface it would seem that the 80286 would wash away the limitations that were vexing PC users. The 16-bit 80286 is capable of support TK megabytes of random access memory and can actually provide applications with TK gigabytes of virtual memory. The chip is specifically designed to run more than one application at a time. This so called "protection" feature means that if program A and B are both running on the 80286, and program A crashes, program B doesn't. On the 8088, one bad instruction in any program would take out all programs and files in the machine, a dangerous situation, to be sure when users are loading their PCs with multiple programs and dealing in critical corporate data.

But sadly, none of these wonderful capabilities were available on the PC-AT. It carried almost all the limits of DOS and the PC. New versions of DOS for the AT had incremental improvements, but never touched any of the hot properties on the 286 chip.

1.3.3 The root of the problem: two personalities.

Here is why: Intel engineers built two personalities, called modes into the 80286. To support 8088 programs, the 80286 had a real mode. It was called real because random access memory on the 8088 is accessed by real memory addresses, just as personal mail goes to a real address that corresponds to a single house. To handle the new chip's new capabilities, Intel created the protected mode.

In this mode, the 80286 could protect programs from each other's mistakes, hence the name. This mode also could use a scheme for RAM called virtual memory, in which memory addresses are contained in tables, more akin to the addressing of letters to post office boxes, rather than actual houses. Each matrix of post office boxes could represent several different streets; each box could, at various times,

hold letters for more than one house. Virtual addressing is the reason the 80286 can support so much more memory than the 8088.

This all sounded great in the engineering room, but when the 80286 came out, big problems could be seen right away. First of all, the 80286 could only be in one mode or the other, but never both. Even worse, it turned out that virtually none of the existing base of DOS applications would run in the protected mode. None of them addressed memory in the way the protected mode required, and almost all used technical shortcuts in DOS that the protection features of the 80286 rendered absolutely illegal.

In other words, techies discovered when the AT came out that you could have DOS applications or the 80286's nifty new capabilities, but you couldn't have both. With the AT, IBM and Microsoft opted for compatibility and held off providing access to the 80286's power. Then, they headed into the lab to figure out a way to provide both compatibility and new features in a single 80286 operating system.

Years later, after innumerable false starts and disappointments, the result of these labors is OS/2.

1.4 How does OS/2 get around the 80286/DOS Catch-22?

OS/2 is actually two operating systems in one. One system--called the DOS Compatibility Environment--runs in the 80286 chip's real mode and provides an emulation of the 8088/DOS system. It is not DOS and it does not use an 8088, but it comes quite close to looking like a PC to a DOS based application--with some scary exceptions that we'll go into later.

The second operating system within OS/2 runs in the 80286's protected mode, bringing developers and users all that memory and power.

This two-headed system resides behind a single user interface. In early versions of OS/2 this is called the Screen Manager and looks something like IBM's Topview. In later versions of OS/2--Microsoft's version 1.1 and IBM's OS/2 Standard Edition, and beyond--the interface is the Presentation Manager, a Macintosh-like windowed, graphical screen system.

Existing DOS applications run in the Compatibility Environment, and old DOS commands are supported there. New OS/2 applications will run in the protected mode environment, where literally hundreds of new OS/2 commands will be available.

This may sound like a simple enough solution. Why did it take years? Because what's going on behind the screen is technical brinksmanship at its best. In order to overcome the 80286's design limitation and provide both real and protected environments at the "same time" OS/2 actually performs a hard reset of the chip whenever the user switches modes. In simple terms this means the operating system turns the chip off and then starts is up again in the other mode.

Think about that for a second. That's like saying you stop your own heart and then grab your own resuscitator and start it again. If DOS applications had not taken the world by storm and OS/2 didn't have to accommodate them in the 80286 generation, this system would be infinitely less complicated and probably would have been on the market 18 months ago.

1.5 Why is protected mode important?

The question that arises in the face of this tale of technical legerdemain is why not just skip the protected mode of the 80286? Why not simply continue to develop the real mode 8086/88 architecture or skip over the 80286 entirely to the most recent Intel chip, the 80386, which is available now and has far fewer compatibility problems than its predecessor?

The most straightforward reason is that software vendors can no longer deliver the kinds of programs their customers are demanding in the memory and performance confines of the 8088.

1.5.1 The need for serial multitasking.

Research over the past several years has found that so-called power users, the leading edge PC workers, use their computers in a way known as serial multitasking. This means that they don't use a single program. Instead, they want to use numerous programs that interact together. They only work on one program at each moment, but want to be able to switch back and forth between programs whimsically and instantaneously.

In order to accomplish this kind of easy inter-program switching, a computer must be able to hold numerous programs in memory at one time and keep track of all their needs and limitations. The original PC running DOS is simply not designed to do this. The protection functions built into the 80286 are essential to perform this more complex kind of personal computing; mutlitasking on an 8088 or in the 80286's unprotected "real" mode is like doing trapeze somersaults without a net.

So, any vendor who wants to be able to respond to the desires of those users who have driven every significant advance in PC history, has to get out of the single-tasking 8088 environment and into one that provides the RAM expanses and operating sophistication to provide rich multitasking.

1.5.2 Are the 80286 and OS/2 the only protected mode options?

No. Neither the 80286 nor OS/2 represent the only path that offers access to protected mode features. Intel's 80386 processor, which was released DTK, provides all of the protected mode benefits of the 80286 and much more besides. to begin with, the 80386, having been finished after the problems with the 80286's protected mode were discovered, uses a different method of emulating the 8088 environment that allows both protected and real mode operations to take place simultaneously. In other words, the problems that caused all the delays and complexity of OS/2's dual personality vanish on the 80386.

In addition, the 80386 can support even more RAM than the 80286--TK megabytes to 16 megabytes --at faster clock speeds. It is simply a much more powerful chip than the 80286 and at the same time is more compatible with existing architectures.

So why not just forget about the 80286 and move on to the newer better chip? Numerous reasons. To begin with, IBM eats its dead. Since Big Blue has been selling 80286 systems for some months and telling customers that eventually they would get access to all the chip's features, IBM is duty-bound to deliver on its promise. Since early 1987, IBM has been saying in public that it would give its customers access to

the 80286's protected mode as a way of opening up RAM and multitasking possibilities. Where IBM speaketh, the industry goeth.

Even without IBM's backing, however, the protected mode of the 80286 would still be valuable. For all its power, the 80386 processor is still quite immature. Its powers are so vast and complex that comprehending them, let alone exploiting them, will take years. The chip is so far beyond both the 80286 and 8088/86 that massive engineering must go into firing it up.

For instance, the first generation of machines to use the 80386 processor, the so-called super-ATs that began appearing in late 1987, can't utilize all of an 80386's operating modes. They use it merely to emulate an 80286; the only advantage to the user is greater speed because of the 80386's higher clock rate. The situation is similar enough to the fenced-in unveiling of the 80286 to be sobering. It will certainly be no earlier than the very end of 1988 before all the pieces necessary to fully exercise the 80386's unique abilities appear in public. Getting the bugs out of them will extend well into 1989, possibly even 1990.

This leaves the 80286 as the most reliable, easily exploitable big-RAM, multitasking option available for almost two years. That's a long time in the computer business.

However, beyond that timeframe the chances are excellent that the 80386 will become the long range platform for all kinds of microcomputing, including future OS/2 development, through the 1990s. Anyone with a serious interest in OS/2 should track the progress of the 80386 processor closely. In time, this operating system and chip will make beautiful music together.

On the software side, users today have a number of viable options for reaching the 80286s protected mode. Probably the best known is UNIX, an extremely sophisticated operating system originally developed by AT&T. UNIX is what is known as portable software, that means it can be easily moved from one architecture to another and easily customized to the foibles of different systems without losing its essential characteristics.

When the 80286 came out, UNIX was swiftly redesigned to take advantage of the chip's full menu of protected mode benefits. UNIX has been able to offer just about everything OS/2 intends to offer for more than a year.

UNIX carries some heavy negative baggage, however. First, it is incredibly abstruse and complex. Developed by recundite techies, the system is filled with incomprehensible commands and quirks. Getting UNIX to work right all but demands a computer science degree. On top of that, until very recently, UNIX could not run the DOS applications that stood at the heart of the PC's success. UNIX offered a devil's bargain, gain the protected mode and lose all your software. That is the same limitation that OS/2 has laboriously been designed to avoid.

Still, UNIX is a proven product is is actually being used in a few companies today to provides protected mode multitasking on 80286 machines. And, over the past few months, products have come out that allow DOS applications to run as "guests" in a UNIX environment; this isn't quite genuine compatibility,

but it does allow applications to be preserved. And, OS/2 itself bears more than a passing resemblance in many of its central concepts to UNIX. It would not be outrageous to say that OS/2 takes the technical concepts underlying UNIX and gives them a DOS-flavor and a human face.

Beyond UNIX, a number of small companies, such as the Software Link of Atlanta and TK have rushed to market with their own proprietary operating systems for the 80286 protected mode. While technically, these products work fine they have the near fatal drawbacks of not being backed by either IBM or Microsoft. Since the industry standard for 80286 operating system will come from the big guys, these products will almost certainly be consigned to also ran status.

1.6 The price you pay for protected mode.

For all the benefits it brings, the protected mode of the 80286 is not without its limitations. Some of these derive from the leap away from the familiar real mode way of doing things, but others are intrinsic to the new environment itself. In computing, as in life, there is no free lunch.

1.6.1 New applications are needed.

The biggest price managers and users will pay for OS/2 is not in the operating system itself. For all its new sophistication, OS/2 will not be incomprehensible to anyone familiar with DOS. The big shift will come in the applications that run under the new system. As we will discuss more fully in later chapters, beneath the operating environment that the user sees and the manager manages, OS/2 contains an application development system the likes of which has never been seen in microcomputing before./ The intent of this rich application command set is nothing less than the gestation of a whole new generation of software, with capabilities far beyond those of today's 123 and dBase.

It is not by accident that Microsoft and IBM have spent 1987--the year in which OS/2 was announced but not yet delivered--hosting innumerable seminars, not for users, but for application software developers. It is critical for OS/2's success that developers make use of the new tools that lie beneath the system's surface. Only breath taking applications will make the complexity and overhead of OS/2 worthwhile to corporate managers and users.

But of course new applications, particularly huge and complex ones, will bring a heavy burden of training, file management, installation and support to all those who wish to use them. As we said, no free lunch.

1.6.2 New concepts of programming and the way PCs are used.

While compilers are already available for OS/2 under C, the programming language most favored by corporate and commercial developers today, other high level languages do not yet have compilers, including BASIC. These programming tools will certainly appear in time, but when they do, they are likely to have fare less use and impact than they had in early PC days.

OS/2 is a C-oriented operating system. Even more than that it is its own application environment. As a result, our ideas of what application programs are and how they are written is likely to change.

In the old PC days, for instance, it was easy to think of an application program as being the machine that ran it. When the program was installed it took over the PC. In OS/2 this can never be the case (except in real mode, which apes the olds days). No program owns the machine. All programs must share, they must be "well behaved." This forces fare more strictures on PC programmers than they have ever had before. We have heard tales of some PC greybeard techies who have thrown in the programming towel in despair over the lack of hacking freedom in the new OS/2 PC world.

Along with these changes in thinking about programs, come corollary changes in the idea of how PCs are used. Just as, looking inward, OS/2 sees the machine as shared among many programs, looking outward, it sees the PC as a workstation communicating, sharing and being directed by many outside forces. LANS, mainframes, PBxes, telephone connections, shared applications and other links too numerous to mention. As DOS was designed with a standalone single purpose, single user, independent PC in mind, OS/2 is crafted with a linked, shared, multipurpose, group-oriented PC in mind.

These new ideas will force adjustments on all of us who work closely with desktop computers.

1.6.3 Cost.

To move from the philosophical to the practical, OS/2 is going to cost a bundle. The system itself isn't so expensive but, as with the French menu, the accompaniments will suck your wallet dry.

For instance, just to get OS/2 up and running on a workstation requires an AT-class machine minimum. No more cheapie 8088-PCs. And, that pricey 80286 machine needs at least 1.5 megabytes of memory just to wake up in the OS/2 morning. Two megabytes is preferable and more if you want to start getting fancy. At current prices that much memory will run you TK. Then there are monitors to support the eye catching VGA graphics supported under OS/2. These are going for TK % more than today's EGA color screens. What about a big hard disk drive for the many files required by multitasking programs and the wide array of OS/2 services. More money out of pocket.

And then there's software. All indications are that OS/2 applications will run from 25-50 % more expensive than the current generation of offerings. This reflects the intense r and d effort that has gone into building these snazzy packages. While they may be worth the money, given the exciting possibilities for what can be accomplished in the new environment, they are certainly going to strain PC acquisition budgets.

In short, given the costs involved, managers will have to see big real, demonstrable benefits from a move to OS/2. Otherwise, such a shift would be pure fiscal irresponsibility.

1.6.4 Schizophrenia

Finally, there is the reality that OS/2 does not represent a new industry standard. It represents another industry standard. The existing DOS standard shows no signs of going away. Its role may change and it may have to grudgingly move from center stage, but it will still be around. That means that managers must accept a dual standard world, something many, we think, hoped they had left behind when the PC standardized microdom years ago.

Some managers even see a three-standard environment, with Apple's Macintosh as the third leg of the stool.

If grief expands exponentially with each standard that must be supported, micro managers are facing a lot of headaches with the inexorable emergence of OS/2.

1.7 Is protected mode the only difference between OS/2 and DOS?

No way. As soon as it became obvious that creating a workable 80286 operating system was going to be a technical bear that would take a lot of time, Microsoft and IBM decided they might as well take advantage of the situation to bring some long overdue new concepts into the desktop computing picture.

The Presentation Manager, for instance, represents a major shift in the way applications interact with users. Instead of each program having its own look and feel, its own user interface, the OS/2 Presentation Manager serves as the link between the user and all his applications.

Another set of shifts involved tools for application programmers. DOS basically stayed out of an application's way. Apart from handling interaction with peripheral devices, it had little to offer and less to say to a well-constructed application. OS/2 on the other hand has been stuffed with snazzy new routines application programmers can use.

An example is Pipes. Pipe commands allow the output of one program or program process to be piped as the input to another program or process. The second process may be part of the original application or a different application. OS/2 offers dozens of similar new concepts for application creation.

OS/2 has also taken advantage of the time it took to get past technical hurdles to provide an extraordinarily rich and supple system for handling memory and taking advantage of the 80286 protected mode's protection features. These offer the first true multitasking environment on PCs outside the tiny world of UNIX.

While the real mode/protected mode conundrum refused to yield to elegant solutions, the structure OS/2 built up for its protected mode operation is extremely modular and elegant. Application developers or hardware manufacturers can pull out and modify or replace whole chunks of OS/2 to provide critical functionality or differentiation. As a result, OS/2 will be able to interact with peripherals--such as high density videodisks--that haven't even been invented yet.

With all of these changes and improvements--which we will examine in detail in later chapters--OS/2 looks and acts much more like a minicomputer operating system than like the old DOS. It would be fair to say that IBM and Microsoft used the long gestation period of OS/2 to build a system that radically changes the way desktop workstations run. Through OS/2, they have migrated the best ideas of a generation of mainstream office computing onto individual desktops.

Chapter 2. A Manager's Guide to the Basics of OS/2

For anyone proficient with DOS, working with OS/2 will be like interacting with a child who has returned from his first years away from home. You find many warmand well known qualities; you will have a general sense of familiarity about what you see. However, without any question, DOS has come back in its OS/2 form all grown up, filled with many ideas new to you and sporting behaviors and attitudes from other spheres of influence. To understand what is similar and different about OS/2's operation we must delve briefly into the operating system's inner workings.

2.1 What are the principal parts of OS/2?

Envision the internal structure of DOS as the sun and planets in a solar system. The large central body is command.com, the DOS command processing file. The small planets are DOS commands that derive their capabilities from the dominating central file. In DOS, command.com is the center of the universe; it controls virtually every significant aspect of the operating system's performance. Apart from this file, all other DOS commands are independent, unrelated, like planets floating in the ether. This kind of highly centralized structure is efficient, but incredibly inflexible. Almost any change or improvement to the operating system requires modification of the command file. This has limited DOS growth. With PCs interacting with many other kinds of computers today, this kind of inflexibility is no longer acceptable. So, OS/2 brings with it a new, layered, modular internal structure. The difference is akin to the gap between a hand built barn out back (DOS) and a skyscraper (OS/2).Between application programs at the top and hardware at the foundation, OS/2 sandwiches three interacting, but quite distinct operating levels: the dynamic link layer, the kernel and device drivers.

Figure 2.1

THE PARTS OF OS/2

And all OS/2 was divided into three parts: The DOS compatibility mode, the protected mode and the optional LAN Manager. The structure of the DOS compatibility mode includes a ROM BIOS compatible with DOS 3X software and a close emulation of the native DOS environment. In the protected mode, however, things work differently. Applications talk to an API described by dynamic link libraries; preeminent among these is the Presentation Manager, a graphics based user interface included in OS/2 version 1.1; The dynamic link libraries interact with the kernel of OS/2, which oversees trafficking, I/O and multitasking. Beneath the kernel sit device drivers that describe the underlying hardware. The LAN Manager, an optional program that links to the OS/2 kernel and extends it functions over local area networks, includes libraries that provide network application protocol services, and a transport layer which can support numerous schemes. IBM's Extended Edition of OS/2 will add to this model, libraries for database management and communications.

2.1.1 Dynamic Link Layer

The dynamic link layer (DLL) provides OS/2`s application program interface (API), the face the operating system turns toward the user and the software he works with. Programs the run under OS/2 interact much more fully and directly with this subsystem level than they do with the command processing files. This is in marked contrast to DOS, where the command file controlled everything directly. The most immediately notable manifestation of this change is a wholesale move in OS/2 way from the number

based interrupts that applications used in DOS to a system of named calls to the dynamic link API. As we will discuss later, this move away from interrupts, which are rife in DOS programs, may prove one of the more vexing problems managers must face with OS/2.Modularity, as you will see, is a watchword with OS/2, so not even the dynamic link layer of the operating system is an unbroken chunk of code. Instead, the DLL is comprised of separate dynamic link libraries (DLIBS). These libraries provide API services for specific territories within OS/2, providing services within that area to applications and translating application requests into formats that OS/2 internal code can handle. These libraries are not inextricably linked to OS/2; that's where the dynamic aspect of DLL comes from. They are either loaded into memory at run time or just when they are called.

OS/2 comes with many dynamic link libraries installed. Her are brief overviews of three of the most important:

VIO

The VIO Library decribes the screen environment that applications can use in OS/2. The API established by VIO is in the form of function calls that application programs can use to request OS/2 services. These functions are listed in Table TK. As video devices change--such as when the Presentation Manager debuts in OS/2 1.1, the VIO library can be updated without affecting other aspects of OS/2.

Table 2.1 OS/2 Video Functions

INT 10H			
FAMILY	**EQUIV.**	**NAME**	**DESCRIPTION**
		VioDeRegister	Deregister video subsystem
		VioEndPopUp	Deallocate pop-up display screen.
		VioGetANSI	Get ANSI status
F		VioGetBuf	Get logical video buffer
		VioGetConfig	Get video configuration
		VioGetCp	Get video code page ID
F	03H	VioGetCurPos	Get cursor position
F		VioGetCurType	Get cursor type
		VioGetFont	Get font table address
F	0FH	VioGetMode	Get display mode
F		VioGetPhysBuf	Get physical display buffer
		VioGetState	Get video state
		VioModeUndo	Cancel mode wait
		VioModeWait	Wait for mode change
		VioPopUp	Allocate pop-up display screen
		VioPrtSc	Print screen
		VioPrtScToggle	Print screen key operation trap
F	08H	VioReadCellStr	Read cell string
F		VioReadCharStr	Read character string
		VioRegister	Register video subsystem
		VioSavRedrawUndo	Cancel save-redraw wait
		VioScrollLock	Lock the screen

F	07H	VioScrollDn	Scroll down
F		VioScrollLf	Scroll left
F		VioScrollRt	Scroll right
F	06H	VioScrollUp	Scroll up
		VioScrUnLock	Unlock the screen
		VioSetANSI	Set ANSI mode on or off
		VioSetCp	Set video code page ID
F	02H	VioSetCurPos	Set cursor position
F	01H	VioSetCurType	Set cursor type
		VioSetFont	Set font
F	00H	VioSetMode	Set display mode
0BH		VioSetState	Set video state
F		VioShowBuf	Display logical buffer
F	09H	VioWrtCellStr	Write cell string
F	0AH	VioWrtCharStr	Write character string
F		VioWrtCharStrAttr	Write character string with attribute
F		VioWrtNAttr	Write N attributes
F		VioWrtNCell	Write N cells
F		VioWrtNChar	Write N characters
F	0EH	VioWrtTTY	Write a TTY string

F=Full family-mode support

Besides providing the equivalent of DOS BIOS-level video output functions, OS/2 also allows direct access to either the logical or the physical video buffers.

MOU

MOU sets parameters for mice that work with OS/2. Because OS/2 will eventually be graphics based, much as Apple's Macintosh now is, the mouse will be a much more important tool for the new system than it has been with DOS. Microsoft and IBM assume that many more applications will use mice in the future than do today, so they have provided a much richer set of mouse functions in OS/2 than DOS offered. OS/2's mouse API functions are listed in Table TK.

Table 2.2 OS/2 Mouse Functions

INT 33H		
EQUIV.	**NAME**	**DESCRIPTION**
	MouClose	Close mouse device
	MouDeRegister	Deregister a mouse subsystem
O1H	MouDrawPtr	Draw a pointer
	MouFlushQue	Flush mouse event queue
	MouGetDevStatus	Get mouse device status flags
	MouGetEventMask	Get mouse event mask
	MouGetHotKey	Get mouse hot key definition
	MouGetNumButtons	Get number of mouse buttons
	MouGetNumMickeys	Get number of mickeys per centimeter

	MouGetNumQueEl	Get number of mouse event queue elements
03H	MouGetPtrPos	Get mouse pointer position
	MouGetPtrShape	Get mouse pointer shape
	MouGetScaleFact	Get mouse scaling factors
	MouInitReal	Initialize real-mode mouse driver
00H	MouOpen	Open mouse device
05H06H	MouReadEventQue	Read mouse event queue
	MouRegister	Register a mouse subsystem
02H	MouRemovePtr	Remove mouse pointer from a screen area
	MouSetDevStatus	Set mouse device status flags
0CH	MouSetEventMask	Set mouse event mask
	MouSetHotKey	Set mouse hot key definition
04H	MouSetPtrPos	Set mouse pointer position
09H,0AH	MouSetPtrShape	Set mouse pointer shape
	MouSetScaleFact	Set mouse scaling factors
	MouShellInit	Initialize shell linkage
	MouSynch	Synchronize mouse subsystem

KBD

KBD handles keyboard interaction in OS/2. DOS had a keyboard buffer, but KBD is to that software construct as a Lear Jet is to a model T. DOS allowed applications to more or less have their way with the PC keyboard. The new hands-off approach embodied in KBD functions is one of the biggest shifts PC application developers will have to get used to in OS/2. The KBD functions are shown in Table TK.

Table 2.3 OS/2 Keyboard Functions

DOS		
FAMILY EQUIVALENT	**NAME**	**DESCRIPTION**
F INT 16H,00H	KbdnCharlin	Read character and scan code
	KbdClose	Close logical keyboard
	KbdDeRegister	Deregister Keyboard subsystem
	KbdFreeFocus	Free keyboard focus
F	KbdFlushBuffer	Flush keystroke buffer
	KbdGetFocus	Get keyboard focus
F INT 16H,00H	KbdGetStatus	Get keyboard status
	KbdGetCP	Get keyboard code page ID
	KbdOpen	Open logical keyboard
R INT 16H,00H	KbdPeek	Peek at character and scan code
	KbdRegister	Register keyboard subsystem
	KbdSetCP	Set keyboard code page ID
	KbdSetCustXt	Set custom translate table
	KbdSetFgnd	Set foreground keyboard priority
		Set keyboard status

	KbdSetStatus	
	KbdShellInit	Initialize shell
F INT 21H,0AH	KbdStringIn	Read character string
	KbdSynch	Synchronize keyboard access
	KbdXlate	Translate scan code

F=Full family-mode support

R= Restricted family-mode support

I/O

OS/2's I/O DLIB includes functions that more closely resemble the DOS API than other parts of OS/2. In a general sense you could say that the functionality of DOS has largely migrated into a single DLIB in OS/2. If, at some point in the future, the uses OS/2 is being put to require a different file system from that of DOS, this DLIB could be replaced without affecting the rest of the system.

Table 2.4 OS/2 File I/O Functions

INT 21H			
FAMILY	**EQUIV.**	**NAME**	**DESCRIPTION**
	ODH68H	DosBuffReset	Commit file's cache buffers
F	3BH	DosChDir	Change current directory
F	42H	DosChgFilePtr	Change file's read/write pointer
		DosCLIAcess	Request CLI/STI privilege
F	3EH	DosClose	Close file
F	41H	DosDelete	Delete file
F		DosDevConfig	Get device configuration
F	44H	DosDevIOCtl	Perform I/O control function
F	45H,46H	DosDupHandle	Duplicate file handle
R	5CH	DosFileLocks	Lock file section
R		DosFindClose	Close directory handle
R	4EH	DosFindFirst	Find first matching file
R	4FH	DosFindNext	Find next matching file
F		DosGetMessage	Get system message with variable text
F		DosInsMessage	Insert variable information into message
F	39H	DosMkDir	Create directory
F	56H	DosMove	Move file
F		DosNewSize	Change file size
R	3CH,#DH	DosOpen	Open file
		DosPhysicalDisk	Request disk partition information
F		DosPortAccess	Request I/O port access
F		DosPutMessage	Put message to specified file handle
F	47H	DosQCurDir	Query current directory
F	19H	DosQCurDisk	Query current disk
R		DosQFHandState	Query file handle state

F	57H	DosQFileInfo	Query file information
F	43H	DosQFileMode	Query file mode
F	36H	DosQFSInfo	Query file system information
		DosQHandType	Query file handle type
F	54H	DosQVerify	Query disk file verify setting
F	3FH	DosRead	Read file
		DosReadAsync	Read file ansynchronously
F	3AH	DosRmDir	Remove directory
F		DosSearchPath	Search path for file name
F	0EH	DosSelectDisk	Select default disk
R		DosSetFHAndState	Set file handle state
F	57H	DosSetFileInfo	Set file information
F	43H	DosSetFileMode	Set file mode
F		DosSetFsInfo	Set file system information
	67H	DosSetMaxFH	Set maximum file handles
	2EH	DosSetVerify	Set or reset verify switch
F	40H	DosWrite	Write file
		DosWriteAsync	Write file asynchronously

F = Full family-mode support

R = Restricted-mode support

Reflecting the similarity of the DOS and OS/2 file systems, this group of functions has the greatest percentage of equivalents in both operating systems.

These basic DLIBS will be expanded upon by Microsoft, IBM, other hardware companies and application developers. A CAD/CAM vendor might create a CAD graphics primitive DLIB to speed his program's performance. An information services vendor might create an optical disk DLIB to facilitate information retrieval from the platters. One of the great advantages of DLIBS is that they can be updated independent of the operating system as a whole. The API can be enhanced almost constantly simply by substituting new DLIBS for old ones or adding to the DLIB pool. The most inescapable example of this is the changing user interface of OS/2. In OS/2 version 1.0 the user interface is the Session Manager a character-based, menu driven screen system. In version 1.1 and all subsequent versions the interface is the Presentation Manager, a graphics-based icon driven screen system. This huge change in the look and operation of OS/2 is accomplished simply by installing new DLIBs, containing the Presentation Manager commands and modifying other systems to interact with Presentation Manager services. As other new ideas emerge--how about a voice driven user interface?--OS/2 can change to accommodate them with far greater ease than DOS could ever muster.

2.1.2 The Kernel

The kernel is OS/2's nerve center. It handles memory management, task scheduling, interprocess communication, file input-output chores, timer services, environment management and basic text messaging facilities. The kernel, in other words, is the landlord for the multi-tenant OS/2 environment. As software that spends most of its time managing internal resources and relatively little directly interacting with users or hardware, OS/2's kernel is much more like mainframe and large minicomputer

operating systems than it is like DOS. While command.com could be virtually swallowed up by powerful applications, the OS/2 kernel code always remains aloof from any single application. It serves the system's interests and the user's interest (as expressed through config.sys instructions and OS/2 user commands) far more than a particular program's. The kernel is of little direct importance to OS/2 managers, because it is so isolated deep within the operating system's confines. Even application programs interact with the kernel only through the DLIBS. So, except for deep tech systems programs, the kernel's value lies only in what it allows the layers on either side of it to provide developers and users.

2.1.3 Device Drivers

The DLL and DLIBS provide the interface between OS/2's kernel and applications. Device Drivers are the link between the kernel and underlying hardware. No hardware talks directly to OS/2's kernel. Everything from the system clock to a token ring interface board must be described for OS/2 by a device driver. Device drivers have been a standard part of recent versions of DOS, but the sensitivity of OS/2 to device drivers is far greater than in DOS. For instance, even though OS/2 is designed to run on both AT-clas machines and IBM's PS/2 micros, early versions of the software would not run on PS/2s. The device drivers containing the technical details on that system were included. Nor would OS/2 for ATs run on most AT-clones. However, MS-DOS and PC-DOS would run across all those variation swithough requiring special drivers for each. In the early months of OS/2 adoption, confusion over device drivers is likely to be public problem number one. Each new graphics board, memory add-on, network card will require a complete and bug free device driver of its very own before it will work with OS/2. Existing products will need new device drivers for the new OS, and in some cases may not be migratable into the promised land of protected mode. Device drivers are an essential ingredient for a multitasking system such as OS/2, because when many programs are sharing a system, none of them can have complete control of the hardware. That would wipe out the operation of all other programs in the machine. In OS/2 the programs can share the device drivers, but since one and only one driver has exclusive interaction with the hardware below it, no program can hog resources at the expense of others.

2.2 How OS/2 Works

The internal operation of OS/2 is radically different from that of DOS. In other to understand the new services this operating system can provide, it is essential to have at least a general familiarity with its strange new workings.

2.3 Multitasking

Multitasking is one of those computer concepts subject to numerous definitions, all different and all correct. Like the elephant of the five blind men, multitasking's description depends upon one's point of view. OS/2 has segregated the different levels of multitasking's meaning rather neatly. To the user multitasking means that a machine allows him to have available for possible use many options at one time. He can move among these serially as fast as he wishes, and can at times run multiple operations in the background with concentrating on a single operation in the foreground. To the application developer, multitasking means that his program can't dominate the user's or system's attention, and must conform to new rules of conduct with data, peripherals and other software. To the operating

system, multitasking demands a huge "middle management" of information reception, tracking and response, about the status of the hardware and all the software contending to use it, as well as a steady ear cocked toward in unpredictable demands of the user. For the hardware, OS/2 means the ability to slice performance time and capacity among a changeable, contentious mob of program and operating system demands. Multitasking means hardware must act like the old vaudeveille performer who kept dozens of plates spinning simultaneously on precarious stick perches. With its modular design, OS/2 provides discrete concepts and services to handle the multitasking needs of each of these computer constituencies.

2.3.1 How OS/2 handles multitasking.

Let's work from the chip up to understand the multitasking system of OS/2. In order to provide access to more than one application in extremely close time proximity, the microprocessor has be be able to switch its attention quickly from instructions for one program to those of another.

The chip needs help from operating system software to accomplish this "slicing" of its processing time. Software that helps the processor allocate its time is called a scheduler or dispatcher. The shift of chip focus from one set of instructions to another is called context switching. This name derives from the fact that the chip must switch away from the first job, without losing its context--the set of data and chip settings in place when processing stopped. If context isn't preserved, the chip can never switch back to job one, and multitasking becomes impossible.

Multitasking systems today use two different schemes to control chip scheduling:event driven and preemptive. An event driven scheduler, demands rigid operation and precise performance from all programs and initiates switches between them according to a predetermined patterns built into the applications. For instance, applications might give up control every time they asked for interaction with a peripheral and go to the end of the line of programs waiting for chip time. The goal is to require that applications pass off chip control frequently enough that none dominates and none suffers such massive performance degradation that it becomes useless. This works splendidly in a system with many short transactions requiring high efficacy, such as transaction processing. But put a calculation intensive program in such a system and it could hog the chip totally all afternoon.

A pre-emptive scheduler places greater control in the hands of the operating system. In these schemes, the operating system checks out what the chip is doing frequently, usually at a predetermined interrupt signal based upon ticks of the computer's internal clock. At each check, the system saves the context of what is on the chip, and then looks over a list of programs that are ready to run. The operating system can then pass control to the next program in line. In a pre-emptive system, applications cannot be certain when their processing will be interrupted and the operation of each program becomes inherently less efficient than in even-drive multitasking. But no one application can take over a pre-emptive system; everybody is forced to share brief periods of chip access.

In chips such as the Intel 80286, an additional element in scheduling is present--protection priority. Protected mode chips set up levels of protection for different kinds of jobs. The higher the protection level of the job, the higher the priority the job has in getting chip time and services. In OS/2, the kernel

software stands at the highest possible protection status. It gets priority access to the chip and can run any 80286 instruction. Other kinds of software stand at lower protection levels; they must wait for the kernel to get out of the way for chip access, and they are embargoed from using some chip instructions that might get in the way of safe multitasking. For example, OS/2 kernel can tell the chip to write to an input/output hardware port. No application can issue that instruction, since it might mess up the operation of another program in the system.

Put briefly, OS/2 is a pre-emptive, priority based multitasking environment. Pre emptive scheduling was chosen because it serves the user's quixotic needs better; regardless of what the user is doing, all active programs are performing useful work; none are running at absolute top speed, none are dead. Since humans operate mushc more slowly than computers, the speed drop off is probably not a significant impediment, while a comatose application which can't get any chip time easily could be. Priority is used to enhance OS/2 control over hardware and to make multitasking as controllable as possible.

2.5 Table of OS/2 Multitasking Functions

INT. 21			
FAMILY	**EQUIV.**	**NAME**	**DESCRIPTION**
		DosCreateThread	Create thread
R	4DH	DosCWait	Wait for child process termination
		DosEnterCritSec	Enter critical section
R	4BH	DocExecPgm	Execute program
R	00H,4CH	DosExit	Exit program
		DosExitCritSec	Exit critical section
		DosExitList	Specify exit function list
		DosGetPid	Get process identification
		DosGetPrty	Get process priority
		DosKillProcess	Kill process
		DosMonClose	Disconnect device monitor
		DosMonOpen	Connect device monitor
		DosMonRead	Read from monitor structure
		DosMonReg	Register monitor buffers
		DosMonWrite	Write to monitor structure
		DosPtrace	Process debugging trace
		DosResumeThread	Restart thread
		DosSelectSession	Select foreground session
		DosSetPrty	Set process priority
		DosSetSession	Set session status
		DosSleep	Delay process execution
		DosStartSession	Start session
		DosStopSession	Stop session
		DosSuspendThread	Suspend thread execution
		DosSystemService	Request special process services
		DosTimerAsync	Start asynchronous timer
		DosTimerStart	Start interval timer

		DosTimerStop	Stop interval timer

R = Restricted family-mode support

Multitasking is one of the major attractions of OS/2. Functions are provided to control tasks at three integral levels: sessions, processes, and threads.

Upon this technical foundation, OS/2 has built a construction of conceptual tools that programmers can use to take advantage of multitasking.

2.3.2 Processes

In the old days, when micros were small and life was simple, all an operating system had to do was deliver an application to the processor and go out for a cup of coffee. The application took over control until the user was through with it, at which point the application closed itself and called the operating system back into play from down the hall. With multitasking, however, everything goes up an exponential level in internal complexity. All applications are subject to sudden interruption right in the middle of their operation.

In OS/2's preemptive system nobody gets more than a short tet-a-tet with the processor. This means that the operating system can't just drop off an application and take off. It has to be able to send the chip small chunks of the application, just enough to be processed in a pre-emptive cycle, and then pass the results of these discrete steps along to other program chunks that are waiting their turn to run. Consequently, OS/2 has to offer application programmers definitions for how they can break their code into easy-to-swallow chunks that the operating system's chip scheduler can understand.

The first step OS/2 takes away from the old idea of what an application program is and toward a multitasking model comes in the notion of processes. In OS/2 a process consists of an executing program and all the resources required to run it These would include memory areas, description tables for chip status (to encure context maintenance) and other system support. An application might consist of one process or of many. An on-screen calculator program, for instance, may be just a single process.

The process begins when the program is called from the command line and ends when the user exits. A word processor, on the other hand, might have a backbone process for opening and closing files, formatting and other housekeeping and then separate processes for printing, mail-merge, spell checking and so on. When an application consists of numerous processes, the backbone program is called the parent process and all the other processes it contains are called child processes. OS/2 initiates the parent process, but the parent can control its own children. Processes represent a much more modular, flexible and elegant method for organizing programs than DOS required. However, processes still don't achieve the micro-chunk break up multitasking requires. For that, OS/2 tasks us within the processes, where we find threads.

2.3.3 Threads

Each OS/2 process contains one or more threads. A thread is a defined execution path within a process. The OS/2 chip scheduler, knows nothing about processes and only schedules threads. In other words, threads are the mini-chunks of program that OS/2 actually multitasks.

Every OS/2 process has a primary thread that is loaded when the process is called. This thread can then call other threads from within the process. Threads get just enough independent systems definition from OS/2 to make them functional, but most of their operation is controlled by their parent process and shared among many threads .OS/2 can support up to 255 separate threads at any one time for all active processes. The number of threads that can be sustained within any single process is set in the OS/2 config.sys file, which is discussed in Chapter 5.

At any moment, a thread is either running or blocked. Only one thread actually runs at a time on the microprocessor. The priority threads have for getting CPU time is divided into three classes, each with 32 priority levels. The highest thread priority class is time-critical class. All threads in this class are higher than any in the other two classes. A thread from a high speed data link is an example of a time-critical program chunk; if it doesn't execute now, the data coming in could be lost. Regular or normal class is the second priority group. Most application program threads and OS/2 keyboard commands reside here. OS/2 has included some tricks for squeezing performance from regular level threads. These are controlled by new commands in the config.sys file and are discussed in Chapter 5.

At the bottom of the heap stand idle-time threads. These program chunks only get chip time if no other threads are requesting it. Code that might fall in here would be a print spooler or an automatic disk backup utility or a program set up in OS/2 to run "in the background," without accessing the screen. Whatever programs are loaded, whatever processes are running, deep down inside all OS/2 is doing is gathering and scheduling threads.

2.3.4 Sessions/Screen Groups

The OS/2 user, though, doesn't really want or need to know about threads, processes and other techie mumbo jumbo. The multitasking the user sees in OS/2 comes via Sessions, which are also sometimes called Screen Groups. A session consists of one or more processes that share the same display screen and keyboard. When a session is created, which will be discussed fully in Chapter 8, OS/2 creates a virtual screen and a virtual keyboard for any processes it contains. When the user wants to interact with that session, OS/2 takes the virtual devices and activates them. The user can switch screens, keyboards and processes with the mere flick of a couple of hot keys. (The specific methods for calling and switching between sessions will change somewhat with various versions of OS/2. See Chapter 8 for more on the care and handling of screen groups.)

OS/2 can support 12 screen groups at once. One of these is automatically the province of the standard user interface. In OS/2 version 1.0 this is the Session Manager screen; in future versions it will be the home of the Presentation Manager. When a user opts to load the OS/2 real mode DOS compatibility environment, the second screen group is reserved for DOS programs. The DOS screen group carries a heavy load of performance baggage other screen groups do not. For instance, when a user shifts from one protected mode screen group to another, all other programs automatically keep running in the background. But whenever a user switches to the real mode screen, all protected mode screen groups--including time critical stuff like communications-- is frozen dead.

Protected mode processes only restart when the real mode screen has been left. (In OS/2, by the way, real mode programs will not run in background.)The other ten OS/2 screen groups can each have their own command processors, user interfaces and processes. If OS/2's cmd.exe is used as the command processor for these sessions, then each of them can only support a single application per screen. OS/2 can only run its windowing interface in one screen group, not all. But future command processor options may make multiple application screens possible in all sessions.

A program can be run in OS/2 without access to the screen via the detach command, which is explained more fully later in this chapter. Essentially, detach places a program on permanent background status. This is not something that would be done with most applications, since it would result in chaotic screen displays. Detach would work best with programs that talk to other programs, such as communication port monitors or compilers. Detach may also be one means for running pop-up programs in OS/2, which is discussed in Chapter 3.

Managing screen groups will become a major consideration of working with OS/2. As DOS users had to be educated in simple file handling and directories, then subdirectories, then network directories, they will now need to be aided in the logical creation of screens, keyboards and programs.

Figure 2.2 BUILDING OS/2 APPLICATIONS ONE THREAD AT A TIME

The principal operating units of applications in the multitasking OS/2 environment are called processes. These are units of program operation along with all the computing resources required to run them. Each process is made up of numerous threads, small processing units that can be handled by the processor in one brief moment before the OS/2 multitasking scheduler turns its attention to the next task. If, in the example above process A and B are both active, OS/2 might run thread A1, then B1, A2, then B2; Process A is complete so thread B3 would come next. When B is complete, OS/2 might turn its attention to process C, which is running in the background and run thread C1.

The user sees OS/2 processes gathered together in virtual screens called screen groups or sessions. The OS/2 session manager controls the access of these groups to the physical display screen.

2.3.5 How are multitasking applications designed?

Another consideration managers should keep in mind here is that, while file formats and commands will almost certainly migrate easily from DOS to OS/2 versions of applications you use, the internal structure of the programs will be changing significantly. Early in OS/2's development, some programs will use a basic core set of OS/2 functions called the Family API to merely shift their existing product to protected mode operations. But in conjunction with version 1.1 and the Presentation Manager you will begin to see true OS/2 versions of applications. These will be much different creatures internally from the DOS programs we use today. Managers who have become old hands at dBase internals, Xywrite formats and other valuable technical arcana of the products they use will must be prepared to start learning a whole new technical lexicon for the OS/2 world.

2.4 Interprocess Communication

One of the great technical challenges of a multitasking environment is figuring out how Thread A can tell Thread B what it is up to, when neither thread knows beans about the other. Obviously, if both threads are demanding control of the printer, oblivious of the other's needs and desires, and the operating system has no way to mediate their dispute, inefficiency runs rampant. OS/2 includes interprocess communication methods user and developers can use that allow the operating environment to let applications share resources or keep them out of each other's way.

2.4.1 Clipboard and Dynamic Data Exchange

Users have two methods at their disposal in OS/2 for moving data from one program to another. Both are tied in with the OS/2 user interface and are discussed more fully in Chapter 8.The Clipboard is a memory area available through the OS/2 screen into which marked sections of screen data can be stored. Information can be stored into the clipboard from one application on screen and picked up into another application. In this way a graphical can be drawn in a graphics package, stored to the clipboard and then brought up inside a word processor based report. Dynamic Data Exchange (DDE) is a Microsoft developed protocol for automating data flow between applications. DDE, which has been included in Microsoft Windows for some time, has not been widely used. But it could become more popular as part of OS/2. With DDE a user could automatically have certain data ported say from a spreadsheet to the corresponding text report, so that numbers in both update in tandem.

2.4.2 Pipes

These rather limited interprocess tools aren't the meat of OS/2, however. The real power lies in tools within the system designed to let developers integrate interprocess cooperation deep within their programs.

A pipe is a software construct (which is similar in concept to, but not the same as the OS/2 Pipe command modifier) that allows one thread to send a stream of information to another thread through a neutral memory location. The first thread writes to the pipe; the second thread picks up from the pipe. Pipes will be found often in highly synchronized communication flows such as those between OS/2 parent and child processes.

2.4.3 Queues

A queue is another OS/2 construct in memory, but unlike pipes, the queue doesn't flow. It's more like a post office box or public notice board. One thread can leave a message in the queue and other threads can then read it. Queues are designed for use within families of threads; a process owns the queue, and only threads of that process get full use of its services. For efficiency, a queue doesn't actually get data messages. Instead it receives a notation of a memory address where the message is stored. This means that a large message can be stored in a queue without taking the time of copying it.

Figure 2.3 HOW QUEUES WORK

One of the principal benefits of OS/2 queues, especially for large interprocess messages, is that the information isn't copied between processes. The receiving process opens the queue when it desires information from other processes. Then, any other process can open the queues by name and leave

messages there. The messages consist of a 16-bit number, whose meaning is defined within the application and a pointed to a shared segment where the message itself is stored.

