A New Generation of Operating Systems for Personal Computers: OS/2 2.1

Stefan Zollner
Ames Laboratory and Department of Physics and Astronomy, Iowa State University, Ames, IA 50011
(December 13, 1993)

A number of sophisticated operating systems (OS's) for personal computers (PC's) have appeared over the past year, such as IBM's OS/2 2.1, Microsoft's Windows NT, and others. I will discuss the common design of these new OS's and explain how they will change working with a PC in comparison with DOS. In particular, I will investigate how these OS's can help with the various tasks encountered in a condensed-matter physics laboratory. As an example, a detailed evaluation of OS/2 2.1 will be presented, including the potential and current limitations of this OS as well as hardware considerations.

XXXX

I. A CRITICAL REVIEW OF DOS AND ITS LIMITATIONS

When I was a freshman in college back in 1981, the university microcomputers had a Z-80 central processing unit (CPU) running at a clock speed of about 5 MHz, an external floating-point unit, 64 kB of RAM memory, and two 8-inch floppy disk drives as mass-storage devices. The IBM-compatible personal computer[1] (PC) in my laboratory today has an Intel 80486DX2 CPU running at 66 MHz, a built-in numerical coprocessor and cache, 20 MB of RAM, and an IDE hard drive with 340 MB of storage space. While the computer hardware has changed dramatically over the past ten years (without the corresponding increase in price), the operating system (OS) most often used to control the hardware has more or less remained the same. This is about to change.

DOS[2], the most popular OS for PC's, much like its predecessor CP/M, is a single-user, single-tasking OS. It was designed for a CPU we now consider archaic and does not support most of the advanced features of the 80486. DOS provides the user with a simple command shell, it can load executable programs, and allows them to access the peripheral hardware (display, keyboard, disks, etc.) through interrupts to the BIOS (basic input-output system). As a UNIX programmer once told me, DOS is not really an OS, but merely a task loader.

Since DOS transfers control of the computer to the executable, the program can do whatever it desires with the hardware, for example send characters to the screen or format the hard drive. DOS does not supervise the program and cannot intervene in case of a fatal programming error. In many cases, when a program crashes, the three-fingered salute (Control-Alt-Del) is the only way for the computer user to stop the program and reboot. It is obvious that such an OS makes it easy for computer vandals to create viruses, sections of code that will attach themselves to the program or OS, wait up to a specified date, and then carry out whatever task they have been designed for, for example destroy the data on the hard disk.

DOS has other limitations, see Fig. 1: It was originally designed for a computer with a 16-bit CPU and 1 MB of RAM. The upper 360 kB of high memory were set aside for the OS and the hardware (for example as video memory), and the lower 640 kB (conventional memory) were made available to the program, except for a small area at the bottom used for DOS. While this was a great advantage compared to the 64 kB that CP/M could access, most modern computers have much more than 1 MB of RAM. Most computer hardware, for example the mouse, ethernet network adapter, disk drive, SCSI (small computer interface) adapter, or IEEE-488 card require a device driver to be loaded in the CONFIG.SYS file of DOS and reduce the amount of available memory. (If you are lucky, all of these device drivers can coexist without affecting each other, but I have seen cases where, for example, a video driver will not run when the network adapter is activated.) Once you add some memory-resident programs (TSRs), such as an antivirus program, a RAM drive, the DOSSHELL, and the MIRROR, SETVER, and DOSKEY utilities, many large programs can no longer run.

While the most recent releases of DOS (since DOS 5.0) have reduced some of these drawbacks by allowing to load some device drivers in the upper memory area, I was still not able to run my xray simulation program when the TCP/IP software (for local-area network connection) was active. In practice, you need a number of different versions of CONFIG.SYS for different tasks. When switching between different activities (for example from manipulating and plotting a large spreadsheet with experimental data to connecting to the university mainframe computer to do a theoretical simulation of the experiment) you may have to edit CONFIG.SYS and reboot your computer. This is clearly not acceptable.

In order to access more than 1 MB of RAM, a number of schemes (expanded and extended memory) were designed. Memory managers such as EMM386, either part of the OS or third-vendor software, promise to squeeze as many bytes as possible out of your upper memory area and to provide access to the extended memory. (Expanded memory is obsolete with modern motherboards.) In practice, however, it is not that simple: In order for a program to access extended memory, it has to follow a certain standard (for example DPMI, DOS protected memory interface). These standards are somewhat ill-defined and vendor-dependent and change constantly to keep up with the rapidly changing hardware, and a program written by a graduate student five years ago may leave you stranded with the message "Out of Memory" no matter how many SIMMs (memory chips) you plug into your computer. If you do not wish to rewrite the program to follow the new standard (just to be obsolete again in a few years), the only remedy will be to strip down your CONFIG.SYS file to the bare minimum to free up a sufficient amount of conventional memory.

Let us summarize: The memory available to most older and home-made programs is the conventional memory of 640 kB minus the memory consumed by device drivers that could not be loaded into the high-memory area between 640 kB and 1 MB. Only modern commercial programs can access the extended memory above 1 MB. Why does it have to be this way ? The reason for this architecture is that DOS was originally designed for the 16-bit processor 8086 which could only access memory in segments of 64 kB in size, up to a maximum of 1 MB using 20 address lines.

