

INSIDE

EDITORIAL

**The Trouble with Standards .....Page 2**  
Developers, especially smaller ones, know that standards can help them greatly. Can X/Open convince vendors that standards can help them, too?

COLUMN

**Forum Update..Page 13**  
Detailed coverage of the Executive UniForum Symposium, "On the Road to Commercial Unix," cosponsored by Patricia Seybold's Office Computing Group and /usr/group.

NEWS ANALYSIS

**CMU developers found Transarc with IBM's help • AT&T's enhanced C++ Release 2 has clarity, but not much else • CimLinc's ID forms processor can act as a front end to databases • Uniplex for the Sun .....Page 20**

# UNIX IN THE OFFICE

PRODUCTS • TRENDS • ISSUES • ANALYSIS

## Thoroughly Modern Unix

*The Shape of New Operating Systems*

By Laure Brown and Tom Portante

**N**OT LONG AGO, one of our analysts, elbow-deep in research, kept stumbling over the term "modern operating system." It leapt out from articles, from the mouths of Unix gurus she interviewed, even from her own notes. And along with it came some intriguing prospects: power, next-generation applications, performance, flexibility, scalability, advanced programming tools, portability... you get the picture. Higher computing plateaus require better (*continued on page 3*)

WE ORIGINALLY intended this editorial to be about the fact that OSF and Unix International became members of X/Open, but we changed our minds (we're allowed). When we put on the hat of the applications developer in the Unix environment, we can't get too excited about this event. Yes, it is important, and yes, it will help—in the long run. But, for the applications developer working hard to provide next-generation software, there is a more important and immediate concern. Despite the advantages of Unix in terms of portability, all is not rosy for developers.

Why the problems? Many of these software developers are small businesses with limited resources. Therefore, when they begin to port their applications to various platforms, they are frustrated by the proprietary hooks that vendors place in their operating systems to gain the strategic edge. One developer recently spoke of the problems she faced when trying, at the request of a customer, to port an application to an early release of AIX for the 370. After many months of effort, she discovered that the job couldn't be done. In the meantime, a lot of time and money had been wasted. We believe that this developer isn't alone.

Thus, we revised the topic for this editorial to read: X/Open as a unifying force. If X/Open can continue to leverage its position as the common denominator, it could help developers to have a common operating system for porting applications. This goal will not be easy to achieve, however. Despite the clout that X/Open has gained in recent years, the standards environment in which vendors must compete does not lend

• E D I T O R I A L •

# Operating System Standards and Applications

All is not rosy for developers.


By Judith S. Hurwitz

which will involve over one hundred users and ISVs, will handle issues such as user interface, connectivity, systems administration, portability, development tools and languages, and human-computer interface, to name a few. ISVs are particularly concerned about the impending onslaught of graphical user interfaces. Vendors are not only facing the prospect of writing applications to a bit-mapped display, but also the possibility of having to write to many different interfaces. The consensus of this group should provide X/Open with the ammunition to convince vendors that it is in their best interest to conform to standards at the operating system and network levels.

The problem remains: How soon will all of this standardization take? How much longer will our friend with the small software company be forced to write and rewrite software for 200 different versions of Unix? For companies like this, the solution cannot come too soon. We only hope it doesn't come too late. ☺

itself to conformance. For example, will IBM and Digital really be willing to forego proprietary hooks in their operating systems? How will these companies react if their customers began to port their strategic software to other hardware?

The dilemma is clear—and so is the challenge for X/Open: to convince vendors that they have more to gain from standards than they have to lose. In June, X/Open will take a step in this direction by sponsoring a meeting in which sophisticated end users and independent software vendors (ISVs) will tell the organization what its requirements are for the next five years. The meeting,

	<b>Publisher</b> PATRICIA B. SEYBOLD	<b>Editor-in-Chief</b> JUDITH S. HURWITZ
	<b>Managing Editor</b> RONNI T. MARSHAK	<b>News Editor</b> DAVID S. MARSHAK
<b>Senior Editors</b> MICHAEL D. MILLIKIN JUDITH R. DAVIS Telephone: (617) 861-3926	<b>Contributing Editor</b> GARY J. NUTT Telephone: (303) 444-0341	<b>Sales Director</b> RICHARD ALLSBROOK JR.
<b>Associate Editors</b> JOHN R. RYMER LAURE BROWN	<b>Circulation Manager</b> DEBORAH A. HAY	<b>Customer Service Manager</b> MICHELLE MANN
148 State Street, Suite 612, Boston, Massachusetts 02109 Telephone: (617) 742-5200 FAX: (617) 742-1028		

## • MODERN OPERATING SYSTEMS •

(continued from page 1) operating systems. And while work is being done in operating systems design, the commercial market hasn't seen much of it. But what plagued our analyst—and what has prompted this article—was less a question of timing than of definition. Just what is a modern operating system? What's not modern about today's commercial Unix operating systems? What are we missing?

Operating systems have adapted over the years to the changes in quality, quantity, and the nature of the resources they must control and the services they must offer. Today they face the demands of standards, distributed network computing (DNC), and object management. DNC is most notably driving Unix to its next generation. Operating system designers confront the challenges of multivendor and multi- and parallel-processing machines running distributed applications over high-speed networks. While Unix has been tinkered with to let it play in these environments, the hacking has exacted a performance toll. New solutions have been called for.

Operating system R&D—especially in the academic world—is a source of innovation for Unix. Some valuable prototypes have been under development for years. The Mach operating system, developed at Carnegie Mellon University, is perhaps the most well-known. But Mach is certainly not in a class by itself. Other research projects are equally instrumental, among them the V kernel at Stanford University and the Sprite system at UC Berkeley. Stanford has taken a novel approach with V. It is a distributed system that's been designed from scratch and is not intended to be compatible with Unix (although most simple programs can be ported). V's Versatile Message Transaction Protocol (VMTP) is gaining attention because it provides very speedy remote file-read operations. Sprite is a reimplement of BSD 4.3. Its focal point is the file system, which makes a network of workstations appear to the end user as a single machine.

However, operating system design is an international effort, not a North American phenomenon. Europe has made some worthwhile inroads. There is, for instance, the Chorus system, originally developed at INRIA (Institut National de Recherche en Informatique et Automatique) in France, and the Amoeba system, developed at Vrije University and the Centre for Mathematics and Computer Science, both in Amsterdam. The goals of these systems are similar to those of Mach. They support DNC with sophisticated techniques of memory management and process control, and with small, modular kernels. They don't aspire to be fully-featured operating systems; their kernels are minimal, and additional services are implemented outside the kernel (see "Operating System Developments in Europe" pp. 4-5).

The systems we've mentioned (and more abound; these really make up just an operating system sampler) are at various stages of development, and none are ripe. They're all considered experimental systems. Be that as it may, Mach has been pushed into the commercial limelight, mainly because of its affiliations with NeXT.

The following pages serve as a road map through the design of modern operating systems. We refer along the way to Mach as an example of the advances being made in operating system architectures, not because Mach has the best solutions, but because of its commercial impact. In the process, we also wound up with a road map of Mach. But that's okay. Together, the maps should put the future of operating systems into perspective: One shows you where systems are headed; the other shows you one way of getting there.

## The Face of a New Operating System

The progress being made in the realm of operating systems has to do with functionality and structure. As mentioned, DNC is driving functionality. Structurally, the momentum is toward microkernels as opposed to the monolithic environments in place today.

## Functionality

**DISTRIBUTED NETWORK COMPUTING.** The distributed horsepower necessary to develop next-generation, multiprocessing, distributed applications under Unix requires an operating system specifically designed for multiprocessing and distribution. Furthermore, complex, interconnecting computing paradigms make security, data integrity, and reliability critical issues. What is needed is an appropriate balance of distribution and security.

**WHY NOT UNIX?** Frankly, the level of performance in today's Unix isn't high enough for heavy-duty, commercial distributed applications. Most Unix systems don't support thousands of

simultaneous network connections. The typical means of communication in Unix (e.g., pipes, sockets, and signals) are not the most efficient ways to conquer demanding commercial applications.

Furthermore, Unix was written to operate on uniprocessors. Vendors who have pulled off multiprocessor versions of Unix have done so by creating complex extensions that deal with Unix's architecture. As each vendor extends Unix with its own set of services, a lack of consistency emerges. What a system like Mach offers is something (continued on page 6)

---

*The progress being made  
in the realm of operating systems has to  
do with functionality and structure.  
DNC is driving functionality.*

---

# Operating System Developments in Europe

## Snapshot Views of Amoeba and Chorus

**W**E'VE BEEN TALKING a lot about Mach, the most commercially visible U.S. research operating systems. Although Europe's developments don't have the same degree of recognition, they are estimable. In many ways, France's Chorus and Amsterdam's Amoeba are following in the footsteps of Mach. Each system has unique features, but when you look at the three side-by-side, you'll see an overlapping of purpose and design. Each has a system call interface to Unix but also stretches Unix for programmers who want to build more meaty applications. Furthermore, each features a small kernel and caters to modular architectures. Chorus and Amoeba are considered less mature in some aspects than Mach (or V or Sprite, for that matter). All the same, we thought it a good idea to sample these European contributions.

### Distribution with Amoeba

**AMOEBEA DESIGN: AN OVERVIEW.** Amoeba's structure is object based. Each object in an Amoeba system has a "capability," which determines access rights (i.e., who can use the object and what that user can do with it). Every object in the system carries its capability around with it as it moves from

site to site. Objects are manipulated by services scattered throughout the system. However, users need not be concerned with communication implications; the operating system takes care of interprocess communications using transactions.

**The Kernel.** The Amoeba kernel is small and sits on each processor in the network. Like the Chorus nucleus, it's only responsible for multiprocessing and interprocess communication—both locally and globally. The kernel handles transactions, sending and receiving messages (including splitting long messages into packets), server location, setting timers, handling retransmission, etc. Just about everything else (process management, file system, system accounting—all the traditional operating system functions) is done outside the kernel.