2.4.4 Shared Memory

Under certain circumstances two threads can share access to a single memory location. For example, the location of the message noted in a queue becomes a shared memory segment, since more than one thread can access it. Only data in memory, not program code, can be shared in OS/2. There is a 64 Kbyte limit on shared memory size. Shared memory would be useful in situations where a number of threads need access to a single database. For instance if the content data of a spreadsheet is stored in shared memory, the graphing, printing, calculating and other processes of the program can allget at it without making their own time and memory wasting copies.

2.6 OS/2 Interprocess Communication Functions

Fuctions	Description
SEMAPHORES	
DocCreateSem	Create a system semaphore.
DosOpenSem	Open an existing system semaphore.
DosCloseSem	Close a system semaphore.
DosSemRequest	Obtain ownership of a semaphore.
DosSemSet	Set an owned semaphore.
DosSemClear	Unconditionally clear (or release) a semaphore.
DosSemSetWait	Set a semaphote and wait for it to be cleared (blocks the current thread until the next DosSemClear occurs).
DosSemWait	Wait for a semaphore to be cleared.
DosMuxSemWait	Wait for a number of semaphores to be cleared.
PIPES	
DosMakePipe	Create a pipe.
SHARED MEMORY	
DosAllocShrSeg	Allocate a shared memory segment (named) to a process.
DosGetShareSeg	Enable a process to access a named shared memory segment allocated by another process.
DosAllocSeg	Allocate a segment of memory.
DosGiveSeg	Give access to a segment.

QUEUES	
DosCreateQueue	Create a queue.
DosOpenQueue	Open a queue for the current process.
DosWriteQueue	Add an element to a queue.
DosReadQueue	Read and remove an element from a queue.
DosPeekQueue	Retrieve but do not remove an element from a queue.
DosQueryQueue	Return the size (number of elements) in a queue.
DosPurgeQueue	Purge a queue of all elements.
DosCloseQueue	Close a queue.
SIGNALS	
DosSetSigHandler	Define a routine to handle a signal.
DosFlagProcess	Set a process external event flag in another process.
DosHoldSignal	Disable or enable signal processing for current process.

2.4.5 Semaphores

Semaphores are software badges of ownership over some system resource. When, for instance, a thread seizes the use of a bit of data in memory or the use of a system table, the thread "owns" the semaphore for that resource. No other thread can get at the resource, until the semaphore is no longer owned. Semaphores can protect sensitive data or program code from corruption. If a thread needs to read a database record, it doesn't want to be interrupted until it is done. Even an update to that record should wait until the reading process is complete. The semaphore provides this protection.

2.4.6 Signals

In DOS the key combinations Ctrl-Alt-Del and Ctrl-Break were signals the operating system always listened for. When they occurred, they could override whatever else was going on. In OS/2, signals can be generated by specific key combinations, functions and constructs within processes. As with DOS, the principal use of OS.2 signals is to note an external intervention. But they can also be used in reaction to a program condition that requires a fast, critical special response. For example, in a system where memory is crammed full, a communication process might have a signal set up to go off whenever if received notice of an incoming message that could not be captured without freeing more RAM. The signal suspends existing operations and flips into play an emergency file closing, RAM opening routine. When the message has been received and stored, the signal can return the system's status to what is was previously.

2.5 Memory Management

Memory is another ticklish subject for a multitasking operating system like OS/2. In a single program environment, the operating system and its hardware snatch a specified chunk of memory for their own uses. Everything else becomes open season to the installed application. It can shovel data and code into the memory addressed pretty much at will. But in a multitasking environment, imagine the chaos in Program X sends vital code to memory address 0, while Program Y, blissful in its ignorance, stores a temporary file in the same location. Program X crashes, the temp file is lost and the whole machine may die. Obviously, this can't be tolerated. As a result, crafty memory management is an integral part of OS/2's design. While the technical details of how OS/2 handles memory are interesting and important for understanding the product, the essential task here is quite simple--OS/2 is set up so that applications only talk to tables of memory addresses, never to memory itself. Only OS/2 knows where each number in the memory tables is actually stored in physical memory. This mode of memory management is called virtual addressing.

Table 2.7 OS/2 Memory-management Functions

INT. 21H			
FAMILY	**EQUIV**	**NAME**	**DESCRIPTION**
R		DosAllocHuge	Allocate huge memory block
R	48H	DosAllocSeg	Allocate segment
		DosAllocShrSeg	Allocate shared segment
F		DosCreateCSAlias	Create code segment alias for data segment
F	49H	DosFreeSeg	Free segment
F		DosGetHugeShift	Get shift parameter for huge blocks
		DosGetInfoSeg	Get address of system variable segment
		DosGetSeg	Get access to segment
		DosGetShrSeg	Get access to shared segment
		DosGiveSeg	Give access to segment
		DosLockSeg	Lock segment in memory
		DosMemAvail	Get size of largest free memory block
R		DosReallocHuge	Change size of huge memory block
R	4AH	DosReallocSeg	Change size of segment
F		DosSubAlloc	Allocate memory within segment
F		DosSubFree	Free memory within segment
F		DosSubSet	Set memory sub allocation size
		DosUnlockSeg	Unlock segment

F = Full family-mode support

R = Restricted family-mode support

Besides allocating and freeing blocks of memory, these functions provide the various levels of memory sharing available in a protected-mode environment

2.5.1 Virtual Addressing

As we mentioned earlier, the protection scheme of the 80286 microprocessor requires that applications talk with the operating system about memory locations, never directly to RAM hardware. The program

talks to a "virtual" address maintained by the operating system, which keeps the actual physical locations of memory in secure, discrete tables.

2.5.2 GDTs and LDTs

OS/2 sets up two kinds of tables for memory address storage. For the system as a whole, OS/2 establishes a Global Descriptor Table (GDT). Each task in the system also receives a Local Descriptor Table. The memory addresses available to the task are defined by combining the two tables. These results in a unique table for each task in OS/2, which means a unique address area for each. Each OS/2 memory table can reach a maximum size of 64KB, in the form of 8152 8-byte descriptors, each of a memory segment of 64KB. So, the two tables for each task, produce two x 8152 x64 kb. This is just about 1 gigabyte of potential memory. However, the 80286 can only handle 16 megabytes of installed memory at any particular moment. So the 1 gigabyte "virtual" memory space must be built up of swappable 16 megabyte units.

2.5.3 Segments

As noted above, OS/2's memory tables can define no more than 64kb of memory in one address block. This 64kb memory segment represents the largest contiguous memory area OS/2 can support. Such a set up is known as a segmented memory model. It means that programs for OS/2 must be loadable in 64kb modules or less. When applications are called, they tell OS/2 what memory segments must be loaded into RAM for the program to function. Then, as the program runs, its processes can order OS/2 to swap segments in RAM with others still on disk. As more and more applications are simultaneously installed within OS/2, more and more segments are left on disk. The OS/2 memory management functions within the kernel oversee the efficient swapping of segments from disk for all applications active in OS/2 muti-tasking environment. Some of the parameters of this disk-memory swapping can be set in the config.sys file, as is described more fully in Chapter 5.

2.6 The sum of the parts

The sum of all these OS/2 capabilities is an operating system that is something like an iceberg--most of its bulk lies beneath the surface. To the user, OS/2 will not seem remarkably different than DOS, at least in terms of concepts and commands, as well will explore more fully in the next chapter. However, beneath the user shell lies a system infinitely more complex and rich than DOS could ever imagine to be.

Table 2.8 OS/2 Miscellaneous Functions

INT 21H			
FAMILY	**EQUIV.**	**NAME**	**DESCRIPTION**
F		BadDynLink	Called on bad dynamic link
F		DosBeep	
F		DosCaseMap	Case-map character string
R	59H	DosErrClass	Classify error code
R		DosError	Enable hard error processing
		DosFreeModule	Free dynamic link module
F		DosGetCollate	Get collating sequence table
	66H	DosGetCp	Get process code page

R	38H	DosGetCtryInfo	Get country-dependent information
F	2AH,2CH	DosGetDateTime	Get system date and time
	65H	DosGetDBCSEv	Get dual-byte character set
F		DosGetEnv	Get environment string address
F		DosGetMachineMode	Get processor mode
		DosGetModHandle	Get dynlink module handle
		DosGetModName	Get dynlink module name
		DosGetProcAddr	Get dynlink proc address
F	30H	DosGetVersion	Get OS version number
		DosLoadModule	Load dynamic link module
F		DosScanEnv	Scan environment strings
	66H	DosSetCp	Set process code page
F	2BH,2DH	DosSetDateTime	Set system date and time
F	25H	DosSetVec	Specify exemption handler

F = full family-mode support

R = restricted family-mode support

Some of the most useful functions do not fit into any previous groups. Except for dynamic link functions, most are DOS functions equivalents or extensions.

It is this rich heart of application programming tools and modular system structure that will determine how important OS/2 will become for corporate computing. The chances are excellent that it will become the most important desktop operating environment in history, far eclipsing DOS. Managers and users, the aphorism goes, don't buy operating systems; they buy the applications that run on them. By being able to deliver more powerful, flexible and easy to use applications via micros than ever before, OS/2 should soon become the standard platform for personal computer development. To paraphrase, one of those lovely gentlemen of Watergate fame: If you grab them by their applications, their hearts and minds will follow.

Chapter 3. Change and Challenges of the OS/2 User Environment

Getting the most from OS/2 will require managers to get used to new properties for familiar DOS elements and new possibilities from abilities native to OS/2. While much remains the same, nothing is really quite as it was in this new computing universe. Starting from OS/2's, most familiar component, it's not-quite-DOS compatibility environment and working toward it's radically new user interface, in this chapter we will trace the perplexities of the OS/2 user environment.

3.1 Types of OS/2 commands

OS/2 retains the familiar DOS user command set and adds to it a wide array of new capabilities. In addition to new commands, OS/2, in protected mode, expands upon the DOS command line characters. Making it easier and more powerful to group commands on a single line or interact between commands.

A good way to begin understanding OS/2 command line operations is to break OS/2 commands down into two large families: internal commands and external commands.

3.1.1 Internal Commands

OS/2 comes with two separate command processing files, one for the real mode, one for the protected mode.

COMMAND.COM, the DOS mode command processor should be familiar from DOS. CMD.EXE, is the new command processor for the protected mode.

Internal commands are those instructions that are built into either or both of these command processors. Although internal commands can be entered independently at the command line, they aren't individual files. The well-known DOS DIR command, for example, is an internal command. There is no file called DIR.EXE or DIR.COM.

Because they are built into the command processor, internal commands cannot be readily expanded or modified by managers or users. They are fixed. And, since they are automatically loaded into memory along with the command processor, they execute immediately.

In OS/2, DOS mode internal commands are built into COMMAND.COM; protected mode internal commands are built into CMD.EXE.

The OS/2 internal commands are:

break	echo	ren
chcp	exit	rmdir
chdir	for	set
cls	goto	shift
copy	if	time

date mkdir type

del path ver

detach pause verify

dir prompt vol

dpath rem

3.1.2 External Commands

External commands are those instructions supported by independent files. The format command is supported by the file FORMAT.COM, which can be seen on any DOS directory. As individual files, external commands can be added or upgraded without altering the command processors. External commands can also be added by the user when he builds his own command files, which have the extensions .COM or .EXE.

When an external command is typed at the command line, OS/2 looks for a file with that name to run. OS/2 first looks for a .COM file, then a .EXE file and the a batch file (.BAT in DOS mode, .CMD in protected mode). This means that a batch file named FRED.BAT will never run in a directory where a file named FRED.EXE exists, because OS/2 will always run the .EXE file first. The batch file must be renamed or moved to a different directory and invoked with a path, which we will discuss in more detail later.

OS/2/s external commands are:

ansi fdisk patch

append find print

assign format recover

attrib graftabl replace

backup helpmsg restore

chkdsk join sort

cmd keyb subst

command label sys

comp mode tree

diskcomp more xcopy

diskcopy

3.1.3 Internal Commands and Pathnames

A pathname is an addition to a command that allows it to search a selected set of directories or subdirectories.

Since internal commands are part of the command processor, they are always in memory and don't need to be called from disk. So, pathnames are not important in calling internal OS/2 commands.

But pathnames can aid the operation of four internal commands: COPY,DIR,DEL AND TYPE. These well-known commands have long been part of DOS and rely heavily on pathnames because they find and operate on files or groups of files. Paths operate with these commands identically in OS/2 with the way they have in DOS.

3.1.4 External Commands and Pathnames

As separate files, external commands are called from disk whenever you type them at the command line. OS/2 will search for these files first in your current working directory and then according to instructions you have given with the PATH command. The PATH command is part of the batch files that OS/2 automatically loads when the computer is turned on--AUTEXEC.BAT in DOS mode and STARTUP.CMD in protected mode. These files and the use of the path command are discussed in more detail in Chapter 5.

Since external commands are file based and sought out by paths, its a good practice to keep them all in one directory. The OS/2 installation procedure, also discussed in Chapter 5, automatically does this for you.

3.2 DOS MODE COMPATIBILITY: NOT REALLY DOS

Your first adjustment to the new micro world will be coping with the DOS mode of OS/2. While many of your DOS programs will run without any problems, the DOS mode is not completely 100 percent "pure" DOS, but emulation and close cousin.

IBM and Microsoft have stated that most "well-behaved" DOS programs will run in the DOS mode of OS/2. Some of your old DOS applications won't run at all, while others will take on some interesting new quirks. Of course, no software publisher will admit to ill-behaviour, so the only sure method is to test all your DOS applications under DOS mode before you are satisfied they will run.

Table 3.1 DOS Features in OS/2 Compatibility Mode

Supported	Not Supported
File Sharing (always in effect)	DOS 3.2 Network features.
Documented DOS services	Most undocumented DOS services.
ROM data area	Reprogramming 8259 interrupt vectors.
ROM BIOS interrupt services	Direct calls to ROM BIOS.
Hardware interrupts	Realtime clock interrupt.
VDI and CON device drivers	Block device drivers.
Spooler interrupt (INT 28H)	Some functions of the INT 24H

	Critical Error Handler.
Reprogramming the 8253 clock/timer	Reprogramming the disk controller.
Reprogramming the COM ports	Reprogramming the DMA controllers.

Many of you use a variety of terminate-and-stay-resident programs, such as Sidekick. Under OS/2, there is no such thing as a TSR program in the DOS sense, because all protected mode programs can run in the background and "pop-up" when you want them to, merely by switching to their screen group. Automatic popping-up within a protected mode screen group is possible, but technically far more daunting than in DOS. See Figure 3.1 for an idea of how it is done.

Figure 3.1 POP-UP PROGRAMS IN PROTECTED MODE

In protected mode, no program can get at hardware as is possible in the real mode world of DOS. Terminate and stay resident programs in DOS accomplish their quick pop-ups on the screen by studying keystrokes as they come from the keyboard, looking for their activating combination. This can't be done directly in OS/2, since the keyboard, like all other hardware is sealed away from applications. To accomplish a pop-up program within a single screen group in OS/2 requires construction of a keyboard device monitor for each application. Keystrokes are copied by OS/2 into the monitor areas and read there by the applications. When an application sees the proper key combination it can come alive. Another difference, though: the application in protected mode must ask permission from OS/2 for a piece of screen. It can't simply blast its way into view via interrupts, as it would in DOS.

Up to 12 programs can execute together, although only a single DOS application can run in the DOS mode. We go over the implications of this in Chapter 6 in more detail.

Sidekick does run in the DOS mode, however, so if your software is at least as well-behaved as Sidekick they will run. However, many vendors will probably rewrite their TSR programs to run in the protected mode, so life might actually be easier to manage these applications under OS/2 than under DOS.

However, the DOS mode is not completely a picnic. There are some memory differences between using pure DOS 3.3 and running in the DOS mode of OS/2. Some of the OS/2 commands work differently between DOS and protected modes, and some commands work differently between the DOS mode and DOS 3.3.

Finally, your communications software and serial devices will require some new tricks to work in the DOS mode. Let's look at each of these issues.

3.2.1 Memory differences between DOS 3.3 and DOS mode

OS/2's DOS mode provides for much less memory than the full 640 k bytes possible. This is because of the way OS/2 is constructed and the rules under which the DOS mode size is allocated.

Figure 3.2 ALLOCATION OF MEMORY BY OS/2

OS/2 splits its 16 megabyte address space between real mode and protected mode. The real mode address space is still 1 megabyte, as it was in DOS, but the usable RAM area is actually less than the 640 kilobytes possible with DOS. This is because all device drivers for peripherals must be located in DOS mode memory so the devices can be accessed by programs from either mode.

OS/2 splits its available memory between the DOS and protected modes. Up to 12 programs may share the protected mode RAM space, but this space is completely segregated from the space used by the single DOS application. And any memory that is used by the DOS mode is subtracted from the total available memory installed in your machine.

A good analogy is the duplex apartment. If the family living upstairs wants to add on another bedroom, they are going to have to take room away from the family living downstairs. Same with OS/2 and memory. If you want more memory for DOS mode applications, you will have to take away memory from that available for protected mode applications.

There is a second problem with DOS mode memory usage, and that concerns how OS/2 loads its device drivers.

OS/2 needs to load all of its device drivers into the lower memory space. This is because the drivers are used by sides of your system, the DOS side and the protected mode side. You wouldn't want OS/2 to load these drivers into high memory, i.e., the memory above 1 M bytes. This is because this memory is only available to the protected mode, so no DOS applications could make use of the drivers to access any system resources, a serious handicap indeed.

These drivers take up memory, and take memory away from the potential 640 k bytes available to run applications. For example, the minimum five drivers needed by version 1.0 of OS/2 Standard Edition takes up about 60 k bytes of RAM. If you need drivers for a mouse and for a serial port, that takes up another 30 k bytes and DOS itself takes another 30 k bytes. This leaves you a maximum of about 520 k bytes of RAM left for applications.

But wait, there is yet another constraint. OS/2 has some fairly strict rules in terms of allocating the size of the DOS memory space. We will get into the specifics in Chapter 5 when we discuss the RMSIZE configuration command. However, what is significant for this discussion is that the DOS size is based on how much RAM is installed on your system board.

Even if your machine has megabytes of RAM installed, what is critical for the DOS size is what is actually socketed on the system board? IBM sometime refers to this as base memory size, and you can either run the AT Advanced Diagnostics or open up your machine and count the chips on the system board to find out what this number is.

If your computer has less than 640 k of RAM installed on its system board (such as the older models 239 and 99 IBM ATs and most Compaqs), you will not be able to get anywhere near the full complement of 640 k of memory to run DOS applications. This means if you only have 512 k on your system board to

begin with, you are left with less than 400 k of usable memory after you load DOS and the device drivers.

The moral of this discussion on DOS memory size is this: If you have a large spreadsheet or database that you can just barely squeeze into your 640 k DOS machine today, chances are very likely that you won't be able to run this spreadsheet or database in OS/2's DOS mode.

3.2.2 Command usage between DOS and protected modes

There are many differences in command usage between the two modes of operation in OS/2 and between OS/2 and DOS. There are OS/2 commands that operate differently between DOS and protected modes, or have different syntax. There are commands new to OS/2 which only run in the protected mode, or only run in the DOS mode. We cover all three types briefly in this section, which assumes that you have a working familiarity with the existing DOS command set. A complete guide to OS/2 commands is included in Appendix A.

One of the first and foremost differences in OS/2 commands centers on those commands which are part of configuring your system. Because there are numerous changes to these commands, we cover them separately in Chapter 5.

There are other commands which have different syntax when they operate in DOS or protected modes. These are:

CHKDSK: This command displays the amount of free memory available. When run in DOS mode it displays free memory in terms of RAM and disk storage. In protected mode, only disk storage is displayed.

DIR: More than one path can be specified when this command is run in the protected mode. For example, in the protected mode you could issue the following directory command:

```
DIR \OS2 \123 \DBASE
```

This would result in OS/2 reporting on the directory contents of those three directories, in the order you specified. In DOS mode, only one directory at a time can be reported, similar to what currently is available under DOS 3.3.

DEL: More than one path can be specified when this command is run in the protected mode.

HELP, HELPMSG: These commands invoke the help facility in OS/2. The commands work in both DOS and protected modes of OS/2, but there are different help messages and explanations displayed depending on which mode you are in when you invoke the commands.

MD, RD: More than one path can be specified when this command is run in the protected mode.

MODE: This command has some significant changes between DOS and protected modes. We discuss this completely in section 3.1.4 below.

TYPE: More than one path can be specified when this command is run in the protected mode.

VOL: More than one path can be specified when this command is run in the protected mode.

There are some commands which only run in DOS mode:

APPEND: This command tells DOS where to look for data files outside the current directory. OS/2 uses another command, DPATH, for this function in protected mode.

ASSIGN: This command assigns a drive letter to another disk drive, and is not supported in the protected mode.

COMMAND: This starts the DOS command processor. OS/2's protected mode uses its own command processor, called CMD.

EDLIN: The line editor only runs in DOS mode.

GRAFTABL: This allows the extended character set to be displayed. OS/2 uses its own graphics device drivers. [true?]

JOIN: Joins a disk drive letter to a specific path. There is no equivalent in protected mode. [true?]

SETCOM40: Sets the communications port parameters. OS/2 uses the MODE command and its communications device drivers for this task.

SUBST: This command substitutes a drive letter for a directory name. There is no equivalent in protected mode. [true?]

There are also some new commands which only run in the protected mode of OS/2:

ANSI: This command sets the support for the extended ANSI keyboard and screen characters. This command can be placed in a batch startup file, with syntax ANSI=ON. In DOS mode, ANSI support was set with a device driver in CONFIG.SYS. If you want to support ANSI in both modes, you will need both commands.

CMD: The protected mode command processor, the equivalent of the DOS mode's COMMAND.

CREATEDD, TRACE, and TRACEFMT: This commands are used for diagnostics and system traces and are not part of usual operations.

DETACH: This command starts up a non-interactive process fully in a background screen group. This command could be used to start up utility programs that don't require any user input.

DPATH: This locates data files outside the current directory and is the protected equivalent of the APPEND command.

ENDLOCAL, EXTPROC, and SETLOCAL: These are three new protected mode batch commands. SETLOCAL sets the drive, directory, and other environment variables that are local to a particular batch

file where the command was issued. The global variables are restored after the ENDLOCAL batch command is issued. EXTPROC defines your own external batch file processor, instead of using the standard CMD processor of OS/2.

FDISK: The hard disk partitioning utility only runs in protected mode of OS/2.

KEYB: This command is used to specify a non-U.S. keyboard layout and only runs in protected mode under OS/2.

SPOOL: This command starts up the protected mode print spooler. All print requests are intercepted by the spooler and saved to disk. This ensures that several print jobs, sent to printer from different protected mode programs, can remain distinct from each other.

START: This starts a protected mode program.

A third set of OS/2 commands operate in both modes, but run into problems when you try to issue them across a computer network. When you try to run them over a network you receive an error message. Network sensitive commands are:

- chkdsk
- diskcomp
- diskcopy
- fdisk
- format
- label
- recover
- subst
- sys

3.2.3 Command usage between DOS mode and DOS 3.3

Several commands have different usage between DOS mode in OS/2 and DOS 3.3. The great majority of DOS commands are unchanged running under the DOS mode of OS/2. However, a few commands have slightly different syntax, and some commands are no longer supported under DOS mode.

Here is a summary of the changes in command usage:

APPEND: OS/2's DOS mode doesn't support the /X parameter, which was used to search and find files before looking in the current directory [true?].

COMP: Under OS/2, two files can be compared even if they have different sizes. DOS 3.3 required both files to be the same size.

DISKCOMP: OS/2's DOS mode does not need either the /1 or /8 options, since the command automatically checks for the number of sides and sectors when it does its comparing.

DISKCOPY: OS/2's DOS mode does not need the /1 option, since the command automatically determines the number of sides when it does its disk copying.

FORMAT: The options /1. /8, and /B options are not supported by the DOS mode. The /S option will create a system diskette for OS/2, even when it is run under DOS mode.

LABEL: OS/2's DOS mode does not support deleting volume labels.

MODE: This command has a variety of differences under OS/2's DOS mode. See section 3.1.4 for further details.

PRINT: The options /U, /M, /S, /P, and /Q are not supported in either the DOS or protected modes of OS/2.

SYS: This command will always create a system image for OS/2's command files, even if it is run from the DOS mode. You also cannot transfer system files to a diskette with any existing data; system diskettes must be a newly formatted diskette.

XCOPY: The /W option is not supported in either DOS or protected modes under OS/2.

EXE2BIN, FASTOPEN, FDISK, GRAPHICS, FILES (in CONFIG.SYS), KEYB, LASTDRIVE, NLSFUNC, SELECT, SHARE, and CTTY commands are not supported in the DOS mode.

3.2.4 Using the MODE command

The MODE command has been substantially changed in OS/2. It has several new features, and several older DOS 3.3 features are no longer supported.

CODEPAGE: No longer supported are the commands to set code pages for non-U.S. users. These commands are now incorporated into the CONFIG.SYS commands DEVINFO and CODEPAGE. See Chapter 5 for further details on how to configure your system with these commands.

PRINTERS: There are certain changes in the MODE command concerning support for serial printers. Under DOS 3.3, you used a "MODE LPT1=COM1" command to redirect printer output to a serial printer. You also used a ",P" option on the MODE COM command in DOS 3.3 to support for continuous retries on time-out errors. If you wanted to stop the continuous retry, you needed to type a CTRL-BREAK.

Under OS/2, the SPOOL command supports both parallel and serial printers, and replaces the DOS 3.3 redirection commands using MODE. (Chapter 5 goes into more detail on how to use the printer spooler.) The CTRL-BREAK key combination is also no longer supported in OS/2, either for serial or parallel printers. The ",P" option is only supported under DOS mode for serial printers (but is supported for parallel printers under protected mode when you use the "MODE LPT1" command).

IBM recommends you carefully manage your serial devices, particularly printers, if you will be using them in both modes. This is because under DOS most applications just went directly to the printer without checking to see if the device was already in use by another application. This could be the case because of the support for multitasking under OS/2.

To avoid difficulties, IBM recommends you wait until your protected mode applications are finished printing, then switch over to the DOS mode, issue the SETCOM40 COM1=ON command, do your printing, then issue the SETCOM40 COM1=OFF command, and switch back to protected mode.

ASYNCHRONOUS MODEMS: OS/2 uses its own communications port device drivers, which are set at startup in CONFIG.SYS with a "DEVICE=COM01.SYS" command. However, these drivers are only valid for protected mode and not for DOS mode.

You will need to use the SETCOM40 command, just as you did under DOS 3.3 to set up comm port drives under DOS mode.

The comm drivers only support the port itself, and not any attached devices. You will still need to set up the communications port parameters (such as the type of parity and number of stop bits) with the MODE COM1 command. This command has slightly different syntax between DOS and protected mode usage. Under protected mode, you may also set particular communications features, such as data terminal ready, ready to send, and handshaking. These features cannot be set under DOS mode.

We cover modem usage in more detail in Chapter 6, when we address how OS/2 operates programs in background in both DOS and protected modes.

DISPLAYS: Using the MODE command to control the type of display mode works much the same way it did under DOS, with a few differences. First, in protected mode the MODE command only affects the current screen group. You may have several screen groups, each with their own display mode, if you so choose. Second, the ",T" option to display a test pattern is no longer supported under either DOS or protected mode of OS/2. Finally, the ability to shift a display right or left a few characters is also no longer supported under either DOS or protected mode of OS/2.

3.2.5 Other DOS/DOS Mode Differences

Version specific DOS programs won't run in the OS/2 DOS environment, because its version number is 10. [true?]

Device drivers written for DOS, especially block device drivers that were typically used for such things as disks and tape drives, will not work in the compatibility environment. Nor will many character device drivers. This could greatly limit the availability of resources to your DOS mode programs. OS/2 style device drivers should be able to handle calls from both DOS mode and protected mode programs, but they will have to be written with OS/2 in mind. You can't assume that DOS device functions will migrate.

3.3 Working with the OS/2 Command Processors

As we discussed earlier, OS/2 uses a single user interface for both its DOS and protected mode personalities. Consequently, the DOS mode command processor, COMMAND.COM and the protected mode command processor CMD.EXE, both use the same command line and support the full range of OS/2 command modifiers, options and symbols.

3.3.1 Redirection

Generally, input to an OS/2 command will come from the keyboard and the response will appear on the screen. But this does not have to be the case. You can call input from a file or output the result of a command to a file. You can pipe output from one operation as input to another or filter one command's output before sending it on its way.

These changes in the flow of command input and output are known as redirection.

Redirecting Output

To send the output of an OS/2 command to a file, rather than to the screen, use the greater than sign [>] at the command line.

If you typed the command:

```
dir
```

a directory would appear on screen. If, instead, you typed:

```
dir > book
```

The directory listing would become the contents of the file called BOOK. If that file doesn't yet exist, OS/2 creates it. If the file does exist, OS/2 replaces its existing contents with the results of the command.

If you want the result of the command to be added to an existing file, rather than replacing it, use two greater-than signs [>>]

```
dir >> book
```

adds the directory listing to the contents of the BOOK file. If BOOK doesn't exist, [>>] still creates it.

Redirecting Input

Since the greater-than sign redirects output to a file, its nicely logical that the less-than sign [<] redirects input from a file.

Let's say we have created our file BOOK, with a directory listing as contents. Now, we want to create an alphabetically sorted version of the list. We could use the OS/2 sort command (about which more in a moment) and redirection as follows:

```
sort < book > booksort
```

This tells OS/2 to direct the contents of the BOOK file to the sort command and then direct the result of the sort command to a new file called BOOKSORT.

3.3.2 Filters and Pipes

A filter command takes input, modifies it, and then outputs it. OS/2 has three filter commands. They are:

- find which searches for a string of test in a file
- more which displays the contents of a file one screen at a time
- sort which sorts a file's contents alphabetically

Filters can be used independently, with their output redirected into files as noted in the example above. More complex interactions of filters are possible by using pipes.

A pipe allows the output of one command to be used as the input for another command. In OS/2, piping is not done with a command, but with a symbol, a vertical bar [|] that separates commands.

Here are some examples of the pipe symbol in use:

```
dir | sort
```

The output of the DIR command is piped to the sort command, resulting in a sorted directory listing being printed to the screen.

```
dir | sort |  more
```

The DIR output is piped to SORT, and the SORT output is piped to MORE; this produces a sorted directory listing displayed one screenful at a time.

```
dir | sort > booksort
```

This pipes the DIR output to SORT and then redirects SORT's output to a file named BOOKSORT.

3.3.3 Grouping Symbols

The command options discussed so far work in both modes of OS/2. Another set of command line symbols provides extended functions within the protected mode only.

In protected mode, these symbols allow you to group two or more commands on a single command line. So, they are called protected mode grouping symbols.

&: The Command Linking Symbol

The & symbol performs the commands on both sides of it, working from left to right

Examples:

```
dir a: & dir b:
```

OS/2 will display on screen the directory of drive a:, followed by the directory of drive b:

```
del chapter.one & del chapter.two
```

OS/2 will delete two files; first CHAPTER.ONE, then CHAPTER.TWO

```
format a: & chkdsk a:
```

The a: drive is formatted and then the chkdsk command is run, producing a report on the drive's status and system memory on screen.

&&: The And Symbol

When && is used between two commands on a single command line, the command to the left is performed first. If that command operates successfuly, the command to the right is performed. If the command on the left does not work, the command on the right will not run.

Example:

dir chapter.one && type chapter.one

If filename CHAPTER.ONE exists, the && symbol will execute the second commmand and type it. It the file doesn't exist, type won't be run.

||: The Or Symbol

The || Symbol performs one of the two commands on either side. If the command on the left works, the command on the right is not performed. If the lefthand command doesn't work, then || will move on to the right hand command.

Examples:

If you can't remember the exact name of a file you could type:

dir fred.txt || dir fred.doc

If the first name doesn't exist, || will try the second.

Here is another example:

del trash || ren fred trash

If the file called TRASH exists, then this command would delete it. If the file is not found, then the second part of the command would come into operation, and the file FRED would be renamed as TRASH.

^: The Character Symbol

We have used the following characters as command line symbols:

& && | || > >> <

But what if we need to use one of those characters as part of a filename or as output of a command? The ^, when placed before a command line symbol. robs the character of its special meaning to OS/2.

echo goodbye > farewell

This command line creates a file named FAREWELL with one line in it: goodbye.

```
echo hello ^>farewell
```

This command line types the following message on the screen:

```
hello>farewell.
```

An interesting philosophical sentiment.

3.3.4 Command Grouping

In addition to all the grouping symbols noted above, OS/2 allows you to organize internal and external commands on the command line using parentheses ().

The parentheses ensure that OS/2 runs your commands in the order you desire. OS/2 processes command symbols in the following order:

^ Character Symbol

() Command Groupings

<,>,>> Redirectors

| Pipe Symbol

&& And Symbol

|| Or Symbol

& Command Linker

For instance:

```
dir a: & dir b: | sort > dirsort
```

This command sequence displays directories for drives a: and b:, but only pipes the directory for b: to the sort command and then into the file DIRSORT. If we want to pipe both the a: and b: directories to SORT, we can type:

```
(dir a: & dir b:) | sort > dirsort
```

Care must be taken with command grouping. The following command sorts both the directory listing and the contents of a file into a single new file.

```
(dir contact.lst && type contact.lst) | sort > contact.srt
```

3.4 New User Interfaces

The DOS user interface, if you think about it, was emptiness. A blank screen with nothing on it but:

A>

DOS was shy. It barely made itself known when in control of a PC. And when an application came on hand, DOS disappeared altogether from the user's view. The application became a whole new environment for the user to work in.

In OS/2 this situation is fundamentally changed. To begin with, OS/2 offers not a single user interface but at least two, and probably more as time goes on. And each of these is a richer, more intensive, more conversational, more controlling environment, by far, than the DOS A> prompt.

The ultimate goal of these new OS/2 user interfaces is to transform the PC interaction from one that is dominated by the look and feel of individual applications, to one dominated by the consistent look and feel of the operating system. When OS/2 is in control of a PC the user should know it and should receive a great deal of information from the screen about what is available to him from his machine. And when an application is running under OS/2 the operating system should still in be in charge--and should look like it's still in charge--or at worst be a mere hot key away from the user's attention.

For managers, this means that the OS/2 user interface will become a major battleground for user PC interaction. The user interface of DOS needed to be explained; the user interface of OS/2 will need to be managed. It is far more than simply a device for prompting user commands. In essence, the OS/2 user interface, like that of Apple's Macintosh, will become the place where workers work; it will be their PC desk, their microcomputer office.

As a result, selecting the right user interface and setting it up properly for workers will become a bell weather for successful management of OS/2. This is a new experience for DOS-oriented managers. Commands structure and operation shift with OS/2, but retain strong ties to DOS. The user interface picture with OS/2, however, has no precedent in PCs, with the possible distant exception of the rarely used Macintosh. It's a whole new realm of experience.

NOTE TO ED: The following sections will be completed upon receipt of the beta version of OS/2; which will have Presentation Manager included.

3.4.1 The Screen Manager

3.4.2 The Presentation Manager

3.4.3 Example of the Care and Feeding of Screen Groups

How will this notion of screen groups work in the reality of office computing? Let's try an example:

Let's say you have the following four programs running in your machine:

Figure 3.3
Screen diagram of multiple applications
(Non-Presentation manager)

1. First you have a Lotus 1-2-3 version 3 spreadsheet running in the protected mode in background. This means that the spreadsheet is not visible on the screen, but the recalculation that you started before you placed it in background is still running. You can check to see if the recalc is done by periodically returning to this screen group.

2. Next is a network communications application, also running in the protected mode but currently in foreground. It is visible on your screen.

3. Thirdly you are compiling a C program using the new C compiler. Again, this is running in background in protected mode, so you need to periodically return to this screen group to check if your program has compiled.

4. And a DOS word processor, which you have placed in background. This means that it's operation is suspended.

Now, what order does OS/2 assign to execute all of these programs? This depends on several factors, including the priority level assigned to each program and the amount of resources each program requires.

The operating system has three different scheduling priority levels, which you can think of as high, medium, and low. High priority tasks are attended to immediately, as well as more frequently than the others. Typically, these are applications that require constant care and attention, such as communications programs, or applications using the printer. When time is available at the CPU, OS/2 schedules this time to one of these high priority tasks. In our example above, program 2 is given the highest priority.

OS/2 also schedules those tasks for the highest priority that have performed the most recent input or output activity. This is because the operating system thinks that those programs doing I/O are more likely to want to do more, and thus require more attention and more CPU resources.

Medium priorities are those tasks running in background in the protected mode. In our example we gave the spreadsheet in program 1 medium priority, because we needed the recalcs done as soon as possible. Low priority tasks are those which get done last, after all the foreground and other medium-priority background tasks are satisfied. We gave the compile low priority because the spreadsheet recalc was more important than the compiled program.

DOS mode programs don't enter into this scheme at all. DOS mode is suspended when it is in background, unlike OS/2's protected mode programs. There are many reasons for this suspension, but the most important one is because the operation of the DOS mode will interfere with the other programs running in protected mode. We discuss other reasons for this in chapter 6.

NOTE TO ED: Same example modified for multiple view Presentation Manager screen group will go here.

Figure 3.4
Screen diagram of mutliple applications
(Presentation Manager)

So, looking at this situation, you as a manager have much more to cope with than simply what programs look like on the screen. The screen in OS/2 becomes a reflection of the arrangement of processes going on within the machine. You will have to decide if a particular pop-up utility should be linked to a particular screen group, should be common to all, or should be a screen group of its own.

Do you want to train you users to rely upon the hot keys and menus, which are easier to understand but slow? Or do you want to press them ahead to command based selection of screen groups, which is more DOS like and complex but far faster in daily operation?

And at the center of the screen situation lays the big question: Do you want to migrate all of your users to the Presentation Manager? Just some? None of them until an application demands it? If so, will the Screen Manager provide with you enough OS/2 functionality to make using the new operating system worthwhile? Or, will you stick with DOS until the combination of OS/2 Presentation Manager and new applications makes a move inescapable?

Tough questions. We will tackle these and other OS/2 quandries later in Chapters 7,8,9 and 10. But first, let's start at the very beginning. For now, let's see how to get OS/2 up and running on PCs.

Chapter 4. Managing the Move to OS/2 from DOS

In Chapter 3, we discussed how OS/2 has some challenges in store for you. Most of the software you are presently using will have to rewritten to take advantage of its powerful capabilities.

But what are your next steps? How do you get from the world of DOS to the world of OS/2, without getting ulcers and spending too much unproductive time in the conversion?

This chapter shows you what you need to look for in converting applications to OS/2: where the potential time-bombs lie, what type of machines you should and should not consider running OS/2 on, and whether you should stick with DOS for certain applications. We discuss what the changes to protected mode really mean in your day-to-day operation of your computer.

We also will mention sample strategies for you to pursue. We have two groups of strategies: The first set are for those of you who need to make a decision on your own, independent of any corporate standards. This would apply if you worked in a small business with relatively informal PC standards, or if you needed to make a decision for your own use of OS/2.

The second set of strategies is geared more towards those of you who are microcomputer managers and corporate policy-makers, rather than individuals. These strategies should help you shape overall corporate direction, and help you decide on how to change existing corporate standards in the face of OS/2.

4.1 THE OS/2 DECISION TREE

You will face many decision points in your move from DOS to OS/2. We have summarized these points in Table 4.1.

Table 4.1 OS/2 Decision Tree

1. How nasty is my potential OS/2 application?

    Does it write directly to the screen?

    Does it take control over machine resources?

    Does it rewrite the CONFIG.SYS file automatically?

    Do data and program files have the same time/date stamps?

2. How hardware specific are my potential OS/2 applications?

    Do they require specific monitors and communications adapters?

    Can they run on fast CPU clock settings?

3. Can my potential OS/2 applications run on DOS 2 or earlier versions?

4. Do any of my potential OS/2 applications have special DOS networkable versions?

5. What happens when my software runs in the background?

    Do I use any communications software?

    Does my program intermingle inputs from the keyboard and from disk files?

6. Does my software run in its own small window under Windows 2.0?

7. Do I use many different resident programs, and are protected-mode versions of these programs available?

8. Do I use any non-IBM hardware, such as hard disks, monitors, serial adapters, and memory boards, in my machines?

4.2 AVOIDING OS/2 ULCERS: UNDERSTANDING YOUR DOS SOFTWARE

The first step in managing the OS/2 transition is deciding whether OS/2 will benefit your applications. In other words, will the goodies of multitasking and larger memory addressing be worth the effort to convert your existing DOS programs?

By "your programs" we mean both the ones you have purchased from software vendors as well as the ones you have written yourself. While you obviously can't rewrite 1-2-3 on your own (unless you are one of the authors), the decision process is the same regardless of whether you purchased or wrote the software.

If the answer is yes, you need to think about rewriting your application for OS/2 (or purchasing the OS/2 version of the application). If not, then you should be satisfied with sticking with DOS.

If you have some applications that should be converted and some that shouldn't, then you need to look carefully at using the DOS compatibility feature of OS/2. (This is the subject of the next section.)

To make this decision intelligently, you first need to assess how much work it will be to convert your software. As we mentioned in the last chapter, there are a number of important differences between the construction of DOS and OS/2 software programs.

Most of these differences have been caused by developers wanting to squeeze every bit of performance possible from their programs. To get this performance, such as faster screen response or quicker file access, these developers had to make use of undocumented DOS functions or else avoid DOS entirely and address hardware directly from their software programs.

This was acceptable behavior in a world where only a single application would run at once. Under the multi-tasking world of OS/2, however, this is definitely not allowed.

Now it is time to pay the piper and restore order. This means that most existing DOS applications will need to more carefully follow the rules of the road to run under OS/2.

But how can you determine the damage if you don't even understand assembly language programming and never wrote anything as much as a BASIC program? Relief is possible. You can decide which of your existing applications should be converted to OS/2 and which shouldn't by just answering a few simple questions. This way, even the non-programmer can avoid OS/2 ulcers.

4.2.1 QUESTION 1: HOW NASTY IS MY APPLICATION?

If your application has broken all the rules under DOS, you have a job ahead of yourself. OS/2 is very strict, and you will spend some effort rewriting your application to meet its demands.

As we mentioned in the last chapter, applications cannot address hardware devices directly; they must go through device drivers and the "kernel" part of the operating system.

If your application directly writes to the screen, such as creating its own cursor shape or attributes, this must be rewritten for OS/2.

If your application takes control of the various machine services, such as the disk controller, the clock timer, the interrupts, the serial or parallel ports, you will have to rewrite the code to remove these violations under OS/2.

Grabbing these things without the knowledge of the operating system creates a problem under multitasking environment: If one program has control over these resources, they won't be available for anyone else to use. This is why programs must make use of the operating system in OS/2.

Another indicator of a potential problem is a program which likes to rewrite the CONFIG.SYS file when its installed, to include special drivers. Some of these drivers may need modification to work under OS/2.

A final indication of a messy program is one which intermingles its data and program code. How can you tell if this is the case?

One easy way is to look at the time and date stamps of the .COM and .EXE files, and compare them with the date stamps of the other files on your disk. If these are the same, then probably your data and code are in separate files. If the dates for the .COM and .EXE files are more recent, then chances are there is some data contained in your program code.

These nasty applications will be the hardest to convert to OS/2.

4.2.2 QUESTION 2: HOW DEPENDENT IS MY APPLICATION ON SPECIFIC HARDWARE?

If your application requires specific hardware to do its job, life will be difficult under OS/2. As we mentioned earlier, OS/2 isolates your programs from specific hardware. Any attempt to link the two together will be problematical.

For example, if you have programs that require certain types of monitors, such as high-resolution monitors, chances are you will have problems running these under OS/2.

If you have an enhanced color monitor and found that in the past resident programs caused problems for certain applications, this is a good indication that these applications might be headaches under OS/2. The problem is caused by most of the registers on the enhanced adapter being write-only. In other words, it is difficult for a program to restore the original state of the screen once it is interrupted by another program.

Another example is where your application depends on a particular CPU clock-speed. If you have trouble running your application on the faster ATs and clones, you will have trouble converting this application to run under OS/2. Anything that is timing-sensitive must be removed or rewriten under OS/2.

4.2.3 QUESTION 3: DOES MY SOFTWARE RUN ON EARLY DOS VERSIONS?

If your software was written long ago, chances are it used an early file system, called file control blocks, to manipulate its files. Newer versions of DOS use another system called file handles, although they still support programs which make use of it. However, OS/2 only works with file handles, so any old application which makes use of the file control blocks will need to be rewritten.

A simple test is to find a copy of DOS 2.0 and see if your software will run under this older operating system. If not, then it uses the newer file handles approach and your conversion effort is somewhat minimized.

