DECEMBER 1993

Chicago Questions and Answers

Microsoft is continually enhancing its Windows operating system product line to deliver easy to use yet powerful products that exploit the latest advances in microcomputer hardware technology. There is a great deal of interest in and speculation about the "Chicago" project, the technology development effort which will deliver the next major release of Windows for the mainstream desktop and portable PC. The purpose of this document is to answer the most common questions that customers have voiced about Chicago.

- **What is Chicago and how does it compare to the Microsoft® Windows™ 3.1, Windows™ for Workgroups and Windows NT™ operating systems?**
  Microsoft has a family of operating system products designed to fully utilize the range of PC hardware available in the market today, while providing a consistent user interface for end users and a programming environment for developers. Windows 3.x and Windows for Workgroups 3.x on MS-DOS® are designed for mainstream portable and desktop PC platforms. Windows NT is designed for the high-end business and technical workstation platforms and Windows NT Advanced Server is designed as a server platform.

  Chicago is the code name for a development project that will produce the successor to Windows 3.x and Windows for Workgroups 3.x. The Chicago project encompasses a variety of important new technologies that will make personal computers running Windows easy to use, and that will provide a more powerful multitasking system and a great platform for communications. Decisions about how those technologies will be packaged will be made later in the development cycle and will be based on customer and business needs.

- **What is Cairo? How does Chicago compare to Cairo?**

  Cairo is the code name for a development project that will produce the successor to Windows NT. Chicago and Cairo will produce complementary products that will continue to provide a consistent user interface and programming environment across the entire range of PC hardware platforms.

- **Why does Microsoft have multiple Windows operating system products? Wouldn't it be simpler to just have one product? Does that mean ISVs have to decide between different operating system products when writing applications?**

  There are two distinct design points for operating systems platforms. One is centered on the mainstream system, and the other is centered on the high-end system. It is not possible to have one operating system implementation that fully exploits the broad range of hardware available today. At the low end (currently represented by products such as the HP Omnibook and entry-level desktop machines), the primary design goal is to keep the operating system small and fast and to keep usage of machine resources to a minimum. At the high end (for example, a dual-processor technical workstation), the product would need to fully support multiprocessing and advanced 3-D graphics as well as be capable of running technical applications that use maximum machine and system resources.

  Over time, low-end machines will become more powerful, and over time, some of today's high-end features will migrate to the low end. In addition, some technical innovations will appear on the mainstream Windows system first, largely because of the timing of product releases, and because some features are focused on end users and ease of use. The Win32 API assures developers that, whichever system they target today, their applications will be able to run in the future as the platform evolves. Thus, while Chicago and Cairo may leapfrog one another with some features, depending on release cycles — e.g., Chicago will sport the next major advance in the user interface, with Cairo inheriting it in its release a few months later — the general principle over time is that the high-end product will be a superset of the functionality offered in the mainstream product. Any deviations from this principle are temporary, due to variations in the product release schedules.

  For ISVs and for development purposes, however, Microsoft has just one Windows platform, defined by the Windows-based 32-bit API, Win32. By following a few simple guidelines, ISVs can write a single application (executable) that runs on the Windows operating system product family. If they wish, ISVs can target specific operating system products because the functionality they provide is important to their particular application, but that is not a requirement.

  This situation is very much like the Intel® microprocessor product line. At any point in time, the Intel product line offers multiple products targeted toward different PC products, ranging from the 80386SL for low-end portable products to the Pentium™ microprocessor for high-end workstations and application servers. What defines those products is the Intel instruction set, which enables applications to run on all Intel chips, even though the underlying implementation at the transistor level may be very different across the Intel product line. There are also some instructions offered on the Pentium chip that are not on the 80386SL, but ISVs would have to go out of their way to make their products run on only Pentium. And over time, Pentium will

become more mainstream, just as the 80486 has become the mainstream microprocessor today, and technologies developed at the low end, such as System Management Mode, will be implemented on the high end as well.