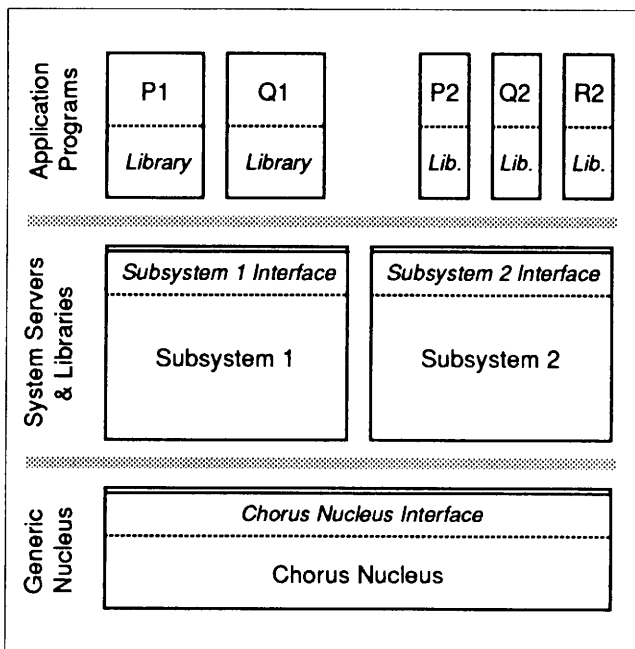
**Transactions.** All interprocess communication in Amoeba is done by transactions. Clients send request messages (which carry the object capability and operation code) to servers that perform the operation and respond with a reply. Processes are divided into subprocesses that enable servers in an Amoeba system to handle multiple requests and enable clients to handle multiple transactions.

**Single Address Space.** The final pertinent characteristic of Amoeba is that it lets multiple processes operate within a single address space. The system uses a task cluster model for implementing multiple threads and to allow servers to work on several requests simultaneously. The model contains:

- Tasks, or lightweight processes
- Clusters of tasks
- Segments, the part of the system's virtual address space in which a cluster executes

The kernel manages clusters and provides communication between tasks. There are no preemptive tasks within a cluster; one task has to be finished before another can begin. This won't quite hack real-time processing, which relies on preemptive capabilities.

**COMMERCIAL IMPACT.** We can't predict how much the solutions found through Amoeba will affect the commercial market. It's still considered very much experimental technology. We assume that future Unix systems will adopt transactional capabilities to better handle distribution. Perhaps Amoeba will stimulate more transactional activity in the industry.



Chorus architecture.

## The Chorus Distributed Operating System

Chorus began life as a research project at INRIA in 1982. However, since 1987, it's been a product of Chorus Systems, where it continues to be refined. Chorus developers have rewritten the operating system four times, searching for better performance and the semantic accuracy of its Unix interface. However, Chorus is not intended to be strapped to Unix. At the moment, it maps to System V, but other interfaces—including one for OS/2—are in the works. Chorus is written mostly in C++, so it's portable and it allows for classes and inheritance. (Perhaps some quick-and-dirty definitions are appropriate here. A class is a general category of similar objects. Objects created within a class, by definition, inherit the basic attributes common to that class.) It's completely distributed; it runs on a variety of multiprocessor; and it has real-time capabilities.

**A SKETCH OF CHORUS.** Chorus has three main characteristics:

- A minimal nucleus (a.k.a. kernel) that can underlie a variety of operating systems, providing distributed processing and communication
- Real-time services with multithreading and fixed-priority preemptive scheduling
- A machine-independent, modular architecture that supports parallel and multiprocessing systems

**Influences.** If you look closely at Chorus, you'll see shades of related research projects. For instance, it handles message-passing, as does Stanford University's V System. Its distributed virtual memory and threads are similar to Mach's, and its network addressing uses ideas from Amoeba.

**Architecture.** The focal point of Chorus is its nucleus, which is basic and controls only distributed processing and communication. The nucleus lies underneath the host operating system and extends it (see illustration at left).

The Chorus architecture is layered, and it separates functions that are usually part of the kernel (i.e., process management, file system, and device management) into an independent set of software services. Once these services split up, the system becomes more modular, portable, and scalable.

At a glance, the Chorus nucleus seems more simple than current Mach kernel implementations. But the Chorus nucleus is overlaid at boot time with System V support code—as are today's versions of Mach—that runs in kernel state and shares memory with the Chorus nucleus. In this sense, running Chorus is not very different from running a commercial Mach system; both have a basic kernel layer and both support Unix compatibility within the kernel (as opposed to the more pure Mach

version, which is completely independent of BSD and shares no memory with it.)

**The Nucleus.** The Chorus nucleus has both local and global responsibilities (see illustration below). On a local level, it has three components:

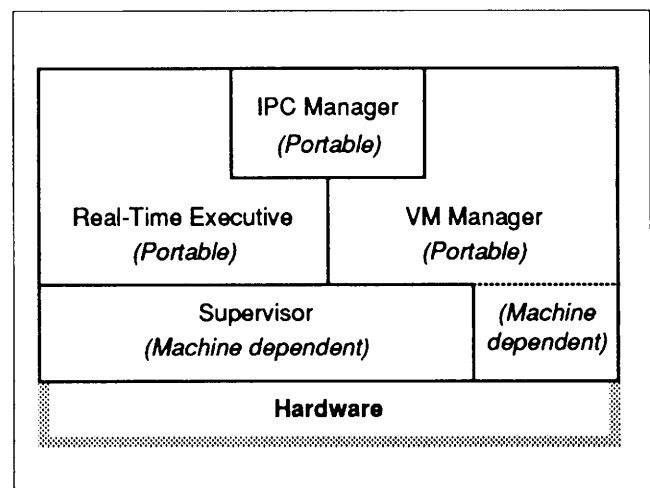
- Real-time Executive, which provides preemptive, fixed-priority scheduling and synchronization
- Virtual Memory Manager (VM Manager), which directs local memory allocation and structures virtual memory address spaces
- Hardware Supervisor, which provides dynamic loading of external events (such as interrupts, traps, and exceptions)

On a global level, there is the IPC Manager, which issues messages transparently to any node in the system. The IPC Manager keeps track of where messages are going and delivers them via RPCs and asynchronous message passes. Sometimes, external system servers are called upon to support various network protocols.

The Chorus nucleus structure hides its distributed nature from those using it. Local services compute locally, and global services rely on the cooperation among nuclei to cope with distribution. The IPC is the only communication tool; all sites use it rather than dedicated protocols.

**LICENSING.** Chorus Systems is licensing Chorus at three levels. Each level includes source code for Chorus V3.1.1, development utilities (i.e., debuggers and the Chorus simulator), user documentation, implementation documentation, and services (maintenance, training, and support).

Chorus Systems is located at 6 Avenue Gustave Eiffel, 78182 Saint-Quentin-En-Yvelines Cedex, France. Telephone 33 (1) 30 57 00 22.



Chorus nucleus structure.

(continued from page 3) less confining. The functionality necessary for distribution and multiprocessing is the very essence of the operating system itself, not an exclusive collection of extensions.

## MACH

### MULTIPROCESSING

Unlike standard Unix environments, where particular processes typically run on single processors, modern operating systems were meant to run in multiprocessor environments. The kernel itself is responsible for distributed processing and interprocess communication. At the core of Mach are two sets of abstractions for multiprocessing (see below). The first set describes how the kernel executes processes and manages memory; the second describes its communication techniques.

#### MACH KERNEL ABSTRACTIONS:

##### Execution And Memory Management

Task:	An address space and collection of system resources
Thread:	The basic unit of execution within a task
Memory object:	Unit of virtual memory that can be mapped into the address space of a task

##### Communication

Port:	A protected communication channel
Message:	A typed collection of data objects

**GETTING THERE FROM HERE.** The key to multiprocessing is parallel processing and better memory management and communication facilities.

**Execution and Memory Management.** The prototype operating systems we looked at have advanced mechanisms for parallel processing and memory management. Memory is treated as a single address space, which means that applications can access memory without knowing it's distributed.

Most Unix configurations don't support parallel processing on shared-memory multiprocessors. They should. For effective DNC, several processes need to run at the same time. Many commercial systems use multithreading (described below) to accomplish this—but Unix systems don't. The multiprocessing Unix systems you see today depend on very com-

plex layers of software outside the kernel. However, most experts agree that parallel processing should be the kernel's responsibility.

## MACH

### PARALLEL PROCESSING AND MEMORY MANAGEMENT

**TASKS.** Mach separates the typical notion of a Unix process into tasks and threads. The task, which encompasses threads, is the primary unit of the system's shared resources. Tasks include sets of virtual addressees along with the access rights to processors and ports.

**THREADS.** Threads run within a task. They are light-weight processes—basic units of CPU—and they can execute simultaneously. In a multiprocessor environment, threads of a single task can execute concurrently, each on a separate processor. All threads within a task have access to the task's physical resources, including files and resource memory. To put the task/thread paradigm in familiar terms, think of a Unix process as a single-threaded task.

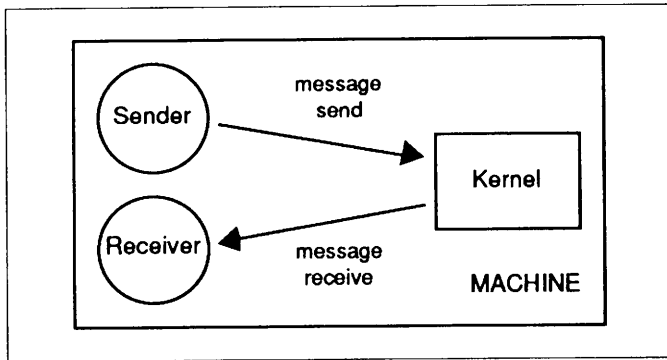
**MEMORY OBJECT.** Included in tasks are sets of virtual addresses along with the access rights to processors and ports. Address spaces are assigned by the operating system to particular tasks. Each address space is represented as a memory object. Data pointed to by a memory object can be provided and managed by either the kernel or a user application.

**Interprocess Communication.** Object-oriented message-passing is apparently the only way to go. There needn't be any knowledge about the content of message objects to pass them between nodes, and it doesn't matter whether an object is local or distributed. A well-implemented IPC mechanism will support this dynamic form of message-passing.

The newly formed Object Management Group (OMG) may help steer the course here. Its goal is to create an object-oriented, application-to-application communication foundation based on distributed computing. Once the foundation is in place, programmers and users can take advantage of modules no matter where they are or who created them. Perhaps we'll soon have the equivalent of integrated circuits for software: morsels of code that can be used over and over again.

Communications must also be secure. In many cases, IPCs tackle security with the concept of object capabilities or access rights, which control the permissions of a particular object: who can operate on it and at what level.