4.2.4 QUESTION 4: DOES MY PROGRAM RUN ON A LOCAL AREA NETWORK?

This is fairly straightforward. Any networked version of your software won't run under OS/2, either in the protected or DOS modes. You will need to rewrite it to take advantage of one of the OS/2 network operating systems, such as the Microsoft LAN Manager.

4.2.5 QUESTION 5: WHAT HAPPENS IF THIS PROGRAM RUNS IN BACKGROUND?

Under OS/2, all programs will be running in the background at one time or another. This may be a problem for your application, depending on how it is written.

If it is a communications program, you will have problems. Most DOS communications programs need to know what is happening with the modem or SNA line at all times, and cannot run well in the background. These will be harder to rewrite for OS/2.

Other programs, such as a spreadsheet or word processor, may work fine in the background. These will be less difficult to rework for OS/2.

One way to tell is to look at how the program gets its inputs: Does the program always expect keystrokes from the operator, or always expect input from data files, or a combination of the two? If the software requires some combination of data files and keystrokes, this will be harder to convert to an OS/2 application.

4.2.6 QUESTION 6: DOES MY SOFTWARE USE LOTS OF GRAPHICS?

OS/2 has two versions, one without the Presentation Manager (version 1.0) and one with (version 1.1). The Presentation Manager is similar to Windows in appearance, but uses very different applications

programming interfaces. Table 4.2 summarizes some of the important differences among running DOS 3 with Windows, using Windows/386 for 80386 processors, and OS/2 with the Presentation Manager.

Table 4.2 Differences in Multitasking Environments

Environment	DOS/Windows	Windows/386	OS/2 v. 1.1
Multitasking	Limited	Yes	Yes
No. of screen	groups ??	??	12
Background execution	No	Yes	Yes
Memory address	single 640 k	Multiple 640 k	16 M
CPUs supported	8086/286/386	80386	80286/386
Protection	None	among virtual machines	among tasks

The combination of DOS and Windows offers limited multitasking capabilities. Programs are suspended when they run in background [check]. Programs share the 640 k of memory among themselves. This is unlike OS/2 and Windows/386. The former allows all protected-mode programs to share the full amount of memory, while Windows/386 lets each program access to a full 640 k session of its own. Windows and DOS can run on a variety of machines, while Windows/386 and OS/2 can run on 80286 and 80386 machines (Windows/386 obviously only runs on '386-based machines.)

Windows/386 uses the 32-bit instructions and memory, while the others use either 8-bit or 16-bit memory.

Finally, OS/2 offers protection among each task running in the machine, while Windows/386 offers protection only among each virtual machine, each machine running a single application. Windows/DOS offers no such protection, meaning that individual applications can crash the entire machine.

If you are an experienced Windows developer, most likely you'll have little trouble converting your application to run in the Presentation Manager.

How can you tell if you are using software which is Windows-compatible? The easiest way is to look for a filename with a .PIF extension (program information file). This file contains the information needed by Windows to run the software properly. Another way is to install Windows on your PC and see if you can run the application in less than a full-sized screen. Those applications that need to display in a full-screen use their own graphic interfaces, are not compatible with Windows, and will be the hardest to convert to OS/2. This is because the entire graphics logic will need rewriting.

If your program has a similar appearance to Windows but doesn't use the actual Windows interfaces, you will have some effort involved in the conversion. You won't need to rework the "look and feel" as much, but you will still need to change the code involving the programming interfaces that are used in OS/2.

4.3 HOW LONG SHOULD I STICK WITH DOS?

Now you have a better understanding of the effort involved in converting your application from DOS to OS/2. But is it worth it?

Should you stick with DOS, and forget OS/2 entirely? Or run the program in DOS mode?

One factor is how many DOS programs you want to keep around. Remember, only one DOS program can run at a time in DOS mode. If you can't live without your full constellation of resident programs, and these programs won't be or can't be converted to run under OS/2, you have an easy decision: stick with DOS.

A second factor is how often you require multitasking, large memory spaces, and other goodies that OS/2 provides. If you are mostly satisfied with your present configuration, by all means stick with DOS. There will continue to be improvements to DOS, according to both Microsoft and IBM developers.

Another factor is what percentage of communications software do you use frequently? If you mostly work with word processors and spreadsheets, then you most likely will have fewer problems running these programs in the DOS mode than if you spend more of your time with terminal emulators and local area networks.

Microsoft has committed to keeping DOS mode as close as possible to DOS. Their developers have said that even Sidekick will run in DOS mode. This means, for your program to run in DOS mode, it shouldn't break any more rules than Sidekick does.

But the DOS mode of OS/2 is still an emulation of DOS and not actually the real thing. As we mentioned in chapter III, there are some differences between "real" DOS and the OS/2 DOS mode. As we said earlier, our guidelines are merely hints of problems you might experience. The only sure way to tell is to try your DOS application in the DOS mode and see what problems you may experience.

The transition will be especially difficult if you manage a large group of PC users, each with their own set of skills and requirements. More than likely, some of these users will want to convert to OS/2 immediately, while others will be more reluctant. This will be more of a problem if your corporation does not use the latest versions of software for all of its employees, since some of the earlier versions might not run properly in the OS/2 DOS mode.

4.4 AVOIDING OS/2 ULCERS: UNDERSTANDING YOUR PC CONFIGURATION

Your next step along the road to OS/2 is to carefully assess your existing PC configuration. Once you have identified which applications are potential candidates for OS/2, you next need to identify which individuals will use these applications and what are the best machines for them to run these on OS/2.

OS/2 is designed to be as compatible as possible with your existing DOS and PC environments. Nevertheless, there are some things that are likely to give you OS/2 ulcers, such as non-standard hard disks, IBM Monochrome monitors, enhanced memory adapters, older DOS versions, and improperly installed serial adapters.

In addition, there are also a few things that can cause problems if you plan on using the IBM Extended Edition version of OS/2, such as non-IBM host file transfer and differences in the choice of high level language APIs.

Let's review each of these topics and suggest ways to avoid these ulcers before it's too late.

4.4.1 NON-STANDARD HARD DISKS.

If you bought your PC from one vendor and your hard disk from someone else, or if you upgraded your hard disk yourself, chances are you might have some trouble installing OS/2 on your machine. OS/2 is very fussy about non-standard disk drives, and you may need to obtain a set of driver files from the hard disk manufacturer to get OS/2 up and running properly.

IBM's OS/2 uses one of two driver files for hard disks. One, called DISK01.SYS, is used for the IBM ATs, and XT/286 machines. The other, called DISK02.SYS, is for the PS/2 family. At installation time, these files are copied to the root directory of your hard disk. Only the appropriate one is needed, but it must be left in the root directory, since when OS/2 boots up this is where it looks to find its drivers.

Unlike other ".SYS" files which use "DEVICE=" statements in the CONFIG.SYS file, the disk drivers do not need to be specified this way: OS/2 automatically knows to go look for it in the root directory.

If you use another version of OS/2 you will have different disk drivers included with the software.

So how do you tell which disk driver is the correct one for your equipment? You can't very easily, other than trying to install the software on your machine. However, the symptoms are relatively obvious: if your machine has trouble booting or reading files from the hard disk, chances are good the driver file is the wrong one. This is why non-standard disk drives may bring about your first OS/2-related ulcer.

Your best bet is to first try to install OS/2 on a computer which still has its factory-installed equipment. Once you have gotten OS/2 running, you can experiment with other computers with the non-standard equipment. Another alternative is to contact the computer vendor and determine if a driver is available for your configuration.

4.4.2 NON-STANDARD MONITORS AND ADAPTERS.

Your second ulcer will be over the type of monitor and adapter card you are using in your machine. As with the disk driver, IBM uses a standard device driver for monitors called SCREEN01.SYS (for ATs and XT/286s) and SCREEN02.SYS (for the PS/2s). Again, this driver must be in the root directory, and does not need to specify in the CONFIG.SYS file.

IBM only supports graphics monitors for its version of OS/2, including its Color and Enhanced Color adapters and all of its PS/2 Video Graphics Array (VGA) display adapters, including its VGA monochrome graphics monitor.

If you use the enhanced graphics monitor and adapter, you will need to put a "DEVICE=EGA.SYS" statement in your CONFIG.SYS file. You can put in a path name and specify another directory for this driver, unlike the other drivers already mentioned which must be in the root directory. For example, if

you specify "DEVICE=C:\os2\ega.sys", OS/2 will look for the driver in your \OS2 directory of your hard disk.

If you use anything else, including the IBM monochrome monitor and its adapter card, or a Hercules adapter card, you will have some problems running IBM's OS/2. This is because OS/2 is a graphics-based operating system, and requires a graphics monitor and a graphics adapter.

While IBM's version of OS/2 won't run, this may not be true for other computer vendors' versions. They may choose to support IBM's monochrome monitor on a variety of graphics cards, such as Hercules. You would need to contact either the computer maker or Hercules in this situation and obtain the appropriate driver files, and replace the SCREEN drivers with the correct file.

What if the vendor doesn't have a driver? Then you might be out of luck, unless you want to write your own.

To stay away from this ulcer, you might to stick with the recommended IBM graphics equipment in your initial installation of OS/2. Once working, you might try contacting the vendor of your monochrome graphics adapter for the appropriate driver.

4.4.3 ENHANCED MEMORY BOARDS.

Some enhanced memory boards, such as AST's RAMpage and Intel's ABoveBoard, can make use of both extended and expanded memory. This is another ulcer in the making. OS/2 only uses extended memory for its protected mode operations, while some existing DOS programs, such as 1-2-3 and Framework, can make use of expanded memory.

OS/2 will not support any DOS program using expanded memory in DOS mode. If you have large spreadsheets or databases and rely upon this extra memory to run your applications, you have a problem.

There are two choices: use the protected-mode application (if available from your software vendor) in place of the DOS one, or else stick with DOS and the expanded memory and forget about running OS/2.

A minor ulcer involves switching all those memory cards between expanded and extended memory addresses. Most of the older cards use switch settings or jumpers located on the card, rather than software programs, to select the type of memory. This means you'll have to take apart every machine and change the switch settings on each memory card, a time-consuming process.

4.4.4 OTHER NON-STANDARD EQUIPMENT.

There are other ulcers associated with other non-IBM equipment in your PC, such as mice, clock/calendars, print spoolers, and non-IBM keyboards. As with the disk drive and screen display, IBM's OS/2 will only work with true-Blue equipment.

There are three additional driver files which must go in the root directory for all OS/2 users. Again, as with the disk and screen drivers, you do not need to specify these in the CONFIG.SYS file.

The drivers are CLOCK01.SYS, KBD01.SYS, and PRINT01.SYS for the ATs and XT/286s, and CLOCK02.SYS, KBD02.SYS, and PRINT02.SYS for the PS/2 family.

If you have any non-IBM equipment in your machine, it may or may not work with OS/2. This could include special clock/calendar cards and keyboards. Unfortunately, the only sure way to know is to try them out.

If you use a mouse, the appropriate mouse drivers must be specified in CONFIG.SYS. Like the EGA driver, these can have their own path. Only the Mouse System's PC Mouse, the Microsoft mice, the Visi-On Mouse, and IBM's PS/2 Mouse are supported in the IBM version of OS/2. (See section 4.5.2 for further information about which mice is supported. Chapter 5 describes the installation of the mouse drivers in detail.) Two device driver statements are actually needed: one for the specific mouse and one called POINTDD.SYS, which provides the actual pointing support.

All print spoolers must use the OS/2 SPOOL command, rather than any other third-party printer spooling device. This command sorts out the different print jobs sent to your printer, so that the jobs are kept separated. The command works with programs that run in the DOS mode as well as in OS/2.

There is also a special series of driver files associated with the BIOS for the PS/2 family. (The ATs and XT/286s have this BIOS coded in the actual chips in the machine.) These files are also needed in the root directory: ABIOS.SYS, F80000.BIO, F80100.BIO, FC0400.BIO, and FC0500.BIO.

4.4.5 OLDER DOS VERSIONS.

Another ulcer involves the appropriate version of DOS that you are using. The cause is the subtle differences in the commands between earlier DOS versions and OS/2. For example, DOS 3.3 and OS/2 both make backups of your hard disk differently than earlier DOS versions. This means that backups made in DOS 3.1 should be kept physically separate from backups made by OS/2 and DOS 3.3. (OS/2 can restore files from backups made in older DOS versions, however.)

In general, it is a good idea to keep current with DOS versions, if for no other reason than to have access to the newer features and functions. However, the newer versions of DOS always require more memory, which means less is available to run applications. For this reason, some of you have decided to stick with older DOS versions. (As of this writing, the current version is DOS 3.3.)

Before converting to OS/2, you may want to first upgrade to the most current version of DOS 3.3 on all your machines (see section 4.6.3 below) to make the transition easier.

To complicate matters, some of you may have different vendor's DOS installed on non-IBM machines, or else use IBM DOS on non-IBM machines. This can be a real OS/2 headache. As mentioned earlier, only the computer maker's version of OS/2 is guaranteed to run, due to differences in disk, monitor, and other drivers. Make sure you obtain the correct version of OS/2 from your computer maker, or else at least test the configuration before you recommend it for your corporation's mix of hardware.

4.4.6 IMPROPERLY INSTALLED OR NON-STANDARD SERIAL BOARDS.

In the past, getting the serial communications equipment to work with your printers or modems was never very easy. This becomes a major ulcer with OS/2.

First, you need to make sure the "DEVICE=COM01.SYS" (or ATs and XT/286s) or "DEVICE=COM02.SYS" (for the PS/2 family) is specified in the CONFIG.SYS file. You can put a path name in this device statement, like the EGA.SYS driver, if you wish. For example, if your CONFIG.SYS file read "DEVICE=c:\os2\com01.sys", OS/2 will look for your driver file COM01.SYS in your \OS2 directory.

Without this specification, the serial port won't work properly under OS/2. Only one serial driver is required in OS/2, so you should make sure that there aren't duplicate or conflicting entries in the CONFIG.SYS file for this assignment.

IBM's version of OS/2 is supported only on the IBM adapters: the AT Serial/Parallel adapter and the PS/2 Multiprotocol and Dual Asynchronous adapters. If you are using something else, such as a Hayes internal modem, your serial equipment may not work with IBM's OS/2. The only sure method is to test it yourself. Again, you would need to contact your board-maker to obtain the appropriate OS/2 serial driver.

Another problem is caused by incorrectly setting the serial ports on your computer. On the IBM AT, there are two serial ports: COM1 and COM2. (The PS/2s can use COM3 as well.) These need to be set correctly for the proper software interrupt (respectively interrupt 4 and 3) and I/O address (respectively 3F8 and 2F8 hex), which on non-IBM boards is done with jumper or switch settings. A good practice is to check the equipment and switch settings and make sure they are properly set. Of course, this means opening up the machine and checking the appropriate adapter.

4.4.7 EXTENDED EDITION ULCERS.

There are other ulcers unique to the IBM Extended Edition of OS/2, particularly the communications manager. These are uniquely IBM products. However, many of you use non-IBM communications gear, such as DCA's Irma boards to communicate with IBM mainframes.

One of the more powerful features of micro-to-mainframe communications software is the ability to write high level language API programs which can automatically read information from the mainframe session and update mainframe and PC files. These HLLAPI programs, as they are called, are a primitive form of program-to-program communications, and follow strict rules.

Unfortunately, not all HLLAPIs are created equally, and in fact, most HLLAPIs differ in subtle ways in terms of the number of commands supported.

OS/2 will use the command set available in the IBM Entry level PC 3270 Emulation program, version 1.1. If you used any HLLAPI calls that aren't part of this version, you'll have to rewrite them.

Another ulcer is caused by micro-to-mainframe file transfer programs which use non-IBM programs running on the mainframe to maintain the link. These are strictly not supported under OS/2.

What is supported is an IBM product created for the 3270 PC file transfer, and since become something of a standard. The product, called IND$FILE, is the only file transfer program supported under the Extended Edition of OS/2.

Several non-IBM vendors, such as DCA and Attachmate, have written DOS programs for their emulators which talk to this IBM program. If you are using one of these you should not have any problems converting to OS/2 EE. If not, you have two choices: The simplest path is to run your emulation software in DOS mode. However, chances are good the program will not function correctly, since communications applications are problematical when they are suspended in background.

If your application won't run in DOS mode, you will need to convert any existing file transfer applications over to the IBM IND$FILE product, and run the OS/2 EE instead of your old DOS-based emulator.

4.5 SAMPLE STRATEGIES FOR INDIVIDUAL USERS

Once you have identified which machines and which applications are best suited for OS/2, you next need to formulate a strategy for the transition. We have divided our discussion into two sections. This first section is for those of you who need to make a decision on an individual basis. You should read this section if you have little contact with other PC users at your job, or work for a small or very decentralized business where decisions are more of an individual nature.

Individuals have a combination of several different strategies for OS/2. You may want to explore several strategies simultaneously, or may want to investigate strategies sequentially, depending on your needs. Let's explore each one in detail.

4.5.1 Ignore OS/2 and continue using DOS 3 exclusively

Many of you, faced with the complexity and challenge of OS/2, may elect to forget about using the protected mode entirely and remain with DOS 3. Microsoft and IBM are committed towards providing you with DOS upgrades and improvements. And DOS certainly is sufficient for the vast majority of your applications.

You should carefully consider several factors before upgrading to OS/2. These factors include cost, your knowledge and comfort level with DOS, how you use your existing applications, and the potential applications developed for the protected mode.

OS/2 will be costly for those of you without the right equipment. Those of you that don't have a 80286-based systems won't be able to run OS/2 and will need to consider the cost of a new AT-class system. OS/2 will also need lots of RAM and mass storage, again costly items for those of you without. OS/2 applications will require a graphics monitor, again increasing the price tag for users of monochrome, non-graphics, and monitors.

A second factor is your knowledge of DOS and your comfort level of using existing DOS commands. Many of you are comfortable with DOS commands, but many of you don't need to be: you want to get your work done using applications software, rather than by becoming DOS-jockeys.

However, those of you who are very uncomfortable with DOS will be even less comfortable with OS/2. If you have hard enough time understanding what is happening when one application is executing, you will be even more confused when several applications are running at once.

DOS is complex enough for some users. Many of you have not even mastered some of the newer DOS commands, such as XCOPY, APPEND, or CALL, let alone the oldest ones such as DIR, COPY, and BACKUP. Users who aren't interested in learning new commands or new features should stay away from OS/2.

The number of new OS/2 commands, small in overall number, is deceptive. All OS/2 commands have different implications in a multi-application environment, and some have markedly different functionality when running in the protected mode. So, while OS/2 users will still face the same DIR and COPY of old, the way these commands are used will require some learning for existing DOS users. Some of the commands will not have any analogs to the DOS world, another complicating factor. (See chapter 3 for a complete description of OS/2 commands.)

A third factor is how you use your existing DOS applications. Those of you who are comfortable with executing single applications, perhaps even those who have gotten used to several terminate-and-stay-resident (TSR) programs, might be better off in DOS. If you are frustrated because you have to keep flipping between your word processor and your spreadsheet, you should consider OS/2. If you are satisfied with the power and procedures of your existing applications, you should probably stay with DOS.

One of the great promises of OS/2 is its increased RAM capacity. Protected-mode programs can use up to 1 G byte of RAM. It does this by using 16 M bytes of actual physical memory, and swapping programs to disk to access the remainder. But you should think carefully about upgrading to OS/2 if all you want is more RAM for your applications. A better choice would be to take advantage of the Lotus-Intel-Microsoft expanded memory specification, commonly abbreviated LIM.

The latest version of LIM (4.0) allows you to run load both programs and data in the above-640 k byte region. (Older versions only loaded data into the higher memory.) However, some older memory boards may not work with the specification. Nevertheless, using LIM might be a better strategy if all you want is more RAM for your applications. Of course, your application must be written to take advantage of this increased memory area.

OS/2 provides a great deal of power. But users must take advantage of the power. Those individuals interesting in sticking with their present applications might be better off remaining with DOS rather than upgrading to OS/2.

A final factor depends on your view of future protected mode developments. OS/2 applications are few in number for the next several years. Most of them are mere "ports" from existing DOS applications, rather than any completely new programs. Some software developers have stated that they have begun huge OS/2 development efforts, scaling back any future DOS applications. Others have stated that they are taking a more conservative approach. Most software developers are still uncertain.

4.5.2 Start learning Windows now

If you decide that OS/2 is for you, there still are many other decisions in terms of the timing of your equipment purchase. Perhaps the most important decision is to begin to learn Microsoft Windows.

While version 1 of OS/2 will be completely character-based, version 1.1 draws heavily upon a Windows-like user interface. Users who lack any experience working with windowing software might want to consider purchasing a mouse and Microsoft Windows to get an early start.

An early start is useful because Windows differs greatly from the classic DOS "A>" prompt. Executing Windows goes beyond just typing commands, because now you manage several different tasks at once, each working out of a separate window. With straight DOS commands, this is more difficult or nearly impossible.

Windows and OS/2 version 1.1 will differ in many programming respects (see 4.2.6). For example, the two programs will have different applications programming interfaces (APIs) and different applications-specific commands. This is significant for programmers and software developers.

Those programmers anticipating working with OS/2 who have never used Windows should start now, even considering these different APIs. At least using Windows is a good first step for these neophytes.

However, these programming commands are not important for end-users. Most end-users will never be involved in programming Windows or OS/2 API function calls.

More important for end-users is the similarity of the interface and Windows navigation commands to the OS/2 presentation manager. In terms of this user interface, the two programs are almost identical. Again, those of you who are new to Windows should consider the software if you plan on using OS/2.

Part of purchasing Windows is buying a mouse. While Windows will work without a mouse, it certainly isn't the best strategy. Using the keyboard with Windows is cumbersome and takes several keystroke combinations to do what a mouse can do easily and quickly. Mice are not expensive.

If you decide to purchase a mouse, understand that not all mice are the same. Furthermore, as we mentioned earlier, IBM's OS/2 supplies drivers for specific mice and keyboards, so not all mice (and all manufacturer's PC keyboards) will work with all versions of OS/2.

IBM's OS/2 will support only the following mice:

- IBM's PS/2 mouse for PS/2s,
- Microsoft's bus, parallel, or serial mice, for ATs and XT/286s,
- Microsoft's serial mouse for PS/2s,
- Visi-On's serial mouse, for either ATs, XT/286s, or PS/2s,
- Mouse System's serial PC Mouse, for either ATs, XT/286s, or PS/2s.

Serial or parallel mice require separate serial or parallel ports available on the PC; bus mice use their own adapters which fit inside the PC. IBM's mouse attaches to its own port on the PS/2 system board.

Each has its own advantages and disadvantages: For example, serial mice may not work with DOS applications in the compatibility environment but do not take up an extra bus slot that bus mice require. You should consider the pros and cons of each choice before buying the most appropriate mouse.

Keyboards are another matter. If you are one of those users that has gone out and purchased a third-party keyboard, you may be in for trouble when using OS/2. By third-party, we mean that the manufacturer of your keyboard is not the same manufacturer of your PC system.

This adds several new dimensions to the compatibility issue. Before, you were concerned if an off-brand PC ran 1-2-3 and some graphics programs. Now there is an immensely more complex problem to consider.

Since OS/2 drivers are so mouse-specific, you should avoid purchasing any other brand of mouse; even it claims compatibility with the above list. An off-brand mouse might save a few dollars in the short run but create much aggravation in the future. If a computer retailer cannot demonstrate that a particular piece of hardware (not mentioned above) works with OS/2, don't take their word that it does and don't buy it.

4.5.3 Give up your favorite terminate-and-stay-resident programs

Most TSR programs will not work well under OS/2, which is perhaps a good enough reason to give them up now before the addiction is any greater. Everyone has his or her own favorite program, such as Sidekick, Superkey, or even more complex programs with a small resident component, such as Carbon Copy.

As we said before, DOS programs will execute in OS/2's compatibility environment, but become suspended when protected-mode OS/2 applications are invoked. This means that any program that depends on timing, such as sending a file via a communications program at a specific time, will send the file at the wrong time, since the program's internal clock won't be running when it is in suspended animation. You need to reset the clock when you return to DOS mode.

This does not mean that OS/2 can't run any communications or timing-dependent program, however: Rather, these programs can run if you are willing to be careful how you do it. For example, asynchronous communications programs such as Crosstalk will work fine, provided that you do not switch out of the DOS mode for extended periods of time while you are on-line and connected to another computer.

4.5.4 Purchase graphics monitors and adapters

All OS/2 applications will require a graphics monitor and graphics adapter. Those of you still working with monochrome monitors without any graphics capabilities will need a new monitor and perhaps a new monitor adapter to work with OS/2. Those of you who have the IBM Color displays might want to upgrade to a higher-resolution monitor.

Even though OS/2 requires a color monitor and adapter, version 1.0 of OS/2 will only display text characters. Users will have to wait until version 1.1 before getting any graphics. This might influence

your purchase decision for a monitor. New, even higher-resolution displays are certain to be announced by the time version 1.1 becomes available.

Before you buy a new monitor you should first determine if it will work with OS/2. OS/2 will use special software drivers for monitors, just as it does with mice. Just as we warned you away from off-brand mice, we also warn you away from off-brand monitors and adapters. Make sure that the brand of monitor you choose is certified to work with the version of OS/2 that you intend to buy.

IBM's version of OS/2 will only work with the 5154 IBM Enhanced Color Display, the 5153 IBM Color Display, or any IBM PS/2 monitor (the 8512, 8513, or 8514 color displays or the 8503 monochrome display). IBM's OS/2 will work only with a regular color adapter, an enhanced graphics adapter, and the PS/2 4050 and 8514/A display adapters. You will get different resolutions and capabilities, depending on the choice of equipment.

[perhaps we could put a chart in here detailing this stuff.]

Other vendors' versions of OS/2 will work with different monitors and adapters. But if the hardware vendor does not supply the drivers, you could be in trouble.

(See chapter 6 for a complete list of supported hardware.)

4.5.5 Purchase software-configurable RAM or CPU boards

As we have mentioned several times, OS/2 will take advantage of lots of RAM. But, more importantly, just loading OS/2 itself will use lots of memory without running any applications.

IBM recommends the following RAM as a MINIMUM for running different versions of OS/2:

- For OS/2 version 1.0 without running DOS 1.5 M bytes
- For OS/2 version 1.0 with DOS applications 2.0 M bytes
- For OS/2 version 1.1 without running DOS 2.0 M bytes ??
- For OS/2 version 1.1 with DOS applications 2.5 M bytes ??
- For OS/2 EE version 1.1 without running DOS 3.0 M bytes
- For OS/2 EE version 1.1 with DOS applications 3.5 M bytes

These minimums are based on many different factors. We recommend installing 4 M bytes of RAM if you are going to be serious about taking advantage of OS/2, no matter which edition you intend to use. Given that most machines have less than 1 M byte of RAM at present, almost every user will need to purchase additional memory to run OS/2.

Memory boards come in many different shapes and sizes, and the choice to the average user is nothing short of confusing. We cover these decisions and the realities of OS/2's RAM requirements in more detail in chapter 6, but for now you should realize that one of the more expensive parts of the OS/2 upgrade will be buying more memory, and lots of it.

4.5.6 Upgrade mass storage capacity

OS/2 will need lots of storage, not only to store the initial set of operating systems software but also the various protected-mode programs. While it is certainly possible to run OS/2 without a hard disk, you certainly wouldn't want to do so. Playing the swapping floppy game becomes even more tiring when you want to run five different applications at once in memory. We recommend you purchase the largest hard disk you can afford. Certainly, 30 M bytes is the minimum working size.

The reason is the size of OS/2 itself, along with the size of protected mode applications. The many OS/2 software files themselves take up about 4 M bytes of storage. (Not all these files will be needed by everyone, but still the amount of storage required by the average user is phenominal.) Users with only 10 M byte hard disks, or fairly full 20 M byte ones, will want to upgrade or add new 30 M byte ones.

If you are comfortable taking apart your computer, then installing a new hard disk drive is a relatively easy task. If you have never taken apart your computer, then you should go to a local retailer and have them do it for you.

You should determine if you have a full-height hard disk or a smaller version. Full height models can be replaced with newer models that have larger storage. Half- or third-height models leave room for additional compact drives to be added.

You can also make more room if your machine has two floppy drives by removing one of the floppy drives and replacing with a hard disk.

If there is no room inside your PC, a variety of external drives are available, although not from IBM. This raise once again the issue of compatibility.

The exact brand name of the hard disk and disk controller will be critical for OS/2, perhaps even more critical than mice and display brands. OS/2 may not boot from some off-brand hard disks. Again, special drivers are used for specific equipment, and the buyer must be certain that these will be supported by the particular version of OS/2.

4.5.7 Purchase either faster AT-clones or IBM PS/2s

Given our warnings about running OS/2 on non-standard equipment, many of you by now are wondering if perhaps the best bet wouldn't be to just buy Blue. This is certainly an expensive strategy, and should only be considered if you plan on working with other PS/2 features which presently are not found on any other PC at present. These features include using the multi-processing capabilities of the PS/2 microchannel architecture or the new Video Graphics array found on the PS/2s.

However, these are dubious arguments at best. Most of the more prominent clone vendors, such as Compaq and Zenith, have licensed versions of OS/2 from Microsoft which run on their own hardware.

Either of these two seems to be the best alternatives to IBM's OS/2, for those of you that are not interested in buying IBM hardware. But we cannot make this same statement for all other clone vendors, due to the afore-mentioned differences between hard disk controllers, display adapters, mice and keyboards of the other clones.

Currently, as we go to press, only Compaq, Zenith, and Tandy have licensed OS/2 from Microsoft, in addition to IBM.

4.5.8 Purchase 80386-based CPU

A slightly different issue is deciding whether or not to purchase a 386-based machine, such as IBM's PS/2 model 80 or Compaq's Deskpro 386. Both machines will run OS/2 without any problems, and probably run it faster than on most 286-based machines.

Those of you interested in maximum future flexibility should consider the 386. Eventually, an operating system will be written to take advantage of its features. For heavy CPU-intensive work such as compiling programs or recalculating spreadsheets, they are the fastest machines around.

However, future '386 machines may run at even faster clock speeds or offer completely new functionality not even dreamed of in the model 80 or Deskpro 386. Thus, users buying existing '386 machines run the risk of their machines becoming obsolete even before they can be fully utilized.

One advantage that most commercially available software does not yet take advantage of the 32 bit architecture of the 80386. This is the ability to run multiple DOS sessions concurrently, something lacking in the architecture of the '286 chip. However, there are no plans for any version of OS/2 to be able to do this. OS/2 is limited to running a single DOS session at a time.

Microsoft has hinted at an "OS/3," an operating system to take full advantage of the 80386, but as of this writing no product or available has been announced.

OS/2 has the same set of features and functions on the '386 as it does on the '286; it just runs faster. If speed is not an issue for you at present and you think other PCs will be more attractive by the time a '386 operating system is available, we suggest you stick with the older '286 technology for the time being.

4.6 SAMPLE STRATEGIES FOR MANAGERS AND APPLICATIONS DEVELOPERS

The last section concentrated on strategies for individual PC users. There are a series of different strategies if you must decide on overall corporate microcomputing policy, if you manage a group of computer users, or if you develop your own OS/2 applications.

Here the decisions are more complex. Let's take a look at the sequence involved and provide some sample recommendations.

4.6.1 Decide to stick with DOS and forget OS/2

After reviewing all the promises and problems of OS/2, you may decide it is better to forget this new operating system entirely. Most of your employees or customers may be perfectly happy with running their single-tasking DOS applications. Your employees/customers may not need any or most of the new features of OS/2.

To make this decision intelligently, you will need to weigh the cost of upgrading your equipment from DOS to OS/2 and the expected strategic business benefit to be gained from OS/2.

The benefits are tough to calculate precisely. You need to approximate how much time people spend switching among applications. Do you have some employees that are constantly moving from their word processor to their spreadsheet or from a database to a communications program? If so, these would be good candidates for OS/2.

Secondly, you should look at how many of your employees are using expanded memory boards for large spreadsheets and databases. These could also be candidates for OS/2, depending on what kinds of applications they are using and whether they are satisfied with the implementation of the expanded memory on their existing DOS software.

Third, you need to carefully evaluate networking and communications applications in your company. Many of the more interesting OS/2 applications will be based on IBM's Extended Edition software, which integrates communications and database functions into OS/2. You may want to look at how you currently use these DOS applications and if you could benefit from the increased integration under OS/2.

We realize that there are no strict formulas and algorithms to make these benefits more precise. It is a judgment call. Ultimately, you need to look closely at the business benefit of OS/2: will OS/2 make my corporation more profitable? More competitive? More closely tied to my customers?

Once you have looked at the benefits, you need to consider the costs of upgrading to OS/2.

Some of the costs are relatively easy to calculate: As we said earlier, we recommend 4 M bytes of memory to run OS/2. You should use this figure in your cost calculations.

You will also need mice, graphics monitors, extra hard disks, and upgrade non-80286 machines, not to mention the cost of OS/2 along with any new protected-mode software as well.

You may not want to upgrade everything all at once, in which case you should follow some of the other strategies we have below.

While these costs are relatively straightforward, there are others which are more difficult to calculate. These include the cost of training, the lost productive time in making the transition, and the cost of continued support for OS/2.

If your company likes to keep its support staff lean and mean, then OS/2 may not be appropriate for a corporate-wide upgrade. The transition will be involved and could overwhelm a small staff. You might need to buy outside trainers and consultants to help smooth the move to OS/2.

There are many factors to an OS/2 upgrade. You will need to know the precise configuration of each machine intended for OS/2, since IBM's version may not work with non-IBM parts. Your support staff will also need to look at each hard disk and see if there is enough room to install OS/2.

You may also need to formulate stricter standards for your PC configuration, in terms of mice, monitors, keyboards, memory boards, and hard disk structures. If you have these standards in place already, this

will not be a difficult task. If you don't, the process will require some staff time and needs to be included in the overall cost calculations.

When you have assembled your final numbers, compare the costs with the benefits. Perhaps a phased-approach, where only a few "power users" get those initial copies of OS/2 would be best, or perhaps the costs exceed the benefits so much that you might want to stick with DOS for the next few years.

4.6.2 Decide to stop buying 8088s to prepare for transition

You may want to hedge your OS/2 bet: stop buying equipment that you know will be difficult to upgrade, but in the meantime continue using DOS until the latest and greatest OS/2 applications force your hand.

The first decision is to stop purchasing 8088-based systems, such as the original IBM PC. While IBM stopped selling these models long ago, many clone manufacturers continue to make 8088- and 8086-based PCs.

These older machines will be difficult to upgrade, since OS/2 requires the 80286 or 80386 processor to run. This means that you will either have to replace the entire machine or purchase upgrade boards, such as Microsoft's Mach 20 or Intel's Inboard/386, which have memory and either an 80286 or an 80386 processor on them.

The add-in boards offer some opportunity, but they have several problems. The 8088 machines may not have any extra adapter slots for the boards: most of these machines came with 5 slots, three of which are usually taken up with monitor, memory, and floppy disk cards. If you have a modem or a network board this leaves only one other slot.

The add-in boards come with connectors for hard disk drives, along with installed memory. But you are limited to the memory socketed on the board. This makes upgrading memory difficult, if not impossible later on, unless you buy a new board with more memory installed.

Another problem is power. The early PCs had relatively under-sized power supplies, so you will probably need to purchase higher-capacity ones.

A final problem is the version of OS/2 itself which will run on these "turbo" boards. IBM does not recommend its version of OS/2 on any of these add-in boards, and you most likely will need to buy a copy of OS/2 from the turbo board maker. This could increase your support time and costs.

It almost seems not worth it to upgrade the older machine: By the time you add up the cost for an add-in board, memory, and power supply, and then factor in the time it will take to upgrade all these machines, it is almost cheaper to purchase a new '286 machine with all the right pieces in place to begin with.

4.6.3 Standardize on VGA graphics monitors

If you are going to support OS/2 users throughout your company, one place that you can make supporting them easier is by standardizing on the latest IBM Video Graphics Array (VGA) high-resolution

monitors. These new monitors, first introduced by IBM in April 1987, offer the highest resolution and color clarity of any IBM monitors.

There are several reasons to choose the VGA series. First, the VGA adapter is integrated into the PS/2 equipment for all of the new 8500 series of monitors. This makes supporting the monitors easier, since you directly attach the monitor to the PS/2 without the need to worry about what kind of display adapter is inside the machine. (IBM does sell an 8514/A display adapter to display higher resolution graphics and text on its 8514 color monitor. However, you do not need this extra adapter to run the 8514 display at lower resolution.)

Second, IBM sells an adapter for its ATs and older PCs which can connect to the VGA monitors. This means that the VGA can run on all your machines, something not possible with the older IBM monitors. You can have a single standard of graphics support for OS/2.

Third, using the VGA monitors making changing from monochrome to color monitors easier. Since the adapter is the same (either the PS/2 integrated adapter or the one for the older AT-style bus), all you need to do is to change monitors. No need to open the machine, adjust switch settings, and the like.

Fourth, the VGA line of monitors is the only IBM monochrome graphics monitor supported under OS/2. If some of your users want monochrome monitors and want to run OS/2, the VGA monitor is their only choice.

Fifth, the VGA monitors do not require any special device drivers in the CONFIG.SYS file, unlike the enhanced graphics monitors. This makes them easier to support and makes the CONFIG.SYS files more universally applicable to your overall user community. If you were to have a mix of enhanced and VGA monitors running OS/2, you would need to support two separate CONFIG.SYS files: one with the driver and one without.

Finally, the VGA monitor works well with multitasking software, unlike the enhanced graphics adapter (EGA) and its monitor. Some applications, according to IBM's documentation, may not be able to restore the screen to its original state when you switch from DOS to a protected mode application and then back to DOS again. This is because the software program cannot remember what the original screen looked like, due to the way the memory on the EGA was constructed.

4.6.4 Standardize on mice

Another place to simplify support for OS/2 is by using the same mouse on all machines. If you have a mix of PS/2s and older PC ATs, this means you are limited to serial mice from Microsoft, Visi-On, or Mouse Systems.

You should determine your strategy based on how many mice you already have in your corporation, and whether you want to replace them with a new standard. Mice are relatively inexpensive, so this should not be a big obstacle for implementing OS/2.

However, unlike monitors, you cannot completely standardize your support for mice. IBM's OS/2 uses one set of mouse driver files in its ATs and XT/286s, and another in its line of PS/2s. If you have a mix of

this equipment, you will need to maintain two separate CONFIG.SYS files, one with each configuration. (See chapter 5 for more details on which mouse driver is required.)

While all of the Presentation Manager applications will work without mice, most likely users will find that mice help make them more productive in navigating about an application.

If you are a software developer, you should test your software with one of the approved IBM OS/2 mice for compatibility.

4.6.5 Decide to start buying new-style RAM boards and newest DOS version

OS/2 uses extended memory, rather than expanded memory. But to make your support life easier, you may want to hedge your bets and purchase new memory boards which can switch between the two modes in software, such as the ones for the Micro Channel bus. Older boards used in the "classic" PC AT bus had a series of switches which controlled the memory configuration.

If you have any machines with these boards, you will need to pop the covers off, take out the board, find the manual with the switch settings documented, change them, and replace. This could easily be a time consuming operation, especially if you are going to be doing this for hundreds or thousands of computers.

You should also purchase the most memory you can afford, preferably 3 or 4 M bytes as we mentioned earlier. As we mentioned earlier, we will go into more detail in chapter 6 as to the specifics of what to look for in RAM boards.

You'll also make support simpler if everyone in your corporation is using the same version of DOS. The newer versions of DOS are closer to the features and functions of the OS/2 commands, such as the changes made in the BACKUP, RESTORE, and XCOPY commands. This makes supporting a mix of DOS and OS/2 users easier, since the commands have the same functions in both environments.

4.6.6 Decide on PS/s or classic ATs for OS/2

IBM's PS/2s have many differences from the "classic" AT line. They use smaller disk drives. The BIOS is different. They require different adapter boards and monitors to work with the microchannel and VGA electronics.

Some of you have decided that these differences are minimal and are recommending PS/2s as the new corporate standard. This means that you will have to support two configurations of OS/2 for your company: one configuration with the drivers for the classic AT family, one for the PS/2 family.

Supporting two series of drivers is not the worst thing in the world, but it will require some careful planning. For example, you may want to include both sets of driver files on every machine's hard disk, to make locating things in time of trouble easier. You could distribute different colored floppy disks for the different machines.

Or you could decide to standardize on the classic AT architecture and buy only non-IBM AT compatibles, such as the Compaq 286 and 386 machines.

Some of you have already made this decision and are buying the clones. If so, you need to decide what brands of computers you should purchase and what brands of OS/2 you should purchase.

If you bought different versions of DOS with your clones (such as Compaq DOS for your Compaq's and Zenith DOS for your Zeniths) and your support staff is comfortable with this strategy, you probably should continue to buy different versions of OS/2 to match your computers.

However, some of your computer vendors may choose not to license OS/2 from Microsoft, in which case you might have a problem running another vendor's OS/2 on these machines. The only way to know is to carefully test the software with the machine.

Another factor in the decision whether to go clone or not is to examine the role that IBM's Extended Edition will play in your corporation. Extended Edition is an IBM-only version of OS/2, and will not be available from non-IBM CPU makers. However, it will run?? We hope?? [check this out] .....

If Extended Edition is going to be a big part of your corporate future, or if you have always bought Blue in the past, then your strategy should be to stick with the PS/2 models 50 and above.

Buying the PS/2s might reduce support costs over the long term, since the machines do not have any switches to change and screws to lose. Again, you need to look at how you presently support end-user computing.

4.6.7 Identify key applications and users that could benefit and purchase OS/2 toolkit

Your last decision-point should look at the individuals involved in your corporation and how they could benefit from using OS/2. You should identify a few select applications, end users, and departments which have the most promise for OS/2.

Factors to consider include the ability to network PCs and communicate between PCs and the mainframe, using large memory-intensive programs such as spreadsheets and databases, and people who need to look at several screens at once, such as programmers.

Chapter 5. Getting OS/2 Up and Running

In this chapter, we will discuss how to install OS/2 on your computer and make the necessary changes to get up and running in the new operating system. OS/2 has a tremendous flexibility -- but along with that flexibility is much confusion over how to properly install and configure your equipment. We discuss each of the OS/2 configuration parameters and provide some sample batch files for you to try out as a first-time OS/2 user.

5.1 What is the minimum configuration needed to install OS/2

In previous chapters, we have touched upon the types of equipment supported by OS/2. Briefly, let's review the minimum configuration needed to run IBM's OS/2, in terms of processor, display, memory, and other peripherals.

First, you need an IBM machine with either a 80286 or 80386 processor is required to run OS/2. This means either an IBM AT, XT/286, or IBM PS/2 model 50 or better to run OS/2, of course. While neither we nor IBM recommend it, you may be able to run IBM's OS/2 on other machines, such as Zeniths and Compaq's. (Of course, Zenith and Compaq both sell their own versions of OS/2, designed to run on their machines.)

However, mixing OS/2 and hardware vendors are a risky proposition. OS/2 is very fussy about what particular hardware is installed. For our discussion for the remainder of this chapter, we assume you will be working with IBM equipment. Figure 5.1 shows what we recommend for equipment.

Figure 5.1 System Requirements for OS/2

For optimum performance, an OS/2 system will require an 80286 or 80386 processor, a graphics monitor and adapter, 4 M bytes of memory, a 40 M byte hard disk, and a high-density floppy drive.

Your machine should have either the standard or enhanced IBM color graphics adapter and monitor attached. If you are using a PS/2, any of the IBM VGA monitors, including the PS/2 monochrome monitor, will work with OS/2. You may also run OS/2 with a PS/2 monitor and the IBM VGA 8514 adapter on an AT or compatible. IBM's OS/2 may not run with non-IBM displays and display adapters.

You will need at least 1.5 M bytes of memory installed to run OS/2 standard edition. We recommend at least 4 M bytes of memory for most OS/2 situations. Your memory board should be configured for extended memory (the memory associated with the VDISK virtual diskette program), rather than expanded memory (memory associated with the paged LIM-style programs).

We also recommend a hard disk of at least 30 M bytes of storage. While you certainly can run OS/2 on a floppy-based machine, you really wouldn't want to go through the pain and suffering involved. And IBM recommends hard disk machines as well.

OS/2 will work well with either a parallel or serial printer.

No mouse is required to run OS/2, but if you install a mouse you should use one of the ones mentioned in Chapter 4 (from either IBM, Microsoft, Visi-On, or Mouse Systems).

As we said in chapter 4, you should plan on using IBM communications adapters, such as the IBM Multiprotocol Adapter/A for the PS/2 machines. IBM's OS/2 may not run on non-IBM cards, such as a Hayes internal modem or an Irma board.