■ **When will Chicago ship? When will Cairo ship?**

Chicago is scheduled to ship in the second half of 1994. Cairo is scheduled to be released in the first half of 1995.

- **What is Daytona? When will it ship?**

  Daytona is an interim release of Windows NT that is scheduled to ship this spring.

- **Major new releases of operating system products have in the past been significantly delayed. How will you make your projected shipment date for Chicago?**

  Chicago will be released when customers tell us it is ready. The way to make shipment dates is to hit your intermediate milestones. To date, Chicago has been making its milestones with the release of the first Preliminary Developer's Kit (PDK) in August and the second PDK in December. Feedback from beta releases beginning in March will tell us more precisely when in the second half of 1994 Chicago will ship.

- **If Chicago ships before Cairo, how will users of Windows NT obtain the new functionality in Chicago?**

  Any new functionality offered in Chicago will be made available to customers of Windows NT through the release of the Cairo product.

- **What are the key benefits and features of Chicago? What features will Chicago *not* have?**
  For customers, Chicago will present a major step forward in functionality on mainstream desktop platforms by providing a system that is easy to use, offers responsive multitasking performance, and provides a great platform for communications. Ease of use will be delivered through the Plug and Play architecture and an improved, intuitive user interface. Chicago will be a complete, integrated protect-mode operating system that does not require or use a separate version of MS-DOS, implements the Win32® API, and provides pre-emptive multitasking and multiple threads of execution for 32-bit applications. The communications capabilities of Windows will be enhanced with integrated, high-performance networking, built-in messaging, and features such as Remote Network Access and File Synchronization designed for mobile and remote computer users.

  Chicago will also be a hassle-free upgrade for the current installed base of Windows-based users. Chicago will be compatible with most current applications and drivers for MS-DOS and Windows, and will provide an easy transition to the new user interface features. The applications performance of Chicago will meet or exceed the performance of Windows 3.1 on 80386 systems with 4MB of RAM running the same applications. For systems with more memory, performance will be significantly improved over Windows 3.1. The setup program will enable customers to uninstall Chicago, assuring customers a way to remove it if they are in any way unhappy with it, and will provide tools for system administrators to customize the configuration of

Chicago.

Chicago will not be processor independent, nor will it support symmetric multiprocessing systems, provide C2-level security, or provide full Unicode support. These features cannot be delivered on the mainstream platform in the near future while still meeting the performance and resource targets necessary to create a compelling upgrade for the huge installed base of users of the Windows operating system. If these features are important to a customer, Windows NT is the product to deploy.

- **What different packages will you have for Chicago?**

  Decisions about packaging the different technologies being developed as part of the Chicago project will be made later in the development cycle and will be based on customer and business needs. One option is to provide a base Chicago package with some add-on packages that deliver functionality required by specific market segments. This is much like the situation today in which the user of Windows 3.1 can upgrade to Windows for Workgroups by acquiring the add-on package that adds the 32-bit file system and 32-bit networking enhancements to Windows.

- **Since the term Chicago is a code name, what will you call the product(s) that you will eventually release?**

  Decisions about names will be made after we decide on a packaging plan.

- **What will happen to the MS-DOS product line?**

  Microsoft will continue to enhance MS-DOS as long as customers require it. Future versions will be derived from the protected-mode technology developed in the Chicago project. Current MS-DOS–based applications and drivers will continue to be compatible with new versions of MS-DOS.

- **Your performance goals on 4MB platforms sound very ambitious, considering all the functionality you're adding to Chicago. How will you achieve those goals?**
  Chicago will implement new working set management technologies that will optimize the use of memory on low-configuration systems. The networking, disk and paging caches will be fully integrated. Protect-mode device drivers will be dynamically loadable, to ensure that only the drivers that are immediately needed are consuming memory. More components of the base operating system will be pageable. Great attention will be paid to effective page tuning, including hand-tuning source code.