It is also worth noting that DOS is a 16-bit OS. The code generated by most compilers as well as system function calls move data 2 bytes at a time, even when run on a PC with a 32-bit CPU and a 32-bit bus. The corresponding loss in performance is almost 50%, in particular for certain applications, for example numerical calculations with 32-bit real numbers. There are some 32-bit add-ons to DOS that try to fix this shortcoming, in particular the 32-bit GNU DOS extender EMX (by Eberhard Mattes) and the Win32 subsystem by Microsoft, enabling Windows 3.1 to run some 32-bit applications.

II. TAKING ADVANTAGE OF THE ADVANCED FEATURES OF THE 80386 PROCESSOR

Let us now compare the DOS memory architecture described above with the flat-memory model of modern operating systems using advanced features of the 80386 CPU[3]. The 80386 is a 32-bit CPU with (at least internal) address and data buses of 32 bits. That means that (at least theoretically) the CPU can access a maximum of 4096 MB.[4] How many of these 32 address lines are actually connected to the memory depends, of course, on the architectures of the motherboard and the bus, but that is of no concern to the OS. Most PCs on the market today allow at least 128 MB of RAM (8 SIMMS with 16 MB each) to be installed directly on the motherboard. The flat-memory model used in OS/2 2.1 gives up the segment:offset concept used with the older 16-bit members of of the 80x86 CPU family. Instead, in order to access a particular structure in memory, its 32-bit starting address is indicated on the bus. Therefore, all 4096 MB of physical memory are equivalent. Large arrays of more than 64 kB in size or jumps to locations in remote areas of memory create no problems for the OS. As a consequence, device drivers, the code of the OS itself, dynamical link libraries, and executable programs can be loaded anywhere in memory. Clearly, any OS based on the 32-bit flat memory model has a great advantage over DOS. The first major[5] 32-bit OS was OS/2 2.0, released in April of 1992. At the time of this writing (August 1993), Microsoft just released Windows NT, also a 32-bit OS, whereas IBM completed OS/2 2.1, where most major problems of OS/2 2.0 have been fixed and new features were added (32-bit graphics engine, Windows 3.1 and multimedia support, and support for non-IBM hardware). The OS codenamed Chicago on the other hand, currently under development by Microsoft as a replacement for Windows 3.1 on low-end client workstations, is a 32-bit OS, but still includes the 640 kB barrier, see Table I. (There are rumors that DOS 7.0 will also have a 32-bit flat memory model, which will bring some compatibilty problems.)

The price we have to pay for the 32-bit flat memory model is an increased size of the programs, since four bytes are now necessary to indicate an address in memory instead of two. This is not a major problem since memory prices have fallen dramatically. You will now understand, however, why such enormous resources are required to run these modern OS's (a huge hard disk with at least 200 MB and at least 8–16 MB of RAM for OS/2 2.1, even more for Windows NT).

There is another difference in memory usage between DOS and OS/2 2.1. Under DOS, memory is real (or physical) memory. If a program needs, say, 5 MB of memory, there is no way to run the program on a machine with only 4 MB of RAM without rewriting the code. Under OS/2 2.x, this is not true, since physical memory is treated as a shared resourced. Any process (program) running under OS/2 2.1 has a virtual process address space of 512 MB. The virtual space above 512 MB is reserved for the OS. When virtual memory is requested by a process, the OS allocates a segment of virtual memory in this virtual process address space. This segment is only loaded into RAM when needed. If there is no free (physical) RAM available, the OS selectes some virtual memory (belonging to the same or a different process) that is not in use and saves it on disk in a file called SWAP-

PER.DAT. This process is called swapping, and is similar to virtual memory architectures on mainframe OS's like VM or VMS. This memory management mechanism is very useful for multitasking or loading large programs, but will eventually lead to a downgrade in performance, if too little RAM is available. Although the minimum amount of RAM required for OS/2 2.1 is 4 MB, performance will improve dramatically with a total of 8 MB. For some applications, such as the multimedia presentation manager or networking programs, 12 to 16 MB (or even more) are recommended for optimum performance.

This memory swapping procedure is completely transparent to the user and greatly simplifies numerical calculations on the PC requiring large amounts of memory. Diagonalization of a 1000 by 1000 complex matrix requires a workspace of roughly 16 MB. It is possible to perform this diagonalization on a PC with 16 MB RAM, since OS/2 2.1 will handle the swapping, but it will take much longer than on a PC with 32 MB, where swapping is reduced.

Before we discuss how Windows NT and OS/2 2.1 use other advanced features of the 80386 CPU for multitasking and memory protection, let us first look at DOS-based multitasking: The more recent versions of DOS already contain some mechanism for multitasking by using the DOSSHELL or Microsoft Windows[6]. The user can activate several application programs (for example a spreadsheet and a scientific graphics package) at the same time and switch between them. It is also possible to have an application (for example an extensive numerical calculation) run in the background and gain access to the CPU whenever the CPU is idle. The problems with DOS-based multitasking are severe, however: Since DOS was not designed as a multitasking OS from the beginning, problems may arise when a program erroneously (for example when a pointer or array subscript gets out of control) accesses a memory area belonging to a different program or, even worse, the OS itself. The users of Windows 3.0 often experienced the infamous user application error (UAE). The only remedy was to exit Windows (unless you were hung completely) and reboot the computer in order to restore system integrity. Microsoft quickly responded to these flaws by selling their customers Windows 3.1, a new version in which some of the problems were fixed. The remaining errors were renamed to general protection faults (GPF), therefore the UAE messages have disappeared. Clearly, multitasking under DOS is not the perfect solution.