5.2 Deciding how to install OS/2

Once you have the right configuration, your next step is to decide how you want to run OS/2. You should first consider whether or not you want to boot your machine from the hard disk or the floppy. By "boot" we mean the ability to turn on your computer and load the initial operating systems software automatically. If you have a machine with a hard disk and you can access your programs once you turn it on without the need for any floppy diskettes, your hard disk is directly bootable.

Making bootable disks means that certain system files must be placed on the disk at the beginning of the disk's memory area. Usually this is done at the time of formatting, or when you install a new version of the operating system.

There are many reasons influencing a decision to go with either floppy or hard disk bootability:

For a floppy boot, you maintain the original DOS boot files on your hard disk. If you later change your mind and decide that OS/2 is not for you, you can return to the DOS world relatively unscathed and with minimal fuss. Another advantage is that you can better isolate problems encountered with OS/2 by putting all the boot files on a floppy diskette.

However, a hard disk boot makes it easier to work with OS/2, since you do not have to keep any floppy disks around with the boot programs. Once properly installed, the OS/2 bootable hard disk won't require as much effort as with a floppy boot. Floppy boot disks also can wear out quickly, which is also something of an inconvenience.

Let's look at both methods.

5.2.1 Running OS/2 from your hard disk

If you need to test a variety of OS/2 applications for your corporation, or if you are convinced that OS/2 is the operating system for you, then we recommended this method.

If you are concerned about complete DOS compatibility, and have some troubles running DOS programs in OS/2's DOS mode, then you can leave the DOS programs on your hard disk and create a separate DOS boot disk. When you need to boot with DOS, you insert this boot disk in your machine's A: diskette drive and you can bring up "pure DOS."

You can switch this easily between DOS and OS/2 because the file structure of your hard disk remains the same between DOS 3 and OS/2. The same file allocation tables are used by both operating systems, and OS/2 reads and writes files the same way that DOS 3 does.

If you plan on booting DOS sometimes and OS/2 other times, we recommend you upgrade DOS to the latest version, as of this writing 3.3. There are several reasons for our recommendation. First, the OS/2 commands are closest to the most current DOS version. This keeps the confusion down when you

switch operating systems. Second, if you run into any problems, IBM will be most helpful in resolving things if you are using the most current version of DOS. Finally, older DOS versions, especially DOS 2 and before, use a different file structure and do not support the AT line of computers, or networks, or other more recent PC innovations.

IBM has assumed that you will boot from OS/2, and has created an installation program which automatically copies all the necessary files onto your hard disk. As we get into the nitty-gritty of the OS/2 installation, we point out what the automated program is doing and what parameters you might want to change later on when you become familiar with OS/2. We also have some sample batch files at the end of the chapter which illustrate how to boot both DOS and OS/2, should you decide to go this route.

5.2.2. Running OS/2 from a floppy diskette

However, there is another choice. You may wish to continue to operate your hard disk under DOS, and create a separate boot disk for OS/2. You would choose this path if you are unsure of how much a role OS/2 will play in your computing life, or if you have yet to create or purchase any OS/2 application and just want to take OS/2 for a test drive.

You can choose to install OS/2 to boot from a floppy with the automated installation program. The steps involved are summarized in Table 5.1.

Table 5.1 Steps needed to run OS/2 from a floppy

1. Create your \os2 and \spool directories
2. Copy files from all four OS/2 program diskettes to the \OS2 directory.
3. Do a diskcopy and create another OS/2 disk #1.
4. Create CONFIG.SYS file shown in the text
5. Add the startup files and CONFIG.SYS to the new diskette.
6. Reboot, you should be able to run OS/2.

The first step is to create the subdirectories \os2 and \spool. Next, you should copy all the OS/2 program files into the \os2 subdirectory. While you can boot OS/2 from a floppy, all the associated files that are needed to run OS/2 still need to reside on your hard disk.

Next, you should make a copy of the first OS/2 program diskette. Use the DOS command DISKCOPY to make you copy, since this will make sure that all the system files are copied and that the copy will be able to boot.

Once this copy is made, put the original OS/2 program disks away in a safe place.

Finally, you need to add the startup files, along with a CONFIG.SYS file, to this boot disk. We describe what commands are needed in each of these files in the sections below, along with sample files in section 5.9.

5.2.3 First steps and installation overview

Making your hard disk bootable is relatively easy with IBM's OS/2. As we said earlier, OS/2 uses the same file structure that is used by DOS 3.0 and above, so other than the system files and new command files, very little else needs changing.

OS/2 comes with three program diskettes and one installation diskette. You can purchase either a 3 1/2 diskette version or a 5 1/4 diskette version. Both packages come with high-density disks (either 1.2 M bytes on the 5 1/4 diskettes or 1.44 M bytes on the smaller 3 1/2 diskettes). You should purchase the version which matches the configuration of your A: drive, since you can only bring up the new operating system from that drive.

You will want to make sure that your hard disk has at least 3.5 M bytes of empty room on it. OS/2 program files take up about 3 M bytes of storage. The extra room is used for temporary storage during the installation process.

As we mentioned earlier, if you are going to boot OS/2 from your hard disk, IBM has automated the installation process as much as possible. The following steps will help you get started. If you are going to boot OS/2 from a floppy, you should use the procedure outlined above in Table 5.1.

But before you start the process, make sure that your root directory does not contain any DOS command files, other than COMMAND.COM.

If you have been running your hard disk with all those DOS command files in your root directory (such as FORMAT.COM and DISKCOPY.COM), now is the time to clean house. You should copy these files to a separate \DOS subdirectory (if you think that you still want to boot your machine in true DOS rather than use the DOS mode feature of OS/2), or you may delete them.

You may have to delete these DOS command files from other directories on your hard disk, depending on your applications installed. (For example, early versions of Lotus 1-2-3 copied DOS program files into the subdirectory used to run the 1-2-3 programs.)

OS/2 has its own version of these commands, some with the same file names, and you don't want to get things confused right off the bat.

Once this is done, all you need to do is put the installation diskette in your A: drive and turn on or reboot your computer.

The automated installation process goes through the following steps. First, it asks you whether you want to format your hard disk. Formatting the disk will erase all your existing files on your disk. If you have already been using your computer and have lots of programs on it, you want to select the other option, to just copy the OS/2 system files onto your hard disk.

If this a brand-new computer, or if you are willing to completely wipe the slate clean and start all over, then you should reformat your disk.

Next the program asks you if you want to rename your AUTOEXEC.BAT and CONFIG.SYS files in your root directory. OS/2 will use new ones, and it suggests defaults of .BAK extensions for both files. This is a good idea for the present time: we will come back to what you need to do to preserve your existing DOS startup routine later.

The next step is to copy files from the installation diskette onto your hard disk. Again, this is done automatically. The installation program places about 50-odd files in your hard disk's root directory. These files contain the program libraries and standard device drivers for your machine. Some of these can find other homes; others must stay in the root. We will go over what you can and can't move later.

Once done copying the program files from the installation diskette, OS/2 asks you to reboot your machine with the standard CTRL-ALT-DEL three-key combination. Then it asks you to place each of the three program diskettes in drive A: and proceeds to copy them to your hard disk. It creates a new subdirectory, called naturally \OS2, to do this.

Once the files have been copied, the program asks you to select the appropriate country (if you are using your computer outside the U.S.), type of mouse, and whether you will be using any serial devices. These choices will influence what drivers OS/2 sets up in your CONFIG.SYS file. You can view your choices, change them if you'd like, and then store them. (More on changing your configuration in the next section.)

The automated installation process is finished, but you have some cleaning up to do. Reboot your computer and you should be able to bring up OS/2. Watch the screen carefully for error messages. Any device drivers OS/2 has trouble installing will be displayed at this time. Make a note of these problems and refer to our discussion on troubleshooting in Chapter 7.

If OS/2 loads successfully, congratulations!

If it doesn't load, you might want to read Chapter 7 now and outline your plan of attack.

5.3 What is new with your protected mode hard disk

If you were successful in installing OS/2, you should be able to now explore the changes on your hard disk accomplished by the automated installation program. There are several basic changes, both to the root directory and several subdirectories.

Your root directory now contains over 50 files, along with any subdirectory entries that you have from your days of using DOS. As mentioned earlier, OS/2 requires certain programs and drivers to run. These must be installed in the root, since before OS/2 is completely loaded into memory it does not know where to find these files.

Here's a listing of a sample root directory:

```
OS2        ®MDBO¯<DIR>®MDNM¯        6-19-87   9:51p

SPOOL      ®MDBO¯<DIR>®MDNM¯        12-24-87   1:24p
```

```
4201     DCP      17069  10-21-87  12:00p
5202     DCP        404  10-21-87  12:00p
ANSICALL DLL       3637  10-21-87  12:00p
AUTOEXEC BAT         96  12-24-87   1:53p
BKSCALLS DLL       5704  10-21-87  12:00p
BMSCALLS DLL       2576  10-21-87  12:00p
BVSCALLS DLL      31744  10-21-87  12:00p
CLOCK01  SYS       2762  10-21-87  12:00p
CLOCK02  SYS       2762  10-21-87  12:00p
CMD      EXE      57648  10-21-87  12:00p
COMMAND  COM      25564  10-21-87  12:00p
CONFIG   SYS         300  12-24-87   1:38p
COUNTRY  SYS      14632  10-21-87  12:00p
CPISPFPC DLL     108598  10-21-87  12:00p
DISK01   SYS      18616  10-21-87  12:00p
DISK02   SYS      18616  10-21-87  12:00p
DMPC     EXE       2472  10-21-87  12:00p
DOSCALL1 DLL       8709  10-21-87  12:00p
DTM      DLL       2222  10-21-87  12:00p
FORMATS  TBL        590  10-21-87  12:00p
HARDERR  EXE      16288  10-21-87  12:00p
ISPD     MSG       1542  10-21-87  12:00p
ISPM     MSG        500  10-21-87  12:00p
KBD01    SYS      16945  10-21-87  12:00p
KBD02    SYS      16945  10-21-87  12:00p
```

KBDCALLS DLL 7232 10-21-87 12:00p

KEYBOARD DCP 85917 10-21-87 12:00p

MONCALLS DLL 7351 10-21-87 12:00p

MOUCALLS DLL 5701 10-21-87 12:00p

MSG DLL 6578 10-21-87 12:00p

NLS DLL 5162 10-21-87 12:00p

OS2INIT CMD 81 12-24-87 1:24p

OSO001 MSG 64808 10-21-87 12:00p

OSO001H MSG 83964 10-21-87 12:00p

PRINT01 SYS 7683 10-21-87 12:00p

QUECALLS DLL 11238 10-21-87 12:00p

SCREEN01 SYS 1583 10-21-87 12:00p

SCREEN02 SYS 1583 10-21-87 12:00p

SESMGR DLL 24262 10-21-87 12:00p

SHELL11F AII 150 10-21-87 12:00p

SHELL11F AIF 524 10-21-87 12:00p

SHELL11F EXE 31676 10-21-87 12:00p

SHELL11F PRO 157 10-21-87 12:00p

SHELL11F CNF 201 10-21-87 12:00p

SHELL11F LIB 41472 10-21-87 12:00p

SPOOLCP DLL 9828 10-21-87 12:00p

STXTDMPC DLL 12569 10-21-87 12:00p

SWAPPER DAT 655360 12-24-87 1:39p

SWAPPER EXE 4150 10-21-87 12:00p

VIOCALLS DLL 13981 10-21-87 12:00p

```
VIOTBL    DCP      52150  10-21-87  12:00p
```

What are all these programs doing there?

Let's go over them in groups. Files ending in a .SYS are the driver files which are not specified in CONFIG.SYS but loaded automatically by OS/2. The ones with an "01" in their names, such as KBD01.SYS, are the files for the IBM AT and XT/286 family of computers. The ones with "02" in the file names are for the PS/2 family. Obviously, you only need one set, but the installation program copies both just be sure it has gotten them all.

Files ending in .DLL are the dynamic-linked libraries that OS/2 uses for many of its tasks. As mentioned earlier in Chapter [], these libraries are loaded only when needed by other programs, hence the term dynamically linked.

Files ending in .CMD are batch files for protected mode programs. They have a different extension to keep them separate from the DOS mode .BAT batch files.

The SHELL files are used by the OS/2 Program Selector in the first version of standard edition. (We have illustrated a directory with version 1.0 of OS/2. Version 1.1, which includes the Presentation Manager, has its own set of SHELL files.)

COMMAND.COM is still the DOS mode command processor, and CMD.EXE is the new OS/2 command processor. Notice how much larger it is.

There is a special file called FORMATS.TBL which is an ordinary ASCII file listing all the OS/2 system files. These files are copied from the root directory of the source diskette or hard disk when you specify the FORMAT /S option. This special file is needed because OS/2 has more than just the three system files used by DOS. If the files specified in the FORMATS.TBL are not in the root directory, you will get an error message.

AUTOEXEC and CONFIG.SYS are still in the root, but their rolls have changed somewhat. AUTOEXEC is only used to start up the initial DOS mode session. (OS/2 uses STARTUP.CMD to start up the initial protected mode session, and a special command file to start up each subsequent protected mode session.)

There are also two files named SWAPPER. The .DAT file contains the programs which OS/2 swaps out of memory and onto your hard disk, when you need to swap programs out of memory, either because you have too little memory or you are running too many programs. More on swapping later on. Notice that you haven't done anything with your machine and yet the SWAPPER.DAT file is already many thousands of bytes.

There are two new subdirectories created by the installation program: an \OS2 and \SPOOL subdirectory. These are used for the remainder of the OS/2 external program files and for the printer spooler, respectively.

If you do a directory on \SPOOL, you'll see that it is empty. OS/2 places a disk image of any files that you send to the printer in here until the print job is finished spooling. When the job is finished, OS/2 removes the file.

Let's look at a sample \OS2 subdirectory:

```
®MDNM¯
.          <DIR>       12-24-87   9:51p
®MDNM¯..         <DIR>       12-24-87   9:51p
INSTALL     <DIR>       12-24-87   9:53p
INTRO       <DIR>       12-24-87   9:57p
ANSI     EXE       9984  10-21-87  12:00p
ANSI     SYS       1694  10-21-87  12:00p
APPEND   EXE       6258  10-21-87  12:00p
ASSIGN   COM       1683  10-21-87  12:00p
ATTRIB   EXE      29080  10-21-87  12:00p
BACKUP   COM      49216  10-21-87  12:00p
BASIC    COM        585  10-21-87  12:00p
BASICA   COM      36253  10-21-87  12:00p
CHKDSK   COM      49232  10-21-87  12:00p
COM01    SYS       8758  10-21-87  12:00p
COM02    SYS      13366  10-21-87  12:00p
COMMAND  COM      25564  10-21-87  12:00p
COMP     COM      33056  10-21-87  12:00p
CREATEDD EXE      48032  10-21-87  12:00p
DISKCOMP COM      38400  10-21-87  12:00p
DISKCOPY COM      39456  10-21-87  12:00p
DOSCALLS LIB      29184  10-21-87  12:00p
```

```
EDLIN    COM       8135  10-21-87  12:00p
EGA      SYS       2110  10-21-87  12:00p
EXTDSKDD SYS       1877  10-21-87  12:00p
FDISK    COM      37824  10-21-87  12:00p
FIND     EXE      27424  10-21-87  12:00p
FORMAT   COM      47344  10-21-87  12:00p
GRAFTABL COM       7112  10-21-87  12:00p
HELP     BAT        444  10-21-87  12:00p
HELP     CMD        439  10-21-87  12:00p
HELPMSG  EXE      27408  10-21-87  12:00p
JOIN     EXE      21040  10-21-87  12:00p
KEYB     COM      12636  10-21-87  12:00p
LABEL    COM      26372  10-21-87  12:00p
LINK     EXE      91880  10-21-87  12:00p
MODE     COM      54932  10-21-87  12:00p
MORE     COM      48354  10-21-87  12:00p
MORTGAGE BAS       6380  10-21-87  12:00p
MOUSEA00 SYS      16950  10-21-87  12:00p
MOUSEA01 SYS      16438  10-21-87  12:00p
MOUSEA02 SYS      16438  10-21-87  12:00p
MOUSEA03 SYS      16438  10-21-87  12:00p
MOUSEA04 SYS      16438  10-21-87  12:00p
MOUSEB00 SYS      17462  10-21-87  12:00p
MOUSEB01 SYS      17462  10-21-87  12:00p
MOUSEB02 SYS      17462  10-21-87  12:00p
```

```
MOUSEB05 SYS      17462  10-21-87  12:00p
PATCH    EXE      37132  10-21-87  12:00p
POINTDD  SYS       5886  10-21-87  12:00p
PRINT    COM      28492  10-21-87  12:00p
RECOVER  COM      36928  10-21-87  12:00p
REPLACE  EXE      33568  10-21-87  12:00p
RESTORE  COM      54896  10-21-87  12:00p
SETCOM40 EXE       8654  10-21-87  12:00p
SORT     EXE      29946  10-21-87  12:00p
SPOOL    EXE      77370  10-21-87  12:00p
SUBST    EXE      21024  10-21-87  12:00p
SYS      COM      32056  10-21-87  12:00p
TRACE    EXE      10474  10-21-87  12:00p
TRACEFMT EXE      64432  10-21-87  12:00p
TREE     COM      28928  10-21-87  12:00p
VDISK    SYS       4662  10-21-87  12:00p
XCOPY    EXE      40656  10-21-87  12:00p
```

Most of these files look like the ordinary DOS command files, and apart from the changes needed to implement them in protected mode, they operate the same. (See Chapter 3 for a description and review of the differences between DOS and OS/2 commands.)

The automated install program has created two subdirectories in the \OS2 directory: \OS2\INSTALL, and \OS2\INTRO. The INSTALL directory is for the dynamic-linked library files and programs part of the automatic installation process themselves. The INTRO directory is for the programs which introduce you to OS/2.

5.4 How to clean up after the automated installation program

Once the automated installation procedure is done, you now take over to fisnish the job of configuring your machine. While you certainly can run your machine without any further effort, many of you may want to clean house and delete files which you do not need, and otherwise reorganize your hard disk to your own liking.

This is also a good exercise in learning how to configure OS/2 and a good introduction in some of OS/2's capabilities.

The installation program copies all possible files to your hard disk, including extraneous driver files.

The first set of files you can delete are the files in the \OS2\INTRO subdirectory, which are part of the "Introduction to OS/2" tutorial. These can be deleted once you have gone through the tutorial, or else sight unseen if you are an experienced DOS user. The tutorial is very basic, with information on how to use the OS/2 commands such as DIR, TYPE, COPY, and so forth.

Next to get the axe are the extraneous mouse driver files. Delete all the MOUSEXXX.SYS files from \OS2 EXCEPT the one required to run your system. If you do not use a mouse, then you can delete all of them.

Next you may want to delete the remaining driver files in the root directory which do not pertain to your system. If you are running a PS/2, you don't need the .SYS files DISK01, SCREEN01, CLOCK01, KBD01, and PRINT01, for example. (Or, if you are using a "classic" AT, you don't need the files with the "02" designation.)

If you are at the OS/2 command prompt, you can enter the following command to get rid of all of them at once:

```
DEL DISK01.SYS CLOCK01.SYS PRINT01.SYS KBD01.SYS SCREEN01.SYS
```

Make sure you use spaces between each file name. You have just demonstrated the simplest form of how OS/2 can do several tasks at once. (See page 2-61 of the IBM Standard Edition manual for more information on the Delete command.)

As a final chapter in your driver cleanup campaign, you may want to move any other driver files (such as for the enhanced graphics monitor or non-IBM equipment) to another subdirectory, such as your \os2 subdirectory. If you do this, you will need to change the configuration specified in the CONFIG.SYS "DEVICE =" statements. See section 5.7.13 below for further details.

If you want to at least cut down on the root directory's file overload, you can place all the .DLL files in the \OS2\INSTALL subdirectory by copying them this way:

```
COPY C:\*.DLL C:\OS2\INSTALL\*.*

DEL C:\*.DLL
```

We have other suggestions to clean up your root directory, and will mention them later when we discuss how the specific OS/2 configuration commands work.

5.5 How to make changes to your configuration

Now that you have cleaned house, it is time to tinker with your configuration, particularly with the CONFIG.SYS file. There is a great deal of information in this file, and you will probably want to experiment with various changes before you settle on the best configuration for your system.

You should be familiar with how to use a text editor or else the DOS mode EDLIN line editor to make these changes.

You did have a chance to make changes to your configuration during the automated installation procedure. However, that was a one-shot situation, coming fairly early in your understanding of OS/2. We recommended that you wait until after the installation dust has settled before you go in and starting moving furniture around.

Before you go any further, we suggest you review section 4 of the OS/2 Standard Edition manual for more information on these configuration commands that you are about to muck with.

There are two different sets of configurations for CONFIG.SYS parameters. One set is the "standard" set of defaults that are specified internally by the workings of the operating system. By default, IBM means that unless told otherwise, this is what OS/2 will assume for your configuration settings. If one of the configuration parameters is misspelled, or incorrectly specified, or omitted completely, OS/2 will set this parameter to the default. If you don't have any CONFIG.SYS file present in your root directory, OS/2 will set all configuration parameters at the default settings.

There are two standard types of defaults, one for systems which boot from floppy diskettes, and one for systems which boot from hard disks. We will mention the differences, although many of the parameters have the same defaults regardless of where you have booted your machine.

There is also another set of settings which we call the "guideline settings." These are automatically installed in your CONFIG.SYS file as part of the automated installation procedure. These are merely suggestions by IBM and can be changed to suit your own particular needs. To make things confusing, some of the guidelines differ from the internal OS/2 default settings. Again, we will mention the differences and try to clear the air of any confusion.

By now, you probably understand that you should not take anything OS/2 does for granted. If you are uncertain about the value of a parameter, you should always specify it in your configuration file to remove any ambiguity.

We have summarized these configuration commands and what each one does in Table 5.2.

Table 5.2 OS/2 Configuration Commands

Command	Parameters*	Purpose
BREAK	ON/OFF	Sets Ctrl-Break checking for real mode only.

BUFFERS	number	Number of disk buffers in memory.
CODEPAGE	code	Selects a language-specific code page.
COUNTRY	code	Selects country-dependent conventions for displaying time, date and currency.
DEVICE	file name	Installs a device driver.
DEVINFO	device,file	Prepares device to accept code page tables.
FCBS	number	Number of File Control Blocks that can be
IOPL	YES/NO	Allows or disallows access to I/O hardware if a process requests it.
LIBPATH	path list	Specifies locations of Dynamic Link Libraries
MAXWAIT	seconds	Maximum idle time before a process receives
MEMMAN	[NO]SWAP [NO]MOVE	Sets options for virtual memory management.
PRIORITY	ABSOLUTE/ DYNAMIC	Sets scheduling options.
PROTECTONLY	YES/NO	Enables running a real-mode sessiwn
PROTSHELL	file name	Protected-mode command processor.
RMSIZE	kilobytes	Size of real-mode partition.
RUN	filename	Start up a process at system initialization.
SHELL	file name	Real-mode command processor.
SWAPPATH	path	Location of file for swapped-out memory.
TIMESLICE	multiseconds	Minimum and maximum timeslice.
THREADS	number	Maximum concurrent threads system-wide.

* parameters in uppercase are entered literally; parameters in lower case indicates values entered by the user.

5.5.1 Default configuration for machines booting from hard disks

Let's look at each collection of parameters. First we show the default settings for those machines which start OS/2 from hard disks. The configuration parameters themselves are in ALL CAPS, while their values are in all lower case (or in numbers):

BREAK = off

BUFFERS = 3

COUNTRY = 001

CODEPAGE = 437

FCBS = 16,8

IOPL = no

LIBPATH = c:\

MAXWAIT = 3

MEMMAN = swap, move

PAUSEONERROR = yes

PRIORITY = dynamic

PROTECTONLY = no

PROTSHELL = dmpc.exe shell11f.cnf shell11f.exe cmd.exe

RMSIZE = (see note below, section 5.7.9)

SHELL = command.com /p

SWAPPATH = c:\

THREADS = 64

TIMESLICE = 32,248

5.5.2 Default configuration for machines booting from floppies

The floppy-disk defaults are the same, except for the following three parameters:

LIBPATH = a:\

MEMMAN = noswap, move

SWAPPATH = a:\

5.5.3 Guidelines produced by the automated installation program

Next let's look at the guidelines produced by the automated installation program. These will be slightly different, depending on what kind of configuration you have specified during the question-and-answer period prior to your automated installation. Let's illustrate one set of guidelines, for an AT which starts from its hard-disk. Other examples are included at the end of the chapter in our sample configuration files.

PROTSHELL = dmpc.exe shell11f.cnf shell11f.exe cmd.exe /k os2init.cmd

LIBPATH = c:\;c:\os2;c:\os2\install;

BUFFERS = 30

MAXWAIT = 3

MEMMAN = swap,move

PROTECTONLY = no

SWAPPATH = c:\

THREADS = 64

SHELL = command.com /p

BREAK = off

FCBS = 16,8

RMSIZE = 640

RUN = c:\os2\spool.exe /d:lpt1 /o:lpt1

DEVICE = c:\os2\com01.sys

You'll notice that some commands have different guidelines from their default settings (PROTSHELL, BUFFERS, LIBPATH) and some new parameters have been added (DEVICE, RUN).

There is a lot of stuff here to figure out. Those of you that thought FILES and BUFFERS were too much are going to have to spend some time here. If your machine seems to be running well at this point and you want to remain in the dark as to what each of these parameters does, fine. You can skip ahead to the next chapter at this point.

Those of you that are curious, or who want to learn more about tweaking your configuration, should continue on, however.

5.6 Configuration Checklist

To assist you with making some intelligent choices in how to change your configuration, we have prepared a Configuration Checklist. The checklist gives you a broad overview of your configuration, and gives you a roadmap for what specific OS/2 configuration commands you need to change.

We describe eight basic decisions you'll need to make before you can configure your system. As you can see, OS/2 has lots of different parameters. The Checklist will help you understand the "big picture" before you get down into the guts of the CONFIG.SYS file:

5.6.1 Do you want to run both DOS and protected modes?

Your first decision is whether you want to run both DOS and protected modes together in the same computer, or just use the protected mode. Initially, most of you probably will want to run both, running some of your existing DOS software when you bring up OS/2. This affects the commands PROTECTONLY and RMSIZE.

5.6.2 What will be your swapping strategy?

The commands MEMMAN and SWAPPATH select how to configure your system so that protected applications can share the same physical memory. You can swap programs out of RAM and into a file on your hard disk if you want to take advantage of this feature.

5.6.3 Where do you want to locate your protected mode and DOS mode shell and processor files?

In DOS days, you usually never had to worry about where your command processor file was located. You usually took it for granted that it would be in the root directory. Under OS/2, this is more complex and you need to worry about which subdirectory houses your Presentation Manager user interface programs, along with where both the DOS and OS/2 command processors are also located.

Commands required for this navigation are PROTSHELL and SHELL.

5.6.4 Where do you want to locate your dynamically linked libraries?

OS/2's library files can be called by any running program, and are loaded dynamically, as needed by a program. This command to show OS/2 where you have put the libraries is LIBPATH.

5.6.5 What device drivers are needed to run your system?

OS/2 is very fussy about individual devices that need attachment. We cover this in a separate section in Chapter 6, but the command is DEVICE in the CONFIG.SYS file.

5.6.6 Are you running your machine in another language?

IBM has built-in flexibility for OS/2 to make it easier for international users to display the appropriate character sets, including special characters and accent marks.

In previous DOS versions, international users had to put a variety of commands both in CONFIG.SYS and in their AUTOEXEC.BAT files to install the character symbols for their language. OS/2 has consolidated this information into these three CONFIG.SYS commands: CODEPAGE, DEVINFO, and COUNTRY. We will not get into the specifics of how to install these commands on your system, but refer you instead to the IBM OS/2 documentation.

5.6.7 What other protected mode commands do you need to set?

There are various other OS/2 protected mode commands which need some tweaking to increase your system's performance. These include BUFFERS, IOPL, PRIORITY, MAXWAIT, TIMESLICE, and THREADS.

5.6.8 What other DOS mode commands do you need to set?

And there are two DOS-mode only commands which include FCBS, and BREAK = ON, both leftover from the days of DOS 3 and applicable only to DOS-mode programs.

5.7 CONFIGURATION COMMAND SETTINGS

Let's go over each configuration command and explain what each does and how to match the command with your system's requirements. We also indicate the internal OS/2 default settings and the automated program's suggested guidelines and whether each command is valid in both DOS and OS/2 modes or only in one mode.

5.7.1 PROTSHELL (OS/2 mode only)

You must tell OS/2 where certain system files are located, regardless of how you boot your machine. This is a something very different from the old days of DOS. If you ran DOS from your hard disk, you placed the system files on the root directory with a FORMAT/S command and off you went. If you ran

DOS from a floppy, you would format a floppy with FORMAT/S or else you could use the SYS A: command to get the system files placed on the floppy.

OS/2 is not so simple, and requires commands in the CONFIG.SYS file to keep everything straight.

The PROTSHELL the command tells OS/2 where to loads the user interface shell program along with the OS/2 command processor. Because this command has nothing to do with the DOS mode of your computer, it is only valid when you are running in the protected mode. This command has a different automated guideline setting (PROTSHELL = dmpc.exe shell11f.cnf shell11f.exe cmd.exe /k os2init.cmd) from the default OS/2 internal setting (PROTSHELL = dmpc.exe shell11f.cnf shell11f.exe cmd.exe). Let's explain what each of these items in the PROTSHELL parameter means:

DMPC is the Standard Edition user interface, and the remainders of the line are its companion programs to execute this interface. CMD.EXE is the OS/2 command processor, and the /K option tells OS/2 to load whatever is the following command file every time you select a new OS/2 session from the OS/2 Program Selector shell. The protected-mode command processor consists of the file CMD.EXE and two associated hidden files, IBMBIO.COM and IBMDOS.COM. This is similar to the way DOS worked, with the system file COMMAND.COM and its two hidden files (which have the same names IBMBIO.COM and IBMDOS.COM, although are much smaller in size).

Note that the installation program puts a guideline command file OS2INIT.CMD as part of the PROTSHELL command, while the actual OS/2 internal default is not to use any command file to start-up each OS/2 session. IBM's guideline OS2INIT.CMD looks like this:

```
PATH C:\;C:\OS2;C:\OS2\INSTALL;

DPATH C:\;C:\OS2;C:\OS2\INSTALL;

CALL HELP ON
```

These set up the search paths for external OS/2 commands (PATH) and for data files (DPATH). The CALL command runs a command file (called HELP.CMD, located in your \OS2 directory) which turns on a simple help bar at the top of your screen. This command file looks like this:

```
@echo off

: SCCSID = @(#)help.cmd          5.1 87/06/17

if %1.==. goto msg

if %1 == on goto yes

if %1 == off goto no

if %1 == ON goto yes

if %1 == OFF goto no
```

```
if %1 == On goto yes

if %1 == oN goto yes

if %1 == OFf goto no

if %1 == OfF goto no

if %1 == Off goto no

if %1 == oFF goto no

if %1 == oFf goto no

if %1 == ofF goto no

helpmsg %1

goto exit

:msg

helpmsg

goto exit

:yes

prompt $i[$p]

goto exit

:no

cls

prompt

:exit
```

If you don't like the help bar at the top of your screen, you can eliminate do one of several things. First, you can change the "PROMPT $i[$p" statement in the above command file. The $i parameter is the one that produces the help bar. You can also remove the CALL statement from the OS2INIT.CMD file, or make any changes to the HELP.CMD file to provide your own helpful messages or startup routines. Of course, you can change everything that we have listed here: the name of the files and the commands in them. This is just what the automated installation program sets up for you.

You can modify this PROTSHELL command line (along with the command file itself) in several ways. If you want to get rid of even more programs that are cluttering up your root directory, you could specify a path name for each and every program name in this command line.

For example, if all the programs are in the \OS2 directory, the command would look like this:

PROTSHELL = \os2\dmpc.exe \os2\shell11f.cnf \os2\shell11f.exe cmd.exe /k \os2\os2init.cmd

You can also use whatever name you'd like as the startup command file, and make whatever changes you'd like to the command file.

5.7.2 SHELL (DOS mode only)

Just as PROTSHELL tells OS/2 where to look for its protected mode command processor (among other things), SHELL tells OS/2 where to look for the DOS-mode command processor, otherwise known as COMMAND.COM. Note that this COMMAND.COM, while the same name as the DOS processor, is a special file which comes included on your OS/2 program diskettes.

The SHELL command is similar to the old DOS SHELL format, and the default is SHELL = command.com /p.

If you want to specify someplace other than the root directory, you need to put the full path in the SHELL command as well as in the SET COMSPEC command in your DOS-mode AUTOEXEC.BAT start-up file. If your COMMAND.COM is located in your \os2 subdirectory, the two commands would be:

SHELL = c:\os2\command.com /p (which goes in CONFIG.SYS) and

SET COMSPEC = c:\os2\command.com (which goes in AUTOEXEC.BAT)

5.7.3 LIBPATH (OS/2 mode only)

This command tells OS/2 where to look for those dynamic-linked libraries. Since this concept is foreign to DOS, LIBPATH is only valid when your machine is running in the OS/2 protected mode. If you just followed our instructions, you moved them to the \OS2\INSTALL subdirectory, which you see is one of the subdirectories specified in the command line. If all goes well OS/2 should be able to find them there the next time you boot.

You probably will change this list over time, as you install other OS/2 applications with their own libraries. Just remember to either copy all the .DLL files to one of these subdirectories, or else add another entry in the list.

Once again, this command has one setting which is the internal OS/2 default (LIBPATH = c:\ or a:\, depending on whether you start from a hard disk or floppy, respectively) and one which is the guideline installed by the automated installation program (LIBPATH=C:\;C:\OS2;C:\OS2\INSTALL;).

5.7.4 BUFFERS

This is an old friend from the DOS days: the 512 k byte blocks of RAM that OS/2 uses to read and write data that is less than a whole disk sector. Buffers can vary between 1 and 100. The automated installation program uses a guideline of 30 while the internal OS/2 default setting is 3.

If you have lots of programs running at once in your machine, you may well want to up the ante over 30 to get better performance.

The BUFFERS statement is used in both the DOS and protected modes of operation. Notice that there is no OS/2 equivalent of the FILES command in the CONFIG.SYS file. The FILES command is no longer used by OS/2. See section 5.7.14 for an explanation of why this is the case.

5.7.5 MAXWAIT, PRIORITY, and TIMESLICE (OS/2 mode only)

These commands are used to determine how OS/2 splits up the action at the processor level. MAXWAIT is the maximum amount of time in seconds a thread will wait before getting bumped up in PRIORITY level. TIMESLICE refers to the minimum and maximum amounts of time in milliseconds that the processor time can be allocated to run particular tasks.

MAXWAIT can vary between 1 and 255 seconds, with 3 seconds the default along with the guideline value selected by the install program. If a program's threads have not received any action by the CPU for this number of seconds, it gets a boost in priority from OS/2. If you have many tasks running in your machine together, the MAXWAIT parameter is one way to ensure that no single program will be left out in the cold, waiting for processor cycles while everyone else is enjoying the party.

PRIORITY can either be set equal to dynamic (the default) or absolute. The OS/2 internal default is to assign priorities dynamically according to the whims of the operating system and what other programs are in charge at the moment. If set to absolute, OS/2 will not vary the priorities as they are running.

Finally, TIMESLICE has two values, one for the minimum (greater than 32 milliseconds) and one for the maximum. The maximum value must be greater than the minimum value specified but less than 65536 milliseconds. The syntax and defaults of the command are:

TIMESLICE = 32,248

IBM recommends keeping the minimum parameter set at 32. Unless your software tells you otherwise, we recommend you use the defaults. This command is not inserted into your CONFIG.SYS file by the automated installation program, so the effectively the guidelines match the default settings.

5.7.6 MEMMAN and SWAPPATH (OS/2 mode only)

These commands select how to configure your system so that protected applications can share the same memory. Because OS/2 allows for virtual memory of 1 G byte, but only controls physical memory up to 16 M bytes, your system needs to know how to swap things out of memory and onto a hard disk when protected-mode programs exceed your machine's physical memory.

If you don't have a hard disk, you don't want to do this swapping. The reason has to do with the poor performance of floppy disk i/o, along with the fact that there is very little room on a floppy to store the excess program code.

But if you do, and you decide to swap, then you should be aware that occasionally OS/2 will be checking your hard disk to see what it has put out there. This can be initially somewhat disconcerting (if you spend a lot of your time looking at the hard disk light), but don't worry: it is perfectly normal. (OS/2 will also copy files to your hard disk during the printing process. This is also normal and part of the SPOOL command. See section 5.7.11 and chapter 7 for more details.)

SWAPPATH describes the actual path that will be used to temporarily store files when information is swapped out of memory and to your hard disk. IBM has this path defaulting to your root directory of your hard disk (c:\) or to the root directory of your floppy drive (a:\).

The swapping program creates a file called SWAPPER.DAT, and places it in your root directory. This file can grow to take over your hard disk as programs are swapped in and out of memory.

We are uncomfortable (for purely aesthetic reasons) with having such a huge file in the root directory, and recommend that you change the SWAPPATH parameter as follows:

SWAPPATH = c:\spool

(More on spooling below.) If you do this, you can delete the SWAPPATH.DAT file in your root directory, but only after you reboot your machine. This is because the present swap data file is in use by OS/2. Any files in use by the operating system can't be deleted by you. But the next time you boot your computer, OS/2 will create a new SWAPPATH.DAT file in your \SPOOL directory, and then you can delete the one in your root.

Memman command is more complicated, and has three sets of choices, depending on your configuration and potential applications: swap, move; nowswap,nomove; and noswap,move. Swap permits segment swapping, while move permits storage compaction. If you swap segments out of memory, compacting them is automatically enabled.

Systems which start from floppy diskettes should use MEMMAN = noswap, move. This is because there is no sense in swapping out programs from memory to a floppy diskette -- performance would be awful. Likewise, hard-disk systems could make use of the swap file, so their configurations should use MEMMAN = swap,move.

IBM recommends that systems being used for process control, or other time-dependant tasks, may want to use MEMMAN = noswap,nomove as their configuration. This is because the swapping and compaction of segments takes some time away from the processing of other tasks in the system.

Here is a chart to keep everything straight:

		MOVE	
		No	**Yes**
	No	Process Control (MEMMAN=noswap,nomove)	Floppy-based systems without hard disks (MEMMAN=noswap,move)
Swap			
	Yes	(Not a legal combination)	Hard disk-based systems (MEMMAN=swap,move)

Because these concepts are not part of the DOS world, the SWAPPATH and MEMMAN commands are only used when your machine is executing in protected mode.

5.7.7 THREADS and IOPL (OS/2 mode only)

These two commands are also used to configure special OS/2 activities in your machine. Again, since they have no analogues in DOS, they are only used when you are running in the protected mode.

THREADS refer to the maximum number of threads, or independent actions, which can run in your machine at once. It may vary between 32 to 255, and the default (both by OS/2 internally as well as the installation guideline) is 64.

IOPL allows input/output privilege to be granted to running protected mode programs. This bumps up the priority of running programs somewhat, so they can directly communicate with I/O devices, such as a communications or printer port. The default is IOPL = no and this command is not installed automatically into your guideline CONFIG.SYS file.

5.7.8 PROTECTONLY

If you still want to run some of your existing DOS applications, you will need to set this command to no to run both real and protected modes. You do this by setting PROTECTONLY= no in the CONFIG.SYS file. This setting is both the default and the selected guideline from the automated installation program.

IBM cautions you not to be so quick to forget about your DOS world and set this parameter to "yes." The caution is based on a lack of a protected-mode text editor, making it harder to change your configuration without a DOS mode session.

The only way to change configurations is rather messy: First, you need to rename the CONFIG.SYS file to something else, such as CONFIG.OLD. Then reboot at which point, since PROTECTONLY defaults to no, you would get back into DOS mode and could make changes to the CONFIG.OLD file with EDLIN, the line editor, or one of your own DOS text editors.

The consequences of selecting PROTECTONLY = no means that part of your system memory will be set aside for DOS. This is done with the RMSIZE command below.

5.7.9 RMSIZE (DOS mode only)

This command sets up how much memory the DOS mode will use and is only vaild if you have set PROTECTONLY = no. Since most of you are used to running software in the full 640 k bytes of memory,

we recommend setting RMSIZE to 640. However, a lower value can be used if your existing DOS applications can all fit in a smaller memory area.

OS/2 takes the memory required to support its DOS mode "off the bottom." In other words, the first RAM to be allocated is allocated to DOS mode applications. Any additional memory left-over is allocated to the protected mode. Each mode has its own memory area; they are not shared -- although all the PROTECTED mode applications share the same memory space.

The RMSIZE parameter is calculated with a complex formula. It is the lower of the following two numbers: either the total system memory minus 512 k bytes, or either 512 or 640 k bytes, whichever is installed in the lower memory address range. For example, suppose you have 1.5 M bytes of memory, with 640 k installed as base memory and the remainder as extended memory. Then OS/2 will default RMSIZE to 640 k bytes of memory for DOS applications. If you only have 512 k bytes (or less) installed on your system board, then regardless of what RMSIZE parameter you use you will only get 512 k for your DOS session.

This can be a painful realization, especially if you were counting on running OS/2 with a fully-fledged 640 k DOS session. Actually, even if you could get the full 640 k with RMSIZE, you still would have less than 640 after you load in all your device drivers. This is because OS/2 loads all drivers first, since they need to be used by both protected and DOS mode sessions.

5.7.10 BREAK and FCBS (DOS mode only)

These two commands have the same usage as their DOS 3 versions. BREAK = on lets the DOS-mode command processor stop a batch file from executing whenever a CTRL-BREAK is pressed. (This is always the case in protected mode and cannot be changed by you.) The default is BREAK = off.

FCBS has two arguments: to the number of file control blocks that are allowed to be opened concurrently (from 1 to 255) and the number that cannot be closed automatically (from 0 to the first number). Like BREAK, this command is only relevant for DOS mode and has no effect on protected mode operations.

The default is FCBS = 16,8. Defaults of both commands are inserted into the guideline CONFIG.SYS file as part of the automatic installation.

5.7.11 RUN

This command executes a program specified in the command line at system startup. It is similar to the command file executed by the /k option of the PROTSHELL command, with one important difference. The file named in the RUN command is begun before the user interface is executed, while the one named in PROTSHELL is begun after the interface is loaded.

This is true regardless of the order of the RUN and PROTSHELL commands in your CONFIG.SYS file, so if RUN comes after PROTSHELL (as it does in the guideline configuration installed automatically), it still gets processed first. Also, all RUN commands are processed after all other DEVICE and other configuration commands, but before the PROTSHELL command.

Placing a program in the RUN line means, therefore, that it must do its own user interface. Also, if there is any disk error handling, it must suspend the OS/2 error handling while it checks for its own errors.

More than one RUN command can be placed in CONFIG.SYS, in which case they are processing in the order specified.

IBM uses the RUN command to process its print spooler. If you ask for the print spooler during the automated installation, the following command will be placed in your CONFIG.SYS file:

```
RUN = c:\os2\spool.exe /d:lpt1 /o:lpt1
```

This means locate the SPOOL program in the \os2 subdirectory, and use the print device located at LPT1 (where a program thinks it is printing to) along with the output print device located at LPT1 (where OS/2 is actually going to send the print job). As we mentioned earlier, the installation program also creates a \spool subdirectory to temporarily store the spooled print jobs until they have finished printing.

If you wanted to spool a serial printer, (or what you might have called redirecting output to a serial printer) you would change the output device to a serial port, as follows:

```
RUN = c:\os2\spool.exe /d:lpt1 /o:com1
```

(You would also need to define the communications port device driver in your CONFIG.SYS file, by using the statement:

```
DEVICE = c:\os2\com01.sys
```

You may also need to specify the MODE command in the initial startup file specified in PROTSHELL. See Chapter 3 for details on MODE.)

5.7.12 OTHER CONFIG.SYS PARAMETERS

CODEPAGE, DEVINFO, COUNTRY: As mentioned earlier, these control the display of international character sets. None of the three are placed in your CONFIG.SYS by the automated program, unless you indicate you are an international user. The defaults are CODEPAGE = 437 and COUNTRY = 001.

DISKCACHE: This sets up control information for the disk caching utility, which is only available on the PS/2 family. This utility speeds up disk input/output, by temporarily writing and reading frequently-used information to an area of RAM storage set aside as the cache. The size of the cache is specified in k bytes, from 64 to 7200 k. This cache is maintained in RAM, so anything specified in this command takes away system memory from programs.