**Virtual Memory.** Virtual memory is certainly no new idea. Support for virtual memory has become (continued on page 8)



Local message communication in Mach.

## MACH MESSAGE-PASSING

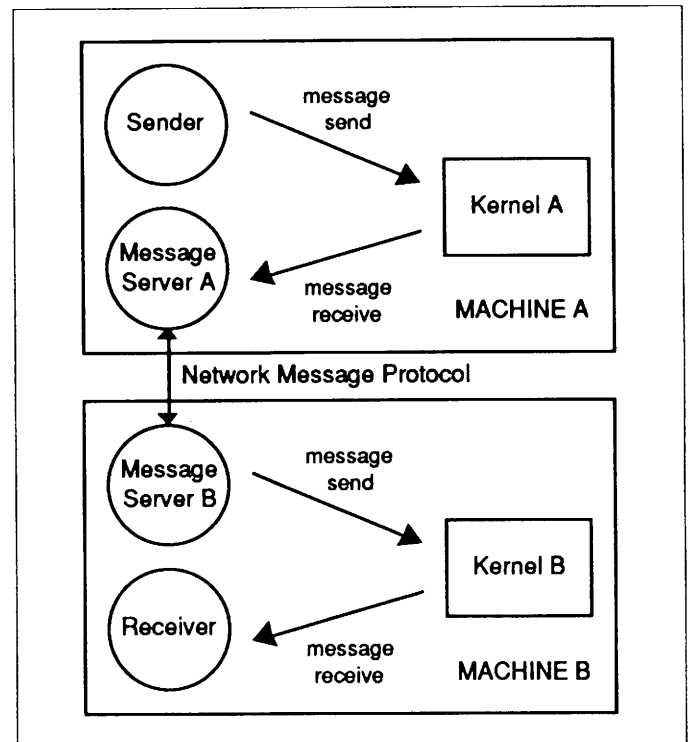
While communication and control processes in Unix take place through a variety of mechanisms such as pipes, signals, pseudo-terminals, and sockets, Mach's approach has been to simplify the landscape. The basic means of message transportation in Mach is the port.

**PORTS.** Ports are Mach's communication vehicles: They send and receive messages. You can think of them as queues for Mach objects (e.g., tasks and threads); a "send" operation from a task adds a message to the queue, and a "receive" operation takes it off. There are also access rights involved with Mach's porting mechanism. For example, to receive a message from a port, a task must have receive rights. Ports can have several senders but only one receiver.

**MESSAGES.** Messages are collections of data: simple data, typed data, or pointers to other objects within the Mach system. Tasks and threads communicate by sending message objects via ports. This encapsulation of data structures controls tasks and threads throughout a multiprocessor or distributed physical environment.

The Mach kernel has no knowledge of networks. As far as the kernel is concerned, messages are always passed between tasks on the same host. Communication is provided by separate message servers that run on each machine (see illustrations above and at right). When a message is sent to a remote port, it's intercepted by the message server and is then forwarded appropriately.

Often, operating systems based on this sort of message-passing have severe efficiency trade-offs. For the most part, Mach has avoided them by extending the facility with virtual memory management (described below). You can move entire files and even address spaces via a single message with all the efficiencies of a single memory remapping.



Network message communication in Mach.

## MACH VIRTUAL MEMORY

Generally speaking, there is usually a very close correspondence between the efficiency of a system's memory management and the physical architecture of the computer. In contrast, Mach makes very few assumptions about underlying memory hardware. Instead, it draws a clear and inviolate distinction between information that must be specific to the computer's architecture and all the other encoded memory management instructions that reside in machine-independent data structures.

This approach results in a number of benefits. One is that Mach deals with memory objects rather than directly with the hardware, and all primary and secondary memory can be dealt with as a single-level store. This means that gigabytes of memory can be mapped as though they were part of the machine's core. Executing tasks and threads can access these addresses—via messages being sent to ports—without the potentially confining and I/O-bound buffer caches of most existing Unix systems. A second benefit of machine independence is the ability to perform within both distributed- and parallel-processing environments.

With Mach, sharing at the page level can occur at write time. Rather than having to copy data into memory, you can logically copy information by using (*continued*)

# Distributed Transaction Processing

## *Mach and the Camelot Project*

**S**HOULD YOU EVER find yourself engrossed in a discussion about modern operating systems (hey, you never know), you'll hear "distributed processing" mentioned over and over again. A technology that supports distributed applications is transaction processing (TP). Transactions help programmers cope with the problems of distributed applications (e.g., failures and concurrency). They also let programs reach data from multiple sites in a single play. Consider, for example, withdrawing money from an out-of-town automatic teller. One transaction executes the withdrawal at that remote site *and* updates your account at home.

You'd think Unix would be great for transaction processing. It's hardware independent; it's a standard operating system; it's easy to program; and it runs on inexpensive terminals instead of special purpose machines.

But Unix isn't robust enough to take on the demands of distributed TP. Yes, there are Unix implementations of TP, but it's only been managed by modifying the operating system. Basic Unix doesn't have sufficient large-scale, commercial network capabilities, and crash recovery can be slow. Furthermore, Unix is single-threaded and doesn't support parallel processing on shared-memory multiprocessors. In essence, Unix just needs too much software (e.g., for permanent data, crash recovery, transactional locking, security, and distributed operations) to efficiently support distributed transaction management.

### **MACH**

#### **VIRTUAL MEMORY (continued)**

pointers. In essence, this becomes virtual memory for multiprocessors. Only pages that are changed are copied, and only data that is specifically called is transferred to another location. This process cuts down on time and memory.

(continued from page 6) an important element of several operating systems, among them IBM's MVS and VM, Digital's VMS, and Apollo's Aegis. Currently, both System V and most BSD variations of Unix also offer varying degrees of memory-sharing. In BSD 4.3, you can only share a copy of a program. System V's virtual memory is more flexible. Users can map part of a file into the address space rather than bringing it into the buffer memory.

**CAMELOT.** The research community is making significant developments to pave the way for commercial TP under Unix. To wit: the Camelot project, a three-year-old distributed transaction processing facility being developed at Carnegie Mellon. Its designers set out to build a flexible, high-performance facility not only for typical commercial TP applications (like hotel reservation or automatic teller systems), but also for an assortment of distributed applications that access shared data.

Camelot developers began with some notions about transaction facilities:

- Distributed operation is a must.
- Your basic Joe Programmer shouldn't have any problem implementing them.
- TP should accommodate existing applications and standard, open computing environments—you know, the typical mishmash of hardware and networks you find in most organizations.
- TP should support user-defined shared objects and nested transactions to maintain parallelism and to control failure pitfalls.
- Transactions should execute with increased performance.

## Structure

**THE MICROKERNEL.** We could describe the structure of a modern operating system in one word: small. Next-generation kernels offer only minimal functionality for distribution: multiprocessing and interprocess communications. That way, they don't hog all the room in your system. This is completely unlike most of today's Unix systems, where additional functionality has been pumped into the kernel. Unix has picked up a lot of excess baggage over the years. Furthermore, as each vendor has made extensions to Unix, it has lost its consistency. Each implementation has its own idiosyncrasies and winds up looking to the end user more like a proprietary system.

**MODULARITY.** Modularity is the real goal of a smaller kernel. Services live outside the kernel as objects instead of being contained in the kernel itself. The system is then faster and much more maintainable; each service is implemented as an



## Distributed Transaction Processing (continued)

- TP shouldn't be confined to costly, special purpose machines.

Camelot takes care of recovery, synchronization, and communication in distributed transactions. It provides simple interfaces (via macros and C library calls) for running transactions and defines servers that encapsulate potentially large permanent data objects.

**How Mach Fits In.** Camelot doesn't use Unix as its foundation. It uses Mach, and it runs on all the machines that Mach supports. Thus, Camelot has the advantages of Unix (its low cost and wide availability) without getting muddled in performance problems. Camelot developers didn't mess with the Mach kernel; you might say that doing so would have been against their ethics as software writers. (Actually, commercial operating systems with Mach-like features such as AIX and SunOS would need very little tweaking to run a system like Camelot.) Instead, Camelot is a collection of facilities layered outside the operating system.

Mach gives Camelot its basic building blocks for distributed applications:

- Mach's tasks and threads support parallel processing on uni- and multiprocessors.
- Interprocess communication is handled by Mach. Thus, message-passing in Camelot has the advantages of location independence, security, and data-type identification.

individual chunk that's easy to manage and debug. Modularity also allows a module to be reused in various systems and thus saves design, programming, and maintenance cost.

Professor Alfred Spector of Carnegie Mellon University (CMU) has devoted substantial energy to the design of distributed environments, including the Camelot project (see box above). He's among many modularity proponents, suggesting that the real significance of Mach lies in its potential to become a very clean kernel with services layered outside. In fact, Spector has devised a layering model for distributed systems (see illustration, p. 10). His paradigm is conceptual, but it should give you an indication of what systems will look like in the not-too-distant future.

**FLEXIBILITY: OPERATING SYSTEM KITS.** Customizability is another advantage of modularity. Ideally, you can add, change, or remove whatever services you want. Expanding on

- Camelot uses Mach's memory manager for write-ahead logging (WAL), which helps in recovery. (Basically, WAL lets crucial portions of the recovery log be updated in advance.)
- Camelot programs communicate both locally and remotely via Mach RPCs to shared servers.
- Mach allows for virtual shared memory among tasks.

**CONCLUSION.** Camelot's impact lies not only in online transaction processing, but also in distributed programming. TP contains features like failure atomicity (i.e., an "undo" feature; should the hardware crash in the midst of a transaction, it won't go through only partially completed), permanence, and synchronization which help programmers deal with the data integrity, concurrency, and failure issues inherent in distributed applications. Thus, software writers can concentrate on the application at hand and let the transaction facility take care of distributed operations. Camelot is a model for implementing distributed transactions in a Unix-like environment. Its developers suspect that, in time, all Unix systems will use transactions, which will simplify the whole process of developing distributed applications.

(As we went to press, we learned that Camelot has helped spawn a new software company. And it's good to see a worthwhile piece of research being adapted for the commercial market. See "Beyond Academia" in the news section.)

that theme, Dr. Ira Goldstein, vice president of Research and Advanced Development at OSF, brought up the notion of operating system kits. In essence, you can make your system behave like any existing operating system by adding the proper set of services. For instance, you could set up a system that imitates Berkeley Unix or System V by choosing the right kit. Interesting idea. However, common implementation technology needs to evolve before we'll see it happen.