The new OS's OS/2 2.1 and Windows NT use a different strategy using advanced features of the CPU. Most parts of the OS kernel (and some low-level device drivers) run in a CPU mode called ring 0 or supervisor mode, whereas the application programs run in user mode or ring 3, see Fig. 2. The CPU has been designed in such a way that only code in ring 0 (that is only the OS) can transfer control from one process to another. Each ring has its own stack, so the application cannot corrupt the OS stack. A number of privileged CPU commands (for example the dangerous INP and OUTP commands which directly talk to the hardware) cannot be executed[7] under ring 3. If a program wants to access the hardware, it usually has to issue a call to the OS or a device driver. Clearly, this design makes it much harder for a computer virus to carry out its tasks. At this time, there are a number of virus protection programs for OS/2 2.1, but no known OS/2 viruses.

The 80386 CPU also provides a secure memory protection mechanism. An application program consists of several memory segments for code, data, stack, etc. Some of these segments (such as data and stack) are writeable and readable, whereas others (constants and code) are only readable. When the OS (in ring 0) loads the program, the CPU is told specifically (using a segment table) which segments in memory the program is allowed to read or write. Whenever the application gets out of control and attempts to access a memory segment that is off limits, the CPU interrupts the process and returns control back to the OS indicating a memory protection fault has occured.[8] The OS at this point informs the user that the application had to be terminated. The (optional) register information is useful for debugging purposes. In this process, system integrity was never in danger, and other applications were not affected. There is no need to shut down the computer and reboot, like you would have to under DOS or Windows. In particular, Windows 3.1 applications running in OS/2 2.1 can be started in separate process spaces protected from each other. This protection is not available in Windows 3.1 or Chicago, see Table I.

Apparently, this design is perfect for multitasking. In practice, however, it looks somewhat different: No OS for the PC (with the possible exception of DOS 3.3) has ever been free of errors. The early releases of OS/2 2.0 and Windows NT, with a few million lines of new code, are much more complicated than DOS and contain a number of well-known bugs. Using OS/2 2.0, I have encountered a number of cases where the OS hung and had to be rebooted. In the worst case, the OS files on the hard disk became corrupted so badly I had to reformat the partition and re-install the OS, quite a time-consuming process. Using true-blue IBM equipment (without any network cards, FAX boards, modems, or SCSI adapters) OS/2 2.1 runs quite stable, but as soon as you add third-party interface boards you have to load non-IBM device drivers, often available only in a beta-test version for your hardware, that jeopardize system integrity. In particular, early versions of IBM's TCP/IP software for ethernet connection had severe problems, but most of these have been fixed using IBM's mechanism of issuing corrective service disks (CSD's) free of charge to their customers. One should expect that at least the programs that come with the OS (such as the enhanced editor EPM) should have been well tested, but I have encountered a number of GPFs.

III. OS/2 COMMAND SHELLS

When discussing an OS, we not only have to take into account its internal design, but also the methods of interaction it allows the user, that is the command shells and graphical user interfaces (GUI's) it provides. The good news about OS/2 2.1 is that it does not force the user to change his/her way of interacting with the OS. (This would not be true when migrating from DOS to UNIX, for example). OS/2 2.1 has a number of different command shells, see Fig. 2: There is a virtual DOS command processor (VDM), an OS/2 command processor (both either full-screen or windowed), and also the possibility of booting a specific DOS version from a floppy disk image the user can create on the hard disk. Then there is WIN/OS2, OS/2's version of Windows based on the Microsoft Windows kernel, and finally presentation manager (PM), a sophisticated GUI much more powerful than Microsoft's program manager for Windows, including the object-oriented workplace shell (WPS). Other user interfaces are available as add-on products from third-party vendors.

When you first start OS/2 2.1, I suggest you use the mouse to double-click on the "DOS Full Screen" icon in the "Command Prompts" folder. This will clear the screen and leave you with the usual DOS prompt on a black background, just as if you had booted DOS. You can customize your DOS session by editing the AUTOEXEC.BAT or CONFIG.SYS files. Different DOS VDM's can be set up to run specific batch files at startup. A DOS virtual device driver (16-bit) included in OS/2's CONFIG.SYS is loaded whenever a DOS box is started, not when booting. (Under OS/2, a DOS program has access to the OS/2 hardware device driver through a virtual DOS device driver.) Then AUTOEXEC.BAT is executed.[9] Another way to load a DOS device driver (such as a VESA display driver) before starting DOS is to open the settings notebook of the "DOS Full Screen" object (first click the right mouse button when the pointer is over the icon to bring up the pop-up menu, then the left button on the arrow to the right of the "Open" menu option, then on settings), then to click on the "Session" tab, then on the "DOS settings" pushbutton. In the window on the left, click on "DOS_DEVICE". Now you can enter whatever device drivers you want to load for your DOS session in the "Value" entry field on the right. There are many other configuration parameters, for example you can enable the upper memory block and specify the number of file handles and the amount of extended memory a program can access. Because of OS/2's virtual device driver architecture, an OS/2 VDM may make even more conventional memory available to a program than regular DOS. Most DOS programs will run with the DOS command processor of OS/2 2.1, including some laboratory programs referencing memory mapped or I/O mapped I/O devices using the INP/OUTP statements, for example the motion architect package for stepping motor control from the Compumotor division of Parker Hannifin. Text-based DOS programs can be run windowed or in full-screen mode (Alt-Home switches between these modes), but most graphics-based DOS programs (such as a TEX previewer) only work in full-screen mode. Some DOS programs (such as games) require some fine-tuning of the DOS parameters in order to run satisfactorily under OS/2. Information about these settings are sometimes available from the manufacturer (usually not), or from the common sources of information on the internet, such as the USENET newsgroups in the `comp.os.os2` hierarchy, the IBM Almaden gopher server `envy.almaden.ibm.com`, and the mailing list `OS2-L@hearn.nic.surfnet.nl`.

If a VDM cannot handle a program, the last remedy is to run the program under a specific DOS image, either by booting from floppy or from a disk image. This still allows some access to OS/2 (for example to switch back and forth between PM and the specific DOS processor), but gives the DOS program as much access to the processor as possible. Some poorly designed DOS programs will not run well under OS/2. The Wordperfect text editor (Version 6.0 for DOS) falls under this category. This program generates more interrupts to the keyboard than necessary for even the fastest typist in the world. The large number of these interrupts (from the DOS virtual device driver to the OS/2 device driver to the hardware. see Fig. 2) generates too much overhead and brings program execution to a crawl. Other problems are sometimes encountered during printing.

Unfortunately, there are some DOS programs which will not run under OS/2 2.1 at all, in particular some games (such as a flight simulator or Wayne Gretzky's hockey game) and other programs that directly access the hardware. Since an OS/2 VDM emulates an 8086 processor (using the virtual 8086 mode of an 80386 processor), programs requiring a more advanced processor (for example Microsoft Windows 3.1 or the GNU 32-bit DOS extender named GO32) cannot be run there. Some physics laboratory programs designed to control experiments with assembly language subroutines or other special features, for example special device drivers for interface boards (in particular the LabWindows program from National Instruments), may also fall into this category. There are, however, other mechanisms to make laboratory control possible. We are currently working on a program to control an ultrafast laser spectroscopy experiment. Results of this work will be discussed elsewhere.

Let us now discuss the OS/2 command shell: Its command set is a superset of the DOS command set, therefore the migration should be easy. There are some enhancements in this shell, for example the MOVE command and some commands specific to the multitasking support of OS/2. I should also mention the support for extended attributes (EA's), and also for long file names if the HPFS (high-performance file system) has been installed on the hard disk. For floppy drives, OS/2 uses the same FAT (file allocation table) file system as DOS. When an executible compiled for DOS is invoked, the OS/2 shell automatically starts a DOS subprocess to run the program. This is somewhat inconvenient, since the screen is cleared before and after opening DOS box, erasing all output the program writes to the screen. When developing programs for both DOS and OS/2, a compiler capable of producing family mode output (for DOS and OS/2), such as the Microsoft or WATCOM compilers, should be used. One of the most powerful features of the OS/2 shell is the REXX interpreter for batch (CMD) files. The REXX language has a C-like syntax with many OS-specific calls and is similar to the REXX language on IBM's mainframes with the VM operating system. There is even a third-party editor capable of understanding XEDIT macros similar to the VM editor XEDIT.

Normally, the OS/2 shell is started from the graphical user interface by double-clicking, as shown in Fig. 2 and discussed in the next section. It should be noted, however, that the OS/2 shell can also be booted directly (without a graphical shell) from the installation disks or from an OS/2 boot disk. This makes more physical memory available to a program and speeds up execution because of reduced overhead, but still takes advantage of the fast HPFS file system, which could be advantageous for numerical calculations requiring a large amount of memory and fast disk I/O. Multitasking is still available, therefore the user can monitor the progress of a calculation running as a detached process in the background. This mode is also useful for maintenance of the OS and therefore often referred to as an OS/2 maintenance boot.

IV. GRAPHICAL USER INTERFACES

The most drastic change in the use of PC's over the past years has come with the arrival of graphical user interfaces (GUI's), first introduced to the PC by Microsoft Windows[6]. With presentation manager (PM) and the workplace shell (WPS), OS/2's advanced GUI, a PC looks almost like a Macintosh (see Fig. 3), but has most of the powerful features of a UNIX workstation: The mouse is used for certain tasks, instead of issuing commands from the DOS or OS/2 prompt. For example, a program can be started by double-clicking on an icon on the desktop. You can associate all files with a certain extension (say .TEX) with a program (your favorite text editor). There is also the drag-and drop technique: You click on a data or text file with the right mouse button, then hold the mouse button while moving the mouse pointer. You will see how an outline of the selected object moves over the screen along with the mouse pointer. This is called dragging. By releasing the mouse button, you drop the object. If you drop an object onto the desktop or a directory folder, the file will be moved to a new location. Dropping the file onto the shredder or trashcan icon will delete it. You can also drop a data file onto a spreadsheet program, which has the effect of starting the spreadsheet and loading the data file. The drives folder, OS/2's version of Windows File Manager, is also based on the PM philosophy, but its feel, look, and functionality disappointed me. The window for a sample PM application along with its controls is shown in Fig. 4.

The second GUI included in OS/2 2.1 is WIN-OS/2 enabling the user to run most Windows applications under OS/2. In case you were experiencing a number of UAE's or GPF's under Microsoft Windows, you may be wondering about the performance of WIN-OS/2. The answer is that OS/2 2.0 (released in April 1992) only ran programs written for Windows 3.0. There were a number of problems in this release, in particular with printing, and I had programs crashing with GPF's, but OS/2 2.0 was usually able to recover from these errors by shutting down WIN-OS/2. There was usually no reason to reboot at this time. Some fixes are available from the manufacturers of Windows programs, and other programs will run better by fine-tuning the WIN-OS/2 settings.

The new release OS/2 2.1 uses the actual Windows 3.1 code licensed from Microsoft. It therefore runs Windows programs much better and also provides support for Windows 3.1 programs. Multimedia support for CD-ROM players and sound cards is also included in OS/2 2.1. However, most vendors of Windows software do not focus on OS/2 2.1 as a potential competitor to Windows 3.1: Most of them have not tested their software with OS/2 2.1 and will not make statements if their software runs on OS/2 2.1. In most cases, there are no problems, but there are peculiar exceptions: Most Claris software, for example, will not print from OS/2 2.1 because of an uninitialized variable in the code. This only shows up in OS/2 2.1, not in Windows 3.1. In this case, the vendor will not fix the bug, because everything is fine in Windows 3.1, but IBM will not take action either.

How does WIN-OS/2 compare[10] to Windows NT ? I have not tested NT myself, but sources on the internet indicate that NT's support for Windows programs is disappointing. Many programs will not run well or too slowly, and DOS 16-bit device drivers cannot be used at all. Microsoft responds to this criticism by indicating that NT is not intended to be downward-compatible to Windows 3.1, therefore there is no problem. IBM is marketing OS/2 with the slogan "A better DOS than DOS, a better Windows than Windows". While the statement about DOS is not really true (but may come close), experts claim that OS/2 2.1 provides excellent for Windows programs. Its Windows support may better than Windows 3.1, and certainly is better than that of the original release of Windows NT. It seems that NT will find most of its market as a high-end server, whereas OS/2 seems better suited for a low-end (486DX50-16 MB RAM) client station. A comparison between different operating systems, including "Chicago" (Microsoft's codename for Windows 4.0) is given in Table I.

V. OS/2 APPLICATION PROGRAMS

One of the biggest criticisms of OS/2 2.1 and Windows NT is the lack of native applications. While it is true that few OS/2- and NT-native applications[11] have arrived in the stores by now, this may not be a problem for OS/2 2.1, since most DOS and Windows applications will run under OS/2 2.1. Many public-domain UNIX utilities, such as EMACS and the GNU programming environment, have been ported to OS/2.

A number of applets (small applications) are included with OS/2 2.1. These applets are not formally supported by IBM as part of the OS, but demonstrate the capabilities of the OS and, in some cases, are demo versions of third-vendor packages. Some of these applets are so excellent that I use them on an almost daily basis: There is a personal planner package, complete with calendar, alarm, and to-do-list, the enhanced editor EPM, which can be configured and recompiled by the user, and PMCHART, a mini-graphics program, which I use to prepare drawings for the machine shop. Some figures in this paper were also drawn with PMCHART.

Many freeware and shareware programs for OS/2 or WIN-OS/2 can be retrieved using anonymous FTP from software depositories such as `software.watson.ibm.com` (corrective service disks for OS/2, new device drivers, IBM-employee written software, for example T, a quick and tiny editor for OS/2 maintenance boot disks), `ftp.cdrom.com` (largest general OS/2 depository, mirrors the IBM site, also has many more or less useful drivers, patches, games, graphics utilities, documentation, hints, GNU and other UNIX software for OS/2, system utilities, and much more), `extreme.cica.indiana.edu` (large Windows archive), `ftp.uni-stuttgart.de` (EMTEX, a public-domain version of TEX and LATEX for DOS and OS/2, including previewers and drivers for most printers), and `clio.rz.uni-duesseldorf.de` (HP2XX, an HPGL graphics viewer). The contents of the `ftp.cdrom.com` site are also available on CD-ROM from Walnut Creek; call 1-800-786-9907 or send e-mail to `info@cdrom.com`.

The most important single reason for me to switch to OS/2 were its networking capabilities. Most condensed-matter physics groups have a number of PC's. For example, one machine may be dedicated to taking data, another one as a PC for data evaluation, preparation of publications, and as a smart terminal to the university mainframe. A third machine may be located in the major professor's office. It is undesirable to duplicate data or programs on several machines, not only to conserve space, but also to avoid a file to develop into different versions on each machine. On the other hand, every user should be able to access all files he is supposed to have access to from any machine. A situation where the user has to wait for an experiment to complete before he or she can download a file to a floppy disk for transport to a different machine is unacceptable. Data security is also an important issue: A student should not have access to letters of recommendation or other sensitive data on his professor's PC, but access to experimental data or corrected version of papers should be allowed. Last not least, networking simplifies backing up data to tape.

The networking protocol most commonly used in a multi-vendor university environment is TCP/IP, originally developed for the UNIX world. The hardware adapter to run this protocol is usually an Ethernet or Token-Ring adapter. These adapters are available for PC's from a variety of vendors. I have attempted to use TCP/IP programs under DOS, but had to realize that the device drivers were too large to be loaded into the high-memory area of DOS. The drivers left me with too little memory to run my applications programs, in particular after loading Windows 3.1, see Fig. 1.

Under OS/2 2.1, I have all the features of TCP/IP, including TELNET, FTP, GOPHER, X-Windows, and NFS client and server, E-MAIL, and access to USENET newsgroups. Because of the flat-memory model, there are no memory conflicts. I do, however, recommend 16 MB of RAM to avoid an increase of swapping and the corresponding decrease in performance. There are also other networking protocols (some of them designed specifically for the PC). While these are usually not flexible enough to allow access to all possible mainframe OS's (UNIX, VAX VMS, IBM VM or MVS, etc.), they may be suitable in your environment for compatibility reasons. With TCP/IP and its integrated networking support, OS/2 becomes an excellent choice as the OS for a client-server environment on both the client and the server machines. In essence, there is no reason to distinguish between client and server.

Here is a list of my favorite OS/2 applications. Not all of these are OS/2-native applications, but they run well on OS/2 2.1:

- IBM TCP/IP 2.0 for OS/2 (including TCP/IP base, LAMAIL, NFS, and X-Windows server).
- IBM programmer's work bench, including C and C++ compilers, PM-oriented debugger, Workframe/2, an integrated development environment (IDE), linker, tool kit, and on-line programming library.
- Borland C and C++ compilers, linker, IDE, and resource development kit. The performance of the current release is disappointing, but the design looks promising.
- National Instruments IEEE bus drivers (NI-488.2M) for OS/2 2.1.
- HFS Window Works 2.0. Spreadsheet, charting, and address book.
- Quicken for Windows (Intuit). Financial planning program.

VI. HARDWARE REQUIREMENTS AND SETTING UP OS/2 2.1

IBM specifies that OS/2 2.1 will run on any modern PC with a 80386SX CPU or better and at least 4 MB of RAM. However, given the current market situation, I suggest at least an 80486DX33 processor (which you may want to upgrade to 99 MHz next year) and a minimum[12] of 8 MB for a stand-alone or portable PC, 16 MB for a network machine running TCP/IP. (Other features, such as seamless WIN-OS/2, high-resolution SVGA, the HPFS file system, and multimedia also require more RAM, but 16 MB should be sufficient.) This will drastically improve performance (by reducing the amount of swapping) and also avoid problems sometimes encountered on slow machines with little memory. An ISA-bus machine will be less expensive than EISA, but I suggest at least two VESA local bus slots for higher transfer rates to the disk and video adapter cards. An IDE hard drive (at least 200 MB, but a 340 MB drive is only marginally more expensive) will help you stay clear of SCSI incompatibility problems often encountered on the PC. Your video card should be an inexpensive SVGA card with 1 MB of video RAM on the board. (Users of advanced video boards with graphics coprocessors have complained that there are no OS/2 drivers from the manufacturer. They therefore had to run their expensive boards in a crippled SVGA or 8514 mode without getting the improved performance they paid for.) Make sure that your SVGA card comes with drivers for OS/2 2.1 in 1024×768 resolution and 256 colors (including seamless WIN-OS/2 support) or is supported by IBM's drivers.[13] (This machine cost about $ 1500 to 2000 in September 1993. I usually buy a cheap machine with a one-year warranty. After one year, the machine is obsolete and therefore has lost most of its value anyway. If a component fails, it is easy to replace.) Optional items include a network card, SCSI adapter, CD-ROM player, a tape backup unit supported by OS/2, fax board/modem, interface cards to control your experiment (IEEE-488, analog-to-digital converter, indexer card for stepping-motor control, digital I/O) and a parallel laser printer supporting various formats (HP Laserjet PCL, HPGL graphics language, Postscript, IBM PPDS) in at least 300 DPI resolution. With a large number of cards, you may encounter interrupt conflicts with an ISA-bus machine, since cards cannot share interrupt levels. You also may need an expansion chassis if you have more cards than slots in your PC.

OS/2 2.1 is available in three versions: On floppy disks (at least 18), on CD-ROM, or as a a license without media. If there is already a license for DOS on the machine, then an OS/2 upgrade can be purchased for under $100. Installing from floppies may be the easiest way (no problems with unsupported SCSI adapters or CD-ROM players), but using a CD-ROM is much faster. The installation is usually without complications, but special cases may require some planning, or hunting for drivers on the internet. Upgrading from OS/2 2.0 (with or without the service pack) is also painless. It is not necessary to reformat the partition (except when upgrading from a previous beta-test release), and your previous desktop will be preserved.

There is a number of issues to resolve before beginning the installation: If OS/2 is the only operating system to run on the machine, then the hard drive should be split (using FDISK) into several partitions (one for OS/2, one for programs, and one for data) with 50 to 100 MB in size and formatted using the HPFS format. Since DOS cannot access HPFS partitions, a small (50 MB) partition formatted as FAT can be set aside just in case you may want to boot DOS from floppy disk. Make sure to leave about 30 to 40 MB of free space on one partition for the swap file SWAPPER.DAT. If both OS/2 and DOS/Windows will be used, then the dual boot feature can be enabled. In this case, both OS/2 and DOS will reside on the C: partition formatted as FAT. More than two OS's (for example OS/2, UNIX, and Windows NT) can run on the machine, if a 1 MB partition is created for the OS/2 boot manager.

After installing OS/2 and booting for the first time, you should immediately shut down to save the system configuration on the hard disk. (Move the mouse pointer to an empty space on the desktop, click on the right button, move the pointer to the "Shut down..." menu option and click the left button. Never turn off the power without first shutting down, since this will the leave the desktop configuration and the file system in an inconsistent state. Some programs, such as the PMX X-window server cannot be closed this way, and have to be stopped by hand.) Now you can customize your system, for example change your display resolution from VGA to SVGA, install printer drivers, or other program packages (TCP/IP, compilers, applications, etc). Whenever an important step has been completed, shut down your machine and reboot to save your changes to the desktop and make sure the installed programs work properly.

VII. TROUBLESHOOTING OS/2 AND MAINTENANCE

If a problem occurs during installation, refer to your documentation. If this does not help, call the toll-free OS/2 support line. IBM's specialist are very helpful, in particular with installation and configuration problems. A number of USENET news groups or internet mailing lists may also be able to provide assistance. Most problems can be solved by booting OS/2 from the first two installation disks (Disk 0 and 1) and then running CHKDSK from Disk 2. This will correct possible errors in the file system and the extended attributes. A corrupt INI file (OS2.INI or OS2SYS.INI) can be restored using the MAKEINI utility, but call IBM first, since the procedure described in the appendix of the installation guide will destroy your desktop customization. If everything fails, there is only one solution: Reformat the partition and re-install.[14] This will destroy your desktop customization, so you may have to re-install some application programs also. Installing special device drivers not included in OS/2 2.1 or attempting to run some DOS or

Windows programs can be challenging and may be bring your system into an undefined state.

You should expect to find errors in the OS or application programs. There have been hundreds, maybe thousands of errors in OS/2 2.0, and even OS/2 2.1 (mostly a bug-fix release) is not without problems. The most severe problems experienced by many users are usually fixed with corrective-service disks after a few months (a service pack for OS/2 2.1 is supposed to be released at the end of 1993), but you may have to live with problems specific to your environment. If a program stops responding and the OS is hung, type Ctrl-Esc and wait for about a minute. This should bring up a window and enable you to close the application. If this still does not work, reboot by pressing Ctrl-Alt-Del or (if this does not work) Ctrl-Alt-NumLock (twice) and then Ctrl-Alt-Del.

The HPFS file system is much less affected by disk fragmentation than FAT, therefore defragmenting the disk may not be necessary. It is not recommended to run DOS disk utilities on OS/2, since they may destroy the extended attributes specific to OS/2. An OS/2 disk utility package (such as GammaTech) should be used instead. DOS virus scanning programs, on the other hand, can be used. (OS/2 virus programs are also available.)

Unfortunately, IBM provides few suggestions about how to maintain and backup your system. Backing up a multitasking OS is a problem by itself, since some files (such as SWAPPER.DAT) cannot be read and may cause verification errors. I tested a commercial tape backup program, but had to return it, since it would not work and repeated contacts with the manufacturer were unable to resolve the problems. There are some public-domain maintenance programs available on the internet (for example CHECKINI to correct errors in the system INI files, or WPSBKP to backup your desktop customization), but these may be specific to a certain OS/2 version. They also are poorly supported and may cause more harm than good if used improperly. Therefore, the issue of OS/2 2.1 maintenance remains unclear at this time.

---

[1] I am using the term PC to represent any type of IBM-PC or compatible computer, including desktop, tower case, and portable computers. These PCs may have different processors (compatible with Intel's 80x86 architecture including the new Pentium chip), clock speeds, and bus architectures (such as ISA, EISA, IBM's microchannel bus, or the VESA local bus). There are, of course, other types of personal computers (based on different CPU's) not compatible with the IBM-PC, for example the Apple Macintosh or the Hewlett-Packard HP300 series, but these will not be discussed in this article.

[2] In this article, I use DOS as a generic term. Virtually identical versions of DOS are offered by a number of companies (such as Microsoft and IBM) and are usually designed to control any brand of IBM-PC or compatible computer. Some hardware manufacturers (such as Hewlett-Packard) offer versions designed specifically for their own hardware. There are also DOS enhancements such as DR-DOS or 4DOS which provide the user with useful commands omitted in the original versions of DOS. Most manufactures preinstall a version of DOS or OS/2 2.1 on a computer before shipping.

[3] In order to use the new OS's discussed in this article, you need a PC with a 80386SX CPUs or better, including any 80386 or 80486 CPU or the new Pentium (80586) chip.

[4] In a similar fashion, the CPU can also access a hard disk partition with up to 4096 MB in a single step, by putting a 32-bit address on the bus. Compare this with the early days of DOS, where a partition could only be a few MBs in size. Clearly, the 4096 MB limitation already causes problems, since SCSI hard disks with several GBs in size are already on the market today.

[5] There may be some PC-based versions of UNIX that are also 32-bit OSs. However, none of these UNIX versions has been able to gain a dominant market share. It is conceivable (but not very likely in my opinion) that UNIX may become the dominant OS for the PC in the future.

[6] I am using the term Windows to denote Microsoft Windows 3.x (or Windows for Workgroups) or earlier versions of this software. This should not be confused with Windows NT (which is an entirely different OS also by Microsoft) and WIN-OS/2, the Windows 3.x-compatible subsystem of OS/2 2.x. There are, of course, other graphical user interfaces based on a desktop, windows, and folders, such as the Apple Macintosh system or XWindows for UNIX.

[7] My students and I recently wrote a program for stepping motor control under OS/2 2.1, where we directly wanted to access the indexer card using INP and OUTP statements, just like we would have done in DOS. The way this can be achieved without too much difficulty is to link a 16-bit code segment running in ring 2 with the application program, where some privileged statements are allowed. Details will be discussed elsewhere.

[8] The OS/2 error message SYS3175 is frequently encountered and indicates a GPF. This error is caused by the application, not the OS, therefore the vendor or developer should be contacted for a fix. Unfortunately, not even IBM's tech support is responsive in fixing these errors caused by IBM applications (such as the EPM editor, IPMD debugger, or LAMAIL).

[9] The commands in AUTOEXEC.BAT are executed when a virtual DOS processor is started, but not when booting or starting an OS/2 processor.

[10] There is no doubt that its internal architecture makes Windows NT a more advanced operating system than OS/2 2.x, at least in theory. The deserialized GUI message queue, superior handling of interrupts and time slices for multitasking, multiple platform support for Intel and DEC Alpha chips, multi-processor support, and a more secure network file system are currently included in Windows NT, but only in the planning stage for future IBM OSs. In practice, OS/2 has five years of experience and is therefore more reliable. By the time NT is available in a "production" version, these

features will probably also be available from IBM. OS/2 excels with its object-oriented workplace shell (making Windows Program Manager look obsolete) and better backward compatibility for DOS and Windows applications. Recent issues of the journal *OS/2 Professional* discuss these issues in more detail.

[11] OS/2-native applications include the following: Wordperfect 5.2, Stacker (FAT only), DCF/2 (disk compression for HPFS), SAS (data base and spreadsheet), PFS:Works, FAX/PM, FaxWorks, CorelDraw, Borland ObjectVision, Visual REXX, Freelance Graphics, Lotus 1-2-3, Lotus Notes, GammaTech disk utilities, DeScribe (word processor), Ultimedia/2. Not all of these are worth buying.

[12] If you cannot afford 16 MB of RAM, make at least sure you can later upgrade to 16 MB without having to discard your 1 MB SIMMs.

[13] Some manufacturers have their drivers ready when a new OS is released. Others (for example NCR or ATI) take up to a year to deliver drivers. By that time, however, the next OS release will be on the market, which means that you may be without drivers again. If drivers are released on time, they may be a quick port of the DOS drivers not fully supporting the new OS, or a port of

[14] Now it is clear why I suggested to set aside one partition for OS/2 alone. If you have only one large partition and no tape backup drive, you are in trouble.

FIG. 1. Memory contents of a PC with 20 MB RAM when run under DOS 5.0

FIG. 2. Structure of OS/2 2.1 and its various components and sub-systems. The solid line separates code running in user (ring 3) and supervisor (ring 0) mode.

FIG. 3. A sample OS/2 desktop showing the various features of the workplace shell and some PM applications, i.e., printer icons for various formats (emulated by a single IBM 4029-022 printer with automatic switching), information folder, minimized icon viewer, shredder (trashcan), OS/2 system, command prompts, and programming folders, OS/2 shell window, and elctronic mail application.

FIG. 4. A sample Presentation Manager (PM) window illustrating the various programming controls. (Copyright by IBM. All rights reserved. Reprinted with permission.)

TABLE I. Comparison of different operating systems for 80x86 CPU's.

	DOS 6.0	Windows 3.1	Chicago	Windows NT	OS/2 2.1
32-bit OS	NO	NO	YES	YES	YES
640 kB barrier	YES	YES	YES	NO	NO
flat memory model	NO	NO	NO	YES	YES
virtual memory support	NO	YES	YES	YES	YES
graphical user interface	NO	YES	YES	YES	YES
object oriented shell	NO	NO	NO	NO	YES
DOS device drivers loadable	YES	YES	YES	NO	YES
memory protection	NO	NO	NO	YES	YES
preemptive multitasking	NO	NO	NO	YES	YES
deserialized message queue	NO	NO	NO	YES	NO
fixed time slice	YES	YES	YES	NO	YES
multiprocessor support	NO	NO	NO	YES	NO
non-Intel platform support	NO	NO	NO	YES	NO
installable file system	NO	NO	?	YES	YES
recommended CPU (or better)	8088	80286	80386SX	80486	80386
recommended minimum memory	1 MB	4 MB	4 MB	16 MB	8 MB
recommended minimum disk space	2 MB	10 MB	10 MB	200 MB	100 MB
client-server support	NO	limited	limited	YES	YES
best purpose	standalone	client	client	server	any
DOS subsystem	N/A	YES	YES	YES	YES
Windows 3.1 subsystem	N/A	YES	YES	YES	YES