The internal OS/2 default is no cache whatsoever. This is different for the installation program's guidelines, which leave no cache for the AT and XT/286 family of computers. This is because IBM does not support caching on these machines. Either set DISKCACHE=64 or =128, depending on whether you have a hard disk of 20 M bytes or greater than 20 M bytes, respectively.

TRACE and TRACEBUF: IBM has invented these commands to make it easier for service technicians to trace problems with your machine. Once TRACE = on is set, various events can be tracked down with these service people's help. TRACEBUF sets up the size of the buffer, between 1 k and 63 k bytes, which records the problems to a file on disk. These CONFIG.SYS commands work in conjunction with the OS/2 commands TRACE, TRACEFMT, and CREATEDD.

PAUSEONERROR: When this command is set to yes, (this is the default) OS/2 will pause when it finds an error in your CONFIG.SYS file so you can read the error message. OS/2 doesn't continue until you press the ENTER key. The system won't pause for just warning messages, which are not considered error messages for this command.

The automated installation program does not place this command in your CONFIG.SYS file, so that effectively the guideline setting is also "yes."

5.7.13 DEVICE DRIVERS

We are almost finished describing the configuration of your OS/2 machine. However, we have saved the best for last: the use of device drivers. These drivers were introduced with earlier versions of DOS and play an even more important role in OS/2.

Each peripheral needs its own device driver specified in CONFIG.SYS, including mice, enhanced graphics monitors, communications ports, RAM and external disks, and other miscellaneous equipment. Some of the drivers, such as for mice, may already be familiar to you. Others, such as for the communications adapters, may not.

All device drivers locate the particular driver file by full pathname, including subdirectory. They do not need to be placed in the root directory, although this is done by the automated installation program. For our discussion, we show the drivers located in the root directory.

We cover what equipment is specifically supported by device drivers in Chapter 6, but for now let's identify the types of device driver statements you might find in your CONFIG.SYS file as a result of the recommended guidelines as part of the automated installation program.

MOUSE DRIVERS. Mice need two drivers files in CONFIG.SYS, as the sample below shows:

```
DEVICE = c:\pointdd.sys

DEVICE = c:\mousea01.sys, serial=com1, mode=b, qsize= 10
```

The first device driver is for pointing devices, while the second tells OS/2 the specific type of mouse you are using. The phrase "mode=b" means the mouse is active in both protected and DOS modes. The "qzise=10" phrase refers to the size of the queue buffer space which can be allocated for each screen group.

IBM warns you that none of the earlier DOS mouse drivers which were loaded in CONFIG.SYS with a .SYS extension are supported. Other drivers, which loaded by running their own .COM or .EXE program files,

may work but then only in the DOS mode. The best bet is to use one of the supported mice and to load the driver as IBM recommends.

ENHANCED GRAPHICS MONITOR DRIVER. If you have an enhanced graphics monitor, you should have the following driver in your CONFIG.SYS:

```
DEVICE = c:\ega.sys
```

If you are not using a true-Blue IBM EGA on your machine, you might receive an error message when OS/2 tries to install this device driver. IBM only supports IBM equipment working with OS/2.

[check the following graph with IBM]

The EGA driver only supports a mouse in three of the video modes of the EGA: mode 14, with 640 x 200 resolution of 16 colors; mode 15, with 640x 350 monochrome resolution; and mode 16, with 640 x 350 resolution with 64 colors. Does this mean that other modes will work, or just that they won't work with the mouse? Confusion here.

COMMUNICATIONS PORT DRIVERS. OS/2 also requires drivers for its communications ports. This driver supports the actual RS-232 interface itself, rather than anything that is attached to the interface (such as a modem or a serial printer). Separate programs are still needed to tell OS/2 how to support these devices.

If you told the automated installation program to add a communications driver, you will see a statement like the following in your CONFIG.SYS file:

```
DEVICE =c:\com01.sys
```

As with disk and keyboard drivers, the "01" refers to the IBM AT and XT/286 family and supports two ports: COM1 and COM2. A COM02.SYS driver would be used if you had a PS/2 computer, which supports three comm ports: COM1, COM2, and COM3. Note that only one COM driver is needed to support all communications ports on your computer.

Again, as with the EGA driver, if you are not using an IBM asynchronous adapter but someone else's (such as an internal Hayes modem), you might get an error message when this OS/2 tries to install this driver.

RAM AND EXTERNAL DISK DRIVERS. A RAM (VDISK) disk means you use part of your system's extended memory as a "virtual" disk drive, so that programs stored to this RAM disk can execute faster. An external disk driver (EXTDSKDD) lets you access a real floppy disk with a logical drive letter This allows you to install certain IBM disk drives which are external to the main computer cabinet, such as the 5 1/4 drive for the PS/2s. VDISK drivers should be listed after the external drivers, if you want to eliminate confusion over the logical letter assignments of the virtual RAM disk with your real disk drives.

The ways both these drivers are specified are similar to the DOS 3 world. There are many possible specifications of these utilities. The following statements are typical.

DEVICE = c:\extdskdd.sys /d:2 /t:80 /s:9 /h:2 /f:2

DEVICE = c:\vdisk.sys 320,512,64

OTHER MISCELLANEOUS DRIVERS. If you have programs which require the extended keyboard and display support, you will need two commands, one the familiar

DEVICE = c:\ansi.sys

This just provides DOS mode support of ANSI devices. Support under protected mode is under the control of a separate ANSI command which is issued from a command file or at the OS/2 prompt. The command has two conditions, "ANSI = on" (the default) and "ANSI = off" to turn off the support.

There may be other drivers in your CONFIG.SYS file, depending on what non-IBM equipment you have installed. Again, please check Chapter 6 for more specific details.

5.7.14 DOS 3 CONFIGURATION COMMANDS NOT SUPPORTED UNDER OS/2

Three of your favorite DOS 3 configuration commands are no longer supported under OS/2: FILES, LASTDRIVE, and STACKS. Specifying FILES and STACKS are no longer meaningful in the OS/2 world, since several programs can run simultaneously inside your machine in the protected mode. Previously, these parameters were a primitive attempt at multitasking and are now internal to the operating system. The old DOS FILES command set the maximum number of file handles that could be opened at once by DOS. The same was true of the STACKS command. Under OS/2, both the number of stacks and the file handle tables can grow dynamically, as needed by the operating system. This means that the number of open files or stacks are no longer limited by the system.

LASTDRIVE is also no longer valid. With the external drive assignment command (EXTDSKDD), you can define as many logical drives as you wish. OS/2 will assign the next drive letter according to the method shown in the IBM manual.

5.8 USING STARTUP COMMAND FILES AND AUTOEXEC.BAT

Now you have some understanding of the configuration parameters contained in the CONFIG.SYS file. But we are not finished: we still need to look at the three types of startup files you'll be using with OS/2.

There are two different startup files in the protected mode, and one for the DOS mode. Let's look at each one. For further discussion, we recommend chapter 3 of your OS/2 manual.

5.8.1 STARTUP.CMD

The first protected mode startup file is called STARTUP.CMD. As we mentioned earlier, all protected mode batch files are called command files and have a separate .CMD extension to distinguish them from DOS batch files with their .BAT extensions. The .CMD files are still straight ASCII files, with commands processed sequentially just as they are in DOS .BAT files.

We mention the differences between DOS and OS/2 commands in Chapter 3 in detail.

This protected mode startup file is executed only once: when your machine boots up, and only for the first OS/2 protected mode session. The file must be in your root directory.

Your STARTUP.CMD file can start several other protected mode sessions, by using the START batch command. However, it cannot start any DOS mode sessions. The only way to do that is to use the program selector or by hitting the ALT-ESC keys to switch among running sessions.

If you don't have any startup file, OS/2 will bring you to the main screen of the program selector or Presentation Manager. If you run the same software every day, you might want to specify a startup file so whenever you turn on your computer you can go right to your software without typing a keystroke. The automated installation program does not create any startup command file.

We have some sample startup files illustrated in the next section.

5.8.2 INITIAL ENVIRONMENT COMMAND FILE (OS2INIT.CMD)

The second startup file used in protected mode is the OS2INIT.CMD which is specified in the PROTSHELL line in your CONFIG.SYS file. This command is executed every time you bring up a new protected mode session. IBM recommends you place path navigation commands (the PATH and DPATH statements), along with any changes you might need to your environment variables in this command file. You can choose any name, provided it has a .CMD extension, and any location on your hard disk, provided that you specify the complete path name for the file in the PROTSHELL line.

The automated installation program creates these commands in an OS2INIT.CMD file:

```
PATH c:\;c:\os2;c:\os2\install;

DPATH c:\;c:\os2;c:\os2\install;

CALL HELP ON
```

5.8.3 AUTOEXEC.BAT

The last startup file is the trusty old AUTOEXEC.BAT. Since it is a batch file, it is only used to startup your single DOS mode session. It needs to be in the root directory, just as before. [CHeck ans see if when you change SHELL does AUTOEXEC need to follow it along???}

The automated installation program creates the following AUTOEXEC.BAT file in your root directory:

```
PATH c:\;c:\os2;

CALL HELP ON
```

This CALL statement references a batch file HELP.BAT which looks the same as the HELP.CMD file shown above. If you want to change this, perhaps by substituting your old DOS 3 AUTOEXEC file, you can do so.

There is one important difference between the help command file and the help batch file, and that is the way each file has set the prompt for OS/2 and DOS, respectively. Now, you may think that we are

making a big deal here out of something relatively minor, but we recommend you use different prompts to help you distinguish when you are talking to DOS and when you are talking to OS/2.

The easiest way to do this is to use the ANSI screen control escape sequences. If you don't know to use them, you should consult the OS/2 manual for more details. One suggestion is to include the following command line in your AUTOEXEC.BAT file, so that the DOS mode comes up in a different color than the other protected mode sessions.

```
PROMPT [DOS $P$G]$e[1;33;44m
```

[check this out]

However, the default prompt that IBM has set for the DOS mode looks like:

```
C:\
```

and is set with a "prompt pg" statement. For the protected mode of OS/2, the prompt is set with a "prompt $i[$p] command and looks like:

```
[C:\]
```

For more information on using the prompt command, see section 2 of the OS/2 manual.

As we mentioned above, OS/2 does not automatically start any DOS sessions. You still need to go to the Program selector and tell OS/2 to start a DOS session. This means that you can't completely automate any DOS mode applications, unless you insert a bootable DOS disk in drive A and start your machine from this floppy.

5.9 SAMPLE CONFIG.SYS FILES

To help you getting started, we have created some sample batch files and other routines that you may copy. Bear in mind that these are only samples, and you should feel free experiment with any changes on your configuration. Because of the sheer number of configuration parameters, you have tremendous flexibility with your individual PC.

[to come]

Chapter 6. Making OS/2 work for you

You now should have some understanding of the concepts of OS/2, how to install it on your system, and what some of the problems you'll face making the transition from the world of DOS.

This chapter explains how you can make OS/2 work with your existing and future applications, and helps provide some understanding of the implications for running multiple protected-mode applications in OS/2.

6.1 Checklist for application migration

But first let's look at the "big picture" in terms of what steps are needed to migrate DOS-based applications towards full-fledged participants in OS/2's protected mode.

We mention this checklist as guidance; not as assembly language programming instruction. Our aim here is not to teach programming to applications developers but to point out the obstacles and critical concerns for those software users. (See the appendix for a list of other OS/2 books that are more programming-oriented.) Armed with this knowledge, you as a consumer will be better informed to understand how to convert your applications into OS/2.

There are five important items to keep in mind:

6.1.1 Hot key conflicts

OS/2 uses several keystroke combinations for its "hot keys." This means that whenever these keys are pressed, even during running DOS mode applications, they will signal OS/2 to do some task.

The most important combinations are CTRL-ESC and ALT-ESC. These signal OS/2 to switch among its running applications, or to invoke a main menu of possible applications which can be started. (More on these keys latter in section 6.3.) But there are others, such as CTRL-NUM LOCK and CTRL-ALT-PRTSC which pauses the display and send jobs to the printer.

These key combinations can't be used by any protected mode applications for their own functions.

IBM says in its OS/2 documentation that any software program which uses these keystroke combination for actual application functions "may give unexpected results." This means that your existing DOS applications may have some trouble interpreting these reserved keystroke combinations when they are run in the DOS mode of OS/2.

One example of an "unexpected result" could be the application tries to process the keystroke to do its function, then passes the keystroke on to OS/2's DOS mode processor. You might get an error message, or your application might try to execute the command it thinks it got from the keyboard. You may or may not be able to perform the intended OS/2 function.

One place this might be a problem is with any pop-up keyboard enhancers that are terminate-and-stay resident (TSR). These TSR type of programs may or may not work under the DOS mode of OS/2. They certainly will not work under the protected mode.

Actually, TSR software won't be needed under protected mode. Multiple concurrent applications is the rule, rather than the exception for OS/2. Any program (so-designed) can run in the background of protected mode, and call on operating system services whenever it needs those services. Moreover, since the programming rules of OS/2 are stricter than with DOS, these pop-up programs are prevented from colliding with each other and worry about who is in control over the foreground application.

Therefore, while you as a consumer of this type of software will probably need to upgrade to newer, protected-mode, versions, the good news is that you will not lose any functionality of the actual programs themselves.

We suggest trying out these hot-key sequences with all the DOS applications you intend to run in OS/2's DOS mode as early as possible after your OS/2 purchase. This will be one of the first tests for compatibility and will help determine how much conversion is needed to bring the application fully into protected mode.

6.1.2 Changes needed to run family applications

A second item for your consideration is how your application will interact with the operating system. OS/2 has greatly expanded on the number and types of system services available to software applications. These services, called application programming interfaces or APIs, are divided into two groups: those that are specific to the protected mode and those which can work equally well in either protected or DOS modes.

The latter sets of interfaces are called the Family API. Applications which take advantage of just these interfaces and no others can run in any of three environments: "pure" DOS 3.3, OS/2's DOS mode, or OS/2's protected mode.

A few caveats are necessary, however, for the family application to run under both the OS/2 DOS mode as well as under DOS 3.3 on 8088-based processors. First, all Family applications must not use any 286- or 386-specific assembly language instructions. Second, it must be able to fit into 640 k bytes of memory, since that is the limitation under DOS. Third, not all features of all Family APIs are supported. Finally, it cannot take advantage of any of the other OS/2 features, such as multitasking, local environment settings, or the windowing features of the Presentation Manager. This means that Family applications must run in a full-screen window (even when they run in protected mode), similar to the way current DOS applications run in the DOS mode of OS/2.

If these rules are followed, the application will execute fine on any DOS 3.3-based PC, in addition to running under the DOS mode of OS/2.

This multi-mode flexibility is possible because the application is compiled with a special translation mechanism. This piece of software translates OS/2 API calls into their equivalent DOS functions when the application is run in DOS, and passes the APIs intact to OS/2 when they are run in the protected mode.

One alternative open to developers is to check for the particular operating system version as part of the program. This is done with a special conditional statement (called DOSGETMACHINEMODE) which asks what version of the operating system is running. The developer uses this flag to place different functions in the software, depending on where the software is running.

What does this mean for you as a software consumer? You should first determine what flavor you are buying when you purchase a new application: is it only for protected-mode, a family application, or just for "classic" DOS? You may have a choice of two different versions, one for OS/2 and one for DOS. What features are found in each version?

There is more work involved in writing a family application, and the size of the DOS mode executable code will be a bit larger. Microsoft says that about 30 k bytes extra will be needed to include the extra translator software modules, called stub loader and bindings. This means that Family applications will take up more disk storage. They will also need more RAM space to run, and take a bit longer to load, in the DOS mode. However, Microsoft says that the bindings are discarded in protected mode, so the application should actually require less RAM and load faster in protected mode.

Some software publishers have development teams for both operating systems and will continue to produce new versions in both arenas. Others will only put their energy and star programmers behind one operating system. You should contact the publisher's marketing executives and determine what their strategy is before making any of your own decisions on OS/2.

6.1.3 Eliminate ill-behaved DOS applications

Part of the family API process is also making your existing DOS application to behave by the rules and documented, supported DOS functions. Over the years, many software developers have taken a great deal of liberty with these rules and many of them have written software which use undocumented or unsupported DOS functions. Chances are, you have several of these programs in your corporation now. Chances are, you are unaware of which ones play by the rules and which don't.

Any DOS software that makes direct use of hardware will have trouble running in OS/2's DOS mode. Direct calls to ROM BIOS, to the video ROM, or to any other hardware will make life miserable for you. Any application that reprograms the disk controller, or mucks with any other "normal" operation of your PC will also be an OS/2 headache. Finally, any application which uses a block device driver, such as a RAM disk or tape backup device, won't run in the DOS mode.

How can you recognize these DOS villains? One way is to see how your applications operate. Do you have any applications with "keyboard enhancers?" These are modules which speed up cursor control, so that your cursor zips across the screen or your screen scrolls quickly from the top of the file to the bottom. These will need to be turned off or eliminate under the DOS mode of OS/2.

Do you have any super-fast backup utilities which spin the diskette drive continuously, even when a diskette is removed from the drive? These are reprogramming the disk controller, and are verboten under the DOS mode. You'll have to go back to using the standard BACKUP and RESTORE commands until your software publisher rewrites his backup program.

As you can see, it is time to pay the piper. Any program that has been promising faster performance, or something-for-nothing, is going to be an OS/2 red herring. You'll have to replace these programs, or do without them until the software publisher comes out with new versions.

There is another problem with ill-behaved DOS programs, and that is that they can ruin your entire system. Anything that crashes in the DOS mode may prevent you from getting back into the protected mode.

This was no big deal in the days of DOS, when one program at a time was executing. However, in today's world of multitasking, this could be a very serious situation, especially if you have several protected mode programs all going at once. A bad DOS program could stop all your protected mode programs dead in their tracks. You will have no choice but to reboot the entire machine, which could take many minutes particularly if you have many megabytes of memory installed and you have to turn the power off, then back on.

As a corporate manager, this could give you real ulcers. You may not know the precise software configuration for all of your initial OS/2 users, and may not be able to test every permutation of DOS software that people want to continue to run. The only way you might find out about a problem is the first time one of these nasty programs crashes somebody's system. We recommend a careful inventory and testing procedure before you certify something is OS/2-ready for your corporation.

As a software developer, you should carefully go back over code you have developed and check for to ensure that your software is completely kosher, conforming to the strictest set of DOS rules and regulations. If you find some abuses, you should carefully document them and make changes to make them able to run in OS/2's DOS mode.

6.1.4 Time critical versus non-critical

One of the more interesting problems with using the DOS mode of OS/2 is that any DOS application goes into suspended animation when moved into the background. (More on background operations later in section 6.3.) This will cause some problems, particularly if your program is time critical.

We define time critical to mean any software which depends upon explicit timing to do its job. If your software has a built-in clock to display the time of day, this may or may not be time critical, depending on what else depends on the correct running of the clock. If your software needs to know the correct time to execute some task or send something over a modem, this is time critical.

"Time critical," then, does not mean reading the right time off your PC screen so you can make your lunch appointment.

For example, suppose you use 1-2-3, which does display the time in one corner of the screen. If you suspend your 1-2-3 spreadsheet and go into protected mode, when you come back to the spreadsheet you will notice your clock is now off by whatever number of minutes you left it suspended in the background. However, this is not a time-critical problem, unless you use the time function in 1-2-3 to

calculate certain interest payments. In this case, having the wrong time will mess up your spreadsheet's calculations.

Most communications programs are time-critical, because they depend on the processor being able to maintain the communications conversation with a remote computer or modem. Every so often, the modem sends an inquiry to the remote machine: "Hi, are you there? What's happening? Got any data for me?" If your DOS program is lying in suspended animation, this conversation may end and the remote computer may think the line was cut off. This is why IBM states that "applications which communicate through modems or local area networks are not supported" under the DOS mode of OS/2. (For a further discussion of OS/2 and communications, see section 6.3.)

6.1.5 Similarities and Differences with Windows

A final consideration is the "look and feel" or user interface of your applications. OS/2 version 1.1 conforms to IBM's Systems Application Architecture Common User Interface. Many other application developers will choose this interface for their own applications, which means that most of the OS/2 programs will have the look and feel of Windows or the OS/2 Presentation Manager.

While it is not our intent to get into the specifics of Windows programming here, we should mention some of the more important implications of the Presentation Manager in terms of software use and design.

All applications conforming to the Common User Interface will have a series of menus that are pulled down from a top bar. Choices are highlighted by the cursor and chosen either by clicking a mouse, pressing a function key, typing a number, or typing the underlined letter on each menu choice. Dialog boxes will pop up on the screen, asking you to do things or confirm your selections. Windows can overlap each other on the screen, and you can scroll around in each window by pressing function keys or using the mouse in a consistent fashion. All function key assignments will be placed at the bottom of the screen, and similar functions, such as to get help, to exit or to move on to the next screen, will use the same function key assignments (such as F1 for help, f3 for exiting).

[insert Windows graphic screen shot here with description of what each of these things looks like]

Those of you who do not have OS/2 can gain some experience with using Microsoft Windows, version 2.0. Windows is very similar to the OS/2 Presentation Manager, although there are some critical differences.

First, OS/2 executes programs in both DOS and protected modes, while Windows is of course strictly for DOS. This means that the Presentation Manager uses additional features to manage the multitasking, multiple threads and processes, and large memory addressing features not found in Windows or DOS.

There are several new programming changes which incorporate these new features, such as a new graphics library and new ways to structure the data between a main "parent" window and associated "child " windows. We suggest you review the technical details in the "Presentation Manager Programming Reference Manual" from IBM.

Windows is a DOS application program which runs other programs in its environment. OS/2's Presentation Manager is an integrated part of the operating system. As such, it has features which more closely tie it to the operating system, such as the lack of special device drivers and a full-featured user interface shell. Windows required its own device drivers for things such as the clock and mouse; these are now integrated into OS/2 as a whole. Windows uses an "DOS Executive" shell which has fewer functions than the program selector of OS/2.

Third, there are some differences between the two programs in terms of user interfaces, although they are not significant. OS/2 uses the ALT-ESC to switch among running programs. Windows uses either ALT-ESC or ALT-TAB. There is no CTRL-ESC combination to bring up the program selector (see section 6.3 below for further details) in Windows as there is in OS/2. In fact, Windows has no analog to the Program Selector, which displays a list of possible programs along with those currently running in your machine.

The Presentation Manager has a completely new set of application interfaces, although most of these interfaces are just the same ones from Windows with different names. IBM and Microsoft used a new naming convention for OS/2's interfaces to keep them more consistent.

Some of the interfaces have new parameters, to include additional protected mode features. Some of the associated development files have also been modified, and so you will need to change and recompile your applications.

If your current DOS applications do not look like Windows, you can be sure that your software publisher, if he has any plans of running under OS/2, is hard at work changing them to do so. You can also be sure that this is a lot of work.

What this means for you is a great deal of retraining. If you are used to the Lotus 1-2-3 or WordPerfect style of menu "trees" you will have to learn the Windows way of thinking when you run OS/2 version 1.1 and later applications.

There is more to Windows, and the OS/2 Presentation Manager, than just look and feel, however. Both programs contain a powerful dynamic data exchange (DDE) language, so that running programs are able to pass information among themselves, unaffected by anything you do as an operator. Microsoft and IBM have published the specifications of this language, along with demonstration programs.

One of the more powerful demonstrations uses Microsoft Excel, which runs under Windows. This is a spreadsheet program and the demonstration consists of several macros which execute in Excel. The macros control other pop-up windows which are displayed on the screen and interact with each other, passing information back and forth while you merely watch the results.

The demonstration is called "Fish" and shows a fish tank with several species of fish swimming around. Another macro displays small submarines, which shoot torpedoes at the fish.

This bit of whimsy shows off the power of the dynamic data exchange potential of OS/2.

As mentioned in chapter 4, one strategy is to start learning Windows now to gain more experience with this new style of user interface. We would also recommend reviewing the DDE guidelines and purchasing a copy of Excel to test out its capabilities.

There is another implication for you as software consumer. The first "Window's like" version from your OS/2 software publisher will need a significant amount of compatibility testing by your corporation. You will need to test existing data files and make sure that they can function without any problems under the new software, as well as ensure that all new functions work the way they are designed and documented.

Given the amount of code which will change between the existing DOS version and the new OS/2 Presentation Manager version of your software, you may choose to wait until the SECOND OS/2 version is available before you certify the software as recommended for corporate-wide use. This next version will have most of the initial bugs worked out, and the publisher will have gained more experience in running under OS/2.

6.2 Writing batch files in the dual-mode world of OS/2

Now that you have some idea of what is involved in migrating applications from DOS to OS/2, you next stop should be how to write protected mode batch files. As we have mentioned earlier, OS/2 calls its batch files command files, and gives them a new extension, .CMD, to differentiate them from the DOS mode .BAT batch files.

The command files, as we will call them in this chapter, are still ordinary ASCII files, with lines of batch commands in them just like the DOS .BAT cousins. But there are three new commands that command programmers can take advantage of. These commands are described in the Appendix.

The three commands are SETLOCAL, ENDLOCAL, and EXTPROC. These are in addition to the DOS 3.3 batch commands such as ECHO, SHIFT, and IF. (See Table 6.1 for a complete list of all batch file commands.)

Table 6.1 Batch file commands

CALL	Nests batch files
ECHO	Echos text to the screen
ENDLOCAL	Ends local environment settings
EXTPROC	Runs an external command processor
FOR	Runs a command for a given set of items
GOTO	Processes commands after a given label
IF	Sets conditional statements
PAUSE	Pauses execution until user presses key
REM	Includes remarks which are not executed
SETLOCAL	Starts local environment settings
SHIFT	Changes the position of replaceable parameters

SETLOCAL allows you to set environment variables, such as path, disk drive, and directory temporarily, for the duration of the batch or command file. They work equally well in either protected or DOS modes. The temporary environment is ended with the ENDLOCAL command.

EXTPROC loads an external command processor. This is used if you have one of your own and want to use it in the batch file.

Batch file programming will be more complicated under OS/2 than with DOS, since multiple batch and command files can be executing concurrently in your machine. This is why the local environment strings are important, so that your batch and command programs can distinguish between their own files and someone else's.

6.3 How to manage background operations

OS/2 provides a program selector to switch among running programs and to start new programs. If you didn't include any STARTUP.CMD file in your root directory, when you turn on your computer you will first be brought to this selector.

The selector will differ depending on the particular version of OS/2 you are using. IBM uses a character-based selector for version 1.0, while the Presentation Manager version 1.1 uses a graphics-based "windows"-style selector.

[graphic of v1.0 selector, if we are lucky a graphic of PM or Windows selector]

They both operate the same way, though. If you are running a program in either protected or DOS mode, hitting the CTRL and ESC keys together will bring you back to the selector, where you can start a new session.

You may run up to 12 protected and one DOS mode sessions, or execute 256 threads, whichever comes first. Obviously, the number of active threads varies during the execution of your programs, but it is safe to say that you should be able to run at least six or seven programs before you run of available threads.

Once CTRL-ESC is hit, the active session (the one you were just in before you typed these keys) is suspended into background and the selector is brought up. You can start another session or bring an already-running session into the foreground.

Once you are finished running your protected mode session, you can type EXIT to end the session and bring you back to the program selector.

Another option part of the program selector is use of the ALT and ESC keys. Hitting these keys will cycle you through the various active sessions. If you have three protected mode sessions and one DOS mode session, you can sequentially move from one to the next. You will move either from one full-screen session to another with version 1.0 or among windows (for the protected mode sessions) and the special full-screen DOS session in version 1.1.

The program selector is the major difference between OS/2 and Windows. Windows uses different key combinations to switch among running programs (such as the ALT-SPACE) which are not supported by OS/2. [list other differences between Windows and Pres. Mgr. here]

6.3.1 How background operations differs between protected and DOS modes

While you can switch among all of your running sessions, whether they are in protected or DOS mode, the two modes are not exactly the same. There are some differences between operating programs in background between the two modes.

All DOS mode operations are suspended while they are in background, unlike the protected mode programs which continue to execute. Any processing resumes when you switch back into the DOS mode session, with some minor exceptions, as mentioned in section 6.1.

This is why time-critical applications will have problems, because they go into a "frozen" state of suspended animation until you return to them. (See the next section for more details.)

Protected mode programs continue to operate in background -- after all, that was the whole idea behind protected mode. The protected mode program continues to run in background unless it needs some input from the keyboard or something else that only works in the foreground. Depending on how your program is written, it may be able to send information to the screen while in background, too.

This could be either good or bad, depending on what your programs are sending to the screen. Certainly, you would want to design things so that any error messages or similar important pronouncements stayed on the screen and waited for you to come back to them and take notice.

Another difference between the protected and DOS modes is how the EXIT command works. In both modes, EXIT ends the current command processor and returns you to any previously-started one. If you have a program which allows access to DOS while you are running the program, you have used this command.

What happens is the program leaves some code in memory and runs a second copy of the command processor. When you are done, you type EXIT and return back to the code in memory.

Under the DOS world, EXIT used to be an important time saver. It was helpful way to get quick access to DOS in the middle of running an application, without the fuss and bother and extra time needed to reload the application from scratch. However, its role has changed somewhat in protected mode since you now can have as many copies of the command processor (up to 12) running together as you want, and you can quickly switch out of any running program with the ALT-ESC combination.

In protected mode, EXIT will end a running session and return you to the program selector if you have not started a secondary command processor. This is different in the DOS session. You can't use EXIT to end the DOS session and return to the program selector. Instead, EXIT operates as it did under DOS, returning you to a previous command processor (in this case, it is the DOS processor, not the OS/2 processor.) If no previous processor exists, EXIT does nothing. If you want to get back to the program selector, you need to press CTRL-ESC.

Another difference between the two modes is how programs are started to begin with. DOS mode programs can only run one at a time, and to start them automatically you place the appropriate statement in your AUTOEXEC.BAT file, as you did before.

The one wrinkle is that no DOS program can start automatically, without you first selecting the DOS mode processor from the program selector.

OS/2 is set up to automatically execute protected programs, however. As we mentioned in earlier chapters, you need to correctly specify what you want to run automatically in the OS2INIT.CMD and STARTUP.CMD command files.

You also cannot place any DOS mode programs in the list shown in the program selector; only the names of protected mode programs are allowed.

There are two other ways to run protected mode programs, by using either the RUN command in the CONFIG.SYS file or the DETACH command in a batch file. These commands should be used for special background-only programs that don't require any user input, such as the OS/2 print spooler or a job scheduling application.

There is one final distinction between the DOS and protected modes, and that concerns the ability of any ill-behaved DOS applications to bring the entire system down. If one protected mode program crashes, you merely switch out of that session and start another protected mode session with a CTRL-ESC or a ALT-ESC. This may not be possible if your DOS mode application freezes up and prevents any keyboard entry, including the ones mentioned to switch sessions.

6.3.2 How to run DOS communications software successfully

We mentioned earlier the problems involved in running programs which depend on particular clock times or which depend on foreground operations. Given these problems, how can you run your DOS communications software successfully under the DOS mode of OS/2?

There are several tips we can offer. One is to obtain the protected-mode version of the software soon after you purchase OS/2. This way, you will use software designed for running in the background.

If your software publisher is not writing a protected mode version, or has not yet released the software, you are stuck with your DOS version for the time being.

First, you should plan ahead when you will need to use communications. Make sure that very little else will be running in your machine. You should be especially careful if you are mixing DOS and protected mode communications programs. If you have some that run under both modes, do not bring up the protected mode programs when you want to use your DOS mode ones. The two sets might have some irreconcilable conflicts.

Next, you should try to use your communications software in the foreground as much as possible. You should stay within the DOS mode during the entire communications session, especially when you are connected to another computer or modem. If you need to use a protected mode program, use it

quickly, getting in and out of the software without dilly-dallying. If you suspend the DOS session long enough, the remote computer may think you have disappeared altogether and hang up the connection on you.

Finally, you should exit your communications program before you continue doing any protected mode tasks.

6.3.3 A note concerning tape backup devices

Tape backup devices are another situation that requires some careful thinking. While IBM does not currently offer tape drives as options for its line of PCs and PS/2s, several other vendors do, including Compaq. Of course, you need special device drivers to run these peripherals. We will use the Compaq tape drive as an example for this section, since it is one of the more popular ones.

Compaq supplies a separate TAPE.EXE tape utility with its version of OS/2. This utility only operates in protected mode. While the utility can operate in background, Compaq recommends you should always run the utility in the foreground for the best performance and to eliminate the confusion of exactly what is backed up. (More on this in a moment.)

There are two important considerations for using tape drives. One is installing the appropriate device drivers. Some manufacturers will ask that you rename their drivers with the DISK01.SYS file name and place it in the root directory. This raises important management issues, since you must write over the existing disk driver file and now you have at least two different files with the same name. (More about installing the Compaq drivers in section 6.3.4.)

A second issue is how to use the tape software itself. When you ran the DOS version of tape backups, you weren't concerned about what else was running in your machine and what files remained opened while the backup software was running. Under protected mode, other applications could be using files while the backup software is trying to do its backup.

The tape utility has been designed to take advantage of OS/2's larger memory space to improve performance. It will use whatever memory is available for its buffering of data copied from disk to tape, although it needs at least 128 k bytes to work at all.
Compaq also has made two important changes to the protected mode version of its TAPE utility. First, the software will not backup any files that are in use by another program. This is for your own protection. If you expect a full disk backup, you should shut down all your other running applications before running the tape backup utility. (This is also why Compaq recommends you run the utility in the foreground.)

Second, Compaq has extended the number of system files which cannot be restored by the utility to include the swapper file, the OS/2 and DOS command processors, and the OS/2 system files. This means that you cannot backup files from one version of DOS or OS/2 and restore them to another disk with other operating system versions. Compaq has done this to prevent inadvertent corruption of these important system files. This is also another reason why you should use the OS/2 version of the TAPE

utility, since earlier DOS versions do not recognize the OS/2 system files in this special "non-restore" category.

6.4 Devices and their drivers

Now that you understand one of OS/2's more important features, background operations of multitasking programs, we move on to a second concept, that of installable device drivers.

A device driver is a software program which provides reading and writing of data to various input and output peripherals. When an application program requests input to or output from a device, the driver is the software interface that makes it all happen.

The driver is installed at the time a computer is powered on and cannot be changed unless the machine is rebooted.

6.4.1 Differences between OS/2 and DOS device drivers

DOS 3 also used device drivers, and many developers took advantage of them to run mice, tape backup units, larger-than-32 M byte hard disks, and other non-standard peripherals. However, OS/2 drivers are more complex, for several reasons.

First, the drivers must be at home in both the DOS and protected modes of OS/2. The driver needs to manage a data stream coming from both sides of the house. Let's take the example of a printer driver. You could be sending one print job from a protected mode program, then switch over to the DOS mode and send a second print job there. The printer driver needs to manage both data streams correctly, so that pages of one document do not get mixed up with the other. Moreover, the driver must know that the CPU has switched modes (from protected to DOS) and changed over to another application in between the time the first request was made and when it was granted.

Second, the drivers need to handle the implications of many different applications executing at once. This means that while an application sends data to a device it continues to execute, perhaps sending additional data to the device. Or the application that issued the request to the device may be suspended by the time the request is granted.

DOS drivers didn't need to do this. A DOS driver typically began a read or write operation, then waited around until the device was done with its task. While it was waiting, no other programs could execute. Because there was no such thing as multitasking, the driver didn't need any programming to check to see what the current state of the CPU is, or need to surrender control of the CPU to other programs.

For example, a disk driver can receive several read/write requests from several different programs. The driver can decide the most efficient order to satisfy these requests, perhaps using some algorithm which minimizes the amount of seek time (the time it takes the drive heads to move across the disk surface).

Third, the drivers are essentially small programs and as such must follow certain programming rules of OS/2. There can only be two segments, a data segment (which must be first) followed by a code segment. Each segment must be smaller than 64 k bytes. Drivers can access the dynamically linked libraries of OS/2, so this constraint is not as bad as it sounds.

A fourth difference is that the OS/2 drivers cannot use the ROM BIOS services and must contain programming down to the actual hardware level of the device itself. This is because the ROM routines were written mainly for "pure" DOS applications, and can't operate under the multitasking protected mode situation found in OS/2.

This points out a major difference between how DOS and OS/2 operate. OS/2 has no knowledge of disk geometry: how many sectors, heads, or tracks are on a certain disk. DOS did have this knowledge. Under OS/2, it is the job of the device driver to provide this information, and to isolate the operating system from the actual nitty-gritty of the hardware.

Finally, the number and size of OS/2 drivers will influence how much RAM is left over to run any DOS mode programs. This is because the driver is loaded in low memory (under 640 k), right after the part of the operating system command processor. Drivers must be loaded into low memory if both DOS and protected modes are going to use them.

Loading into lower memory, though, takes away RAM left for DOS mode applications. If your system requires many different drivers, you might find that you have much less than 600 k of RAM to run any DOS mode program. This could be a problem if you were used to more RAM to run big spreadsheets or databases.

Because of the many differences between the OS/2 and DOS device drivers, it is very unlikely that any DOS drivers can work under OS/2 without any modifications. This is especially true for block mode drivers, which moves blocks of data at a time. Block mode drivers are typically used in conjunction with disk drives, serial ports, tape backup units, and printers.

A block driver typically uses direct memory addressing techniques. The driver tells the device where the block of data is located in memory and the device then either reads or writes that block directly, without bothering or tying up the CPU.

There is another type of driver called the character mode driver. These drivers are only moving single bytes of data at a time to their devices, such as a light pen, a mouse, or a keyboard. Instead of directly addressing memory, they execute a specific input/output instruction for each byte being transferred to or from the device. As you can imagine, this is a less efficient way of doing things than the wholesale moving of data with block drivers.

Some of the old DOS character mode drivers are still supported under OS/2, but only under some limited circumstances. First, these old drivers can only be used in the DOS mode and will not run in the protected mode. This means that only DOS mode applications (and no protected mode applications) will be able to access this device. Second, the drivers will work only if they polled their devices rather than used a hardware interrupt. Third, the driver cannot initialize the device with any old-style interrupt 21 (hex) functions. Finally, any old mouse or clock character drivers are not supported in OS/2: these devices must use new drivers which can share the device between the DOS and protected modes.

6.4.2 Changing drivers means rebooting

As mentioned earlier, once you have installed your driver you are stuck with that configuration. The only way to change a driver is to rename the file (for the drivers which need to be in your root directory) or change a statement in the CONFIG.SYS file and then reboot your machine. When you finish loading the operating system after the boot, the new drivers should be installed.

[check this, what about driver DEINSTALL command 20???]

Driver management is a difficult proposition, especially when you have a dozen different files, all of them used for different PC configurations. And the only way to tell if a driver works is to turn on the computer and see if it loads all the drivers correctly.

We strongly recommend as a precautionary measure that you give your users a DOS diskette with their drivers on the diskette. You may also want to write a small batch file (we have called it "CONFIG") which will copy the drivers either to the root directory or to the \os2 directory. This way, if one of your employees accidentally deletes one driver, they can be up and running with minimum effort.

Such a batch file will look like this:

```
REM  CONFIG.BAT

REM to restore original driver configuration

copy A:KBD0?.SYS c:\

copy a:screen0?.sys c:\

copy a:printer0?.sys c:\

copy a:clock0?.sys c:\

copy a:disk0?.sys c:\

copy a:*.sys c:\os2\*.*
```

If for any reason OS/2 fails to boot, they can insert the diskette in their machine, bring up DOS, and load the necessary driver files back on to their machine.

6.4.3 Drivers supplied by IBM

We have already mentioned in Chapter 5 how drivers are installed in your computer. Basically, there are two types of drivers, those that need specification in the CONFIG.SYS file with a "device=" statement and those which don't.

The former types of drivers include ones for mice, for external disk drives, and for serial communications ports. The latter types of drivers must be placed in the root directory of your startup disk, since OS/2 only knows to look for them there. These include the driver for the keyboard, screen, clock, printer, and disk.

The CONFIG.SYS drivers can be placed in any directory, provided you specify the path name in the "DEVICE =" statement properly. We recommend you keep them all in the \os2 directory.

There is one final subtlety involved in device drivers. There are two groups of drivers, one for the AT and XT/286 family, and one for the PS/2 family of computers. As you can see in the table below, some of the drivers are the same for both computers, while others use different drivers.

ROOT DIRECTORY DEVICE DRIVER FILES

(All five files are required for every machine)

For AT, XT/286 family	For PS/2 family
kbd01.sys	kbd02.sys
screen01.sys	screen02.sys
clock01.sys	clock02.sys
printer01.sys	printer02.sys
disk01.sys	disk02.sys

CONFIG.SYS DEVICE DRIVER FILES

(Only required if you have specify equipment)

	For AT, XT/286 family	For PS/2 family
ANSI display*	ansi.sys	ansi.sys
EGA Monitor	ega.sys	ega.sys
External Disk	extdskdd.sys	extdskdd.sys
Virtual Disk	vdisk.sys	vdisk.sys
Serial Port	com01.sys	com02.sys
Mouse Pointer	pointdd.sys	pointdd.sys
PC Mouse (S)**	mousea00.sys	mouseb00.sys
Visi-On Mouse (S)	mousea01.sys	mouseb01.sys
Microsoft Mouse (S)	mousea02.sys	mouseb02.sys
Microsoft Mouse (P)	mousea03.sys	(not supported)
Microsoft Mouse (I)	mousea04.sys	(not supported)
IBM PS/2 mouse	(not supported)	mouseb05.sys

NOTES:

* ANSI.SYS driver is only needed for DOS mode support. Protected mode uses "ANSI = on" command which is placed in startup files.

** Indicates type of mouse attachment: S= serial mouse, P=parallel mouse, I= in-port mouse

6.4.4 Non-IBM Drivers

While IBM does not support non-IBM equipment (other than the mice mentioned above) with its version of OS/2, you may be able to run the operating system on certain non-IBM systems or peripherals if you install the drivers yourself. While no book can be completely comprehensive on this subject, we

have tested the following equipment with IBM's OS/2 and found no problems using the indicated drivers.

There are two strategies here, for the root directory and CONFIG.SYS drivers. If you are using equipment which requires a root directory driver, you will need to copy over the existing IBM driver file with the non-IBM file. Or you can rename the IBM file to something else and then rename the non-IBM file to the same name as the IBM one.

For example, suppose you want to run IBM OS/2 on a Compaq with the Compaq tape backup device, booting OS/2 from your hard disk. (Compaq supplies its tape and disk driver together in a file named CPQTAP01.SYS.) You need to issue the following commands.

copy a:cpqtap01.sys c:\	[copies driver file from installation diskette to the root directory of your hard disk]
c:	[changes to c: drive]
ren \disk01.sys ibmd01.sys	[renames old IBM disk driver]
copy \cpqtap01.sys disk01.sys	[replaces with Compaq driver]

The following table lists the non-IBM equipment, its purpose, and any associated commands:

Vendor Device		Driver/Command
COMPAQ	Disk/tape driver	cpqtap01.sys
COMPAQ	Disk only driver	cpqdsk01.sys
COMPAQ	Tape backup command	tape.exe
COMPAQ	Fixed disk setup command	fdisk.exe

[fill in others as we know about them]

6.4.5 A note on re/writing your own

This section is just a beginning step in explaining the very complex subject of device drivers. If you want to get more involved in writing your own or rewriting existing drivers, you should see the following books and articles for more information:

The IBM "Device Drivers Guide" included in the OS/2 Developer's Kit is the complete reference book for driver command syntax and usage. Also, David Schmitt, "Designing Drivers for OS/2," PC TECH JOURNAL, (Part 1) December 1987 and (Part 2) January 1988 are a good summary of the basic steps involved in writing your own drivers. Finally, the [other OS/2 books being published by Brady].

6.5 Managing memory in OS/2

By now you understand that OS/2 offers the prospects of more memory, 16 M bytes of it. There are several issues you should consider in terms of managing this huge memory space. First, what types of memory boards should you purchase? Second, exactly how much memory will you need, and what should you initially install in all your machines?

6.5.1 Characteristics of OS/2 compatibility for new and existing memory boards

While we would like to give our specific recommendations for memory adapters, the marketplace changes too quickly for any book on OS/2 to provide specific timely suggestions. However, we can offer several guidelines for you when shopping for more RAM.

First, get as much RAM on a single board as possible. The fewer boards you need to install (and fewer spare slots you need to occupy) in your machine, the better. Some of you may have machines that have few spare slots. This presents a difficult choice: you can buy some time with a large RAM board now, but eventually you will need to upgrade to a new machine with more RAM and more functions (such as serial and parallel ports) on the system board or more slots on the system board.