If we are to reach the point where we can mix and match and reuse modules on various systems, we need full suites of services (like the model in the illustration, p. 10) that map to a slew of systems, and that's one of the hurdles of newer operating system designs. Without them, such streamlined kernels are limited. As a result, current implementers of Mach, like NeXT, Encore, and Mt. Xinu, are BSD derivatives; lots of Unix code is still in the kernel.

**MACH** MODULARITY

There are two possible paths for the growth and continued development of the Mach operating system. The more expedient route has been the one taken by the earlier adopters of Mach, such as NeXT and BBN. For these, the procedure was, in essence, to go into the current BSD 4.3 Unix kernel and selectively replace chunks of code with Mach kernel coding. The result is a hybrid operating system that offers a sizable share of Mach's increased functionality and efficiency as well as compatibility (to a bug-for-bug level of compatibility) with existing BSD standards.

There is an alternate, more ambitious path that this operating system is taking. It attempts to provide a set of powerful, device-independent primitives for (continued)

**MACH** MODULARITY (continued)

data communication and manipulation—a groundwork for hardware and software evolution. The kernel should be very small, much smaller, in fact, than Unix's kernel. The system's extensibility will rest on having many complex functions that are typically associated with the kernel reside outside that kernel as objects to which messages are passed. With this version, BSD runs completely independent of the Mach kernel and shares no memory with it.

While there are no commercial products built around this version of the kernel, CMU does have a full, binary-compatible BSD 4.3 environment running on top of it. It should go into production use at CMU later this year and is receiving attention from commercial vendors.

**A SCHEMATIC OF "PURE" MACH LAYERING:**

<b>User processes</b>	user interface system services network protocols Unix compatibility file system
<b>Kernel</b>	virtual memory management interprocess communication multiprocessor scheduling

Advanced services (SQL, OODB, etc.)	■ ▲
Advanced languages and run-time	■ ◆
Advanced operating system compatibility	■
File system and recoverable storage	■ ▲
Node and service management	▲
Directory	■ ▲
Local and distributed data types	■
Protection	■
Authentication	■ ▲
Recovery	■
Locking	■
Transaction	▲
Logging	▲
Internode cross-domain invocation	■ ◆
Stream and datagram transport	▲
Reusable abstractions	■
Primitive compatibility	■
Cross-domain invocation	■ ◆
Machine-dependent modules	▲
Device-dependent modules	
Kernel: thread, address space, IPC	
Primitive language and run-time	■ ◆
■ Library      ◆ Tool ▲ Process or Protocol Abstraction	

**MACH** LICENSING

Mach is licensed by CMU. There is no charge for the license and no royalty or distribution fee. Mach can be obtained through the CMU Mach Project, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

**Conclusion: Reality Check**

You'll be hard-pressed to find an operating system quite like the one we've defined. Even so-called modern prototypes are really only research systems. However, this is the way we expect Unix to evolve. And commercial systems are taking notice. Some vendors, like Data General, Arix, and Hewlett-Packard have made significant advances in multiprocessing—even if they don't handle multiprocessing inside the kernel. More systems are offering preemptive scheduling and better virtual memory mechanisms, and modularity is becoming more of a focal point.

**A SMALLER KERNEL IS NOT ENOUGH.** Although the prevailing design tends toward a smaller kernel, focusing only on

System architecture model as seen by Carnegie Mellon professor Alfred Spector. Layers may call layers anywhere below them, and some layers may make upcalls to layers above.

that is a mistake. We've talked about the lack of services. The kind of modular, consistent services that minimal kernels need is not available today. We're in somewhat of a Catch-22. Unix has become a dumping ground for features that are not conveniently implemented outside the kernel. Since the kernel handles them, why would they exist as separate entities? However, without them, these very basic kernels simply don't contain enough functionality.

In the meantime, current Mach implementers have gutted parts of their operating systems and replaced them with Mach code. This is a convenient and practical approach, but it's not really an ideal situation. In fact, Sun's Bill Joy maintains that Mach has been "violently oversold and is bigger, slower, and barely more functional than System V.4. Actually, in many ways, it is less functional." Of course, that's Bill Joy, but he does have a point. He added that the real issues in distributed commercial programs have to do with applications, not operating systems. And he's right. But before the applications appear, the proper languages and tools need to be available.

**LANGUAGES AND TOOLS.** The real catalyst behind next-generation applications will be programming languages. The kind of software we look forward to requires languages that specifically address distribution and object orientation. And they're out there. You hear a lot about SmallTalk and C++, but there are others—Eiffel and Actor, for instance. MIT's Argus and CMU's Avalon are languages aimed at distributed object management. The continued refinement and proliferation of these kinds of languages are really critical to application evolution. Furthermore, programmers need to think in these languages. It doesn't take long to learn C++, but it may take months before its programming principles sink in.

Finally, we need migration tools: tools that ease the transition to applications written in these next-generation languages, tools that help developers effectively distribute existing programs, tools that manage distributed objects. The operating system is just a foundation. What we need are developments beyond the kernel. ●

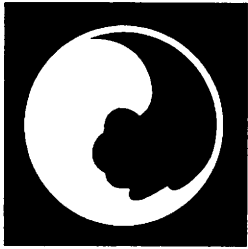
## Reprints!



Companies featured in *Office Computing Report*, *Unix in the Office*, *Network Monitor*, or *P.S. postscript on information technology*\* are often interested in obtaining large quantities of specific articles in

which their companies appear. We reprint the articles in the same style as the newsletter in quantities of 100 or more. If you are interested in reprints, please call Mary Treleven at (617) 742-5200 for estimates.

\*We also provide reduplication services for the cassette tape part of *P.S. postscript on technology*.



## Announcing the Seventh Annual **SEYBOLD EXECUTIVE FORUM**

### *Preparing for the Global Information Age*

#### **OVERVIEW**

As computers and communications weave their web around the world, companies are scrambling for position in the new global information marketplace.

- How will businesses be impacted by interconnected networks with their customers, their suppliers, and even their competitors?
- How will these networks be managed?
- How will organizations be structured?
- Who will be the purveyors of information in the new global information marketplace?
- What form will these services take—databases, computer conferences, videotex, object bases?
- How will the data and telecommunications infrastructure evolve?

Join us as we bring together the top executives from the computer and communications industries along with top executives, technologists, and business planners from a variety of industries to explore the issues and prepare for the '90s.

#### **WHO SHOULD ATTEND?**

**Executives** concerned about preparing their organizations for the '90s. People involved in planning corporate strategies for information architecture and information systems. **Marketing Executives** seeking to understand the impact of technology on the products they market. **Financial Executives** needing a better grounding in the technology directions of the next decade. **Line Executives** who need to understand how their current decisions may be affected by changing technologies. **Business People** who need to know how fast global markets will develop and what technology infrastructures will be in place in what timeframes.

#### **COMPANIES INVITED TO PRESENT INCLUDE:**

Apple Computer	MCI
AT&T	NCR
Boeing	NeXT
British Telecom	Northern Telecom
Citicorp	Olivetti
Digital Equipment	Philips
Federal Express	Siemens
General Electric	Sun Microsystems
Hewlett-Packard	Time-Life/Warner
IBM	Unisys

**October 30, 31,  
November 1, 1989**

The Royal Sonesta  
Hotel, Cambridge,  
Massachusetts

*Sponsored by:*  
Patricia Seybold's  
Office Computing Group

#### **REGISTRATION**

- \$ 895 for registrations that are *paid* by July 14, 1989
- \$1095 for registrations received after July 14, 1989

Advance registration is required. To register, call Deborah Hay at 1-800-826-2424 (in Mass. call 617-742-5200) or send a fax to 617-742-1028.

## • FORUM UPDATE •

# Commercial Unix

## The Moment of Truth

By John R. Rymer

It's no longer a question of whether or not Unix will become a commercial operating system. Unix is an operating system of commercial quality today, as a growing roster of corporate users attests. But when will Unix take its place alongside IBM's MVS and Digital's VMS as a major commercial player?

"On the Road to Commercial Unix," the first Executive UniForum Symposium (April 26-28), provided some answers. The symposium, cosponsored by Patricia Seybold's Office Computing Group and /usr/group, the international Unix organization, brought together 40 speakers and 200 attendees to chart Unix's course toward widespread commercial use, identifying barriers as well as fast lanes.

The symposium arrived at the following conclusions:

- Unix is a viable commercial operating system today.
- Unix's position as the centerpiece of a growing open systems movement is the source of much confusion and misinformation. Openness is, in fact, limited to the choice of hardware. Also, preoccupation with standards may be impeding innovation.
- The economics of Unix has prevented innovation in applications at a level equal to that in the DOS world. A key factor is that users aren't backing their demands for standards-based systems with their budgets.
- The split between the Open Software Foundation (OSF) and Unix International (UI) is rapidly becoming a mere political sideshow. The stage is set for the two bodies to, at minimum, cooperate. X/Open's emergence as a clearing-house of standards—de jure and de facto alike—is promoting convergence on a single set of standards in the Unix world.

- Unix will become an important enabling technology for distributed network computing, object orientation, and other new technologies.

**BEYOND VIABILITY IS MARKET ACCEPTANCE.** It's useful to remember that Unix is 20 years old. This fact implies a maturity that OS/2 certainly doesn't have. It also implies, as noted by Bill Joy, Sun Microsystems' vice president of R&D, that Unix is huge. "Unix has too much [function]," he said. "Ideally, I'd like a white wall to hang my picture on." Joy's dream of starting over with an object-oriented version of Unix has been put on indefinite hold, because the Unix International consortium has become the guardian of the future of AT&T's System V Unix.

Acceptance by commercial users hasn't been easy for Unix. Pete Peterson, CEO of WordPerfect, characterized Unix with an extended metaphor: "If your daughter brought home a boyfriend and the boyfriend was DOS, he'd be a nice kid with a terminal disease. If he was OS/2, he'd be nice but still looking for a job. If he was a proprietary system, he would have been born with a silver spoon in his mouth and never have had to work very hard for anything. If he was Unix, he'd be a hard worker, and his suit would be frayed at the edges."