- **Will Chicago run my current Windows-based applications? How about MS-DOS–based applications?**

Chicago will run most of the current applications for Windows and MS-DOS, as well as new applications written to the Win32 API. Some classes of applications will need to be revised to be compatible with Chicago, such as shell-replacement utilities and file-management utilities. Chicago's new shell provides a complete set of services that is tightly integrated with the operating system components. Shell programs will

need to do more than simply replace components such as Program Manager or File Manager. And file-management utility vendors will want to revise their applications to take advantage of the Long File Name feature that Chicago offers. Microsoft is working closely with shell-replacement and file-utility vendors to enable them to revise their products to add value to and be compatible with Chicago.

- **Will I have to get new device drivers to use my current devices with Chicago?**

  Chicago supports current real-mode device drivers as well as new 32-bit protected mode device drivers. As a result, customers will be able to use their current devices either with their current device drivers, or with new device drivers made available with Chicago. Performance and functionality can be improved if the user installs the new Chicago drivers. Microsoft is making it easier for device manufacturers to deliver new drivers for common devices by defining a more layered, modular device driver architecture. For displays, printers and modems, Microsoft will deliver universal drivers. These drivers will implement common device functionality and expose an interface for device manufacturers to create "minidrivers" that implement the features specific to their devices. This approach was very successful with printers for Windows 3.1, resulting in rapid availability of fast, high-quality drivers for a wide range of printers.

- **Will my current applications perform as well on Chicago as they do on Windows 3.1 today?**

For Chicago to be a compelling upgrade, Windows-based users must experience a level of performance after installing Chicago that meets or exceeds the performance they currently experience running an identical set of tasks on Windows 3.1. Because a large portion of the installed base of users of Windows today have 4MB systems, Chicago must meet its performance goals on 4MB systems. On systems with more than 4MB of RAM, Chicago will offer significantly improved performance.

Understand, however, that there are user and application scenarios today that already use more than 4MB. Users who already require more than 4MB will continue to require more than 4MB with Chicago — and if they are using more than 4MB, they should see improved performance. But they won't get away with using less memory in the future than they do today. It's an important distinction to maintain.

- **You say Chicago will have a different user interface than Windows and Windows NT. When will that user interface be reflected in the beta versions of Chicago?**
  The new user interface will be delivered with the first beta of Chicago, scheduled for March 1994.

- **Won't a new user interface mean a lot of retraining for current Windows-based users? Will the advantages of the new user interface be worth the retraining costs?**

  The user interface being developed for Chicago will offer dramatic gains in ease of learning and ease of use for the broad range of people using PCs today. Instead of mastering different kinds of tools to work with different resources on their computers, users of Chicago will be able to browse for and access all

resources in a consistent fashion with a single tool. This will be much easier than learning separate applications such as Program Manager, File Manager, Print Manager, Control Panel, etc. as users of Windows must do today. A system toolbar that is always accessible will make it much easier to start and switch between full-screen tasks. The implementation of OLE 2.0, with its focus on the user's document rather than on the tool used to create it, and the direct manipulation of data through drag and drop in the user interface, will make working with documents easier and more intuitive.

Current users of Windows will be immediately productive with Chicago and be able to learn the new features of the user interface as they work. Chicago's smart setup technology will use the current system settings to present an initial configuration that is familiar for the current Windows-based user. And for corporate customers and individuals who may not want to make any user interface changes initially, Chicago will enable them to continue running their current Program Manager and File Manager configurations.

- **What is Plug and Play? What benefits does Plug and Play provide?**

Plug and Play is a technology jointly developed by PC product vendors that will dramatically improve the integration of PC hardware and software. It allows a PC to adapt itself dynamically to its environment; devices can be plugged into or unplugged from a machine, without the user having to do anything special — the machine just works. Plug and Play is a general framework that advances that state of the PC architecture by defining how the software communicates with any device connected to the PC.

Plug and Play technology enables installation and configuration of add-on devices without user intervention. Plug and Play will make it possible for a consumer to turn a standard desktop system into a great multimedia machine by just "plugging" in a Plug and Play sound card and CD-ROM, turning on the system, and "playing" a video clip.