However, be aware that boards with more than 2 M bytes of RAM may or may not work correctly under OS/2, depending on how the vendor has implemented the memory drivers in firmware on the board.

The only way to test this is to create a large application, such as a spreadsheet, and see if you can grow the application beyond the 2 M bytes of RAM on your adapter (plus any memory on your system board, typically 640 k or 1 M byte). If you can create this huge monster, your RAM is working correctly.

A second consideration is to look carefully at the memory chips themselves on the board. They should all bear the same label from the same manufacturer. This indicates that the board was populated at one time with the same brand of memory chips.

Many distributors purchase RAM boards with little or no memory and then populated them with the cheapest chips possible. This is true of just about every distributor, even those that are "certified" by the RAM adapter manufacturer. While this practice saves these distributor money and increases their profit margins, it could also create all sorts of RAM headaches for you if one of these cheapie chips fails. This was a minor problem with DOS, but a major one under OS/2 with its megabytes of RAM and tons of chips inside your machine.

Cheap chips can also be a problem if you are using the speedier 10 or 16 M Hz machines. The faster your machine runs, the tougher it is on those RAM chips. This is why you should look carefully at the chips, and ask what the rated speed is (180 nanoseconds or less indicates the faster chips).

A third consideration is to go with the newer-style of single-inline-package memory modules, rather than the older style chips. You can easily recognize the new style of chips, since there are several memory chips soldered onto a small board which snaps into place, usually at a 45 degree angle to the rest of the RAM board. The older chips fit directly into rows of sockets on the board.

The newer chips hold more memory (usually one package contains 512 k bytes, versus 8k or 64k bytes per chip on the older ones) and are more reliable. However, the new chips are also more expensive (in dollars per byte, including the cost of the newer RAM boards as part of the calculation). We think they are worth the investment, especially if you will be using 4 or more M bytes of RAM in your machine.

Another issue is the device drivers for the memory board itself. IBM boards and others closely compatible do not require any drivers. You should try to stick with these boards, rather than any that come with their own drivers on a diskette. [CHECK THIS OUT.]

Another issue is the type of memory supported on some of the older boards. RAM boards run either expanded memory, the Lotus-Intel-Microsoft paged style memory which only runs under DOS (and not under the DOS mode of OS/2) or extended memory, the memory used by protected mode. The two types are mutually exclusive and typically you need to configure your adapter for either style.

Older AT-style boards used physical switches located on the board which had to be changed before the board was installed. Some of the older boards, such as the AST Advantage, could only run in one or the other style of memory, so you will need to determine exactly what memory board is currently installed in your machine before you venture into the land of 16 M bytes.

If you have any machines with switches on the board, you have a chore ahead of you. You will need to pop the covers off, take out the board, find the manual with the switch settings documented, change them, and replace. This could easily be a time consuming operation, especially if the manuals are missing or the vintage board you have in the machine does not match the switch setting chart in the manual.

(Manufacturers are constantly making engineering changes to their boards, and occaisionally they will change switch and jumper configurations without documenting it in their manuals. Your only recourse to getting the right configuration then is to call their customer support lines and tell them the board serial number.)

The better bet is to buy memory boards that are software-selectable, such as the ones for the newer style Micro Channel bus. Being software selectable means you can change their configuration without opening the machine, just by running the diagnostic diskette. Of course, this means buying a PS/2.

[maybe screen shot of the diagnostics here]

Finally, you should never purchase your RAM adapters from mail order houses, unless you have a supplier that is very liberal about accepting returned merchandise. We recommend that at least with RAM boards, you should buy from a local distributor who can provide the necessary support and replacement quickly in case anything goes wrong.

6.5.2 How much RAM is enough

We mentioned in chapter 4 our recommendations for memory usage for the various versions of OS/2. Let's be more specific about those recommendations.

First, these are IBM's published minimums for RAM requirements:

- For OS/2 version 1.0 without running DOS 1.5 M bytes
- For OS/2 version 1.0 with DOS applications 2.0 M bytes
- For OS/2 version 1.1 without running DOS 2.0 M bytes ??

- For OS/2 version 1.1 with DOS applications 2.5 M bytes ??
- For OS/2 XE version 1.1 without running DOS 3.0 M bytes
- For OS/2 XE version 1.1 with DOS applications 3.5 M bytes

These are the real minimums as far as memory, meaning that this is the minimum amount of RAM which can get OS/2 up and running with small applications. This means you will probably run out of memory before too long, especially if you run several applications that start to get greedy about their RAM consumption.

As we have stated earlier, we recommend 3 M bytes for the Standard Edition and 4 M bytes for the Extended Edition as a good starting point for taking advantage of OS/2 applications.

Why so much RAM? For one thing, OS/2 applications will be much larger than DOS ones. Spreadsheets and databases can easily take up 4 or 5 M bytes. Also, many of the background operations and special purpose utilities of OS/2 (such as those associated with local area networks and communications applications) require RAM as well. Another reason is that the OS/2 operating system itself is so much larger than DOS.

But perhaps the best reason for so much RAM is that memory is still one of the best bargains around for PCs, and the cost of memory continues to drop as chip engineering costs have been amortized and volumes go up for the newer, denser memory chips. Finally, taking a machine apart and changing chips is very time consuming, and it is much easier to just install as much as you can afford right off the bat.

Now, unlike DOS, OS/2 is not as rigid about RAM usage. Under DOS every program got its own special address space. When DOS ran out of space at 640 k, you could not run any other applications. OS/2 does not add each program's memory requirements together; RAM is shared among executing protected mode programs, once the device drivers and DOS mode space is accounted for.

6.5.3 LIM 4 EMS vs. OS/2 explained

**********fill in

[do we want to discuss this here or just fold in somewhere else?]

6.6 Managing screen displays in OS/2

OS/2 is almost as particular about its video displays as it is about its memory boards. While we can't list every video adapter here, we can provide some of the characteristics of which boards and screens work and which don't. There are also some special considerations for using the Enhanced Graphics Adapter (EGA).

6.6.1 Characteristics of OS/2 compatibility for new and existing graphics boards

OS/2 sets up a series of rules to manage the 12 protected mode and single DOS sessions. Perhaps one of the more important rules is that a program must save its own video state when you switch among the active screen groups.

This is to prevent you from viewing utter chaos on your screen. In the past world of DOS, programs often ran into problems when they would forget what was on the screen when a user hot-keyed out of the session. Under OS/2, each program must remember.

This presents something of a problem for the EGA, as we mention in the section below, since its registers are write-only.

Under the Presentation Manager, graphics programs use a special programming interface, which is based on IBM's mainframe Graphical Data Display Manager. This means that each video adapter needs drivers to talk to this programming interface, rather than special OS/2 presentation drivers.

However, with earlier versions of OS/2, this is not the case.

[need to clarify and expand]

6.6.2 Special considerations for IBM's EGA

You will have some special problems using IBM's enhanced graphics adapter (EGA) running DOS mode applications. This is because the EGA has some memory registers which are write-only, meaning that the operating system cannot determine what these registers contain.

This is a problem when you switch among the various screen groups. For example, when you switch from a running graphics application in your DOS mode to an OS/2 application, and then back to DOS, you'll see that a different image is on your screen compared to when you last were in DOS mode.

IBM recommends that you don't switch horses in mid-stream. In other words, make sure you finish any tasks in DOS mode before you move on to bigger and better things in protected mode. However, this merely recognizes that the problem exists and not any real solution.

OS/2 tries to restore your screen the way it thinks you left it, but since it cannot read some of the memory registers on the EGA board, it cannot completely do the job.

Some of the non-IBM board makers were aware of this problem and have corrected this so that the operating system can read information to restore the screen status. [ANd who might they be?] Of course, you will need a special OS/2 device driver for this.

Chapter 7. TROUBLESHOOTING

By now you understand that using OS/2 is a bit more complicated than using DOS. Of course, you are bound to run into problems, even if you carefully follow our installation instructions in Chapter 5. Some of the OS/2 manuals are less than crystal clear, and sometimes you will find yourself struggling with a problem for hours or even days.

This is not to suggest that using OS/2 is full of problems, but knowing how people use computers there is always room for "operator error," as it is called. To help you out of your OS/2 rut, we provide you in this chapter with some troubleshooting tips and techniques. We also have included some common situations we have heard from early OS/2 users. While we can't reproduce every possible problem, we have tried to pick the more likely ones.

If your problem is not covered here, you have several choices. First, you can try to contact IBM or your system vendor such as Zenith or Compaq. If your company owns an IBM mainframe, there are several possible ways to get help from IBM, including direct technical support through their ASKINFO on-line electronic bulletin board.

If you have purchased an OS/2 software development kit from Microsoft (or other systems products), you also have access to their on-line technical bulletin board system called DIAL.

On either system you can submit your question electronically and get an answer usually the next day. We have used both systems and found that a straight question usually gets a straight answer, although IBM's set up is easier to understand and to use.

Second, you can try looking through the many pounds of OS/2 manuals for an answer. While this is certainly a daunting task, the manuals are all well-indexed. IBM has done a fairly impressive job at documenting its operating system, and at the very least you might learn about some new OS/2 feature (although it may not be the one you were searching for).

A final attempt is to ask around in your company for an OS/2 expert. Someone else might have already experienced the problem, and found the answer.

If all else fails, you can get in touch with us in one of two ways. If your question can wait, you can send it to us via a pre-paid postcard at the back of this book. Once you fill out the postcard we'll try to respond to the more popular problems in the next edition. Of course, this doesn't provide you with any immediate help, but you will be helping others.

A second method is via MCI Mail. If you have access to this popular electronic mail service, we will try to answer any questions you send us. (Our mailbox names are MEDELHART and DSTROM.) Several other electronic mail systems, such as CompuServe, have gateways to MCI Mail as well.

Our discussion contains several sections: problems with installing OS/2 itself, printer problems, early warnings of compatibility issues, and general operational problems.

7.1 Installation problems

We covered many of the issues involving installation in Chapter 5. Even though IBM has included an automated installation aid in its OS/2 software, there are still many things that can go wrong. Many of the installation problems concern where files are located, and telling the operating system how to point towards them.

IBM's installation copies every possible OS/2 program, driver, library, help text, and utility file it can find to your hard disk. In our discussion in Chapter 5, we have shown you how to weed out your directories and get rid of the files you don't need. But what if you prune too heavily, and delete something you really need?

7.1.1 Troubleshooting tools recommended

Before you begin down the path of true OS/2 enlighten, we should take some time to discuss the minimum set of tools we recommend for rudimentary troubleshooting. No, we are not talking about line monitors, voltage meters, or data analyzers here. Although, if you have any of this equipment and know how to use it you are already ahead of the game.

The tools we are talking about are relatively simple software programs. You should have a DOS 3.3 diskette which you actually have tried booting up and know works with your system. You will need a DOS disk in case you can't boot up OS/2, and want to start to figure out what the problem is with your configuration.

The remainder of our tools can be stored on directories on your hard disk, since the file structure for OS/2 and DOS is the same. However, if you run into a problem with using your hard disk you may need another copy of these programs on a diskette.

A second and very important tool is a line editor. Our choice is EDLIN, not because of features but because it is simple and included in every DOS disk. There are very few commands to learn and it can quickly help change configuration and startup files. Some of you may want to use a word processor instead of EDLIN. If so, make sure you can create ASCII files without any embedded control codes.

Other utilities which will come in handy include the Advanced Diagnostics disk for changing memory and other parameters of your system, and utilities for your particular memory and video adapters. Typically, the hardware vendors include these utilities with their products.

Finally, we also recommend several inexpensive programs such as Norton Utilities, to help look at specific sectors on your hard disk. Norton (along with others) also has several useful things to change file attribute bytes (to uncover hidden files or make a file read-only) and to search for any text string or any filename on your hard disk. With the over 100 files that make up IBM's OS/2, it is easy to lose something!

7.1.2. Putting the right files in the right places

Now that you are fully equipped for any but the most odious situation you should first make sure that everything is in the right place. Your root directory should look like the model we displayed in Chapter

5: the BIOS files, the command processors, and the bare bones device drivers such as the ones for the keyboard and screen. If OS/2 won't even boot, bring up your DOS diskette and look around the root directory for anything missing. OS/2 will generally tell you if something isn't where it should be during its boot process.

One common problem is not correctly specifying your various path commands to show OS/2 to where things are located. As we mentioned in Chapter 5, there are several important path-type commands in OS/2. Most of them are found in the CONFIG.SYS file.

First, there is a path for the dynalink libraries, specified with the LIBPATH command. These libraries contain programs needed by OS/2. We suggest you collect them from all your OS/2 programs and place them in one subdirectory, such as an \OS2\LIB directory. This makes it easier to maintain and keeps your path statements relatively simple.

Next is the user shell, located by the PROTSHELL and SHELL commands for the protected and DOS versions of the shell respectively. If OS/2 can't find its shell it won't be a very happy operating system and you will get this error. We go into more detail about working with the shell later in this chapter.

After this are paths for the swap and spool files. We'll mention them later, but the commands are also in your CONFIG.SYS file (SWAPPATH and the RUN command).

Finally we have are the general, everyday paths for programs and data, specified by PATH and DPATH respectively. If you have gotten these wrong, chances are OS/2 will boot up fine but you'll have trouble running any of your programs unless you are in the same subdirectory as your particular software. This is similar to what you had to worry about with DOS. The major difference is now you have two paths to keep track of.

Depending on how you have set up your system, these commands may or may not be in your CONFIG.SYS file. If you let the automated installation program do its thing, these path commands are in the OS2INIT.CMD file, which starts up every protected mode session and is located in the root directory, unless you have changed its location by changing what is specified in the PROTSHELL command already.

While IBM has tried to make the automated installation program as bullet-proof as possible, there are several places where you could end up with the wrong installation. First, you may need to change the defaults for several configuration parameters. We mentioned in chapter 5 what our recommendations were for these parameters and how to change them.

Second, some of your software might not be OS/2-aware. In other words, some DOS software automatically replaces your CONFIG.SYS file with one of its own. This replacement might not have any of the special configuration commands, or may use drivers which won't work with OS/2.

Let's say you have already made it through the pain and suffering of installing OS/2. Now you go to install your software. Let's assume you use some DOS word processing software which comes with its own mouse drivers. Because this is DOS software, these drivers won't work with OS/2. When you run the automated installation program, it will overwrite your OS/2 mouse drivers with the ones from DOS.

The result is when you boot up OS/2 it will say you have an invalid driver. And all you did was try to install your software.

To fix this problem, you first need your DOS 3.3 disk. You boot up under DOS, replace the mouse driver with the correct one, and make any changes to your CONFIG.SYS file to remove any damage that has been done by your word processing software. That should do the trick.

Unfortunately, there is nothing in your word processing software which will tell you that it is trashing your perfectly good drivers with bad ones. The only way to know is if you get an error message when you boot up OS/2.

7.1.3 Access to swap and spool files

We mentioned above that one of the many paths to OS/2 enlighten is where you locate your spool and swap files. These serve two different purposes.

Most of the time the spool file contains nothing. That is, when you use a print spooler, the software creates a file on your hard disk. When the time comes to print a document, the spooler first moves a complete copy of the document to your hard disk spool file, and then proceeds to print the document from the file. When the printer is done printing, the spool file should be empty. This method reduces the amount of time your application software is involved in printing the document, and also frees up your session to do other things besides printing.

The swap file is another temporary storage dump for the operating system, but this time it contains the bits from active RAM. If you are running programs too big for your britches, the excess code is swapped to your hard disk (if you have enabled this feature in your CONFIG.SYS file with the MEMMAN and SWAPPATH commands). Remember, you can install up to 16 M bytes of RAM in your machine, but you can access up to a gigabyte of virtual memory. The remainder of memory is managed by swapping stuff from RAM to disk.

There is a slight problem with using the swap file -- it continues to grow larger and larger if it needs to. This means that if you run lots of programs and exceed your physical memory limitations, the swap file can take over your hard disk like the butterscotch pudding in Woody Allen's Sleeper.

There is a way out of this dilemma, if you are concerned about the size of the swap file and you still want to swap programs to disk. However, it involves reformatting your hard disk, which means you want to first make a file-by-file backup (if you want to save whatever you now have on your disk).

After making your backup, you first partition your hard disk into two partitions: one for your programs and one for the swap and spool files. This latter partition becomes your "D:" drive. The swap file is limited by the size of the drive itself, and now the rest of your hard disk is safe from being taken over.

How big should you make your second partition? We recommend no more than 2 M bytes, unless you are running many large programs concurrently.

To finish the job you will need to specify the D: drive in the SWAPPATH and RUN statements in your CONFIG.SYS.

One of the interesting situations with these swap and spool files is their influence on the drive light indicator. Those of you that feel comfortable in knowing when your disk is active and when it isn't are in for a rude surprise when you start using OS/2. Sometimes it will seem as if the disk drive has a mind of its own, writing and reading at odd times. This is because of the spooling and swapping going on. There are two other situations which cause OS/2 to look at the disk: needing a dynalink library and running over a network. Because the library files are dynamically used by the system, you may never be able to predict with any certainty when OS/2 will load one into memory.

The moral of the disk light situation is this: if you are the type to be bothered by the light flashing on and off at odd moments, do yourself a favor and put a piece of black electrical tape over the light so you stop worrying about it and let OS/2 let on with its business.

7.1.4 Problems with DOS mode size

Another problem is getting the most space for your DOS mode memory. As you recall from our discussion in Chapter 5, the command which controls the DOS mode memory size is RMSIZE, found in CONFIG.SYS. However, there are some stipulations which we shall repeat here about setting this size.

First, you will never get the full 640 k of memory for running a DOS mode application. The chief culprits, as we mentioned in Chapter 5, are the device drivers. These need to run in the lower memory area, and they squeeze out the room left for any applications.

A second problem is how much RAM is installed on your system board. The RMSIZE parameter is the lower of the following two numbers: either the total system memory minus 512 k bytes, or either 512 or 640 k bytes, whichever is installed in the lower memory address range on your system board. If you only have 512 k bytes (or less) installed on your system board, then regardless of what RMSIZE parameter you use you will only get 512 k for your DOS session.

7.1.5 How much memory is enough

Throughout this book we have said that 4 M bytes of RAM are the bare working minimum for using OS/2. Obviously, everyone's circumstances will be different, but 4 M can quickly get used up by a variety of applications, not to mention OS/2 extensions such as the Extended Edition.

If you run a DOS mode session, you will need at least half megabyte of RAM just for this function. This memory is not used by OS/2 for protected mode applications.

Unfortunately, the only way you'll know whether you have enough RAM is to try to load all your applications into various screen groups and see what happens.

You might have noticed that the OS/2 command CHKDSK has a slightly different display between DOS and protected modes. In DOS mode, the command shows free mass storage along with the free RAM. In protected mode, only the mass storage report is displayed. This means that you have no way of even knowing how much memory is installed and usable in your computer in protected mode. The only way

to tell is to watch the RAM counter when your machine boots up. (This counter is usually displayed at the top left hand corner of your screen when you first turn it on.)

7.1.6 Selecting your user shell

One of the more interesting aspects of OS/2 is selecting the user shell. This is the series of software which sets up a menu of all your applications, and includes a visual picture of your hard disk directories and file system.

Not all versions of OS/2 have the same user shell. In fact, those versions that include the Presentation Manager use a completely different shell from those that don't. This is because the Presentation Manager is part and parcel of the user shell, as we explain in Chapter 8.

You can "roll your own" shell and replace the one supplied by IBM. If you have your own shell programs, you must change the PROTSHELL and SHELL commands in your configuration file.

One recommendation for shells we have is to use different colors for the DOS and protected screens. (See Chapter 5 for further details on how to implement this.)

[maybe add more stuff as we know more about shells.]

7.2 Printer problems

Once you have OS/2 installed and working properly, you still may have some other problems in running your applications. Most likely, one of the first roadblocks you might encounter is using the printer with OS/2.

Printing takes on a slightly different culture in OS/2. With DOS, the printer used to take over: once a document is sent to the printer, everything else stops until it is finished. Third parties developed print spoolers which offloaded some of this waiting, but they had some problems: the spoolers often only worked with a limited number of applications and often conflicted with other RAM-resident programs.

OS/2 attempts to resolve these difficulties by packaging its own print spooler into the OS/2 stew. However, there are some implications for DOS users, especially those who are not used to running a spooler.

First, there is the peculiarity of OS/2's DOS mode itself. All DOS mode operations are suspended when you switch into protected mode. This means if you send a DOS document to the printer and then switch quickly into protected mode, your document may not have had time to be copied into the spooler's memory. You may end up with a partial print job, or the spooler may wait until you return to DOS mode before it prints out your entire document. This is not really a problem, just more of understanding how OS/2 works with the DOS and protected modes.

There are some other problems you might encounter with OS/2 printing. You might not be able to run the spooler to begin with, or experience slower than normal printing. Finally, we have a tip for you if your printer doesn't work at all. Let's look at each one of these conditions.

7.2.1 Finding the spool file

OS/2 doesn't have to use the spooler when printing, but it is recommended, and we have mentioned how to install the spooler in Chapter 5. As we said earlier, the spooler helps free up the time involved in printing files. Also, if several protected mode applications are sending print jobs concurrently, the spooler will be able to sort out the different jobs so they don't get mixed together.

The print spooler command is actually implemented by running a background OS/2 process. You do this with the RUN command in the CONFIG.SYS file. The actual syntax of the command is:

```
RUN = c:\os2\spool.exe /d:lpt1 /o:lpt1
```

This tells OS/2 to run the background process called SPOOL.EXE, located in the \os2 directory. Use the print device located at LPT1, and send it to the output device located also at LPT1.

As we mentioned in chapter 5, you could also use the spool command to redirect output to the serial port, in case you have a serial printer attached to your machine.

When the SPOOL program executes, it tries to open a file with a name which is a combination of today's date and some other unique characters. If your disk is not working correctly, or if you run the spool program from a directory where you have read-only access, the program will not be able to open any files and will give you an error message.

7.2.2 Slow printing

One problem might be slower than normal printing. You may send a file to the printer, only to have to wait several minutes before it starts printing. This may or may not be a problem, depending on how you have configured your system. We suggest you check the parameters specified in your CONFIG.SYS file for the maximum number of threads and parameters such as time slice and priority.

[beef this baby up]

7.2.3 Cable troubles

Let's say you have hooked up your printer to your computer and you have it running under DOS without any problems. You have install OS/2, try to print a document in the protected mode, and nothing comes out. You check all the conditions above and still you can't locate the problem.

One place to look next is with your printer cable. Some of you might be using sub-standard printer cables, which don't have all the leads and pins connected. OS/2 is just as fussy about printer cables as it is with other hardware.

Try using another, newer cable which has all the pins connected, before you start to give up hope.

7.3 Early warnings of compatibility issues

As you have noticed by now, OS/2 brings new meaning to term compatibility. Anything and everything inside your PC is subject to its scrutiny. This is because IBM's OS/2 is specifically tuned to run on IBM systems, not anyone else's. And this could include such things as IBM memory cards, IBM video adapters, and IBM hard disk drives.

While we can't possibly describe all of the compatibility issues here, we can mention briefly some of the major problems you might face when running OS/2 and protected-mode software applications on non-IBM machines.

This list certainly isn't exhaustive, but a good indication of where you will find some intial problems.

7.3.1 Non-IBM systems

Many of you have non-IBM computers and have run IBM DOS on them with no problems whatsoever. OS/2 is an OS of a different color. While Compaq's and Zeniths might run IBM's OS/2 without any trouble, other systems, such as Wang, might not be so lucky.

We suggest you use the same vendor for CPU and OS/2. If your CPU vendor doesn't license OS/2, then you might want to look at running Compaq's or Zenith's version rather than IBM's. This is because these versions of OS/2 are more "generic" than IBM's, and designed to run on the variety of communications cards and video monitors that run in Compaq's and Zenith's.

By "system" we also refer to those of you who have bought 286 or 386 turbo cards. You will need to get a version of OS/2 from the vendor of the card, not from the vendor whose box the card is in. This is because the version of OS/2 has to support the 286 or 386 chips which are attached in a strange fashion inside your machine: Typically, these turbo cards replace the 8088 chip with a ribbon cable connecting to their card. Plain-Jane IBM OS/2 doesn't understand what this cable is all about and will not run with it at all.

7.3.2 Non-IBM communication hardware

Running IBM's OS/2's Extended Edition is another potential problem area. Again, IBM intends for you to use their communications cards with their version of OS/2. If you have Irma boards and Hayes modems, you will have some trouble with the OS/2 EE. There are two problems: first, using the correct OS/2 communications hardware driver.

IBM's drivers may or may not work with non-IBM communications hardware. The only way you know is at boot time, when OS/2 tells you that it couldn't find a particular device or doesn't like the hardware that you have in your machine. You should pay careful attention to any of these warning messages as you boot up your computer.

If your modem or 3270 adapter are not supported by IBM drivers, you can try to use alternative ones. Many communications hardware vendors will supply the appropriate driver files with their products. You should contact them directly and see what is available.

For many of these non-IBM communications products, you will need a special driver which should be available from the comm vendor. The driver is installed in CONFIG.SYS, just like the other OS/2 drivers.

IBM is not responsible for supporting every modem and coax card in the world, and this means you have a new problem: how to manage all the drivers for all the various communications equipment in your corporation.

In some cases, the drivers will have the same name. For example, many of the various serial port drivers are named COM01.SYS. It is hard to tell which one is which, and you should develop some sort of tracking system, or keep them in separate subdirectories or on separate diskettes. Guaranteed one of the first problems is going to be someone who copies the wrong driver file to their system, or renames the driver to be something else. This is where having Norton Utilities is helpful: You can get in and take a look around and see which file is the right one.

A second problem is having a communications card which supports the various OS/2 programming interfaces (APIs). This is something which is not advertised and generally not well known by your vendor sales representatives. For example, the original Irma board doesn't run the IBM 3270 high level language applications programming interfaces without special software from DCA. Without this software, you won't be able to get the OS/2 API support you need on an Irma board.

OS/2 supports what IBM calls the Entry level Emulation, 3270 High Level Language Applications Programming Interface. This mouthful is actually a set of 39 commands which allow PC programs to take control over host sessions and automate the micro-to-mainframe link. Several non-IBM coax products support this interface, including ones from Attachmate and AST Research. As we have mentioned before, you should always try before you buy and certainly before you recommend for your corporation to buy in quantity.

Communications is perhaps one of the more complicated aspects of OS/2, and you should always operate in a slight state of disbelief when a vendor claims it is "100 percent compatible with OS/2 communications interfaces." Get the products and test them before signing up.

7.3.3 Non-IBM video adapters

Another problem is with non-IBM video adapters. This could show up in a variety of places, including in the type of driver files required by OS/2, the way the user shell is specified in the configuration file, and other factors.

IBM's OS/2 does require a graphics monitor and adapter. If you are using a monochrome monitor (other than IBM's new monochrome VGA monitors), you cannot use IBM's OS/2.

Several other OS/2 vendors may support a better selection of monitors and adapters. You should contact them to get the complete selection and then carefully test any combination of video adapter and screen to make sure it works with your version of OS/2.

7.3.4 Memory adapters

section TK

7.4 General operational problems

In this chapter, we have covered some of the more common problems with installation, such as printer and configuration troubleshooting. But there are going to be other times when you are trying to track down answers to some tough questions. Here are a few of the more common situations, including ending background processes, debugging batch files, and keeping track of file and memory usage.

7.4.1 Killing a background process

Neither IBM nor Microsoft currently includes in their commercial OS/2 versions any way to kill a background process in midstream. Once such a process is started, it will continue to run. In fact, a background process is defined as one which does not receive any input or control from the keyboard, so there is no way of stopping it under any user control once it is started.

This is in contrast to the foreground processes, which can easily be terminated by typing an exit command at the OS/2 prompt. This closes out the screen group and returns you to the program selector, just as exit would terminate a secondary DOS processor under DOS.

Unfortunately, OS/2 will not even display what background programs are running in your machine, unlike the foreground programs which are listed on your program selector. So sometimes you may or may not even know that a process is executing in the background, causing you any trouble.

In theory, the whole notion of protected mode is to keep programs from bumping into each other. But OS/2 is still a new operating system, and some of the applications are also still fairly new and untested.

This is mainly a problem in the development environment, where you might be testing a buggy background application. If you start a background process and realize that it is going to be trouble, your only alternative is to turn off your computer and start all over again.

7.4.2 Debugging batch files

Batch file programming under DOS was always something of a curiosity and usually only done in times of dire necessity. Under OS/2, batch files (called command files since they have .CMD extensions) take on greater purpose. They also bring greater potential problems, making troubleshooting more difficult.

While it would be beyond the scope of this book to provide you with all the tools to debug batch and command files, we can offer you some tips.

First, test your command files one at a time, with nothing else running in your machine. This way, if you run into problems, you can isolate them quickly.

Second, test small pieces of code individually, also to isolate the problem.

Third, you should have a set of command files which display your local environment. (This is easily done with a few set and setlocal commands without any arguments to display their settings.) One of the more common problems is that OS/2 keeps local environment parameters, in addition to global ones (as mentioned in Chapter 6). Your command file may be confused as to what is happening in which environment.

Finally, when in doubt about some aspect of OS/2's protected mode or new batch file commands, try testing the command file in the DOS mode by copying it to a .BAT file. You may not be able to execute all the commands in DOS mode, but you may be able to see what is wrong with the file.

7.4.3 Keeping track of file usage

Keeping track of your files under DOS was relatively easy. You could go through your directories with assurance that they understood what the files were and which program created them. This may no longer be true with OS/2. If you execute several programs at once, you will find that many files are actually in use and can't be deleted.

Remember, OS/2 still uses the single-user DOS file system. This means that only one application can open and write to a file. If you try to open the file a second time, you will get an error message saying that the file is in use by someone else. You should then look at the Program Selector and determine which application is using the file. Remember that some background applications, such as the print spooler and the program swapper, also can create and open files on your hard disk.

If a file is marked read-only, then several applications can concurrently read the file.

One of the improvements that OS/2 offers over DOS is its visual file system, part of the Presentation Manager. You can view the actual subdirectory structure as part of the operating system -- something in the past that was only available as a separate add-on third-party program.

7.4.4 Keeping track of memory usage

As we mentioned earlier, it isn't easy to keep track of memory usage in the protected mode. CHKDSK doesn't give you any clues. And the swapping of memory to disk means you can have applications which are bigger than your physical memory.

The only way to know if you have enough is to run your applications concurrently and see what happens.

[add info on third party memory checkers]

Chapter 8. Understanding Advanced Features of OS/2

So far our discussions have only touched upon a very limited part of the world of OS/2. We have purposely confined ourselves to IBM's Standard Edition in the past seven chapters, both to simplify our discussion and to eliminate as much confusion as possible. However, there are many more exciting components of OS/2, and our intent is to briefly mention them here.

Perhaps the most significant component of OS/2 is the Presentation Manager, a jointly developed Microsoft/IBM windows interface. We discuss how to operate the Presentation Manager and what the differences are between the graphics-based and character-based versions of OS/2.

IBM also sells the Extended Edition of OS/2, which includes three other "managers," as IBM refers to them: a Database manager, a Communications Manager, and a Local Area Network (LAN) Manager. This latter feature is one of the more important aspects of Extended Edition. We discuss both the general features of Extended Edition and specifics relating to the LAN Manager in other sections in this chapter.

Finally, we speak briefly about what the significant differences are between various OEM versions of OS/2, such as among the IBM, Compaq, and Zenith versions.

8.1 Presentation Manager

Certainly the most notable component of OS/2 is its user interface, and there are drastic differences between the "A:>" prompt of DOS yore and the full-screen Windows-like OS/2 Presentation Manager.

PM is a graphic window on your applications. When you choose the program selector, a small window pops up in the middle of the screen with a list of applications. You can also graphically view your hard disk subdirectory tree, and run any program by selecting its name from the directory listing.

For users who are familiar with Windows under DOS, operating the PM version of OS/2 won't be very different. To run programs you point to them with a mouse and double click (in other words, click the left mouse button twice in succession). The various sizing boxes, the minimize (making an application the smallest possible window) and maximize (making it almost full-screen) features, the scroll bars (the horizontal and vertical bars which allow you to scroll through your application) are all the same as with Microsoft Windows version 2.0 (but somewhat less similar to earlier iterations of Windows.).

There are some differences, however. Windows had a cumbersome installation procedure, because it was considered a DOS application. PM is part and parcel of the operating system, and installs as the friend of OS/2, rather than its enemy. There is no PM.EXE file which loads the Presentation Manager; it is part of the entire packaged deal of version 1.1.

Actually the PM is a series of dynalink libraries, just like other powerful OS/2 applications "managers." As with all dynalinks, it stays in suspended animation until an application is loaded that needs the library services.

As it happens, the user shell is the first program loaded by the operating system and version 1.1 and later of the shell make calls to the dynalink PM libraries. So, there is a lot going on when OS/2 boots up,

including the loading of the PM. This is why it takes so long for OS/2 to boot: there is a lot of software loading into RAM.

PM's initial menu screen also does not look at all like's Windows's MS-DOS executive screen. [say more on this]

A few other things have changed from the DOS Windows days. ALT-TAB, which is used to switch among windows, is now ALT-ESC in OS/2. [and so forth, TK]

For those of you who love to issue OS/2 commands and miss the good old "C:>" prompt, you can still have your cake and eat it, too: By selecting the CMD.EXE file name from your root directory (or wherever you decide to keep it), you can invoke a secondary OS/2 command processor and quickly get a "C:>" prompt to issue OS/2 commands. You might even want to keep a screen group at the ready with such a prompt for old time's sake or as a convenient way to run OS/2 commands.

There are several different types of applications recognized by PM, we discuss them in the next section. There are also some major differences in how PM and non-PM versions of OS/2 operate, and we outline those for you as well.

8.1.1. Types of Presentation Manager applications

There are four potential types of applications that are recognized by the OS/2 PM. First is the DOS mode application. These run in their own screen group, independent of anything happening in protected mode. They use the entire screen surface. As we have said before, any DOS application is suspended when you switch into the protected mode.

A second type of application is those called "non-windowable." These are protected mode applications which also need their own screen group and require the entire screen surface. These applications need a whole screen because they require direct access to the screen buffer, video memory, and other resources, so they can't share the screen with anyone.

A third type is called a "VIO" (for video input/output) application. These have been written to run in character windows, and are easy to recognize. They don't have any menu bars, graphics (other than simple line character graphics), dialog boxes, or scroll bars. These aren't as nasty as non-windowable applications, because they don't access video memory directly.

A final series of applications is the fully Presentation Manager-ready application. They have all the accoutrements including the scroll bars, dialog boxes, pull-down menus, and graphics images that the others lack. They have been written to the full PM programming interface, and can share the screen real estate with each other without fear of causing pain and suffering to another application.

Ultimately, all your applications with be the full PM style, but in the meantime you will have to put up with managing the various different types of PM applications.

8.1.2 Differences between version 1.0 and 1.1 of OS/2

As you can imagine, there are several differences between the non-PM and PM versions of OS/2, or versions 1.0 and 1.1 as they are officially known. The entire kernel of OS/2 has been changed, along with the user shell. Many of the dynalink libraries have changed, and a completely new set of device drivers is required to work with the graphics interface of the PM.

It almost sounds like a complete rewrite, and this is not far from the truth.

[ED NOTE: WE MAY WANT TO BEEF THIS UP WHEN WE GET OUR HANDS ON MARCH PM VERSION OF SDK]

8.2 Extended Edition

Extended Edition (EE)is almost as much of a jump from Standard Edition (SE) as OS/2 was from DOS. IBM has packed in loads of functions to EE. Perhaps the two most important groups are the ability to organize databases and use a variety of communications links.

The SE was a jointly developed product between IBM and Microsoft, which Microsoft in turn licensed to a variety of PC manufacturers such as Zenith and Compaq. However, EE is IBM's domain alone.

The EE is IBM's first step towards a unified Systems Application Architecture, a collection of protocols and interfaces which allow programs to run on a variety of operating systems across IBM's product line of micros, minis, and mainframes. SAA-compatible products will have a similar look and feel, use the same communications interfaces and protocols, and require a minimum fuss and bother to run on these different operating systems. IBM has chosen OS/2 EE as its SAA-certified operating system for the PC.

EE comes in two distinct versions, just like the Standard Edition: version 1.0 which is a character-based application and version 1.1 (and beyond), which uses the Presentation Manager.

EE is proof positive that operating systems will become more and more integrated into your working life, blurring the distinction between what is considered an operating system function and what is truly the domain of an applications program. Let's take a closer look at these extensions.

8.2.1 Database features

OS/2 EE Database manager is IBM's first try at writing PC database software. What is most interesting about the product is how it is integrated wholly into the overall fabric of the operating system itself, including file management, communications, and programming interfaces.

The database component uses a relational structure. In other words, data is arranged in tables of columns and rows, defined by the user. In many respects, it is a scaled-down version of DB2, very similar to many PC packages such as dBase and Paradox.

The database manager has three important components: a query manager, a set of utilities, and a set of database programming interfaces. Let's look at each of these in detail.

QUERY MANAGER. The Query Manager component handles the user interface, sends queries to the database, and produces reports. There are two separate user interfaces, one using full-screen menus and one using a command-line.

In operation, the database bears a striking resemblance to other PC databases. You can add, delete, and modify data, generate queries and reports, create customized views of the database, index and sort. There is nothing terribly new about all of this, except that it is included as part of the operating system software and that it is coming from IBM.

The maximum size of a PC database is 255 columns or 4000 bytes, whichever comes first. There is no limit to the number of rows, other than your local disk storage. The entire database must reside on a single logical disk, which must be less than 32 M bytes for version 1.0 of OS/2. (Version 1.1 of OS/2 will remove this restriction, however.)

The 4000-byte limit on each row is not as severe restriction, since IBM has included a feature similar to the dBase "note" field. An entry in a database can reference a PC file, which can be up to 32,000 bytes. The file can contain graphics, ASCII text, or other data which needs a large field length.

One twist is that the data can be located either on a PC or on mainframes. In some sense, then, OS/2 EE takes the user friendliness of a dBase or a Paradox and combines it with the power of a PC version of DB2.

Because this was written by IBM, the query manager includes support for Structured Query Language or SQL (pronounced "sequel"). You can create SQL inquiries to your PC or mainframe databases, with lots of user help screens and prompts if you forget your field or database names or structure. Again, this compares to the features of PC/FOCUS or Paradox.

IBM has designed the query manager as part of its Systems Application Architecture (SAA). The architecture spells out a consistent interface for a variety of communications protocols and other components, such as database queries and report generators. These components will eventually be used by all IBM operating environments, including OS/2 EE.

DATABASE UTILITIES. No database program would be complete without a complete set of utilities, and OS/2 EE is no exception. IBM includes the standard fare of file format conversion, backups and restores, table restructuring, and update control utilities. For example, you can convert your Lotus 1-2-3 spreadsheet into an OS/2 EE data file, or vice-versa. You can import and export selected subsets of the entire database to a 1-2-3 file, or even an ASCII flat file.

IBM's mainframe database experience is evident when we look at the utility for update control. Mainframe databases have what is called a "rollback/commit" feature. The database doesn't post any updates until all transactions have been successfully completed. If your machine hangs up in the middle of a transaction, OS/2 will automatically back out all data not committed and completely processed.

This keeps your data integrity high and helps you separate out any garbled updates easily. It is a nice feature that should be included on other PC databases.

PROGRAMMING INTERFACES. A third part of the OS/2 EE database manager is its programming interfaces. Programs will be able to take control over PC datafiles and do in code what you normally do at the keyboard. Some of these interfaces won't be fully specified until later versions of OS/2.

One of the more important extensions to the Database Manager will be the addition of APPC interfaces. A future version of OS/2 will allow APPC function calls to be directed integrated into the database component, so that two OS/2 EE PCs (or a PC and a mainframe) can update each other's files. Multiple applications will be able to access multiple databases concurrently, all without the knowledge or direct keyboard control of a PC operator. This could prove to be quite powerful, once developers catch on to this idea.

Direct support for SQL statements will also be added to the COBOL/2 and PASCAL/2 languages. This means that programmers will be able to write code incorporating SQL queries directly.

8.2.2 Communications features

In addition to these database features, OS/2 EE also folds in the functionality of several of IBM's communications programs, including the 3270 and 3101 emulation programs, IBM's Advanced Program to Program Communications (APPC) and the local area network operating system.

Because the LAN software, called the LAN Manager by Microsoft and the LAN Server by IBM, is so important, we discuss it separately in another section. However, one point to keep in mind during our discussion on OS/2 EE is that any LAN support is only included in OS/2 EE version 1.1 and not 1.0.

There are three different programming interfaces supported, along with a variety of terminal emulators and file transfer protocols. Let's review this cornucopia of choices.

[maybe a chart here, like the Ivory p. 4 287-499 nov 87]

PROGRAMING INTERFACES. There are three different programming interfaces supported by the EE: APPC, SRPI, and an asynchronous interface.

APPC is IBM's strategic direction for cross-systems communications. Programs written to this interface can run other programs, without any user intervention once the programs begin.

APPC has already been adopted by a variety of systems vendors, including DEC, Apple, HP, and Tandem, to name a few. This means that programmers can write software which updates databases distributed on a mixed-vendor network using the APPC interface.

However, this is still far from perfect. Each vendor uses inconsistent APPC interfaces and different subsets of the full 47 APPC command set.

As we mentioned earlier in the database discussion, this is the goal of IBM's SAA. Presently, IBM supports APPC on its two major mainframe operating environments, CICS and VTAM, several of its minicomputers (the Series/1, System/36, 38, and 88), and on PCs.

APPC support is limited by the type of transportation link between systems. IBM presently only supports two: synchronous modems and token ring connections. The type of link is called logical unit 6.2, or LU6.2. IBM's APPC is not possible without this type of transportation, although several other vendors offer their own APPC implementations running over non-LU6.2 links.

The vast majority of corporate mainframe connections are over regular coax cables, rather than using synchronous modems. IBM calls these coax links logical unit 2, or LU2. However, there is an interface which marries the best features of APPC with this more popular transportation system. IBM calls this the Server Requester Programming Interface, or SRPI. SRPI uses a subset of the APPC functions, but is supported over coax-connected terminals.

SRPI is used by IBM's Enhanced Connectivity Facilities, a series of mainframe and PC programs which provide services such as virtual disks and host database access from PCs. In other words, with ECF you can store files on the mainframe as if they were on a huge hard disk, reading and writing to the file across a communications link to the mainframe. ECF was first released in October, 1987, and has quickly become a standard mechanism for providing more intelligent micro-to-mainframe links.

Several of IBM's DOS-based 3270 communications programs support ECF and SRPI, so it is no surprise that OS/2 EE does as well. OS/2 EE supports SRPI over a variety of connections, including coax-connections to 3274/3174 communications controllers along with the standard APPC methods such as synchronous modems and token ring connections.

A third interface is one IBM calls the Asynchronous Communications Device Interface. This is IBM's attempt to standardize on the many different asynchronous line connection control and line characteristics. Hopefully, the interface will minimize the hassles of using different asynch modems and serial ports when running communications programs.

Having these three interfaces makes it easier for both program developers and users alike. Developers can write more powerful communications programs which use these interfaces. This means that we will see communications software which can intelligently take control over a session, without any operator intervention, and extract and download data, for example. Users will benefit by not having to babysit each communications program and not having to learn idiosyncrasies of matching specific communications software and hardware.

OS/2 EE version 1.1 adds two other interfaces to this collection: NetBIOS and IEEE 802.2. While we won't get into details here, you should know that these are the two popular LAN programming interfaces which are supported by a variety of PC DOS networks, including existing IBM software such as the PC LAN Program and the LAN Support Program. These interfaces are important for PCs attached via LANs to IBM mainframes.

Future versions of EE promise additional goodies. IBM has committed to running ECF over LU6.2 links, along with support for X.25 packet switched networks and its 3270 High Level Language interfaces for example.

TERMINAL EMULATORS. OS/2 EE supports several popular terminal emulation protocols. Because OS/2 is multitasking, you can run several different sessions concurrently. However, only one asynchronous session can run, although it can run concurrently with synchronous sessions.+

IBM supports VT100, 3101, and 3270 terminals in its first release of EE. VT100 terminals are often used by DEC minicomputers and are a popular choice as the "lowest common denominator" when communicating between different systems. IBM's 3101 protocol is for more simple ASCII terminals.

Both VT100 and 3101 terminal emulation types are used to connect to information services such as Dow Jones News/Retrieval, the Source, and MCI Mail.

A third terminal type supported by EE is for 3270 emulation, the IBM mainframe standard. Up to five or ten 3270 sessions are supported, depending on what hardware you choose to attach to the mainframe. (Synchronous modems can handle up to ten sessions under EE, while token ring and coax connections can handle up to five.)