So it is for Unix—and for its users, who struggle to come to grips with Unix's rapid progress in the commercial world. Timing is tricky. Salomon Brothers, the big Wall Street firm, was burned two years ago when it made a commitment to IBM's Unix program before the program was mature, related Peter L. Bloom, senior operating officer. Steven A. Ruegnitz, director of Unisys Corporation's Open Systems Marketing Team, remembered jumping through technical hoops while on the U.S. Army's systems staff to meet communications requirements under Unix. There were easier ways in proprietary environments, but, in the Army, you have two choices: Unix or Unix.

Ruegnitz had no choice, but most users do. Still, they too often underestimate the effort needed to implement Unix in commercial applications. There are barriers at every level. Judith Hurwitz, editor-in-chief of *Unix in the Office*, noted that professionals in the typical commercial

data processing shop are afraid of Unix. "They've become accustomed to thinking of the Unix operating system as a scientific and academic operating system not fit to work in the real world," she says.

At the user level, said Paul Ely, president of the Network Computing Group at Unisys, Unix is a shock. "When you approach it, it acts like you're bothering it," said Ely. "It is the misbehaving child of the operating systems world."

**REPORTS FROM THE FRONT.** The fact is that converting a business to Unix is a major effort. William H. Pigott, vice president of MIS at DHL Worldwide Express, is happy with the Unix systems he oversees today. DHL has Pyramid systems in the United States, Pyramid and NCR in the United Kingdom and Africa, NCR in Latin America, and Hewlett-Packard in Europe. Applications can be easily moved to any of these platforms.

But this environment was five years in the making. It all started when DHL's largest shareholder mandated Unix as the worldwide corporate standard. Pigott's staff began writing lots of C code to build under Unix DHL's core business applications. Pigott now says this was a bad idea. Use a fourth-generation language (4GL) instead, he advises. The additional cycles you burn are less expensive than C programmer time.

Clearly, packaged software can reduce the implementation pain of Unix. Applications are plentiful in some specialized areas. Geoffrey K. McDowell, manager of MIS at Four Seasons Hotels Limited, had some 30 Unix packages to choose from when he initiated a three-year program to replace the hotel chain's back office accounting systems. The chain also wrote some of its own software with Zanthé's Zim 4GL. Still, McDowell cautioned, Unix does require more in-house expertise.

Vendors that help will find a receptive market. Four Seasons Hotels uses Hewlett-Packard (HP) equipment and is "smothered" with support, McDowell reports. Even so, however, HP's Unix support at first operated at a deep technical level, which was not appropriate for his staff. HP is adjusting.

**THE POWER SURGE IN UNIX.** Unix users do not sacrifice anything in terms of platform power. Representatives of Pyramid Technology, Sequent, Arix, and Data General assured UniForum Symposium attendees that Unix supports powerful machines. Each of the vendors is migrating from a host-based model of computing to a model that al-

lows workstations to access their machines as servers on a network. Each is investing heavily in networking to support this shift.

These vendors are delivering impressive platforms under Unix. Pyramid's new MIServer is rated at 140 MIPS, and supports up to 64 GB of disk storage. Sequent stresses its performance in relational database applications. Arix's design features independent I/O processors and can be implemented in a variety of chips, including RISC (Reduced Instruction Set Computers) chips.

RISC is the new force in the platform arena. Data General's new processors are based on the Motorola 88000, and Arix can utilize RISC designs. Sun's Bill Joy noted that we've only begun to realize the potential of RISC. The coming of binary interfaces for Sun's SPARC and the 88000 varieties improve software portability. And the power of RISC chips will enable innovation in new applications, said Joy.

**OPEN SYSTEMS: CONVERGENCE AND CONFLICT.** Unix and the open systems movement are closely associated. That's good. But too many users miss the distinction between Unix, the architecture at the heart of the standards-based environment, and the "Unix of a hundred variants" that exists in the real world. This is a potential barrier to commercial acceptance because it raises user expectations. In fact, Unix users do not deal with the cohesive, monolithic entity suggested by the existence of the Posix interface definitions, noted Dr. Pamela Gray, a /usr/group director.

Posix contributes much to Unix's evolution. It is a rigorously defined standard that OSF and Unix International have both agreed to follow. Long-term, Posix also will open up some proprietary operating systems. William Heffner, director of software systems at Digital Equipment, told of a formal plan to allow VMS to service Posix system calls. Completion date? Years from now.

**X/Open: The Best Hope.** But Posix is also a political document. Its many options actually don't help promote convergence in the various implementations of Unix. Fortunately, X/Open's Common Applications Environment (CAE) and the Federal Information Processing Standard (FIPS) based on Posix help by selecting from among Posix's many options. Gray singled out the X/Open Consortium as the best hope for convergence upon a single set of standards in the Unix world.

There are several reasons why X/Open plays a special role in the standards process. First, its

board is balanced between OSF and UI members, with Nokia Data of Finland remaining neutral. Both OSF and UI have agreed to follow the formal and de facto standards recognized by X/Open. (This balance could be threatened by HP's recent acquisition of Apollo Computer. If Apollo ceases to be an OSF board member, OSF will be one vote short of a balance with Unix International on the board. At press time, the situation hadn't been resolved one way or the other.) On a positive note, both organizations have become X/Open members.

Also, X/Open is moving aggressively to become the clearinghouse for Unix user requirements, said Bill Bonin, vice president for North American operations. X/Open's 20-member international User Advisory Council meets quarterly and articulates a direction for Unix development in the commercial market.

**Openness Conflict #1.** Despite these standards, the user seeking openness via Unix confronts a central conflict between one vendor's real value-added and another's superficial differentiators. Paradoxically, too many Unix extensions blow away the user's freedom of choice.

Vendors must innovate because Unix doesn't answer every need. "Today's innovation is tomorrow's standard," offered Digital's Bill Heffner. "We add value to push the state of the art," said William Filip, vice president of the advanced workstation division of IBM. And so, for example, Gould has added important security extensions, and HP has spearheaded internationalization technology. Some of these preserve the user's option to select from many hardware vendors, some don't. Users must acknowledge that vendors will always seek proprietary leverage over standards; no vendor willingly becomes a supplier of commodities.

The solution to this paradox differs for each situation. Users must decide, based on corporate requirements, the degree to which openness is important. Proprietary lock-ins are rife in the Unix market, noted Michael Banahan, president of The Instruction Set Limited, but they may or may not be important in an individual user's scheme of things. If the corporate requirement is security and the best fit is a proprietary extension, it may make sense for a user to adopt it anyway. Keep going back to your requirements, Banahan cautioned users.

**Openness Conflict #2.** The open systems movement stalls after providing users with a choice of

hardware platforms. Unix software isn't open—particularly DBMSs and high-level programming languages. Several users among the attendees expressed frustration at this state of affairs. But representatives of three leading relational DBMS vendors—Oracle, Relational Technology, and Sybase—offered no encouragement to those who hope for the suite of standards needed to make DBMSs from different vendors interoperable.

The situation in 4GLs illustrates the point. An organization chooses Unix because it wants an operating environment that is owned by no single vendor. When the same organization selects a 4GL to speed development of applications, its systems cease being open. Every 4GL is DBMS-specific, and so our hypothetical organization has been locked in—despite having selected Unix.

This is "lock-in" at a level that's higher than the operating system, but it's lock-in nonetheless. 4GLs are the most glaring example, but not the only one in the Unix DBMS market. The DBMS vendors acknowledged that current standards don't support interoperating, heterogeneous DBMSs on a network. SQL alone doesn't provide all of what's needed, and standards that will are in their infancy. The Open Desktop offering from the Santa Cruz Operation (SCO) implements a version of the international Remote Data Access (RDA) standard, called General Communications Architecture. But RDA is not complete yet. Return and error codes, dictionaries, and catalogues are other areas where standards can promote interoperability of databases from several vendors.

But don't expect new standards in these areas soon. Because of fierce competition, major DBMS vendors see no compelling reason to agree on them. Jerry Baker, vice president of Oracle's Unix division, noted that his division did \$100 million in Unix business last year, a 200 percent increase. Growth this year is slowing—to about 100 percent. Oracle just doesn't encounter demands for heterogeneous databases, according to Baker. "Corporations usually standardize on Oracle, and that's that." For the next few years, gateways will be the most prevalent method of connecting different brands of DBMSs.

New standards issues will make the database arena an open systems cul-de-sac for some time. Each of the DBMS vendors at the symposium portrayed transaction processing services, software engineering tools and methods, and support for object-oriented designs as development priorities. Each of these areas requires value added atop the Unix standard.

For example, objects comprise data plus their

semantics, a form that relational tables are not adept at storing. James Black, vice president of New Systems and Environments at RTI, noted that, at minimum, the relational vendors need to expand their catalogues to accept new data types from the object world. They'll also need to define a new query standard to augment or replace SQL. SQL doesn't understand the semantic content of an object. International standards bodies are working on an extension to SQL, but this is a long-term effort. In the meantime, each vendor is working on its own extension to SQL and its own extended catalogues, noted Mark Hoffman, president of Sybase Incorporated.

Patricia Seybold, president of the Office Computing Group, asked the trio if they'd be willing to work with IBM to define a common standard based on SQL. (IBM originally defined SQL.) Only Black displayed any willingness to do so.

**OSF VERSUS UI.** The competition between OSF and UI was worrisome to attendees. The big question was how seriously OSF and its backers would disrupt the progress Unix has been making via AT&T's System V. Moreover, some attendees wondered if OSF's output would be of any consequence in the evolution of Unix.

Digital's Heffner and IBM's Filip both fielded questions about their commitments to Unix. Heffner said VMS and Ultrix have equal resource commitments. Filip revealed that Michael Serranga, former assistant chief of IBM's SAA organization, is now working on AIX. His priority: convergence of SAA and AIX. IBM's goal in office applications is a common set of functions for SAA and AIX, with interconnection between the two environments.

Attendees also wanted to know about the plans at IBM and Digital to incorporate OSF/1, OSF's first operating system release, into their brands of Unix. Heffner said Digital will put pieces of OSF's releases into its Ultrix over time. "The [OSF] virtual memory system is in flux," Heffner noted. "It is impractical to implement OSF/1 wholesale." According to Filip, IBM OSF/1 and AIX 3.0 won't differ in important ways at first, and IBM will move AIX along parallel with OSF's releases, mapping AIX to OSF. Since OSF/1 won't be available to vendors until the end of the year, users won't learn the results of individual vendor's mapping processes until well into 1990.