Plug and Play can enable new system designs that can be dynamically reconfigured. For example, imagine a docking station that enables you to remove the portable system while it is still running so that you can take it to a meeting, and the system automatically reconfigures to work with a lower-resolution display and adjusts for the absence of the network card and large disk drive. Or imagine an IR-enabled subnotebook that automatically recognizes, installs and configures an IR-enabled printer when you walk into the room, so your applications are ready to print to that printer.

Plug and Play can also save development and support costs for the product manufacturer. Today, as many as 50 percent of support calls received by operating system and device manufacturers are related to installation and configuration of devices. With Plug and Play, device driver development is simplified because device manufacturers can write one driver that works across multiple bus types using the Universal Driver Model specified by the Plug and Play architecture. Today, device manufacturers have to include bus-specific code in each of their drivers. With Plug and Play, specific bus configuration data is contained in "bus drivers." Also, operating system preinstallation and configuration are simplified for OEMs because Plug and Play devices will automatically install and configure during setup.

- **What changes to current hardware and software are required to make Plug and Play a reality? How will vendors figure out how to develop new devices with Plug and Play capability?**

  First, Plug and Play is compatible with existing systems, so nothing "breaks" because of Plug and Play. Plug and Play devices can be brought out over time — in fact, this is already occurring — and will work with existing systems.

  To deliver all of the above benefits requires changes to devices and drivers, the BIOS, and the operating system. Three fundamental capabilities are required for a system to provide Plug and Play functionality:

  - A unique identifier for every device on the system
  - A procedure for the BIOS and operating system to install and configure that device
  - A mechanism for the system and applications to recognize that a configuration change has occurred while the system is running

  All the changes to devices and drivers, the BIOS and the operating system are defined by a series of specifications for Plug and Play architecture. The Plug and Play architecture is an open, flexible and cost-effective framework for designing Plug and Play products.

  The Plug and Play architecture was jointly developed by a working group of leading vendors, who reviewed design proposals with hundreds of companies in the industry at conferences and through online forums. Plug and Play can be implemented by any operating system vendor and any hardware manufacturer. In addition to Microsoft, IBM has announced support for Plug and Play in OS/2®.

  The Plug and Play architecture is flexible, because it provides a framework that works on multiple types of bus architectures (ISA, SCSI, PCMCIA, VL, PCI, etc.), and it is extensible to future bus designs.

  The Plug and Play architecture is also cost-effective, because it requires little or no incremental cost for vendors to implement in their products.

- **Won't it take a long time for these changes to be reflected in products?**

  Acceptance of the Plug and Play architecture is widespread, as seen by the rapid progress the industry is making in delivering Plug and Play specifications and products.

  Specifications have already been released for ISA, SCSI and PCMCIA devices, and the Plug and Play BIOS. Additional specifications are in process, including PCI, ECP, VL, EISA, Micro Channel, and Access. The first Plug and Play devices were demonstrated at COMDEX/Fall 1993, representing a wide range of companies and products. Intel has released development kits that enable device and system vendors

to deliver improved configuration capabilities for ISA and PCI systems running with Windows 3.1 in a manner that will provide compatibility with future Windows operating systems. Fully Plug and Play-capable systems (including all Plug and Play devices and a Plug and Play BIOS) will be available in the first half of 1994. These systems will be able to offer complete Plug and Play functionality when combined with Chicago.

- **I've heard that Chicago implements a 32-bit API. Is that API different from the 32-bit API implemented on Windows NT?**

There is only one 32-bit Windows API, called Win32, with ISVs able to use the API set to provide different levels of functionality for Windows 3.1, Chicago and Windows NT. Chicago implements a large subset of the functionality of the Win32 API offered on Windows NT, and extends the Win32 API in some areas. These extensions will be delivered on Windows NT as soon as possible after the release of Chicago.

- **If there are different implementations of the Win32 API available on different products in the Microsoft operating system product line, does that mean ISVs will have to have separate versions of their applications for Windows and Windows NT?**

No. By following some simple guidelines, ISVs can develop a single executable file that runs on Windows 3.x, Chicago and Windows NT. At the recent Professional Developers' Conference, we provided in-depth technical sessions on the proper way to design applications to do so, supplied tools in the SDK to help make such development easier, and showed several applications that ran across the entire Windows family.

- **When will applications be available that exploit Chicago? Won't that take a long time?**

ISVs who are developing 32-bit applications for Windows 3.1 and Windows NT using the
Win32 API and the guidelines we have provided will have applications that are able to run on Chicago immediately. There are already more than 250 Win32 applications available today, and more coming quickly. Other ISVs will wait until Chicago ships to provide their 32-bit applications; usually those applications start coming on-line about 90 days after the operating system ships. Chicago also will support today's 16-bit applications, so users can move to Chicago immediately and upgrade their applications as they become available.

Chicago represents a major market opportunity for ISVs. Chicago will ship on almost all OEM systems soon after it is released, and it will be acquired as an upgrade by a substantial portion of the Windows installed base (the installed base will probably number more than 50 million by mid-1994). Customers who purchase new systems and upgrade their operating systems are the most active purchasers of new software applications. As a result, ISVs have a very significant business incentive to release versions of their applications that exploit Chicago.

- **I've heard Chicago described as a 32-bit operating system, yet I've also heard that portions of Chicago are implemented with 16-bit code. Are both these statements correct?**

Chicago will provide a 32-bit platform for applications by implementing the Win32 API on a complete, protect-mode operating system. Chicago will also run well on

mainstream Windows platforms (which for a large portion of the Windows installed base is a 4MB 80386 system), and Chicago will be compatible with applications and drivers for MS-DOS and Windows. These requirements must be met if Chicago is to meet customer needs and provide the volume to make ISVs successful.

These requirements have driven all the design decisions for Chicago. The resulting design deploys 32-bit code wherever it improves performance without sacrificing application compatibility. The design retains existing 16-bit code where it is required to maintain compatibility or where size is a critical issue but has minimal impact on performance. All of the I/O subsystems and device drivers in Chicago, such as networking and file systems, are fully 32-bit as are all the memory management and scheduling components (the kernel and virtual memory manager). Many functions provided by the Graphics Device Interface (GDI) have been moved to 32-bit code, including the spooler and printing subsystem, the rasterizer, and the drawing operations performed by the graphics "DIBengine." Much of the window management code (user) remains 16-bit to retain application compatibility.

- **If portions of Chicago still remain 16-bit, what happens when a 32-bit application makes a function call that is implemented by the 16-bit Chicago component? Doesn't this slow down 32-bit applications on Chicago relative to 16-bit applications?**

When Win32-based applications call a 32-bit API that is implemented by a 16-bit component of the system, the function call is translated to its 16-bit equivalent for processing by the system. This translation process is referred to as "thunking." Although there is some overhead associated with a thunking operation, the Chicago thunk layer is very efficient. That overhead will be more than offset by the improved efficiency of the linear memory addressing scheme used by Win32-based applications. The overall impact of some "thunking" code is quite modest vs. all the other work the application and operating system have to do.

For end users, perceptions of application performance are based on a combination of the efficiency of the application when executing its own code and the efficiency of the operating system code when the application has called an operating system service. On Chicago systems with adequate memory, end users will experience gains in system efficiency when running 16-bit applications, and they will experience gains in both system and application efficiency when running 32-bit applications.

- **Will I need new networking software to connect Chicago to my network server?**

Customers will require Chicago to connect to their network servers when Chicago is installed, and to offer high-performance, reliable networking functionality. To meet this requirement, Chicago will continue to run existing real-mode networking components. However, we expect customers to want to upgrade to the new 32-bit networking components provided by Chicago. Chicago will enhance the open, flexible, high-performance 32-bit networking architecture offered today with Windows for Workgroups 3.11 that enables customers to mix and match networking components. Chicago will support NDIS 2.0, NDIS 3.0 and ODI drivers, and will provide 32-bit NetBEUI, IPX/SPX and TCP/IP protocols. Redirectors for SMB and NCP-based networks will be included. In addition, Chicago's new multiple-provider interface will make it possible for the

user to view, browse and connect to multiple networks in a consistent fashion.

- **What about NetWare®? Are you working with Novell on NetWare support?**

  Customers will require high-performance, reliable NetWare support the day Chicago is released. To meet that requirement, Microsoft is developing a 32-bit NCP Redirector that is seamlessly integrated with the Chicago user interface, and is encouraging Novell to do the same. Microsoft will offer Novell access to information and assistance to write a Chicago redirector. Novell engineers attended the Win32 Professional Developers' Conference and have been provided access to the Preliminary Developer's Kit for Chicago.

  With this approach, customers should be able to choose from multiple sources for reliable, high-performance NetWare connectivity software when Chicago is released.

- **Will there be a Chicago server?**

  No, not in the sense of a "server product" such as Windows NT Advanced Server. Chicago will continue to improve upon the peer server capabilities offered in Windows for Workgroups by offering additional features for remote installation, control and administration. These features will make Chicago an even better product for an easy-to-use file and print-sharing LAN that is ideally suited as a small-business, small-department or remote office network. Similarly, Windows NT offers peer services as well for the high-end desktop. But for most server applications, and in the sense that most people ask about a "server product," Windows NT Advanced Server is the Microsoft server product.

- **I keep hearing rumors that you are working on a portable version of Chicago. Is this true?**

  No, we are not working on a portable version of Chicago. Windows NT is our portable operating system, and it's already available on high-end Intel, MIPS®, Alpha and Clipper™ machines; it will be available on the PowerPC™ by mid-1994 and on other high-end platforms over time. There is no reason to make Chicago portable. Chicago is optimized for Intel processors, and much of its internal code is Intel assembler, which puts Chicago at the heart of today's low-end and mainstream line. Portability is important for the new generation of high-powered Intel and RISC machines, on which Windows NT runs and for which Windows NT has been optimized. As these new high-end machines become more mainstream, which will happen over time, Windows NT will already offer the power, security, and reliability that users will demand to exploit these new machines.

- **What will Chicago do to make the client operating system more manageable?**

  A primary goal for the Chicago project is to make Windows less expensive to deploy in a corporation. Chicago will include some specific features and enabling

technologies that will make it easier for system administrators to install, configure, monitor, maintain and troubleshoot their Windows-based desktops.

Chicago can be set up from a network server and at the desktop can be configured at the desktop to run locally or across the network. In each case, the administrator can establish a specific configuration for the installation, selecting from a flexible array of setup configuration options. Chicago desktops require only a floppy drive to start up, and paging of components to a swapfile on the network can be disabled to minimize network traffic.

Once Chicago is installed, administrators will be able to centrally configure desktop settings such as file and printer sharing, network access, and passwords. They can remotely monitor Chicago desktops with peer services running to determine what resources are shared, what connections have been made, and what files are being used. Chicago enhances the security provided by Windows for Workgroups to include user-level security. To enable users to access their personal groups, applications, and data from any system on the network, Chicago will provide user profiles.

Chicago will also provide the infrastructure for the delivery of enhanced desktop management services by third parties. A backup agent will be included with Chicago to enable administrators to back up desktop data to a network server. To integrate the desktop into SNMP-based enterprise management systems, Chicago will also include a Systems Network Management Protocol (SNMP) agent and a Management Information Base (MIB) for a number of system resources. The system registry and Plug and Play architecture provide a rich store of data about the software and hardware configuration on the desktop, and this information can be accessed by system management software using a DCE-compliant Remote Procedure Call (RPC) mechanism.

- **What improvements will Chicago offer for people who use a mobile or remote computer?**

Chicago will provide great support for mobile form-factor devices and will make it easy for end users to access the resources of their desktop systems when they are away from their offices. The implementation of Plug and Play in Chicago will support insertion and removal of devices such as PCMCIA cards while the operating system is running. It will also support automatic reconfiguration of dockable computers when they are inserted or removed from the docking station, without rebooting the system. An enhanced version of Advanced Power Management will further extend battery life. The services provided by Windows™ for Pen Computing will be enhanced and incorporated into Chicago, including basic inking and rendering support.

A special focus will be on remote connectivity. Any Chicago-based machine will be able to serve as a Remote Access dial-up server or a remote client for Windows NT Advanced Server, Novell NetWare servers or Chicago peer servers. The same technology will be used for serial cable and infrared connections between PCs. The Remote Access architecture will be integrated with the Chicago networking architecture by using the same network protocols and advanced security features. Remote Access will support wireless devices and allow application developers to make their applications “slow-link aware” to improve the user experience when working on a remote system via modem rather than on a high-bandwidth network. Furthermore, Chicago will provide a simple form of file synchronization and APIs for applications to access the file synchronization services to merge changes when both the source document and

copy have been modified.
Remote e-mail and Microsoft at Work™ fax capability will be included, as in Windows for Workgroups 3.11 today.

- **Will the file synchronization feature in Chicago provide document management capabilities?**

Chicago's file synchronization services are optimized for the needs of the mobile computer user who wants to take copies of documents to a remote location and have them be automatically synchronized with the source documents. It is not intended as a replacement for sophisticated document management systems.

Chicago's file synchronization allows customers to identify files that they want to stay up to date, to change those files, and to have the files automatically updated when the source file is available to the system. The update is performed by replacing the source file with the modified copy at the discretion of the user. If an application writes a "merge-handler," then specific data within the modified and source copies of a file can be merged, to create a new updated copy.

- **You say you have one API with Win32. Does that mean there will also be just one Windows SDK?**

Yes, there will be one Win32 SDK that developers can use to develop 32-bit applications for Windows 3.1, Chicago and Windows NT. In fact, we recently announced a new subscription service, the Microsoft Developer Network Level II that provides developers with not only the Win32 toolkit, but every system toolkit we offer, on a single CD, updated quarterly.

- **What benefits does Chicago offer to developers? What are you doing to make developing Windows-based applications easier?**

The Microsoft Visual Basic® programming system has dramatically streamlined and simplified the development of Windows-based applications, and it will be enhanced to support the development of 32-bit applications for Chicago. Microsoft also is enhancing its Visual C++™ development system and Microsoft Foundation Class tools.

- **Will Chicago include Visual Basic for Applications?**

Visual Basic for Applications will be offered as a separate product.

- **Will Chicago and Windows NT share the same device drivers?**

Generally not, since Chicago and Windows NT have different device driver models. However, since both products support a modular, layered device driver architecture, there are areas of substantial synergy. For example, SCSI miniport adapters for Windows NT will be binary-compatible with Chicago, as will printer drivers and NDIS drivers for Windows NT.

- **Will WOSA services be included with Chicago?**

  WOSA is a general, open framework for implementing multiple back-end services in Windows while providing a single front-end interface for end users. Services in Chicago such as messaging and remote network access are designed according to the WOSA framework. Whether or not support for additional WOSA services, such as ODBC support, will be shipped with Chicago is a packaging decision that will be made later in the development cycle and will be based on customer and business needs. All the WOSA-related toolkits are available today to developers through the Microsoft Developer Network Level II subscription service.

#########

©1993 Microsoft Corporation. All rights reserved.
Microsoft, MS-DOS, Visual Basic and Win32 are registered trademarks and Microsoft at Work, Visual C++, Win32s, Windows and Windows NT are trademarks of Microsoft Corporation.
HP is a registered trademark and Omnibook is a trademark of Hewlett-Packard Company.
Intel is a registered trademark and Pentium is a trademark of Intel Corporation.
OS/2 is a registered trademark and PowerPC is a trademark of International Business Machines Corporation.
Novell and NetWare are registered trademarks of Novell, Inc.
MIPS is a registered trademark of MIPS Computer Systems, Inc.
Clipper is a trademark of Computer Associates International, Inc.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This document is for informational purposes only. **MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.**