Not all VT100, 3101, and 3270 functions are supported by OS/2 EE. OS/2 only supports full duplex operation in 3101 emulation, cannot set top and bottom margins with VT100 emulation, and does not support light pens, mice, or APL characters on 3270 terminal emulation.

Future editions of EE will support other terminals, including the 5250 emulation used for IBM System/36 and 38 minicomputers

FILE TRANSFER PROTOCOLS. OS/2 EE supports three mechanisms for file transfer, the 3270 PC File Transfer program, XMODEM, and straight ASCII transfer.

IBM uses the 3270 PC program on three of its mainframe operating systems: TSO, CICS, and VM. However, your corporate mainframe administrator will need to upgrade this software if you want to perform file transfers under OS/2.

XMODEM is a popular asynchronous file transfer protocol used in many DOS communications packages such as Crosstalk and Relay Gold. OS/2 EE supports a form of XMODEM using 128 byte blocks to transfer files.

Any PC file can be transferred via XMODEM. However OS/2 EE has a third method to transfer files via an ordinary ASCII capture/send commands. This means that straight ASCII text files, without any control or special characters can be transferred with this method. Typically, you would use this to capture information from an electronic mail system, or to send a print-image text file to another computer. This straight ASCII capture is similar to what is found on other communications products such as Crosstalk and Smartcom.

As you can see, OS/2 EE wears several communications hats: part synchronous 3270 terminal, part asynchronous ASCII terminal, part LAN workstation. Several of IBM's existing communications software functions have been rolled into the operating system, at the same time integrating the programming interfaces and file transfer features.

This raises some questions. How do you decide whether or not to purchase EE? Let's look at these questions next.

8.2.3 Checklist for OS/2 Extended Edition purchasers

For purposes of this discussion, we assume you have already decided to buy OS/2 SE. Let's outline what questions you want answered before you go out and buy EE.

WHAT EXISTING PC COMMUNICATIONS SOFTWARE DO YOU USE? You should first inventory all the PC communications products in use in your corporation. This may not be so easy. You should include the three types of products that OS/2 EE competes with: asynchronous, 3270, and LAN operating systems.

Essentially, OS/2 EE is another PC communications product and competes with existing packages such as Crosstalk, Smartcom, and Relay (on the asynch side), Irma and PCOX (on the 3270 side), and NetWare and 3+ (on the LAN side). If you are presently using one of these software products and you are satisfied, you may not want to go through all the hassle of switching to a new package.

However, you may find that there are several products in use in each category in your company and the decision may not be as clear-cut. For example, suppose you use Crosstalk to dial up MCI Mail, Irma to connect to your IBM mainframe, and run Novell's NetWare local area network operating system. OS/2 EE will provide functions similar to all three of these packages. But do you really want to give up three working, functioning products and replace with the mostly unknown and new OS/2?

We suggest you start off slowly, evaluating the micro-to-mainframe leg of the triad first. Compare the features and ease of use, memory requirements, and other factors between OS/2 and your existing micro-to-mainframe products. Determine if a protected mode version of the non-IBM communications software exists, and who in your company would benefit from the multiple sessions or ease of background use that comes with the protected mode territory.

DO YOU USE ANY NON-IBM COMMUNICATIONS ADAPTERS? One important factor in your decision to go with EE is the number and variety of non-IBM communications adapters in use in your company. You will have a nightmare trying to get OS/2 EE up and running on a collection of Irma boards, Hayes modems, and 3Com Ethernet cards. Just as with non-IBM video adapters and RAM cards, you might need to obtain a special OS/2 device driver for these non-IBM communications products. Some vendors may decide not to write such a device driver for OS/2 EE.

However, if your situation is so blue that you bought IBM modems, IBM 3270 coax adapters, and IBM Token Ring LAN cards, you should give OS/2 EE serious thought. The product is designed to run on IBM hardware, and you should have fewer compatibility problems.

HOW COMMUNICATIONS-INTENSIVE IS YOUR ENVIRONENT? If your situation is mostly stand-along PCs with an occasional modem here and there, probably OS/2 EE doesn't make much sense for you. You probably would be better off sticking with your existing PC DOS communications software, and either running it in the DOS mode (with the provisos mentioned in Chapter 6), or upgrading to a protected mode version of your software.

However, if you have lots of communications applications going on, and want to move further along the connectivity learning curve, than OS/2 EE deserves more serious consideration. As we mentioned above, EE can replace three separate functions with one consistent constellation.

If you have an IBM mainframe or use IBM communications products, then OS/2 EE is almost a sure bet. IBM will eventually phase out its DOS versions over time, or else add new features to the OS/2 EE that won't find their way into the DOS world of PC 3270 Emulation, PC LAN, and 3101 Emulation programs.

HOW IMPORTANT ARE APPC AND ECF? If you are presently a heavy user of APPC/PC, then choosing OS/2 EE is almost a done deal. All of IBM's efforts on the APPC PC front are focused on OS/2 EE. EE will also lead the way in terms of IBM development efforts on other APPC implementations. This means if you want to get a head start on APPC, you be well-advised to buy OS/2 EE and get started.

A second side of the APPC equation is Enhanced Connectivity Facilities, or ECF. ECF is the here-and-now version of APPC. It has some of the functions and can run on most of the micro-to-mainframe links used by many corporations. If you are considering ECF, you should also be considering OS/2 EE. If both APPC and ECF hold no excitement for you, than perhaps you might be better off without OS/2 EE for the time being.

WHAT EXISTING PC DATABASE SOFTWARE DO YOU USE? IBM doesn't sell any PC database products, so this decision is easier than the communications choices mentioned above. Chances are, though, you are using something like dBase or Paradox. You need to compare the functionality of what the OS/2 EE offers to the protected mode versions of these databases. Can you convert files easily? Will the protected mode buy you anything important? Can you open several different database files in different windows in the protected mode, or must you run a separate copy of the software in each window?

Of course, if your environment is not database intensive, this is an easy call. OS/2 EE is not for you.

DO YOU USE ANY PC-TO-MAINFRAME DATABASE TOOLS? If you have both PC and mainframe FOCUS, or similar database duos like Oracle, Ingres, or Informix, then you have to look carefully at what the OS/2 EE database functions will buy you. This decision is more complicated because you must evaluate both the database and communications features together.

First, you need to determine what communications software and hardware you are using to bridge the link between PC and mainframe. You should look carefully at who is using these products, and whether they have applications which use both PC and mainframe sides. Perhaps no link at all is involved, and folks at your company stick solely with the PC or mainframe versions.

You also need to find out if your database vendor will offer a protected mode version of his software, and what communications environments will be supported with this version.

Another factor is the brand of mainframe database software. If your mainframe database of choice is an IBM product like DATABASE 2 or SQL/DS, then the choice of OS/2 EE is clearer cut. OS/2 EE is designed for these products, and you will be better off running OS/2 EE than a combination of database and communications products from non-IBM vendors.

Basically, your decision comes down to comparing two conversions: (1) going from DOS versions of FOCUS and Irma (or whatever communications software you are using to maintain the PC-to-mainframe link) to protected mode versions of both products; or (2) going from DOS versions of FOCUS and Irma to OS/2 EE. Is the first conversion less troublesome than the second? It depends on the number of applications and the hardware configuration of your PC environment, and whether an IBM or non-IBM mainframe is running the companion database side.

You could also stick with your DOS application, but we assume if you have read this far you are more interested in taking advantage of the promise of OS/2.

8.3 LAN Manager

As we said earlier, the OS/2 LAN Manager is perhaps one of the more significant items in the Extended Edition collection. (While different vendors refer to the features with different names, for simplicity we have adopted the name "LAN Manager" which is the one used by Microsoft as the generic description of OS/2's networking software.

As we said earlier, IBM only includes LAN Manager features and support for network communications in version 1.1 of OS/2 EE. Let's discuss the features and different OEM implementations of LAN Manager in this section. We also provide a checklist of questions to answer before making any decisions on implementing the LAN Manager in your corporation.

8.3.1 Summary of features

The OS/2 LAN Manager is almost a completely new product, and it bears almost no resemblance to its DOS progenitor Microsoft Networks. In fact, they are so different it is hard to imagine that the two products can operate together over the same network.

Yet Microsoft and IBM have designed them this way, so that workstations (PCs which are using files on other machines, called servers) can operate either with DOS or OS/2 and still use the same network wire and share servers running either DOS or OS/2.

If you know or use IBM's PC LAN Program or other implementations of MS Networks (such as those available from AST, Ungermann-Bass, NCR, and other system vendors), you are already half way towards using the LAN Manager. The syntax of many of the commands is very similar, and in many ways the distance between PC LAN Program and OS/2 LAN Manager is equal to the distance between DOS and OS/2.

There are several important differences between the old and new networks, however. Perhaps the most important issue is that the LAN Manager is a full partner to OS/2, rather than another software application program as PC LAN Program was under DOS.

Being a full partner means that LAN Manager shares many of the application programming interfaces with OS/2, along with other OS/2 features such as multitasking, large RAM space, and all the pipes, queues, and interprocess communication functions that we spoke about in Chapter 2. There is some

elegance in this design: OS/2 processes can communicate with each other among networked computers, almost as they do inside the same PC.

The LAN Manager has in fact added some of its own interfaces to make this interprocess communication more powerful. Almost everything that a user can accomplish with a set of keystrokes is under program control. This makes OS/2's networking services easy to use but powerful to expand and integrate into other applications.

One example of how this programming interface can be exploited is illustrated in OS/2's LAN Manager messaging features. Programs detect when error conditions occur on the network and automatically send messages to the appropriate parties for information. The messages are standard ones that users can send to each other as part of the LAN Manager program, but in this case they are sent by software programs rather than actual users.

These changes are good news for many of you who are still suffering with the PC LAN Program. Network administration will be a lot easier with OS/2, since you have one operating system running on both the workstation and the server. Another plus is that programs will be more transportable in OS/2 between a single-user and multi-user application.

This means that a protected mode application written for a single-user PC can take advantage of network services such as shared disks and printers with almost no modification.

None of this was the case with DOS network applications. No matter whether you used NetWare or PC LAN or something else, it was all very different from DOS and required lots of careful attention to a variety of administrative details. DOS applications software often needed a great deal of modification to work under any network operating system as well.

The LAN Manager also brings new and improved functionality when compared to PC LAN Program, especially in terms of user login, access security, remote program execution, printer management, status reporting and network administration. Let's discuss each of these in more detail.

The first and most noticeable difference between PC LAN Program and OS/2's network services is with how you login to the network and access various network resources. In the old days with PC LAN, you needed to create batch files or use the menus to share individual resources. This was cumbersome and difficult to administer.

With OS/2, the user just logs in to the server, similar to the way it is done with Novell's NetWare or Banyan's VINES. A special login script, prepared by the network administrator, quickly and automatically attaches the desired network resources, such as sharing specific subdirectories and printers. All this is done with one login command.

The network administrator can set up as many profiles as necessary, each for a different group of users. For example, everyone in payroll can have one profile, while the engineers and accounts payable departments can have other profiles. In our example, the payroll profile allows them to review the corporate payroll database, while the other guys are prevented from such access.

A second area is improvements to access security. Network administrators can specify execute-only access to files, meaning that network users can only run programs and not copy or delete them. Previously, PC LAN had only read, write, or create/delete options.

This execute-only option can be useful, particularly if combined with some sort of "software checkout" scheme on the network. The administrator can set up a software program which only allows a specified number of users to run a program simultaneously. This scheme may be required by some applications vendors. Thus, with this option, the limits to simultaneous access can now be enforced by the network administrators.

Another security feature is that passwords are stored and transmitted in encrypted format, so that snooping is made more difficult. A network administrator can change but not view a password (if someone is forgetful, for example), similar to the many way mainframe security systems work.

One feature that OS/2 LAN Manager brings new to the party is the ability to actually run programs in the server, using the server's processor and the server's RAM space. While this is not necessarily a good idea for every network program, it can be useful in some applications, particularly with databases if you execute the database application on the same machine that the files are located. Remote execution can eliminate much input/output traffic, but the tradeoff is a more crowded CPU with slower execution on the server.

Network administrators can specify who can use this remote execution feature and which programs users are allowed to run.

Another enhancement is the LAN Manager's printer management features. Printing was almost painful under PC LAN Program, and most of this has been fixed under OS/2, although there still is some work to set up printers.

Print queues can support multiple printers, so that if one printer is busy a print job can be routed automatically to another. This applies equally well to serial and parallel printers, something that couldn't be said about DOS networking programs.

Print jobs can also automatically wait for a specific printer, if for example you want your file to be printed on legal sized paper. Another feature is that you can install print postprocessors, such as if you wanted to run your print job through a Postscript interpreter on its way to the printer.

One interesting addition is that OS/2 talks to the network users. You are constantly informed about what is going on. For example, when your print job is finished OS/2 tells you. If the server's hard disk is almost full (by whatever definition you decide), the network administrator is altered to this fact. As we said earlier, this is an implementation of the LAN Manager's messaging application programming interfaces, whereby software programs send messages directly to network users.

There is a great deal of enhancement to displaying the status of other network activities, such as who logins to what resource, what resources are available to you [show screen shot which ran in 12/15/87

PCWEEK] and in general, and who else is logged on. In fact, the LAN Manager can produce a variety of audit trails and error logs to assist debugging tricky network problems.

One final feature is enhanced network administration. You can already see that many of the already-mentioned items make it easier for administering the nitty gritty of the network. OS/2 also lets complete remote control over a server from a workstation, with the appropriate password. This means that you can configure the server's hard disk, run programs, delete files, in short do everything you could do at the server's keyboard. If you have multiple servers on your network scattered about in geographically hard to reach places (such as in several cities), this could be a big time saver in debugging problems. An administrator would (almost) never have to leave his or her office.

8.3.2 Differences among Microsoft LAN Manager Implementations

Several vendors have licensed the LAN Manager from Microsoft, including IBM and 3Com. These vendors are free to choose what features they want to implement, but most have implemented the complete set so the various flavors of LAN Manager are very similar to each other, just as the various implementations of Microsoft Networks were also very much kissing cousins.

However, there are some subtleties involved. As an example, we illustrate these differences by comparing how IBM and 3Com have implemented the LAN Manager in their OS/2 products. There are important differences in terms of packaging and how they support various network adapters and networking protocols.

3Com's 3+open includes both workstation and server functions in a single package, while IBM has split the functions into its two software products: the LAN Server software (for network file servers only), and OS/2 Extended Edition (for workstations only).

This packaging difference has more to do with how the previous DOS-based LAN products were sold more than any attempt to be different by the different LAN vendors. IBM sells its network software on a single copy basis while 3Com sells its software on a per server basis.

This split reflects the way both companies sold their DOS networking software in the past. IBM sold its DOS-based PC LAN Program on a single-copy-per-machine basis. In other words, you had to buy a separate copy of PC LAN Program for every workstation and every server on the network. IBM included both server and workstation functions in the product, and you select which configuration you want to run when you install the software. Included in the server functions were electronic mail and messaging, and file and print sharing. Separate products are available for gateways to IBM hosts (the 3270 PC Emulation Program Version 3), bridges to other IBM networks (The PC Network Bridge Program), and other functions.

3Com, on the other hand, packaged its 3+ network product line differently. For a single price, you purchased the software for the server and an unlimited number of workstations. (3Com also has a limited, 5-user version available at a lower price.) You purchased whatever server functions as separate modules, including 3+Share (for file and printer sharing), 3+Mail (for electronic mail and messaging), 3+3270 (for gateways to IBM hosts), and 3+Backup (for backing up 3Com's file servers).

IBM continues to sell its OS/2 LAN software on an individual, per-machine basis. (Quantity discounts are available, but you still must buy software for each individual PC from IBM.)

The one important difference is that the server software has been separated from the workstation software. All workstation functions have been packaged as part of the operating system under OS/2 and require OS/2's Extended Edition (at least version 1.1 or later). Of course, as we mentioned in the previous section, you may also run your DOS machines on OS/2 networks. For this situation, you will need the old standby PC LAN Program (at least version 1.3 or later).

Servers require both the Extended Edition (again version 1.1 or later) along with the OS/2 LAN Server software.

You should notice that we said only version 1.1 of the Extended Edition will support LAN services. All versions of Standard Edition and version 1.0 of Extended Edition do not have any LAN capabilities.

3Com has a slightly different method of packaging its OS/2 LAN products, again based on their intial offerings with their 3+ DOS line.

3+open is the name for the package for OS/2 network software, and includes the entire LAN Manager product. This contains both the OS/2 and DOS versions of the workstation features along with the OS/2 server functions. The first version of 3+open will only include file and printer sharing, but subsequent modules will be sold (as with the DOS 3+ products) which provide the remaining functions such as backup, mail, and SNA gateways.

As with the 3+ DOS products, the OS/2 versions are sold in two different packages, both priced per network server. One price is for a limited network of up to 5 users and another is for an unlimited number of users per server.

This packaging difference means that slightly different installation decisions are required between the IBM and 3Com products. With IBM, all workstation functionality is included in the operating system. Thus, when you install the operating system on your PC you also have installed the LAN workstation features. With 3Com, you need to copy the workstation software individually to each PC from the master series of program diskettes.

The different packaging also implies different purchase decisions. The 3Com server-based pricing policy means that the cost of network software is more predictable than the cost of IBM's, since you don't have to calculate the number of individual PC workstations that will be on each network. For larger networks, the cost of 3Com's software may be much less than the cost of IBM's, since you are not paying for each individual PC user.

There is a second major distinction between the IBM and 3Com versions of OS/2 LAN software. Each company's software supports a different set of network adapters. IBM only supports their own PC Network and Token Ring adapter lines, while 3Com supports the IBM Token Ring adapters along with its own lines of Ethernet and token ring adapters (see chart).

Chart: Network Adapters Supported by OS/2 LAN products

IBM (LAN Server/Extended Edition)	3Com (3+open)
IBM Token Ring I	IBM Token Ring I
IBM Token Ring II	IBM Token Ring II
IBM Token Ring/A	IBM Token Ring/A
IBM Token Ring T&P II*	IBM Token Ring T&P II*
IBM Token Ring T&P /A*	IBM Token Ring T&P/A*
IBM PC Network I	
IBM PC Network II**	
IBM PC Network II/A**	
IBM PC Network Baseband	
IBM PC Network Baseband/A	
	3Com EtherLink I
	3Com EtherLink II
	3Com EtherLink Plus
	3Com EtherLink/MC
	3Com TokenLink
	3Com TokenLink Plus

Notes:

* IBM sells two adapters called the IBM Token Ring Trace and Performance adapters for specialized network management functions.

** IBM sells its PC Network broadband adapters which operate on three different frequencies, both for the PC bus and the Micro Channel bus. (The latter is indicated with a "/A" by IBM and a "/MC" by 3Com.)

A third difference is the choice of networking protocols supported by the two companies. Both IBM and 3Com support a protected mode version of NetBIOS, while 3Com also supports a version of Xerox's Networking Services (or XNS) on its software.

3Com include the extra XNS support to remain compatible with its DOS networking products, which also support this protocol. Since IBM does not support XNS on any of its DOS software, they chose not to include any XNS capabilities.

The support for NetBIOS means that existing DOS software which is written to this protocol, such as multiuser databases or communications gateways, can work over OS/2 networks. NetBIOS is also fairly standard among network applications software, and most products have some support for this protocol.

However, Microsoft improved upon the original, using a more efficient utilization of NetBIOS sessions than with its previous DOS-based products. One of the OS/2 LAN services is a feature called auto disconnect/reconnect which can be especially useful in large networks, or on networks with users who only require casual access. It works this way:

As an example, let's suppose that the maximum number of NetBIOS sessions on your network is 32. A "session" refers to the individual conversation between a network user and a network resource, such as a shared disk or shared modem. Individual IBM and 3Com products vary between 32 and 64 sessions supported.

This means that at most 32 (or 64) individual conversations can occur.

Typically, on DOS networks users login to a server, thereby starting a session, and then may not actually access the server for some time. This could be a problem since they are consuming a network session and perhaps preventing others from gaining access to the network.

OS/2's auto disconnect/reconnect feature will end this session automatically after a certain amount of inactivity, and then reconnect the session when the user requests actual network services. This means that the limit on the number of NetBIOS sessions is not as severe as it was under DOS, since sessions are freed for actual network activity, rather than to reserve the potential for communications.

8.3.3 Checklist for OS/2 LAN purchasers

Now you should have some understand about the power and potential of an OS/2 networking system. However, should you buy any OS/2 network software, or stick with an existing network product? And if you buy into the OS/2 networking philosophy, should you go with IBM's or 3Com's? Let's address these issues in this section by answering several typical questions.

IS AN OS/2 NETWORK RIGHT FOR YOUR SITUATION? You should choose an OS/2 network for many of the same reasons for choosing the OS/2 base operating systems software itself: for greater RAM space and multitasking. These reasons make even more sense in a networking environment, which typically require more memory and have more opportunities for running many different programs together on a server.

Another reason to switch from DOS to OS/2 networks is if you have problems with your DOS applications interfering with various network functions, such as hot key conflicts or background conflicts. You should consider using the OS/2 LAN software as one way to resolve many of these multitasking conflicts.

DO YOU HAVE ANY NETWORKS IN YOUR CORPORATION? If you don't have to worry about mixing existing networks with newer ones, your choices are much easier. You can buy the best LAN operating system for your circumstances, without the need to consider how compatible they wil be with any existing products.

ARE YOU USING NETWARE? Many of you that have LANs in your businesses are using Novell's NetWare operating system. You should stick with NetWare if you are satisfied with the operation and features of your network operating system and if you do not have any plans to use any of the more advanced communications features of OS/2 on your network stations. However, if you plan on using OS/2 for non-networking functions, you should carefully re-evaluate your use of NetWare.

One advantage that OS/2's networking services has over NetWare is that, unlike Novell, the same operating system is used in both the workstation and the server. This could help your support personnel, since they would only be supporting a single operating system (if you move completely over to OS/2).

Novell sells an OS/2 Requester which is a protected mode program that runs under OS/2 and allows workstations to access their own NetWare servers. This could provide a temporary transition between using NetWare's operating system on the server and running OS/2 workstations.

WHAT OTHER EXISTING LAN OPERATING SYSTEMS DO YOU USE?

If you use 3Com's 3+ series of software as your DOS network operating system, for example, you should probably stick with 3Com's OS/2 LAN Manager. Keeping your DOS vendor as a standard when you move to OS/2 is the simplest solution. This is especially true of your existing networks use proprietary adapters, such as 3Com's EtherLink or TokenLink cards in our example.

You have two choices if you decide not to continue to use the same vendor on the DOS and OS/2 sides of the house: First, you can bridge the existing DOS network to the new OS2 network. This is done by putting two different network cards in the same server. A second choice is to replace any vendor-specific cards with others than can run under OS/2. Neither choice is as easy to implement as sticking with your DOS networking vendor.

Our example holds for other LAN vendors that have their own implementation of the Microsoft LAN Manager. The differences among these implementations of LAN Manager are less important than the conversion from one vendor's operating system under DOS to a different vendor's under OS/2. There is one exception to this, and that is if your corporation owns any IBM mainframes or minicomputers.

HOW IMPORTANT IS COMMUNCIATING WITH YOUR IBM MAINFRAMES? If you have an IBM mainframe, the choice among different implementations of Microsoft LAN Manager changes somewhat. While, as we have said, the differences between IBM and 3Com implementations deal mainly with packaging, IBM has several cards up its collective sleeves that could change things for the future. IBM has announced its intention to support Advanced Program to Program Communications in its OS/2 LAN Server software. This support means that programs running on OS/2 servers could communicate with

others running on IBM mainframes. If this support is important, you should question your IBM representative carefully as to their future plans for APPC and OS/2.

8.4 Differences among OS/2 OEM implementations

OEM vendors can freely choose what features they wanted to implement when they license OS/2 from Microsoft. Again, as with the LAN Manager, the similarities are more important than the differences. Every version of OS/2, no matter from whom, has the same set of programming interfaces (221, to be exact), the same basic set of commands and command syntax (such as FORMAT, COPY, etc.), the same command processors, and the same basic libraries. This means that an application which runs on one version should (in theory) run on them all.

But there are some important differences among the various versions of OS/2 sold by different hardware OEMS.

One word of caution, however. The computer field is never static for long, and vendors (including IBM) will make changes often to their software. In addition, many vendors are still debating when to support OS/2 for their systems. This means that you should contact the manufacturer of your computer to first determine what support for OS/2, if any, is available, rather than take our word for it.

As you'll see from our discussion, OS/2 is very picky about the particular style of hardware, the peripherals supported, and down to the particular level of BIOS used by each PC vendor. Each manufacturer has adapted OS/2 to run on specific equipment. Sometimes, this is the OEM's own equipment (in the case of IBM), sometimes it is equipment the vendor thinks likely that its customers use.

IBM is particularly strict about what it supports. While Microsoft has developed OS/2 for the OEM masses, IBM chose not to include this largess of support into its version of OS/2. This will cause you much heartache.

The moral of the story is "don't assume OS/2 is just like DOS and can be installed on almost anything". If you have some strange hardware, you will need to contact its manufacturer and find out not only how to make it work with OS/2 but how to make it work with your specific version of OS/2 (IBM, Compaq, or otherwise).

This makes life miserable for folks who have to support a variety of PC equipment in their corporations, especially those of you who have bought a variety of boards over time. Test as many different combinations of hardware as you can before you are sure that it will work with OS/2.

Let's look at some specifics. Again, these might have changed between when this was written and when you are reading this. IBM's OS/2 does not support any communications adapters which use the National Semiconductor 8250 asynchronous communications chip. There is a simple reason: none of IBM's equipment (neither internal modems or AT serial ports) uses this chip. However, Zenith and Compaq both support this chip in their OS/2 versions, because they have machines which use the chip.

You may think this is no big deal, but the 8250 chip is widely used in a variety of serial ports and modems, including in all the internal Hayes modems. (Actually, Hayes uses its own custom chip set, but the chips emulate the 8250 all too well.) This is a problem if you want to run a Hayes modem on IBM's OS/2.

Compaq and some others uses a different disk partitioning scheme for its hard disks, and thus use a separate disk driver and a different version of the FDISK command in its versions of OS/2. IBM's disk driver and FDISK command are based on its PC DOS 3.3 style of disk partitioning.

Compaq also includes a TAPE utility to run its tape drive for backups. Since IBM does not sell any internal PC tape drives, there is no TAPE command in the IBM version of OS/2.

IBM's product does not support any 80386-based machines with an AT-style internal bus. This is because the only IBM 80386 machine is the PS/2 model 80, a microchannel machine. Some of the other vendors, such as Compaq, have these machines and thus have included this support in their versions of OS/2. This also includes support for a 80287 coprocessor when used on a 80386-based machine. This configuration doesn't work with the model 80 PS/2 (it only works with an 80387 coprocessor). So IBM's OS/2 only supports the 80287 on 80286 machines, such as the model 50 and the PC/AT.

Another implication is that IBM's OS/2 won't run products such as the Intel Inboard/386. You will need to obtain a separate Intel OS/2 version to run on their turbo adapters.

Compaq and others include a disk caching utility in their OS/2 version. IBM's version only includes the utility for its PS/2 machines.

IBM's version only supports its own video adapters. The other vendors are more liberal in their video support, including a wide variety of EGA adapters from such manufactures as AST, Genoa, Tseng, Tecmar, STB, and Video 7, among others.

While we are on the subject of drivers, vendors can choose between implementing device drivers in RAM (as most do) or in ROM. [GIVE AN EXAMPLE TK]

IBM's version of OS/2 supports the different BIOS which are used on its PS/2s. Compaq and the others don't expect you to run their version of OS/2 on IBM's PS/2s, so they don't include this extended BIOS support. This is a minor matter, but an illustration of how nitty-gritty OS/2 can get.

Finally, file names are different for some of the utility files among different OS/2 versions. For example, IBM's BIOS files are called IBMDOS.COM and IBMBIO.COM (just as they were in DOS), while the others vendors use OS2DOS.COM and OS2BIO.COM. There are some other file name differences, including the names of the user shell programs. IBM's installation aid, as we mentioned earlier, installs all drivers on your hard disk, whether you need them or not. Other vendors install a subset of drivers.

Chapter 9. Top Ten Issued Facing Corporations

9.1. Is OS/2 a replacement for DOS or an adjunct to it?

The answer to this critical question depends upon the timeframe you are talking about.

For all of 1988 and probably a good part of 1989, the best OS/2 can hope to achieve is a coequal status with DOS. Beyond that period, however, there can be little doubt that OS/2 with its sophistication and rich programming interface will dominate the future.

The reasons why OS/2 won't run away with desktop computing in the short run are legion. First, of course, are applications. The first wave of OS/2 applications is limited to merely strengthening the operation of well-known DOS programs. WordPerfect in protected mode is much like WordPerfect in DOS. These programs--apart from cases where mega memory is critical--don't provide users with much tangible benefit. They are worthy demonstration projects for the new world of the protected mode, but not really significant advances in serving users.

The real punch with OS/2 applications comes in the Presentation Manager generation. These programs will not appear until late 1988 at the earliest, probably sometime in early 1989. Only then will there be compelling reasons to move large numbers of users away from DOS.

Even when first-rate OS/2 applications become available DOS will still live. If a DOS application solves a real business need and its users are content, why rock the boat. From now until forever DOS will be an easier environment to manage than OS/2. It seems to us that a healthy attitude is to assume you will stay with DOS in every situation, until OS/2 presents a solution so favorable that you would be derelict in your responsibilities not to use it. These are likely to come in the form of SQL databases and front ends and programs that provide cross-system communications functions integrated with other tasks, such as the capability to use APPC protocols to ask for data from within a spreadsheet.

The one place where OS/2 could become a replacement for DOS in the near term is in corporate written applications with large memory requirements. If you have been chafing at DOS memory limits or have had to postpone development of massive applications on the desktop because they won't fit into 640K, OS/2 and its immediately available programming tools could become a development environment of choice very quickly.

For all the spunk and value DOS has, though, in the long run, OS/2 will replace it as the standard operating system for micros. The most fundamental reason this will happen is that OS/2 is a genuine full-dress operating system, the kind of tool a corporation needs to squeeze benefit from a computer, and DOS is not. The number of user situations where OS/2 makes more sense than DOS will multiply in the ever more complex world of microdom over the next two years. The number of OS/2 applications with capabilities far beyond those of DOS programs will snowball. The abilities inherent in OS/2 itself will increase as it migrates from 80286 to 80386 based machines.

In summation, we don't think OS/2 demands that you refit all of your machines and retrain all of your users today. The announcement of a new highway doesn't mean that everybody will stop driving on the

old road tomorrow. There's time, and plenty of good use in the old road. But, we don't think any sane micro manager can ignore OS/2, either. Like that new road, its six lanes wide, lets you move at high speeds, has slick interchanges. The big commercial traffic will eventually succumb to its temptations. If you want your micro management business to last, you'd better take an option on some land near the OS/2 highway.

9.2. Must I put significant resources into studying/working with OS/2 before we make a decision on how we use it?

Yes. We think OS/2 is complex enough and critical enough that making the best use of it will require as much hands-on familiarity as any organization can achieve. Think back to all the problems you had in the early days of DOS, and multiply them by ten. That's likely to be the level of travail that occurs when you make the switch to OS/2.

One reason OS/2 requires advance planning and study is because so many aspects of microcomputing that were tangential to DOS are deeply rooted within OS/2. User interfaces, hardly an issue in DOS, is likely to become its own area of specialization in the world of OS/2. Programming considerations loom large with OS/2, stemming from its close ties to C and its uncertain interactions with other languages, as well as its requirement for new technical tricks. The API of OS/2 so much fuller and more intricate than that of DOS will require its own expert, if you want to tinker with your applications as has become common in the DOS world. Compatibility and user support issues will also loom large in any OS/2 conversion and will vary widely with each installation. What equipment you already have the knowledge level of your users, the network you've installed. All these can affect the way you will introduce and use OS/2 and so must be understood by someone in your organization.

The specific aspects of OS/2 you study, and the amounts of manpower you assign to the task are questions only you can answer. In general, though, we think it takes from 3-6 months to bring a team fully up the OS/2 learning curve. And we believe that prior to shifting to OS/2 you will need as many identified and trained experts in OS/2 as you currently have for DOS. Once you make the switch to OS/2 you will almost certainly need more people, on a per user basis, than you do with DOS. It's simply a more complicated world.

How much effort should I put into working with 1.0? 1.1?

OS/2 1.0 in both the IBM and Microsoft versions is unmistakably a transitional, testing oriented product. It does not represent the full flavor of OS/2. The product was created primarily to give application developers and corporate managers something to look at and plan around, while the rest of the OS/2 pieces were completed.

As a result, we do not recommend that anyone commit his users to OS/2 on the basis of the first version. That's not what it was really intended for. If you have an emergency need to get a group of user into protected mode or a programming project that can't wait, by all means use 1.0 to support that effort. But keep the shifts small, well defined, bounded, and focused on memory benefits and little else that OS/2 brings.

OS/2 1.1 will represent the first appearance of the genuine new age article. The focus of effort, in terms of developing specific expertise and determining your development plans should be around 1.1.

Keep in mind, however, that 1.0 is older, has more experience behind it, more testing and more of the S®MDSU¯E®MDUL¯T (Somebody Else's Troubles) factor. It would seem to make sense to build a core of expertise in OS/2 1.0, to use that basis for evaluating OS/2 1.1, and only then shift all or part of your plans to 1.1 when you are convinced that it is ready and will work for you.

Simply waiting for 1.1 to happen will almost surely put you too far down the slope to be your most effective when it arrives.

9.4. What does the extended edition mean for my users, and what attitudes should I take toward it?

In its early forms--versions 1.0 and 1.1 OS/2 extends the range of the microcomputer operating system into realms where DOS never tiptoed. With the emergence of the Extended Edition, OS/2 expands its sphere of influence into the world of applications. In the Extended Edition the operating system and a number of core applications become deeply intertwined. Specifically, the Extended Edition adds database management and communications functions to OS/2 version 1.1.

This process of intermingling applications and systems software is common on larger computers, and it being spearheaded at the micro level by IBM, primarily to make it easier to integrate micro work with IBM midrange and mainframe machines. The Extended Edition is an IBM product, designed for IBM environments.

So, on the surface it would seem that the Extended Edition should be considered important by those who are in an IBM environment and who use database or intersystem communications applications heavily. Anyone whose situation conforms to this description should study the Extended Edition closely and prepare to make it a central component of desktop strategies.

But the influence of the Extended Edition goes far beyond the IBM product itself. Just as IBM has used operating system products to establish standards for third party developers in the past, it will use the Extended Edition to create standards for application developers. However IBM's Extended Edition Database Manager works, so will dozens of other database products. Whatever protocols and formats are supported in the Extended Edition Communications Managers will become de rigger in all communications products.

Beyond even that, the Extended Edition is important as a standard because it represents the desktop outpost of IBM's highly strategic Systems Application Architecture(SAA). SAA is IBM's plan to build a base that links applications, languages, screens, across all of Big Blue's hardware. The Extended Edition is the PC component of that system. It is designed to interact with other SAA elements, such as VM running on a 9370, in a seamless fashion never before possible from IBM. The development of such a global cross-system environment is so important to IBM and so alluring to many of its biggest corporate customers, that it will cause the Extended Edition to dominate in many companies, even if better performing alternatives are available.

In short, the Extended Edition stands an excellent chance of defining the industry standard for desktop database management and communications as well as traditional operating system functions for the next generation of micro products. This means that the Extended Edition has major potential impact for users who PCclones in non-IBM based corporate computing environments. On this basis, we think the Extended Edition must be considered an essential area of evaluation for all micro managers.

In addition, third party vendors are certain to offers "clones" of the Extended Edition for non-IBM systems. Microsoft, Ashton-Tate and a number of database vendors have endorsed an OS/2 database server that makes standard PC products, such as dBase and Paradox general equivalents to the Extended Edition's database for workstations. DCA, Hayes and other vendors have expressed their intentions of providing communications capabilities under OS/2 that match IBM's. If these third party products are strong enough, and compatible enough with IBM's offering, they could establish an Extended Edition standard for all Intel architecture micros.

To us, this would be the best possible state for users and their managers. If it occurs, it means the Extended Edition--OS/2 with Presentation Manager, database and communication functions built in--becomes the basis for micro operations. When DOS created a non-vendor specific standard that extended only as far as file handling and peripheral access, microcomputing exploded. With a standard that covers everything from memory and screen access to downloading mainframe data, we think micros could make another great leap forward.

So, we think micro managers should back the development of extended editions, not only by IBM, but by every possible vendor, and should encourage their users to see the Extended Edition, not as a loss of freedom of choice, but as the bestowal of a vast new power.

9.5. Which vendor should I purchase OS/2 from--Microsoft or IBM?

Actually, the purchase options for OS/2 aren't between IBM and Microsoft but between IBM and other hardware vendors. Microsoft is developing OS/2 but won't sell it to the public; instead, Microsoft will license OS/2 to hardware vendors who can then customize it for their machines and sell it to their customers. IBM is, obviously, one of those hardware vendors. So are most of the other leading makers of PCs, including Compaq, Zenith, Hewlett-Packard and many more.

In the DOS world, the choice of hardware and operating system weren't particularly intertwined. You could buy any reasonably compatible PC and run most versions of DOS on it. But in the OS/21 world this is not the case, and the decision about who to turn to for OS/2 support begins when you buy your hardware.

First, OS/2--because it is based on drivers that describe the hardware beneath--is far more vendor sensitive than DOS. Each motherboard, each memory expansion, each monitor and peripheral must be described to OS/2 precisely by its own driver of the operating system won't be able to work with it. This means that if you buy hardware from a vendor who doesn't not license OS/2 or at least have the capacity to create his own drivers for OS/2 you could get into serious trouble.

In terms of PC boxes, IBM boxes are the only ones guaranteed to run IBM OS/2, Compaq boxes the only ones certified to run Compaq OS/2, the same with Zenith and other versions that come out. PC bought from vendors who do not have their own versions of OS/2 may not be able to run OS/2 at all, will almost certainly have at least occasional operational bugs if they can run it, and won't be able to keep up with the changes in the operating system as it grows. In the OS/2 world buying hardware isn't a one-night stand as it DOS days, but the beginning of a long lasting relationship, as has been the case with major software vendors for some time.

Beyond the box itself, peripherals and expansion cards also must have compatible drivers. Again, this means that, where OS/2 is concerned buying upgrade hardware is at least in part a software decision and the beginning of a relationship. You are buying price, performance, and a new factor, the ability to provide drivers and to expand, improve and modify them as OS/2 changes. Any less commitment than that from a hardware vendor could leave you high and dry at any moment.

However, this doesn't mean you must become a slave to IBM or any other hardware vendor. It just means you should be careful. To some extent you may still be able to have your cake (IBM software) and eat it too (with clone hardware) just as in the DOS past. IBM OS/2 version 1.0 runs fine on Compaq and Zenith PCs, meaning you can (at least for the moment) still have the comfort of IBM sanctioned operations, with the luxury of less expensive or better performing third party machines. Still, even with these boxes that situation could change with future version of OS/2. And it doesn't work fully in reverse. Compaq and Zenith OS/2 will probably run fine on IBM AT machines, but not well at all on the PS/2s, which are where IBM's future lies.

In summation: Buy OS/2 from a PC manufacturer whose machines you like and whose technical skills you trust, because that company will largely control the future course of your OS/2 offerings.

9.6. What impact will OS/2 have on my relationships with my current vendors?

It depends, of course, upon whom you are doing business with. The broadest impact of OS/2 on vendors is likely to be the creation of two camps. On one side will be those hardware and software companies with enough money and staff to take on the expensive R&D for OS/2 products. On the other side are those vendors who just don't have the oomph to take on this complicated transition.

Virtually all major micro vendors have grown large enough today that they will make the switch to OS/2 as quickly as possible. The top five vendors in any category, the same companies seen in the DOS world as having critical mass, will form the membership of the OS/2 club.

Managers who have based their purchase around these mainstream vendors will find little dislocation in moving toward OS/2. They should be able to follow a progression from existing DOS wares, to family API (protected mode) versions, to Presentation Manager products with file (and/or data) compatibility with DOS versions, to versions that offer or use Extended Edition services and finally to versions that expand all their OS/2 capabilities to take advantage of the full resources of the 80386 chip. For these managers, OS/2 will represent more offerings from traditional vendors.

Trouble arises when, for money saving or other reasons, a manager has based his purchasing around programs from smaller or less full service vendors. As noted above, PC purchased from sources that don't license OS/2 may not even be able to run the operating system. The same goes for boards, modems, drives and other peripherals. On the software side, where a vendor doesn't have the wherewithal to develop for OS/2, his customers are going to be faced with the choice of keeping their vendor and staying in DOS or of starting over with a new vendor and all that entails--new file formats, commands, training, etc.

Another vendor wrinkle in the OS/2 fabric is the emergence of a new class of vendors in the micro world. These companies have built reputations at the mini or mainframe level. Their products have never been able to squeeze into 640K and the workings of micros have always seemed alien to them. In OS/2 they see a familiar looking environment; and in the protected mode they see an operating expanse they can fit in. These companies may very well be able to accomplish benefits in OS/2 that micro-based vendors can't immediately match, and should be taken seriously as new alternatives by micro managers.

We think managers should undertake complete inventories of the products they use and vigorously question their vendors about OS/2 plans well in advance of any decision the company makes on just how to use OS/2. If it looks like a particular product won't migrate to OS/2 in an application where benefit could be achieved if users moved ahead, the manager should begin scouting new vendors and products early.

Even where a specific problem is not evident, now is a good time not to stand pat with existing vendor relationships, but to take advantage of the shift to OS/2 to review vendor offerings overall. Make existing vendors tell you their plans and convince you of their continued value to you. Let the also-rans from DOS days give you their best pitch on how they might sprint ahead in the new environment. Change is in the wind where OS/2 is concerned, which makes it a good time to check out the horizon as regards what your vendors can do for you.

9.7. What impact will OS/2 have on my local area networking decisions and the way I do terminal emulation?

OS/2 is going to create massive short term confusion in the PC-based communications game. Partly this is because of its sheer complexity, and partly because of the many communications hats that OS/2 wears.

OS/2 EE is a terminal emulator for both the mainframe 3270 and asynchronous worlds; it has the workstation operating system for a local area network. And it has the potential for running APPC and a bevy of programming interfaces as well. This means it is at least five different packages in one, and we haven't even mentioned the database features.

If you use IBM communications products now, you most likely will find some not-so-subtle pressures to replace these with OS/2 EE. And if you don't, you'll find lots of problems with incompatibilities between non-IBM communications products and IBM's OS/2. What this means is that life will become complicated.

But there is some hope. In the long run, OS/2 will likely to become the standard platform for workstation-server interaction and one of several options for server systems software. It also has a good shot at the terminal emulation market as well.

Over the past couple of years a general set of standards has emerged for LANs. DOS 3x is the workstation operating system. NetBIOS is the protocol for workstation-server interaction. The server software is either Novell's Netware, 3Com's Ether system or the PC Network/MS-Net product from IBM/Microsoft.

OS/2 is going to throw that neat picture into a cocked hat. To start with the workstation: OS/2 immediately doubles the number of operating system choices for networked PCs. OS/2 or DOS? To make matters even more complicated, in order to interact with a server, OS/2 requires the addition of a separate LAN software module. Microsoft calls this The LAN Manager. IBM includes it in the Extended Edition Communications Manager. So, the workstation operating system choice in OS/2 is actually threefold: DOS, MS OS/2 with LAN Manager, IBM Extended Edition.

At the server, things become even more complicated. Novell, the current market leader, will continue to use its Netware server software to support DOS workstations and to provide some functions to OS/2 workstations. Network applications that use the LAN Manager will be supported by a separate OS/2 applications server that either works with a coprocessor in the Novell Netware server or resides in its own box. Applications that work with the IBM Extended Edition will be supported in Netware directly, by inclusion of support for the APPC communications protocols IBM has said it will use in the EE. You with us so far? Wait, it gets worse.

3Com has become Microsoft's networking partner for OS/2 and plans to build all of its future products around OS/2 and the LAN Manager. This means that 3Plus Open, 3Com's OS/2 product line, will fully support all the functions of the OS/2 LAN Manager and provide DOS level support to DOS workstations. At some point in the future the product line will support IBM's EE protocols, but not right away.

IBM has linked its server software to its OS/2 Extended Edition. When the communications components of the EE are installed, they will talk with the IBM OS/2 LAN Server program on the network server. The LAN Server program will also be able to talk with DOS workstations running the most recently announced version of the PC Network Program. The LAN Server program will be able to work with OS/2 workstations that don't have the Extended Edition in place, but only in a limited way.