**The GUI Cauldron.** The biggest question about OSF concerned the impact of its Motif graphical

user interface (GUI). Though based on toolkit technology from Digital and HP, Motif has the look and feel of Presentation Manager, the graphical interface for OS/2. This commonality raises the possibility of a common interface across industry platforms as diverse as Unix, OS/2, and even IBM's proprietary systems, which will use Presentation Manager as their interface base.

OSF Motif competes with Open Look, which was developed by Sun Microsystems and is backed by AT&T. Lt. Col. Terry Elton, communications and computer systems manager at the U.S. Air Force, complained that the existence of two standards complicates federal purchasing and could impede applications progress.

Filip, stressing convergence, said Motif and Presentation Manager will evolve in lockstep. This common direction isn't as clean as it might be, however. Lurking in the shadows of IBM's planning is NextStep, the interface technology it licensed from NeXT Computer for its AIX platforms. Don't expect any action on NextStep, said Filip, until IBM's negotiations with NeXT's Steve Jobs are concluded. For Digital, the issue is simple: Motif will be the interface on all of its systems, said Heffner.

Will Motif beat out Open Look? This is a question of survival for software developers, although it actually may not be particularly important to users. Graphical user interface is the Unix developer's most critical issue, said X/Open's Bill Bonin, because so much code—half of the typical application's source code—is required just to support the graphical environment. Bonin predicted that "the market" will decide on an interface this year, and most attendees seemed inclined to agree.

The user view of the interface question is different, however. Sun's Bill Joy offered that the interface wars are "much ado about not much. A common user interface [across platforms] doesn't matter," he said. "You can learn a new interface within one minute." What's really important is giving users interfaces that don't call attention to themselves. Little things count, said Joy. The shading employed by NextStep to separate active and passive windows, for example, is very helpful.

Things will get even more interesting in graphical interface. Motif and Open Look are just window systems—platforms, if you will. HP, for one, plans to build additional capabilities atop Motif with its NewWave product, said Willem P. Roelandts, general manager of HP's General Systems Group. NewWave is a window content manager that introduces object-oriented capabilities to a window manager like Motif. Expect NewWave



to be at the heart of innovative network management software products from HP.

**OSF-UI Convergence.** For every worry expressed about the split in the Unix world, however, a voice was raised citing the convergence of the two camps. X/Open certainly holds this promise. And Robert Kavner, president of AT&T's Data Systems Group, held out the possibility of cooperation and even a merger between OSF and UI.

AT&T's recent moves have satisfied the demands of licensees for stable, predictable licensing terms and an impartial development organization open to licensee input. Kavner conceded that OSF was born of AT&T's management blunders with Unix. By allying with Sun, AT&T brought "Dennis the Menace" (Bill Joy and Company) into power, he noted. "Sometimes I feel like Mr. Wilson," quipped Kavner.

The stage is set for OSF and UI to get together. Kavner said they're still talking, and hopes the two can reach an accommodation. Joy blamed "Sun bashing" for the failure of a previous merger proposal, but was confident some accommodation is in the offing. "These things are like the football franchises," he said. "To get accepted by the NFL, you have to first form the AFL, and then get absorbed by the NFL."

Unisys may emerge as the broker of any accommodation. "I'm not sure the OSF-Unix International split is a disaster," said Paul Ely, "but we want a single industry standard." Watch what Unisys and AT&T do with Motif. If they accept Motif as their standard, it will be as a gesture of offering OSF an olive branch, Ely suggested. Unisys's X/Open representatives already support Motif over Open Look, he revealed.

The OSF-UI split was a major concern among attendees at the beginning of the symposium, but, by the second day, it had faded. In a survey of attendees taken on the second day, 65 percent said they didn't care when asked if OSF and UI should merge.

**STANDARDS VERSUS INNOVATION.** Bill Joy was annoyed by the conference's focus on wider standardization as a market-driving force. "The marketplace is not driven by committees," he said. "The marketplace drives itself." The touchstone for developers in the Unix market should be what business users need to be more competitive, not efforts to achieve wider open-system standards. Standards today allow users to choose multiple hardware vendors, and that's enough, offered Joy.

Joy is building applications that don't exist

today, focusing on teleconferencing and voice integration and on an application environment that supersedes the operating system in importance. "I'll go and build something great, but proposing it as a standard may ruin it," said Joy.

Joy's point of view on standards was not widely shared. But his emphasis on innovation struck a nerve with users at the symposium. Users want innovation, and they're dissatisfied with what they find in Unix applications for the office environment. Indeed, Joy's approach was 180 degrees out of phase with the approach of the other vendors who discussed Unix office applications.

Unix office software is dull. Karl Klessig, president of Quadratron Systems Incorporated, and Jeffrey Waxman, president of Uniplex Integration Systems, revealed why.

The state of the art in DOS- and proprietary-based office software—graphical user interfaces, procedural automation, compound documents, voice integration—is not the state of the market, Waxman told attendees. Functions like procedural automation and compound documents will be requirements three to four years hence, he said. For the present, Uniplex and Quadratron will provide baseline functions within individual program modules—like word processing—and strong integration with other modules.

Klessig and Waxman butted heads with Randy Griffin, a vice president at Computer Sciences Corporation (CSC), a big systems integrator, on what users really require. CSC is making a nice business of integrating the most functional individual software packages from different vendors. Griffin's customers aren't willing to settle for the lowest common denominator offered by Quadratron and Uniplex. Yet Waxman and Klessig held out no hope for innovation in their product lines.

What's going on here? Why don't Unix software vendors value innovation as much as DOS vendors do? A big reason is the costs Quadratron and Uniplex bear as major players in Unix. Both spend considerable amounts of money just porting their software to different versions of Unix. Both vendors value having a wide variety of Unix platforms to run on, and Klessig noted that his company puts a lot of effort into data interchange software.

**The Fed Factor.** And then there is "the federal factor." To listen to federal systems managers, government users don't particularly care about advanced office features. As Steven Ruegnitz puts it, 80 percent of the users use 20 percent of the features. However, a closer look at this situation re-

veals that federal procurement rules virtually prohibit federal users from obtaining features like procedural automation. To specify procedural automation in a contract, a federal department has to be able to obtain it from at least three sources, said Ruegnitz. New features generally aren't available from three sources. So, ironically, at the same time the federal government is driving the progress of Unix, it is retarding innovation in Unix applications software!

But the feds are only one factor in the market, albeit a large one. If Unix is to achieve widespread commercial acceptance, office applications software written for it is going to have to improve beyond today's character-based interfaces and features, which are nothing special. Symposium attendees—even one from the U.S. Army—made it clear that they'd use advanced features like group calendaring, procedural automation, and compound documents if they could get them. There's a clear opportunity for innovators.

Conventional wisdom says DOS application vendors will inject new ideas into the Unix market. While this is probably true, Pete Peterson, CEO of WordPerfect, raised the question of how far DOS vendors will go with Unix. When asked what innovations WordPerfect would bring to the Unix market, Peterson replied, "Slick manuals and support." These are important, but note that graphical user interface support is low on WordPerfect's priority list, to say nothing of procedural automation, compound documents, and so forth. Peterson did promise support for Motif, Open Look, and NextStep, but downplayed the importance of graphical interfaces in general.

**UNIX THE ENABLER.** The state of office applications innovation under Unix stands in dismal contrast to the expectations for the operating system. Michael Millikin, vice president and chief technologist for the Seybold Office Computing Group, asserted that Unix will be at the heart of a broad migration to distributed network computing. This means transparent distribution of resources across servers and workstations on networks that are scalable, flexible, and easy to use.

Millikin sees Unix servers as well as proprietary servers on these networks. And he noted that important common technologies are emerging, particularly LAN Manager for both OS/2 and Unix.

Paul Ely of Unisys endorsed this idea with special emphasis on the need for distributed networks to support existing proprietary systems.

Unisys is seeking rapid movement toward the distributed model in both its Unix investments and its development of its mainframe architectures.

Ely believes object-oriented software design will be an important part of achieving the potential of distributed network computing. "We need to be able to build applications easily, and object-oriented programming languages can help," he said. For example, network resources like mail systems, fax servers, and printers should be available to users as black boxes. Users should be able to send a document to a black box without having to worry about how it works.

Unisys joined the Object Management Group (OMG), an industry consortium formed to promote standards in object applications, to help accelerate development in this area, said Ely.

Object orientation also plays a big role in Bill Joy's thinking as he crafts his application environment. The notion that users can get at data not by mucking with it but by asking it questions is powerful in its simplicity, he told attendees. Joy's other major goal: tailorable user interfaces. Users need a wider range of interface metaphors, he said, and they won't get very far if window systems keep thrusting their controls into the foreground.

**CONCLUSIONS.** The first Executive UniForum was loaded with ironies. Is today's innovation tomorrow's standard or tomorrow's lock-in? Users make openness a corporate objective and get locked in by software. And economics forces applications software functionality to the lowest common denominator.

Each of these illustrates important barriers Unix must overcome before it becomes a commercial heavyweight.

First, agreement on more and wider standards will help accelerate the acceptance of Unix. The fact that openness is so limited is not lost on commercial users. The major benefit of moving to Unix is freedom of choice. But hardware is the only domain where this freedom is available, and that's not a wide enough benefit. Wider standards will also reduce the threat of vendor lock-in.

Second, innovation in Unix software must accelerate. Commercial users have been seduced by the flashy qualities of DOS applications. They won't rush to settle for less. A crowd of Unix aficionados confidently dismisses OS/2 as too little too late. This was an oft-heard remark at the symposium, and it signals an overconfidence on the part of Unix vendors. This, in and of itself, is the highest hurdle commercial Unix must overcome. ●

# INTRODUCING

# P.S.

*postscript on information technology*

## Patricia Seybold Launches a New AudioNewsletter

### CURRENT TRENDS IN NONTECHNICAL TERMS

This innovative monthly service, both an audiocassette and a printed newsletter, is designed for you, the technology watcher, and for your nontechnical colleagues. It comes to you from Patricia B. Seybold, publisher of *The Office Computing Report*, *Unix in the Office*, and *Network Monitor*. Every issue presents discussions of current issues and technology trends in nontechnical terms so that you and your associates can find out what is going on and make more intelligent decisions.

### FASCINATING INTERVIEWS

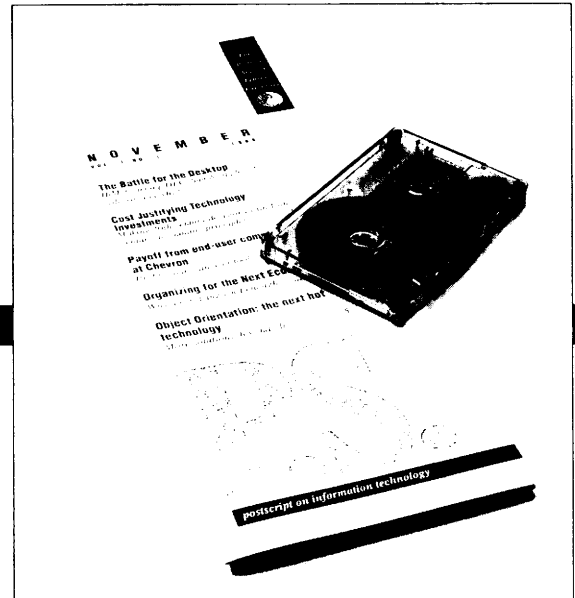
Each month, Patricia Seybold will bring you fascinating interviews on a variety of topics, including:

- Executive Information Systems
- Cost Justification
- Object Orientation
- Evolving Standards
- Desktop Platforms
- End-User Computing
- Managing Change
- Partnering for Progress

She will also give you her personal view of where things are headed. . . what to watch. . . and what you can safely ignore.

### INNOVATIVE AUDIOTAPE REVEALS EVEN MORE

But that's just the beginning. . . because an exclusive audiotape that accompanies each printed edition of the newsletter reveals even more:



- One-on-one interviews with industry insiders
- Technology trends explained in plain English for busy executives who must make fast, intelligent decisions
- Firsthand success stories—how your peers manage change in their corporate culture

## Ordering Information

### CALL FOR YOUR FREE TRIAL SUBSCRIPTION TODAY!

Call Toll-Free: 1-800-826-2424 to order your trial subscription to *P.S. postscript on information technology*, or write to:

Patricia Seybold's Office Computing Group  
148 State Street, Suite 612, Boston, MA 02109.

There is no obligation. We'll send you a FREE ISSUE—both newsletter and audiocassette—of *P.S. postscript on information technology*. Pay \$395 when your bill comes in and you'll receive an issue once every month for a full year—12 issues in all. If you do not wish to subscribe, simply write "cancel" on your bill.

You can also benefit from *P.S. postscript on information technology* by ordering the \$95-a-year newsletter only.

# P.S.

Listen to it in your car on your way to work  
or plug into it as you exercise!

# NEWS

PRODUCTS • TRENDS • ISSUES • ANALYSIS

# ANALYSIS

• DNC •

## Beyond Academia

We've been talking a lot lately about the research developments at Carnegie Mellon University (CMU). In April, it was the Andrew project; this month, it's the Mach operating system and the Camelot project. It's no coincidence. The university has put together some valuable distributed network computing solutions under Unix, some of which have helped usher in a new Pittsburgh-based software company called Transarc. The company is headed by CMU professor Alfred Spector, who has worked closely with both the Andrew file system (AFS) and the Camelot project, and he brings with him a few AFS and Camelot cohorts. Thus, the products to come from Transarc will be modeled after CMU prototypes: file system products à la AFS and transaction processing products à la Camelot (see "Distributed Transaction Processing" in this month's feature).

**IBM'S BANKROLL.** Interestingly, IBM is helping foot Transarc's bill (although to what extent, neither company will say). IBM is becoming quite the venture capitalist these days; this is its seventh venture investment in the past 18 months.

Since Transarc is patterning its products after CMU R&D, IBM's interest in the start-up isn't exactly surprising. IBM has already pumped over \$30 million into the Andrew project, a joint IBM/CMU venture. Furthermore, it plans to integrate Andrew solutions into AIX. IBM also hopes to use Transarc software. IBM already has rights to AFS, and obviously Transarc isn't going to simply replicate the CMU system. The remaining question is how much of an advantage will Transarc file system products have over the AFS original? It's hard to speculate; no products have been announced (not that we expected any; as we went to press, Transarc was only a few days old). However, considering the fact that a few influential AFS-ers are now on staff at Transarc, the refinements could be considerable.

**CMU PROTOTYPES.** The Andrew file system has some distinct advantages over Sun's NFS, the de facto standard. Specifically, AFS's disk-caching scheme makes it a better network decongestant. It also has better security and makes systems administration easier. (For the lowdown on AFS, see Vol. 4, No. 4.) Although Transarc plans to extend these AFS benefits, the company says that its products will support AFS.

Camelot has been licensed along

with Mach to Mt Xinu (Berkeley, California). Since the main Camelot developers are now with Transarc (Camelot is actually Spector's baby), the company plans to keep a consulting relationship with Mt Xinu as long as necessary for the distributed transaction processing facility.

**CONCLUSION.** As distributed network computing continues to flourish, better Unix solutions are vital. That's what caught our eye about the goings-on at CMU in the first place. In the future, we talk in depth about some of Unix's limitations in the DNC arena (e.g., relatively inefficient communication mechanisms and lack of support for multiprocessing). We think Transarc has the potential to really help out here, but we won't know until some products materialize. We'll keep you posted. ☉

—L. Brown

• OBJECT  
ORIENTATION •

## A New Era Begins for C++

Now that developers are getting serious about the C++ object-oriented language, AT&T has decided to do the

CMU Spawns a New Company. Page 20

AT&T Brings a Lot, but not Enough, to C++ Release 2. Page 20

Accessing Multiple Databases with CimLinc ID. Page 22

Uniplex Ports to Sun but Still Doesn't Do Windows. Page 22

same. Later this month, C++ Release 2 will strengthen the language's definition, add to it important new features, and align it more closely with ANSI standard C. At least one other vendor will also endow C++ with a full-fledged interactive development environment.

This is good news for the many developers using C++ to build software. However, Release 2 doesn't give them quite everything they've been asking for. In Release 2, AT&T chose not to address several key issues, including the need for a real C++ compiler.

C++ has suffered only a little from its immaturity. Despite its bare-bones repertoire of functions and tools and its sketchy definitions, the language has gained the widest following among developers of object-oriented software. Among developers of commercial products, C++ has surpassed SmallTalk, the granddaddy of the object-oriented languages, as the preferred language.

The most important trait of C++ Release 2 is its clarity. AT&T is publishing the first commercial-grade reference manual for the language, detailing all of its features. Previously, developers were forced to interpret much about the language's finer points. In addition, Release 2 adds three important enhancements:

- Support for multiple inheritance. This feature allows an object to inherit methods from more than one class of objects, dramatically increasing the reusability of code. C++ Release 1.2 supported only single inheritance.
- Stronger type-checking. This feature reduces the risk that programmers will specify nonexistent or incorrect object types in their designs. Type-checking strengthens the interfaces to objects in a system, which should improve reliability.
- Increased compatibility with the ANSI standard for the C language.

C++ will never be totally compatible, but the closer it is to ANSI C, the better for developers.

Stability was AT&T's primary concern in designing Release 2, says Donald Kretsch, supervisor of C++ development. Multiple inheritance has been added at the urging of licensees, and there's an upgraded I/O interface class of objects. However, AT&T will move slowly, if at all, on proposals for other added features, says Kretsch. New features will be added only if they don't compromise consistency between releases. Thus, although licensees are clamoring for many other extensions to C++—real-time support and multiprocessing support, for example—AT&T is likely to let most of them remain as vendor-specific extensions to the base language for a long time.

*Despite its bare-bones repertoire of functions and tools and its sketchy definitions, C++ has gained the widest following among developers of object-oriented software.*

The disappointments in C++ Release 2 are significant. First, AT&T will not provide interactive development tools, and Kretsch says development of tools is not a priority. That means licensees will have to rely on third parties to provide them. The situation for C++ developers today is grim. "C++ programmers are still working with stone knives and bearskins," says Harris Shiffman, a technical marketing specialist at Sun Microsystems.

Fortunately, more tools should start arriving very soon. This month, ParcPlace Research Systems Incorporated, a leading SmallTalk vendor, will

announce a version of its interactive development environment for C++. The toolset, called ObjectWorks for C++, includes:

- A class browser, which allows programmers to probe the methods supported by existing classes of objects
- An object inspector, which allows programmers to look inside objects to discern their contents and behavior
- An incremental compiler, which compiles code section by section, making corrections easier
- An interactive debugger, which allows programmers to trace C source code to its C++ objects
- A graphical development environment

Sun's Symbolic Programming Environment currently provides these tools for Lisp development, and Shiffman reveals that Sun has plans to port them to C++. He won't, however, reveal Sun's roll-out schedule.

The need for a real compiler for C++ is a nagging issue. C++ source is actually "translated" into C source code, and it is the C code that gets compiled. C++ Release 2 will retain the language's status as a C language pre-processor. "We need to move into a different level of technology with our compiler," concedes Kretsch. An incremental C++ compiler is a priority at AT&T, and it will eventually become part of the core language AT&T licenses. But it won't be available soon.

Next, AT&T plans to address other inadequacies of C++. First, the number of class libraries provided with C++ will slowly grow to include X-Window interface objects, OSI network interface objects, and so on. As they grow, these libraries will reduce the proprietary libraries in the market, improving overall compatibility between different C++-based software. AT&T is also seeking a forum to discuss which standards will help ensure that class libraries from dif-

ferent vendors can interact. Today, there's no guarantee they will.

The next major feature AT&T is likely to add to C++ is support for generic or parameterized types. Generic types define objects that perform generic functions, like searching lists.

A long-range objective for Kretsch's group is determining how the language defines objects that are persistent, or stored for repeated use. Today, persistence is defined by object-oriented database management systems. "We believe persistence has to be represented in the language," says Kretsch, "but we don't know where to draw the line." The simplest solution would be to add a new keyword, like "persistent," to C++. But that leaves open questions of how to address objects as unique entities. Those issues may be better left to individual databases. AT&T's goal, says Kretsch, is to add support for persistence that's strong enough to allow C++ to be used as both a programming and a database language. That would eliminate any need for the equivalent of an SQL in the object world.

All of this represents progress for C++. With all its gaps, the popularity of C++ is amazing. Release 2 promises to strengthen its standing among object-oriented languages. ● — J. Rymer

## • FORMS PROCESSING •

### CimLinc Ties It All Together

CimLinc Incorporated, Itasca, Illinois, offers a product called Intelligent Documentation (ID), which, in its simplest form, is a forms processor. The Unix-based software can replicate any paper form you may use in your organization. Ho hum.

But there are exciting things about ID. Not only can it create screen forms, it can also add intelligence (calculations, logic, etc.) to these forms. Better?

And there's more. ID can also

front-end databases. Not just a database, but databases. And not just Unix relational databases (including, at this time, Oracle, Sybase, Unify, Ingres, and Empress), but image databases and text databases, too. And get this: Each form can access multiple databases. For example, you can have a form for an airplane part. When you enter the part name or number, all the product specs are automatically filled in from, say, an Oracle database. Next, the image of the part displays (ID provides tools to manipulate the image). Finally, the inventory information and price of the part come in from yet another source.

All this is transparent to the end user, who needn't have the foggiest about where this data exists on the network.

**PRODUCT DETAILS.** ID runs on a number of platforms: Sun3, 4, and 386i; Solbourne; and HP's 9000 and 3000 series under X-Window. The software is currently being ported to AIX, also under X-Window, and will be available for the IBM RT sometime this summer.

The product can access any system that can be attached to an Ethernet or Sun NFS network.

Pricing is as follows:

- \$6,000 for the systems administrator product. The systems administrator builds the forms and creates the links to the underlying databases.
- \$4,000 for the professional. The professional user fills in the forms and manipulates the data within them. With the proper permissions, the professional can also use the ID forms as front ends to update the underlying databases.
- \$1,000 for view only. The view only version runs on X terminals as well as on more expensive workstations.

ID is sold both directly from CimLinc and through industry-specific VARs, who design basic forms and procedures for their vertical markets.

**CIM SHELL.** The user interface to ID is CIM Shell, a mouse-driven, graphical environment. CIM Shell is also a keystroke-capture macro facility and a complex scripting language. Unfortunately, CIM Shell's script-builder is designed not for the average user, but rather for the sophisticate or for the systems administrator.

**A MANUFACTURING TOOL.** ID was originally designed for manufacturing environments. An early customer, Boeing Aerospace, used ID to reduce the time needed for welding engineers to fill in a welding specification sheet from 18 hours to 2 hours. The extra time was usually spent tracking down the proper information and querying the databases.

**COMMERCIAL APPLICATIONS.** We see a lot of commercial applications for ID. Any organization with a paperflow and distributed data situation needs some sort of integrating product. ID is impressive in its ability to front-end multiple data sources so transparently.

What CimLinc needs to do now is front-end the scripting language so that average end users can link together forms into a process (procedural automation) and create their own applications. ● — R. Marshak

## • UNIPLEX •

### Porting to Sun

There's something very attractive about a Sun workstation—especially its interface. Sun has always provided the kind of windowing environment and graphical user interface that is just now driving the industry. To meet industry demands, applications developers are striving to adapt their products to slicker platforms. Recently, Uniplex ported its software to Sun machines (the new Sun3 and SPARC workstations announced in April). However, we were somewhat disappointed that Uniplex hasn't taken advantage at all of

Sun's interface.

We've always been fans of Uniplex's integrated office system, Uniplex II Plus. It's never been a window-based system, but functionally, it's impressive—a definite leader in the Unix office system game. It has most everything you'd want in an office system: a relational database based on (and fully compatible with) Informix, spreadsheet, calendar, word processing, rolodex, and calculator. (Our last review of Uniplex II Plus appears in Vol. 2, No. 12.) But we are concerned that the product is not keeping up with the times, as evidenced by the Sun port.

**SUN PORT.** The Sun port simply allows you to run the traditional character-based software within Sun windows. You're still working with vanilla Uniplex. Of course, that does create some consistency; Uniplex users who migrate to Sun can stick with a familiar interface. On the other hand, writing software to Sun workstations without taking advantage of Sun's interface seems like a waste. It would be one thing if Uniplex had a sophisticated interface itself, but it doesn't. The interface is acceptable, but really nothing to write home about. The strength of Uniplex lies more in its integration of mod-

ules than its friendliness.

**ON THE HORIZON.** The current Sun port is actually a transitional step. It's hard to be a serious Unix contender without porting your software to Sun. Uniplex is working on an X.11 version. Last February, Uniplex announced a technology-sharing agreement with IXI Limited (Cambridge, England) that will give Uniplex access to IXI's X.desktop, a graphical Unix shell that supports both Motif and Open Look. The X version of Uniplex is due out this fall. We'll postpone judgment until then. ☉

—L. Brown

**SPREAD THE NEWS!!** And Extend Your Newsletter Subscription One Month **FREE!**

Do you know someone who might benefit from the information in Patricia Seybold's monthly Computer Industry Reports? Please let us know so that we may send them a **FREE SAMPLE ISSUE** to review.

In return for your help, we will add a **FREE** month to your current subscription term.\* Just fill out the information below and send it to us at: Patricia Seybold's Office Computing Group, 148 State Street, Suite 612, Boston, MA 02109. Or call us at 1-800-826-2424 (in Mass., call 617-742-5200). *Thank you!*

CONTACT NAME _____	YOUR TITLE _____	YOUR NAME _____	YOUR TITLE _____
CONTACT COMPANY _____		YOUR COMPANY _____	
CONTACT ADDRESS _____		YOUR ADDRESS _____	
CONTACT CITY/STATE/ZIP _____		YOUR CITY/STATE/ZIP _____	
CONTACT PHONE NUMBER _____		YOUR PHONE NUMBER _____	

**SUGGESTED INTEREST (check one):**

- Office Computing Report
- Unix in the Office
- Network Monitor
- P.S. postscript on information technology

**SUBSCRIPTION YOU WOULD LIKE TO EXTEND ONE MONTH:**

- Office Computing Report
- Unix in the Office
- Network Monitor
- P.S. postscript on information technology

\*Limit one extension per subscription term.

# Patricia Seybold's Computer Industry Reports

## ORDER FORM

Please start my subscription to:

		U.S.A.	Canada	Foreign
<input type="checkbox"/>	Patricia Seybold's Office Computing Report	12 issues per year	\$385	\$397 \$409
<input type="checkbox"/>	Patricia Seybold's UNIX in the Office	12 issues per year	\$495	\$507 \$519
<input type="checkbox"/>	Patricia Seybold's Network Monitor	12 issues per year	\$495	\$507 \$519
<input type="checkbox"/>	P.S. postscript on information technology	12 issues & tapes per year	\$395	\$407 \$419
<input type="checkbox"/>	P.S. postscript on information technology	12 issues per year	\$ 95	\$107 \$119

Please send me a sample of:

- Office Computing Report       Network Monitor  
 UNIX in the Office       P.S. postscript on information technology

Please send me information on:

- Consulting     Special Reports     Conferences

My check for \$\_\_\_\_\_ is enclosed.       Please bill me.       Please charge my subscription to:  
 Name: \_\_\_\_\_ Title: \_\_\_\_\_ Mastercard/Visa/American Express  
 Company Name: \_\_\_\_\_ Dept.: \_\_\_\_\_ (circle one)  
 Address: \_\_\_\_\_ Card #: \_\_\_\_\_  
 City, State, Zip code: \_\_\_\_\_ Exp. Date: \_\_\_\_\_  
 Country: \_\_\_\_\_ Bus. Tel. No.: \_\_\_\_\_ Signature: \_\_\_\_\_

Checks from Canada and elsewhere outside the United States should be made payable in U.S. dollars. You may transfer funds directly to our bank: Shawmut Bank of Boston, State Street Branch, Boston, MA 02109, into the account of Patricia Seybold's Office Computing Group, account number 20-093-118-6. Please be sure to identify the name of the subscriber and nature of the order if funds are transferred bank-to-bank.

IU-0689

Send to: Patricia Seybold's Office Computing Group: 148 State Street, Boston MA 02109; FAX: 1-617-742-1028; MCI Mail: PSOCG  
 To order by phone: call (617) 742-5200

## Topics covered in Patricia Seybold's Computer Industry Reports in 1988/1989:

Back issues are available, call (617) 742-5200 for more information.

### Office Computing Report

#### 1988—Volume 11

- # Date Title
- 8 Aug. Metaphor Computer Systems—A Quiet Revolution
- 9 Sep. Technology That Supports Meetings
- 10 Oct. Document Processing
- 11 Nov. Upper CASE Tools—Getting Users to Help Design Systems
- 12 Dec. Executive Information Systems

#### 1989—Volume 12

- # Date Title
- 1 Jan. Digital's Office Strategy—Going After the Desktop
- 2 Feb. Compound Documents Take the Stage
- 3 Mar. Wang's Freestyle—Not the Paperless Office, But an Office with Less Paper
- 4 Apr. Focus on Training Technologies—Office Systems Training from DG, Digital, HP, IBM, and Wang
- 5 May The LAN Office Emerges—WordPerfect Enters OA Race

### UNIX in the Office

#### 1988—Volume 3

- # Date Title
- 8 Aug. BBN's Slate System—Delivering Compound Documents
- 9 Sep. Distributed Databases—Where Are We?
- 10 Oct. DECwindows
- 11 Nov. DataMaker's Unix—A New Beginning
- 12 Dec. Document Processing with Interleaf *Second Feature*: Apollo and Sun Service and Support

#### 1989—Volume 4

- # Date Title
- 1 Jan. Unisys Ofis Ensemble—In Search of Cohesion
- 2 Feb. A Tale of Two Operating Systems—System V.4 and OSF1
- 3 Mar. FrameMaker 2.0—The Art and Science of Document Processing
- 4 Apr. CMU's Andrew Project—A Prototype for Distributed, Object-Oriented Computing
- 5 May Sybase—The Next Generation

### Network Monitor

#### 1988—Volume 3

- # Date Title
- 8 Aug. Novell—The Juggernaut Rolls On
- 9 Sep. OSI Standards—A Promise Fulfilled?
- 10 Oct. ISDN: We Still Don't Know
- 11 Nov. Data General's PC Networking Strategy
- 12 Dec. 3+Open—3Com Makes Its Move

#### 1989—Volume 4

- # Date Title
- 1 Jan. APPN—Peer-to-Peer Networking Arrives at IBM
- 2 Feb. Bridges & Routers—Dividing to Conquer
- 3 Mar. BBN Builds Big Networks—Designing Private Packet-Switched Networks
- 4 Apr. Northern Telecom DNS—Trying to Build Intelligent Networks
- 5 May NetView—IBM's Enterprise-Wide Manager