If this sounds like a confusing mess, that's because it is. Each of the major LAN players has staked out a slightly different position designed to best fit its strategy. Nobody seems to be in a position here to create a standard in the short term, not even IBM. Which means that managers will have to cope with partial solutions to their networking needs until one or another of these schemes reaches critical mass in the marketplace.

Our best guess for how it will all pan out is this: OS/2 will become the operating system of choice for workstations. However, the LAN Manager won't fly long-term. Instead, IBM's Extended Edition protocols and clones of it will take over the function of getting a PC and its applications talking to the network. At

the other end of the link, we see the idea of a single network server being torn apart in the OS/2 world. In its place, LANs will be overseen by sets of specialty servers. Novell, which is marvelous at file handling, should continue to do well at offering file servers. This task will also be taken on increasingly by VAXes, IBM 9370s and other midrange computers.

These file servers will cooperate with database servers and general application servers. A general application server in an all-IBM environment might run the LAN Server Program to provide the best possible support to EE applications. In a non-IBM system, 3Com software could run on the application server, interacting with EE clones or the LAN Manager on the workstations.

Database servers will run DBMS software that will interact across the network with front ends on the OS/2 PCs. These DBMS server products will surely be SQL-based--since every player has stated that he will use this method of relational database building--but may not be fully compatible with one another, even so, because SQL has many flavors. IBM, Novell, 3Com, Microsoft and many others will have such products. As is usually the case, the IBM product, which won't be ready for release until sometime in mid-1989, will set the long term standard. Until then, the Microsoft/Ashton-Tate system will probably take the lion's share.

If you boil these trends down into questions of day to day operations, they point to tough times for LAN management. First, OS/2 ushers in a difficult period of product decision making that should be handled with great care because it will define the LAN vendors you'll probably be working with for years to come. Even after you're comfortable with what your product choices will be, life remains difficult. You need to plan the integration of OS/2 networks and workstations with your existing DOS products. This means managing two sets of standards at once. Finally, when OS/2 networking is in place, you'll be facing a LAN world of great complexity, with multiple servers, varying levels of workstation services and tight interaction with applications.

Successfully managing the shoals of OS/2 networking may prove the greatest immediate success possible in implementing OS/2 in a corporation.

9.8. Should I standardize on 286 or 386 systems?

Here, we're going to give you our prejudice and then we'll tell you what's practical.

Our prejudice: Go with the 386 and damn the expense. Our reasoning: The 80286, as we discussed in Chapter 1, was a flawed processor when it was released and remains one today. OS/2 has done an admirable job of masking the worst faults of the 80286 but hasn't made them vanish; they are still in there, limiting your options in the future.

Today you can buy an 80286 or an 80386 machine and achieve precisely the same level of functionality from either. OS/2 does not use any specific characteristics of the 80386. The 80386 machine may run a bit faster, but that's really the only benefit in terms of running OS/2 today.

But that doesn't mean the 80286 and the 80386 machine are equal. Today, the 80386 machine provides you with options to OS/2 the 80286 machine can't. For instance, you can look at windows 386 or

Software Link's 386 based multitasking operating system as alternative solutions for some of your users if you have 80386 machines installed. These products, working with many of the tools already out for OS/2, including the latest version of C and MASM 5.0, allow you to begin writing your own corporate programs immediately that do utilize 32-bit instructions, paged memory, easy switching to real mode and other notable features of the 80386.

In the future, the benefits of 80386 machines grow. Microcomputers traditionally have one life. You buy them as leading edge products, use them intensely for a couple of years, and then--right about the time they are depreciated--you have to unload them and buy an new generation of more powerful hardware. The 80386 micros may be the first generation of PCs with two lives. An 80386 machine can have a full product lifecycle running in 80286 mode with OS/2 for the next year to eighteen months. Then--mirable dictu--when 80386 versions of OS/2 become available in late 1989, the 80386 machine can have a whole second life, running in native 386 mode. Machines with 80386 processors may have usefulness that outlives their depreciation, a significant fiscal benefit.

A third, less obvious advantage to 80386 machines, especially for your management and evaluation staff, is that they have 386s in them. A developer with an 80286 machine can learn about the 286 and that's it. A developer with an 80286 machine can learn everything he could on a 286 and has the option to begin noodling around with new 386 features, too. You can better prepare your staff for the future by letting them sample it now.

There is no question in our mind but that the 8088/86 is the chip of the past, the 80286 is a mere transitional chip, and the 80386 is the processor that will dominate microcomputing for a generation.

There, that's our prejudice. But we are practical enough to realize that it's your money and you probably don't have infinite amounts of it. So, from a pragmatic point of view, we'll give somewhat more messianic advice.

We think that, at the least, development and programming staffs should have 80386 machines. The 386 is simply a better processor and they can learn more. Regular users won't get any benefit from an 80386 for as long as two years, so saving money by buying those 80286s is not at all a bad idea. We would recommend, though, that you anticipate upgrading at least your power users, and probably all your knowledge workers to 386s within that two year span. So, you should have in mind firm plans for what to do with the 286 machines you buy today two years from now. Pass them down to secretaries? Offload them on the used market and write off the cost? You should also begin to soften up management to what might look like an unnecessary dual expense. We'd begin referring to 80286 purchases as stopgaps, temporary fixes, maintenance purchases, and the like.

If you have LANs in place, consider making your servers 80386 based machines. This will get your techies working with the processor, bring you an immediate speed advantage, but keep the number of actual 80386 boxes you have small.

Finally, with the future in mind, we would recommend that you make your primary hardware relationships with companies that are at least experimenting with the 80386. You want your vendor to be as ready as you are to make the move when the time comes.

9.9. How will OS/2 affect the way I use my development and support people ?

Basically, you'll need more of them and they will need to be both more formally technical and more specialized than was the case in DOS.

DOS was never really a development environment. It was so sketchy an operating system that users themselves could just about master it on their own. A DOS expert was someone profient at batch files and hard disk management, a fairly limited repertoire. Genuine developers worked with assembler or high level languages.

In OS/2 the operating system itself becomes a potential development environment and the number of support areas vastly increases. Starting from support for users and working back toward pure systems issues, here are the development and support areas we see:

OS/2 installation and maintenance. You will need someone to add all that memory, to run the install programs, write the batch files, set up the screen groups. In DOS days, the cocktail party ratio for user hand holders to users was one to 50. For OS/2 we wouldn't be surprised to see it drop to one to 30.

OS/2 screen management. You will need someone who is expert at the Presentation Manager and any other user shell you plan to work with. The Presentation Manager has an API that would rival many programming languages. Using that tool, controlling it, mediating between applications that are accessing it, allowing users to customize their environments with it, all will require on hand expertise. We think you'll probably need at least one expert per user interface, and probably support staff at the rate of about one per 100 users or one per site.

OS/2 API expert. In addition to the screens, you'll need someone who really knows the OS/2 APIs. You'll need this for writing the utilities, patches, fixes and customizations you need for day to day computer use. You'll also need it as a basis for judging the proper behavior of commercial applications you are considering. Also, you may find that the OS/2 API itself can become the development tool for small applications. Finally, corporate application developers will need an expert to turn to for API subroutine advice.

Hardware and software evaluators. OS/2 brings with it a fresh explosion of hardare and software evaluations. The number of products coming in the next two years will certainly dwarf anything that has preceded them, even the heady early days of DOS. More than ever the value of these products will be in their interaction with other products, requiring evaluations on a company by company basis. In the OS/2 world, one man's solution could easily be another's disaster.

OS/2 networking expert. The LAN Manager, the LAN Server, and all that goes with them will require its own guru for effective implementation. Actually, getting the network up and running will surely demand some support staff, in addition to this systems planner.

If you already have staff in place, this is the year to get everybody to Microsoft University or IBM customer briefings or some other educational seminar. Identify today the roles you'll want your current people to hold in the OS/2 pantheon.

If you've been doing all the thinking yourself up to this point, you had better start lobbying for planning and development help immediately. OS/2 will demand it. If shifting, increasing or acquiring staff is out of the question, begin auditioning consultants to help you handle OS/2 problems.

If you can't afford a consultant, don't move to OS/2. You'll be over your head almost immediately. Going to OS/2 is not a one-person project.

9.10. How will OS/2 affect applications written within the corporation?

Our guess is that OS/2 means there will be more of them. Even where commercial applications are used--and they will remain the bulk of the desktop business, we feel--so much customization, interfacing, connectivity and general fiddling will go on around the vanilla product, that it will become non-generic. We think there will be far less out of the box use of software than there was in the DOS world.

The in-house application work that is done will skew heavily in the direction of C, even for applications where COBOL. FORTRAN and other long used language have traditionally dominated. Under OS/2 we believe C will be the language of choice for every application, except on the rare cases where another language offers an unattainable benefit.

Between C and the OS/2 API, we think the need to write in assembler will diminish. This is also true because developers both corporate and commercial, want to be able to port their programs easily to the 80386 or even to Motorola 68000 based system such as Apple's Macintosh.

At the same time, we think the use of task specific languages, such as SQL for databases, and nonprocedural languages, the DDM in Microsoft Windows, or Lotus' LEAF, will increase enormously. You may actually need to have someone trained to program in the specific languages that are built into each of your major applications.

The last trend we see from OS/2 in corporate program writing is a massive shift of formerly mainframe based application development to PC and PC-based networks. DOS couldn't support the programming tools to generate mainframe compatible code; OS/2 does. Also the increase in workstation networking has shifted the center of corporate program generation away from the mainframe and toward multi-CPU, multi-operating system networked environments. OS/2, with its large memory capacity, well behaved operations and strong network ties, could prove ideal.

So, in conclusion, we think OS/2 will spur corporate software development, increase the importance of C and non-procedural languages and speed the movement to shift corporate applications from mainframes to distributed environments.

Appendix A: OS/2 Command Summary

Most OS/2 commands can be typed with several optional parameters. Options provide OS/2 with extra information about what a command should do where it should seek or send data, or how result should be displayed. When options aren't used with a command, OS/2 either prompts the user for the information or reverts to a default setting. The default settings for each command are discussed in the command summary that follows.

In discussing options here and in the command summary that follows, these type conventions are used:

Style	Example	Meaning
italics	filename	This is a variable that you must provide a name or value for
[brackets]	[\directory]	Information within brackets is optional. Type the information within, not the brackets
... (ellipsis)	directory...	You can repeat an option followed by ... as many times as needed.
separators	copy fred	Separates a command from its option; Usually just a space; occasionally a semicolon or equals sign.

Here are the common options for OS/2 commands. Not all options apply to all commands. See the command descriptions for more detail.

	Option	What It Does
drive:	c:	Convey a disk drive name
path	[/directory][/directory...]	/directory Creates a set of directory names in order of search priority
filename	fred.bat	The name of the file to be acted upon
pathname	/directory/fred.bat	A path with a filename at the end
switch	/p,/s,/w	A / followed by a value sets an option specific to a particular command
argument	ansi [on]	Sets a conditional situation. Usually on or off.
string	abcde	Gives the command a string of letters, numbers, other characters or spaces to act upon.

ansi

Mode:Protected Mode Only

Syntax: ansi [on] or [off]

Discussion: Installs ANSI escape sequence support in protected mode.

ANSI sequences are series of characters that can define functions in OS/2 for some applications.

append

Mode: Real Mode Only

Syntax: To specify directories to search:

append [drive:]path [;[drive:]path]...]

To remove appended path:

append;

Discussion: Sets a seach path that located files outside the current directory other than .exe,.com and .bat files, which can be located via the path command.

Example: To seach data files in a subdriectory on C: called chapters in the book directory as well as in the notes subdirectory of the book directory on A:, type:

append c:\book\chapters;a:\book\notes

assign

Mode: Real Mode Only

Syntax: assign [x[=]y[...]]

where x is the drive OS/2 is currently writing to, and y is the drive you want OS/2 to write to instead

Discussion: The assign command allows you to use drives that your application may not recognize. If the application only recognizes drives a: and b:, you can use assign to read and write files from a harddisk c:. The command typed with no following argument resets all drives to their original assignments.

This command performs a functional virtually identical to that of the subst command. To ensure maximum OS/2 compatibility, use subst instead of assign, wherever possible. Since assign masks the true type of device, don't use this command with other commands that need accurate device information, such as backup, restore, join and print. Also, format and diskcopy ignore assignments.

Example: assign a=c b=c moves all operations from floppies to the hard drive

attrib

Mode: Both

Syntax: attrib [+-r] [+-a] [drive:]pathname [/s]

where:

+r sets the read-only attribute of a file

-r turns off read-only mode

+a sets the archive attribute of a file

-a clears the archive attribute of a file

/s causes attrib to work with all subdirectories, as well as the noted path

Discussion: The attrib command sets two attributes for files. The read-only attribute (r) puts a file on read-only status. The arvhice attribute denotes files that the user wants to be able to backup, restore or xcopy.

Example:

attrib chapt2 would display the attributes for the file of this chapter

attrib +r chapt2 would put the chapter on read only status

attrib +a would put the file on archivable status

backup

Mode: Both

Syntax: backup [drive1:][path][filename] [drive2] [/s] [/m] [/a] [/f] [/d:date] [/t:time] [/L;filename}

where:

drive1 is the drive you want to backup

drive2 is the target drive for the backed up files

and the switches set the following options:

/s backs up subdirectories

/m backs up only files that have changed since the last backup

/a adds backed up files to those already on the traget disk. (Does not work files created by backup in DOS 3.2 or earlier)

/f formats the target disk before backup; floppies only

/d:date backs up only files modified on or after the set date

/t:time backs up only files modified at of after set time

/L:filename logs backed up files in a file of the name specified; if no filename is given a file called backup.log is created in the root directory of the file where the files being backed up are located

Discussion: Backup creates an environment for backing up files among disks of different sizes and types. Dont use backup with drives that have been assigned, joined or sbustituted with assign, join or subst.

Example: backup c:\book\chapter2 a: /m /L:booklog

backs up the the file chapter2 in the book directory, but only if it has been changed since the last update. If the file is updated, it is logged in the booklog file in the c: root directory.

break

Mode: Real Only

Syntax: break [on] or break [off]

Discussion: This command extends the control-C (the control and break keys hit together) break function beyond keyboard, screen and printer operations to other activities of an application, such as disk access. Break [on] broadens the sphere of the function; break [off] limits it to keyboard, screen and printer.

chcp

Mode: Both

Syntax: chcp [nnn]

where:

nnnn is the code page selected

Discussion: This commands selects one or the two code pages you asked OS/2 to preapre in the config.sys file. A code page is a table that defines the character set used by keyboard, screen and printer. OS/2 allows the following code pages to called by chcp:

437 United States

850 Multilingual

860 Portugese

863 French-Canadian

865 Nordic

If you type th command with no value following. OS/2 will list the active and prepared code pages.

Example: chcp 865 sets the code page for Nordic (if it has been prepared by config.sys)

chdir (cd)

Mode: Both

Syntax: chdir [dirve:][path]

Discussion: The chdir command, which can be abbreviated to cd, changes the working directory.

Example: chdir \book\chapter

changes the working directory to the \book\chapter subdirectory

cd \book\chapter does the same thing

chkdsk

Mode: Both

Syntax: chkdsk [drive:][pathname][/f][/v]

where the switches mean:

/f fixes error found on disk; if this switch isn't used, chkdsk tells you about errors, but doesn't fix them.

/v displays the name of each file checked by chkdsk

Discussion: The chkdsk command checks the status of your disk. In real mode, chkdsk also display a report on available system memory; this does not occur in protected mode.

Examples:

chkdsk a:

produces a report on the status of drive a:.

chkdsk a:>report

directs the chksdk report into a file called report; this should not be done when the /f switch is active

chkdsk a: /f

corrects errors found on drive a:

cls

Mode: Both

Syntax: cls

Discussion: Clears the display screen so that only the OS/2 prompt and cursor remain.

cnd

Mode: Protected Mode Only

Syntax: cmd [drive:][path] [/c string] or [/k string]

where:

string is a command or commands that you want to pass to a new command processor

/c tells the command processor to perform the command or commands noted by the string and then return control to the primary command processor.

/k tells the command processor to perform the command(s) in string and then to stay in memory after the commands are completed

Discussion: The cmd command starts a secondary command processor in protected mode. cmd.exe is the standard comand processor in protected mode, but others can be used. They are loaded with cmd and are described to the system with the protshell command in the config.sys file, which is discussed in Chapter 4.

Example: cmd /c dir b:

Tells OS/2 to load the new command processor under the current program, call a directory of drive B:, and then return to the primary command processor. A /k at the end would tell the new command processor to stay in memory after running the command.

command

Mode: Real Mode Only

Purpose: Starts a scondary command proecssor in real mode.

Syntax: command [drive:][path][/p][/c string][/e:nnnnn]

where:

/p keeps the secondary command processor in memory and doesn't automatically return control to the primary command processor

/c string tells the command processor to perform the command or commands noted by string and then automatically return control to the primary command processor

/e:nnnnn Sets an environment size for the new command processor, with nnnnn as the number of bytes, from 160 to 32768

Discussion: The real mode counterpart to cmd. You should always set the environment size when loading the secondary processor in real mode, since it can't be changed after most commands are run.

Example: command /c dir b: calls a secondary command processor under the current application, has it call a directory of

drive b: and then return control to the primary

command processor.

comp

Purpose: Compares the contents of two sets of files

Syntax: comp [drive:][pathname] [drive:][pathname]

Discussion: The files compared by comp can be on the same or different drives, the same or different directories. They can have the same path and filenames as long as they are on different drives.

Example: comp c:/book/chapter/*.txt b:*.bak

compares the files with extension .txt in the /book/chapter subdirectory with the .bak files on drive b: This would allow a user to see if those .bak files` are current or not.

copy

Mode: Both

Purpose: Copies files from one location to another; also appends files

Syntax: copy [drive][pathname] [/v] [/a] [/b] [drive:][pathname]

where:

/v tells OS/2 to verify that the sectors on the target disk have

records the file information properly

/a copies files in ASCII format. In ASCII format the command processor

reads until reaching an end-of-file (cntrl-z) marker.

/b copies files in binary format. In binary format the command procssor

reads the number of bytes specified for the file in the directory

to append files:

copy pathname + pathname [...] pathname

Discussion: If the second drive and pathname are not used, OS.2 autoamtically saves the file in the working directory or the default drive. If a file of the same name is already there, copy won't run. Copy is for small groups of files. If you want to copy all of adirecotyr files or subdriectories, use the xcopy command.

Example: copy c:/book/chapters/chapter1 /v b:chapter1.bak

copies the file chapter1 in the c:/b4ook/chapters subdirectory into the file chapter1.bak on drive b: and the verifies that the copy has occurred correctly

copy c: chapter1 + chapter2 + a:chapter3 b:book

appends files chapter1, chapter2 and chapter 3 and then copies them into a single file called book on drive b:.

date

Mode: Both

Purpose: Tells the system the date.

Syntax: [mm-dd-yy]

Discussion: When entered permanently sets the computer's internal clock.

del (erase)

Mode: Both

Purpose: Deletes files.

Syntax: In real mode:

del[drive:]pathname

In protected mode:

del [drive:]pathname

The difference is the space after the command and before the drive letter. It is required in protected mode.

Discussion: You can use wildcards with del. In protected mode you can specify multiple filenames.

Example:

In real mode:

```
delb: *.bak
```

deletes all files with the .bak extension on drive b:.

In protected mode:

```
del b:*.bak c:*.txt
```

wipes out all .bak files on b: and all .txt files on c:.

detach

Mode: Protected Mode Only

Purpose: Detaches a process from the screen and keyboard so it can run in background.

Syntax: detach command [arguments]

where :

command is any OS/2 program or command that needs no input from the keyboard arguments are any valid arguments for command.

Discussion: Detach is dangerous. Once a process is detached it must end on its own, the user cannot end it. If the parnet process ends a detached child process continues to run as an orphan. Programs not deigned to run in backgroun that are detached can corrupt disk files and trash screen information.

Example: TK

dir

Mode: both

Purpose: Lists files in a directory.

Syntax:

In real mode:

```
dir [drive][pathname][/p][/w]
```

In protected mode:

dir [drive][pathname][...][/p][/w]

where:

/p displays the directory one screen at a time

/w selects wide display, with only file names arranged in columns

Discussion: If you type more than one filename in protected mode. The dir display includes the volume label for all disks referenced, directory statements on all directories referenced, file information on all files and file totoal and disk space reports for all drives.

Example:

In real mode:

dir b: chapter.* /w

displays the name only in column format of all files called chapter, regardless of extension on drive b:

In protected mode:

dir b:chapter.* c:/book/chapter a:murder.axe /p

Results in a directory listing all chapter files on drive b:regardless of extension, all files in the /book/chapter subdirctory of drive c: and the file muder.axe on drive a:, all displayed one screen at a time.

diskcomp

Mode: Both

Purpose: Compares the contents of two floppy drives

Syntax: diskcomp [drive1] [drive2]

where drive1 is the source drive and drive2 the target drive

Discussion: Diskcomp only works with disks of the same size and format. You can use the same drive for both target and source disks and then swap them when promted during the compare.

diskcopy

Mode: Both

Purpose: Copies the contents of one floppy drive to another.

Syntax: diskcopy [drive1] [drive2]

where drive1 is the source drive and drive2 the target drive

Discussion: Diskcopy creates a mirrory physical image on one disk on another. In doing so it will format an unformatted target disk and destroy anything already stored on the target disk. Diskcopy only works with disks of the same type and ignored subst or join commands; it only works with physical drives. For copying files, use xcopy instead.

dpath

Mode: Protected Mode Only

Purpose: Lets you tell an application where it can search for data files.

Syntax: To specify directories to be searched:

dpath [drive][path[[;[drive:]path]...]

To clear all settings:

dpath ;

To display the list of data path directories:

dpath

Discussion: Similar to the real mode append command. dpath is assigned for a single application program's session.

Example:

dpath \book\chapters;b:\archive

sets the search for data files to the \book\chapters subdirectory of the current drive and the archives subdirectory on drive b:.

Exit

Mode: Both

Purpose: Exits the current command processor.

Syntax: exit

Discussion: When you use cmd or command to start a command processor, you use exit to return to the previous command processor. Exit also lets you leave an "application program to move to the OS/2 command processor.

fdisk

Mode: Both

Purpose: Configures a hard drive for OS/2.

Syntax: fdisk

Discussion: When you call fdisk, you get a series of menus to help with hard drive setup.

find

Mode: Both

Purpose: Searches for a string of text in a file or files.

Syntax: find [/v][/c]/n] "string" [drive:][pathname]...]

where:

"string"	is the st of characters you are looking for
/v	displays all lines not containing the string
/c	displays only the number of lines cotaining a match in each specified file
/n	precedes each displayed line with its relative line number in the file

Discussion: If /c is used with /v find displays the number of lines that do not contain the string. /c and /n should not be used together. Inside the string quoatation marks, upper case characters do not match lower case characters. Wildcards are not allowed with find.

Example: find /c "OS/2" c:\book\chapters\chapter1 will display lines where OS/2 occurs in the chapter1 file in the \book\chapters subdirectory on drive c:.

format

Mode: Both

Purpose: Formats a specified disk

Syntax: format [drive:] [/4] [/s] ;/t:tracks] [/n:sectors] [/v:label]

where:

/4	formats a 5.25 inch double-sided disk in a high capacity drive.

/s Works only with 1.2 megabyte of higher drives; copies operating system files listed in formats.tbl to the newly formatted disk.

/t:tracks tracks is the number of tracks on the disk.

/n:sectors sectors is the number of sectors on the disk.

/v:label label is the volume label of the disk.

Discussion: Shouldn't be used with assign,join and subst and doesn't work over a network.

Example:

format a: /s

formats the a: drive and copies operating system files onto it.

graftabl

Mode: Real Mode Only

Purpose: Enables an extended character set to be displayed in graphics mode.

Syntax: graftabl [xxxx] or graftabl ? or graftabl /sta

where:

xxx is a code page number

? displays graftabl options

/sta displays the active character set

Discussion: Works with the code page numbers noted earlier. You can only load graftabl once each time you start OS/2. If you plan to use it,consider putting it in your startup files. See the cchp command and Chapter 4.

Example: graftabl 437

loads the United States character set

helpmsg

Mode: Both

Purpose: Provides background information relating to OS/2 warning or error statements

Syntax: helpmsg messageid

where:

messageid in the unique identifier for the error or warning statement

Discussion: Helpmsg is an online assistance utility for OS/2 users. Each OS/2 message has a unique identifier consisting of DOS followed by four digits. YOu can learn more about that message via helpmsg.

Example: If the error message

DOS0100 File Not Found

is displayed, you can learn more about what it means by typing:

helpmsg dos0100

join

Mode: Real Mode Only

Purpose: Joins an existing disk drive to a specific path

Syntax: join [drive: drive:pathname] or join drive: /d

where:

/d disables a previous join

Discussion: Join lets you access a disk drive via a drive id and path, in place of the drive's physical name. So you can replace references to a: with calls to a specific path on c:. This is useful if your application wants to find information from a specific location you are already using for something else. You can join your program's location to a new path and actually keep material there. To resue the joined drive, you must unjoin from it. Remember that file commands, such as foramt anddiskcopy don't work with join. |Join only works with root level directories.

Example:

join c: a:\book

will place all references to a: to the path c:\book

Keyb

Mode: Both

Purpose: Specifies special keyboard.

Syntax: Keybyy

where:

yy is a two letter country code

Discussion: Valid country codes are:

US United States

UK United Kingdom

GR Germany

FR France

IT Italy

SP Spain

Example: keybsp

calls for the Spanish keyboard

label

Mode: both

Purpose: Creates or changes the volume label on a disk.

Syntax: label [drive:][label]

where:

label is a new volume label of up to 11 characters

Discussion: Lets you name your disks if you wish to. Doesn't work with join or subst and does not allow use of grouping symbols or wildcards.

Example: label c:bookdisk

The volume label of drive c: is now bookdisk

mkdir (md)

Mode: Both

Purpose: Creates a new subdirectory under the current directory.

Syntax: Real Mode:

mkdir [drive:]pathname

Protected Mode:

mkdir [drive:]pathname [[drive:]pathname]...]

Discussion: The difference between real and protected mode is that you can create more than one subdirectory at once in protected mode. Can be abbreviated md.

Example:

In protected mode:

```
mkdir c:\book\chapters a:\book\notes
```

creates the \book\chapters subdirectory on drive c: and the \book\notes subdirectory on drive a:.

mode

Mode: Both

Purpose: Establishes operatinal settings of communications and outer putput devices

Syntax: Parallel printer mode:

```
mode LPTn(:)(chars)(,(lines)(,p))
```

where:

n specifies the printer number: 1,2,or 3

chars specifies the number ofcharacters per line

lines sets vertical spacing, 6 or 8 lines per inch

p places mode resident in memory and attempts constant output to the printer if a time out error occurs

Asynchronous communications mode:

```
mode COMm(:) baud(,parity(,databits(,stopbits)(,p)))
```

where:

m Sets the asynchronous communications ports number, from 1-8

baud Sets the transmission speed: 110,150,300,600,1200,2400,4800,9600 or 19,200

parity Sets the parity check state (error detection): even, odd or none

databits Sets the number of data bits in each aynchronouse packet: 7 or 8.

stopbits Sets the number of stop bits per packet: 1 or 2.

p places mode resident in memory and attempts constance output to the communications port if a time out error occurs

Display mode:

mode display

where:

display sets a video mode recognized by OS/2: 40,80, BW40, BW80,CO40,CO80 or MONO

40 and 80 indicate number of characters per line

BW stands for graphics monitor with color disabled

CO stands for graphics monitor with color enabled

MONO stands for monochrome

Discussion: Apart from simply setting up parameters, mode can be used to redirect output through the devices it controls. If you have a serial printer, for instance, you want to redirect output headed toward the parallel printer port toward the serial port. You can use mode to accomplish that in this way:

mode com1:48,e,7,1,p this sets up the comm port to receive data

mode lpt1:=com1 this redirects output for the first printer port to the serial output

Examples:

Printer:

mode lpt2: 80,8 sets printer 2 for 80 cahracters a line, 8 lines per inch

Asynchronous Port:

mode com2: 96,o,8,1,p sets the second comm port to 9600 baud, odd parity, 8 databits, 1 stopbit, sending loop

Screen:

mode CO80 sets the screen to color display with 80 characters per line

more

Mode: Both

Purpose: Sends data to the display one screenful at a time.

Syntax: more < source

or

source | more

where:

source is a file or command

Discussion: Source is a filter command that accepts input from a pipe or redirected file and then sets it up to display one screenful at a time. More does this by setting up a temporary file on the logged disk; so if that disk it full or write protected, more will not work.

Example:

more < chapters.bk filters the contents of this file through more so it will display one screen at a time

type chapters.bk | more sends the output of the type command through more, so the file is typed one screen at a time

patch

Mode: Both

Purpose: Allows changes to program code.

Syntax: patch ((drive:)pathname)(/a)

where:

/a sets automatic mode, where patches take place without specific command

Discussion: With patch you can place bytes into writable files or at their end. patches are entered at hexadeicmal offsets the command prompts for once it is invoked. Patch should never be used unless you know what you are doing:

precisely where the patch should go and what its affect will be. Patch is similar to debug in the DOS environment.

Example: patch c:/progs/book.exe

[illus here of interactive prompt screens for patch]

path

Mode: Both

Purpose: Sets a search path for external commands

Syntax: path ((drive:)(path) (;)...)

or

path ;

Discussion: Just as in DOS, path establishes a set of directories to search for external commands.

Example:

path c:\book\chapters;\B:\progs\wp searches these two directories,

in addition to the current

directory for external commands

print

Mode: Both

Purpose: Out puts a file to a printer or other output device

Syntax: print (/d:device) (/t) (/c) (drive)(pathname) (...)

where:

/d:device	Specifies the device name, usually LPT1
/t	Deletes all files already waiting to be printed
/c	Cancel mode:removed the preceding filename and all names that follow from the print queue

Discussion: Print allows you to output a file for printing directly from the operating system. This is particularly useful when viewing the contents of .bat or .cmd files and other program listings.

Example: print /d:lpt1 /t c:\chapters\chapt1 prints to the first printer the contents of chapt1, removing all other files waiting in the print queue

prompt

Mode: Both

Purpose: Lets you customize the OS/2 command prompt

Syntax: prompt ((text)(characters)...)

where the following characters:	Produce the following results:
$$	The $ character
$t	The current time
$d	The current date
$p	The current working directory
$v	The OS/2 version number
$n	The default drive
$g	The > character
$l	The < character
$b	The | character
$_	A carriage return/linefeed
$e	ASCII code X'1B' (escape)
$q	The = character
$h	The backspace character
$c	The (character
$f	The) character
$a	The & character

Discussion: In OS/2 the real mode and protected mode prompts are different, to help keep users in mind of where they are. If the prompts are modified, they should still be kept distinct from one another.

Examples: A common prompt is:

prompt pg which displays the current directory, followed by >

a more complex prompt:

prompt Howdy! The current time is thhhhhh_$p $q

Produces the following:

Howdy! The current time is 4:37

C:\Chapters =

The $h, backspaces, erase seconds and hundreths of seconds from the time display. The $_, linefeed, puts the prompt on two lines.

If ANSI escape sequences are supported in your system, you can use them in prompts with the $e character. This prompt:

$e[7m$n:$ee[m

puts prompts in reverse video, then returns to standard video mode

recover

Mode: Both

Purpose:Recovers a file or adisk with bad sectors

Syntax: recover (drive:)(path)filename

or

recover drive:

Discussion: A helpful utility for avoiding unexpected data loss. It allows OS/2 to read a disk sector by sector, saving everthing that hasn't been corrupted and marking bad sectors so they won't be written to again. Recover does not work across networks or with subst or join commands.

Example:

recover b: recovers the entire drive

recover chapter.one recovers the contents of this file

rename (REN)

Mode: Both

Purpose: Changes a file name

Syntax: ren (drive)pathname filename

Discussion: You can use wildcards (.* and ?) with this command, but not drive designations. It cannot move files, just change their names.

Example:

ren chapter.one intro.bk Changes one file name

ren *.rv4 *.fin Changes all files with this extension

This will not work:

ren c:\chapter\chapt.one b:intro.bk

The renamed file must remain in the directory where it started.

replace

Mode: Both

Purpose: Updates files on a hard disk with newer versions.

Syntax: replace (drive:)pathname1 (drive:)(pathname2) (/a)(/p)(/r)(/s)(/w)

where:

pathname1	is the source path and filename
pathname2	is the target path and filename
/a	adds new files found in the source directory, as well as replacing old ones
/p	includes a prompt line asking if you want to replace a file
/r	replaces read-only as well as unprotected files
/s	searches all subdirectories of the target drive for files to replace; incompatible with /a
/w	Waits for a source disk to be insertes before beginning to search for source files.

Discussion: Replace searches the source drive, then replaces files on the traget drive that have the same name. The /a swtich also will copy over files on the source drive that do not exist on the target drive. Replace cannot be used with hidden files or system files.

Example:

replace a:rules.wrk c: /s Updates all files called rules.wrk on

	C:
replace b:*.txt c:\stories /a	Updates all files in this subdirectory contained on a, and adds all .txt files on a:, not found in the directory

restore

Mode: Both

Purpose: Restores files saved with the backup command.

Syntax: restore drive1: (drive2:)(pathname) (/s) (/p) (/b:date) (/a:date)

(/e:time) (/L:time) (/m) (/n)

where:

drive1	contains the backed-up files
drive2	is the target drive
pathname	identifies the file(s) you want to restore
/s	restores subdirectories
/p	prompt for permission to restore read only files or those that have changed since since the last backup
/b:date	restores only files last modified on or before the date
/a:date	restores only files last modified on or after the date
/e:time	restores only files last modified at or before the time
/L:time	restores only files last modified at or after the time
/m	restores only files modified since the last backup
/n	restores only files that no longer exist on the target disk

Discussion: The following files cannot be restored: msbio.com, msdos.com,cmd.exe and command.com. Sys can restore the first two files. Copy can restore the second two.

Example:

restore a: c:\chapters\chapt.one restores the files chapt.one from backup disk a: to the \chapters subdirectory on c:.

rmdir (rd)

Mode: Both

Purpose: Removes a directory.

Syntax: Real Mode:

rmdir (drive:)path

Protected Mode:

rmdir (drive:)path(...)

Discussion: Only empty directories can be deleted. The files they contain must be erase with the del command first. In protected mode, more than one directory can be removed with a single command. You cannot remove the directory you are currently working in.

Example:

rmdir \chapters

set

Mode: Both

Purpose: Equates one string with another for use in programs.

Syntax: set (string=(string))

Discussion: Set searches memory reserved for the current environment and reaplces the first string with the second whenver it is found. This is useful in startup.cmd or autoexec.bat files as well as in some programs you might write.

Example: set file=chapter.one replaces the string file with the name of the file chapter.one

sort

Mode: Both

Purpose: Accepts input, sorts it alphabetically or numerically and outputs it.

Syntax: (source) | sort (/r) (/+n)

or

sort (/r) (/+n) < source

where:

source is a filename or command

/r sorts in reverse order

/+n sorts according to characters in column n.

Discussion: Sort is a filter command and works well with redirection symbols. It is often used with the more command as an output and the dir command as an input.

Example:

sort < contents.txt sorts the table of contents

sort /r <contents.txt sorts the contents backwards

dir | sort /+14 pipes the directory into sort and sorts the 14th column; this is the file size, so the files in the directory will be sorted by file size

spool

Mode: Both

Purpose: Sets up a printer spooler to allow background processing while other OS/2 commands work in foreground

Syntax: spool(drive:)(pathname) (/d:device1) (/o:device2)

where:

[drive:)(pathname) specifies the print spool subdirectory, where the temporary spool files will be created. The default is \spool.

/d:device1 Names the port connected to the printer.

/o:device2 Names the output printer

Discussion: Spool sets up temporary files for your print jobs, queues them, stores them while you are doing other work and then sends them to the printer as soon as possible. To use spool you must specify the /t or /c switches with the print comand.

Example:

spool \output /d:lpt1 Takes the print jobs in the directory

output and print them in parallel

printer 1.

subst

Mode: Both

Purpose: Substitutes a drive letter for a path.

Syntax: subst (drive: drive:path)

or

subst drive:(path) /d

where:

/d deletes a virtual drive

Discussion: With this command you can use a path in commands as if it represented a real disk drive. The path becomes a virtual drive to OS/2.

These commands do not work with subst: chkdsk, diskcopy, fdisk, format, label, recover, sys.

Example: subst d: c:\book\chapters Creates virtual drive d: for this

directory. Type dir d: and you'll

get the /book/chapters directory.

sys

Mode: Both

Purpose: Transfers OS/2 system files from one drive to another.

Syntax: sys drive: (/s)

where:

/s causes all system files listed in the formats.tbl file on the

source disk to be copied.

Discussion: Sys is usually used to make a new disk bootable. It does not transfer command processors to the disk, just the files that make it`recognizable to OS/2.

Example: Sys c: /s copies the system files from your working drive to

c:.

time

Mode:Both

Purpose: Lets you enter or change the time.

Syntax: time (hours:minutes(:seconds (.hundreths)))

Discussion: Time with nothing following causes the current time to be displayed. OS/2 uses a military 24-hour time system. Valid times are:

0-23 hours

0-59 minutes

0-59 seconds

0-99 hundreths

The format of time display is based upon the country deignated in the country command. The United States is the default.

Example:

time 04:56 sets the time to 4.:56 AM

tree

Mode: Both

Purpose: Displays the paths in your directory structure and, optionally, the

files they contain

Syntax: tree(drive:) (/f)

where:

/f causes the files, as well as directory to be shown

Discussion: Tree lets you look over the directory structure of your working drive. It can help you find your way around the disk.

Example: tree displays the directory structure

tree c: /f | more shows the hard drive directories and files

ones screen at a time

type

Mode: Both

Purpose: Scrolls the contents of a text file on the display screen.

Syntax: Real Mode:

type (drive:)(path)filename

Protected Mode:

type (drive:)(path)filename (...)

Discussion: Type only works with text files. .exe or .com files or those from application program produce nonsense on screen. In protected mode you can type more than one file with a single command.

Example:

type chapter.one	scrolls the contents of the chapter
type chapter.one chapter.two	scrolls one after the other
type chapter.one | more	displays the contents one screen at a time

ver

Mode: Both

Purpose: Displays the current version of OS/2.

Syntax: ver

Discussion: Lets you know the vintage of command processor you are using.

Example: Type :

ver

And the response is:

OS/2 Version 5.9 or whatever your current version is

verify

Mode: Both

Purpose: Sets the verify status on and off when writing to a disk.

Syntax: verify (on)

or

verify (off)

Discussion: When writing files to a disk you can ask OS/2 to verify that they are copied correctly. Verify types with no argument displays the current setting.

Example: verify on will ensure that files copied will be verified.

vol

Mode: Both

Purpose: Displays disk volume label or ID, if there is one.

Syntax: Real Mode:

vol (drive:)

Protected Mode:

vol (drive:)(...)

Discussion: Vol must reside on the drive you are working from In protected mode you can ask from more than one vol at a time.

Example: vol c:

xcopy

Mode: Both

Purpose: Copies files, directories and subdirectories between drives.

Syntax: xcopy (drive:)(pathname1) ((drive:)(pathname2)) (/s) (/e) (/p) (/v)

(/a) (/m) (/d:mm-dd-yy)

where:

/s copies both directories and subdirectories that have files in them without it xcopy only works with the current directory.

/e copies subdirectories even if they are empy

/p prompts you yes/no on copying each file

/v sets verify on for xcopy

/a Copies source files with the archive bit set. See attrib.

/m Copies the archive files ,but then turns the archive bit off.

/d:mm-dd-yy Copies source files modified after the specified date.

Discussion: Xcopy is the command of choice when work with drives of different types.

Example:

xcopy a: c: /s /e Copies all files, directories and subdirectories--empty or not--from a: to c:.

BBATCH COMMANDS

call

Mode: Both

Purpose: Calls one batch file from another.

Syntax: call batchfile (argument)

where: batchfile is the batchfile you want to call argument is the command in this batchfile that will be run following batchfile

Discussion: The argument in a call command determines where the orignal batch files resumes processing after the call is complete. If no value is given, the batch file picks up with the next command listed after the call.

Example: call setup will call the setup.cmd or setup.bat file

echo

Mode:Both

Purpose: Turns screen echoing on or off.

Syntax: echo (on)

or

echo (off)

or

echo (message)

where:

message is a line of text

Discussion: Batch commands are echoed to the screen unless echo is turned off. Whe echo is off you can send specific information to the screen with the message option. Echo with no setting displays the current setting.

Example: echo off turns echo off

echo I know

echo this is

echo taking time,

echo Be patient! these send messages to the screen

endlocal

Mode: Both

Purpose: Returns to the settings in place before the setlocal command was used.

Syntax: endlocal

Discussion: See setlocal. This command cancels its effects.

Example:

extproc

Mode: Both

Purpose: You can use this command on the first line of a batch file to name an alternate command processor for the batch file.

Syntax: extproc (batch processor name) (argument)

where:

argument is any valid option for batch processor name

Discussion: If you are using something other than command.com or cmd.exe to handle you commands, name it with this command.

Example: If we had cooked up a control program called strom.exe, we would type at the head of our batch file:

extproc strom.exe

for

Mode: Both

Purpose: Performs a command for a set of items.

Syntax: for %%c in (item(...)) do command

where:

c is any character except 0,1,2,3,....,9

item is a filename or any portion of a command line

command is the desired command

Discussion: Allows you to set up conditional, ineractive statements in batch files. For looks at the variable %%c for every item and applies the comand to each.

Example:

for %%a (*.txt) do del %%a

This puts all files with the .txt extension in the %%a variable and thendeletes them

for %%b (*.fin) do print %%b

This puts all files with the extension .fin in the %%b variable and then prints them

goto

Mode: Both

Purpose: Processes commands starting with the line after the specified label.

Syntax: goto label

where: label specifies another location in the batch file

Discussion: Goto sends the batch file to a label. The label can be any combination of characters and spaces, except semicolons , equal signs or other spearators. Each label must be marked by a colon (:). The colon causes OS/2 to ingore what follows, since is it a label, not a command.

Example:

:begin

echo off

```
format b: /s

if errorlevel 0 goto end

echo Format unsuccessful

:end

echo Format successful
```

if

Mode: Both

Purpose: Performs a command based upon the result of a condition.

Syntax: if (not) errorlevel number command

or

if (not) string1 = = string2 command

or

if (not) exist filename command

Discussion: In the first instance, the condition is true if the exit code for the program processed by cmd.ese was equal to or greater than the variable number. In the second instance, the condition is true if string1 and string2 are identical. These strings cannot use separators or spaces. In the third case the condition is true is the filename exists. The use of the not option causes if to go into action if the condition is false.

Example:

```
if not exist chapter.one echo Sorry, can't help you!
```

pause

Mode: Both

Purpose: Suspends running of the batch file.

Syntax: pause (comment)

where:

comment is any special message

Discussion: When OS/2 sees paus, it suspendsthe batch file, then prints the comment followed by the clause "Strike a key when ready," if echo is set on.

Example:

```
pause Make sure there is a blank disk in drive a:

format a: /s
```

rem

Mode: Both

Purpose: Lets you insert reminder comments in a batch file.

Syntax: rem (comment)

where:

comment is a line of text to document the batch file's function

Discussion: REMs can be used to notate batch files, or(with no text following) to add space to the batch text files for readability. The only separators allowed in the comments are space, tabs or commas.

Example:

```
rem This batch file changes directory and

rem loads the xywrite word processor from disk.

echo off

c:

cd \xywrite

editor
```

setlocal

Mode: Both

Purpose: Lets you set local drive, directory and environment variables for a batch file.

Syntax: setlocal

Discussion: Setlocal lets you establish temporary operating conditions for a batch file. They are terminated with endlocal, or when the batch file issuing sendlocal is terminated. Setlocal commands cannot be nested.

Example: setlocal If you had created a virtual drive d: for

path d: homemade programs and for a time want your

strom.exe batch file path to go there; use setlocal

shift

Mode: Both

Purpose: Lets you change the position of replaceable parameters in batch files.

Syntax: shift

Discussion: Usually, command files, such as batch files, are limited to handling 10 parameters, noted as %0-%9. Parameter %0 is always replaced with the drive location and name of your batch file. That leaves nine replaceable parameters. To get more than this number you can shift new parameters into the %9 position one at a time. There is no backwards shifting and each time you shift you lose the parameter holding the %0 position.

Example: