Patricia Seybold's

VOL. 3, NO. 10
ISSN: 0887-3054
OCTOBER 1988

INSIDE

EDITORIAL

FORUM

NEWS ANALYSIS

UNIX IN THE OFFICE

PRODUCTS • TRENDS • ISSUES • ANALYSIS

DECwindows

X Moves from Classroom to Boardroom

By Michael D. Millikin

BREAKING FREE FROM its academic heritage, X-Window is suddenly a hot topic on the commercial front. Vendor after vendor has lined up behind this evolving standard, each with the stated intention of incorporating it into its systems architecture.

Of all those vendors, however, Digital has burst forth first with a complete implementation of X—DECwindows—that spans its strategic operating systems (a simpler task for Digital than for some). DECwindows, thus, is important to Digital, not just in relation to Unix, but throughout the company's product line. *(continued on page 3)*

• E D I T O R I A L •

The Best of All Possible Worlds

The Unix User Interface Issue Deserves More Than the Easiest Solution.

By Judith S. Hurwitz

USER INTERFACE takes center stage as the Open Software Foundation (OSF) begins its deliberations about which interface will become standard in the Unix arena. It is a confirmation of the growing importance of Unix that user interface under Unix is taking equal billing with user interface under OS/2. The key test for OSF and for the Unix community itself will be how OSF will arrive at this important decision.

THE EASY ROUTE. Two avenues are possible. The most obvious and, in the short run, the easiest would be simply to choose one of the existing (or in development) user interfaces from a single vendor (at last count, we hear that some 39 designs have been offered) and make that the standard.

Digital Makes a Bid. This is the sort of turnkey solution that Digital Equipment is hoping OSF will choose. It is no secret that Digital desperately wants DECwindows to emerge as an industry standard. In some ways, this makes good sense. It would be relatively simple for OSF to accept what Digital has generously offered and, in turn, offer it to its members as one neat package. DECwindows would be an acceptable user interface because it includes all the component parts that OSF requires.

A MIX-AND-MATCH APPROACH. In the long run, however, it makes more sense for OSF to take the best of all the offerings. Many corporations and universities have developed excellent technology on top of the X-Window platform. Therefore, in the best interest of the future of commercial Unix, OSF should take the best components of many of these technologies. Yes, this would be more difficult. OSF would have to play the role of systems integrator, putting the pieces together into a cohesive format. Fortunately, the design of X-Window—a requirement for OSF—lends itself to this type of modularity. For example, OSF could choose one vendor's toolkit, another's widget set, and yet another's icons and look and feel. This approach would solve two problems. First, OSF will appear more diplomatic if it does not select one vendor's solution. The selection of something as important as user interface cannot be divorced from the hot politics of competition. Second, while DECwindows has some excellent component parts, there are other vendors that do some things better.

BRIDGING THE GAP. As OSF makes its choice, it also needs to consider the ramifications of bridges to the PC arena. As OS/2 and Presentation Manager rise to prominence, it will be important that the user interface selected for Unix have hooks into this environment.

If Unix is to become truly commercial, it must accommodate those systems already installed and planned for installation in user organizations. We expect that OS/2 and Unix will be closely linked in the future. At the same time, OSF should take into account the future importance of object orientation. It is inevitable that a graphical user interface is the first step towards a true object-oriented environment. Therefore, OSF should take a careful look at user interface tools that would aid these next-generation environments. For example, the Andrews Toolkit from Carnegie-Mellon University will be invaluable for this type of applications environment.

Our message to OSF, then, is "Take the best of all possible worlds."

Patricia Seybold's Office Computing Group

Publisher PATRICIA B. SEYBOLD

Editor-in-Chief JUDITH S. HURWITZ

Managing Editor
RONNI T. MARSHAK

Senior Editors
MICHAEL D. MILLIKIN
JUDITH R. DAVIS
Telephone: (617) 861-3926

Associate Editor
JOHN N. KELLY

News Editor
DAVID S. MARSHAK

Contributing Editor
GARY J. NUTT
Telephone: (303) 444-0341

Assistant Editor
LAURE BROWN

Sales Director
RICHARD ALLSBROOK JR.

Circulation Manager
DEBORAH A. HAY

148 State Street, Suite 612, Boston, Massachusetts 02109 Telephone: (617) 742-5200 FAX: (617) 742-1028

Patricia Seybold's UNIX in the Office (ISSN 0887-3054) is published monthly for $495 (US), $507 (Canada), and $519 (Foreign) per year by Patricia Seybold's Office Computing Group, 148 State Street, Suite 612, Boston, MA 02109. Application to mail at second-class postage rates is pending at Boston, MA and additional mailing offices. POSTMASTER: Send address changes to *Patricia Seybold's UNIX in the Office*, 148 State Street, Suite 612, Boston, MA 02109.

•DIGITAL•

(continued from page 1)

X Overview

The X-Window system is a high-performance network windowing and graphics system that is hardware independent. X-Window allows users of one system to open up a window onto another node (of either the same or of a different system type) in the network and run an application there, having the output display in that X-Window on the user's screen. Through X, a user can have access to the applications resident on a variety of different systems, provided that each system is able to support the X-Window system and they are networked together.

Because an X environment enables the running of an application on one node with the display appearing on another, it requires a fairly speedy network: Ethernet, for example.

Benefits of X-Window

BENEFITS TO DEVELOPERS AND USERS. The X-Window system offers several attractive benefits to developers and to users.

Transparency. X is a transparent distributed system.

Network Based. X is network based and is built around a client/server model. This allows one vendor's workstation to display the applications running on multiple foreign application nodes across the network. A classic example of this on the technical workstation side is to have a Cray on the back end running a simulation application while displaying the output in a window on the user's desktop machine. Access from the display to the host is transparent to the end user.

The trick will be for vendors to maintain enough basic compatibility among their implementations of X to allow distinctions in the interface, while keeping the need for recompiling to a minimum.

The same architecture is attractive on the commercial side as well, but for different types of applications.

There are other ways to achieve such results, other ways to have relatively puny workstation clients exploiting mesomorphic back-end processors. Regular old terminal emulation (ROT or ROTE, take your pick) is one such method. Terminal emulation, however, does not address interface consistency or application portability. Nor does it exploit the presentation capabilities of the workstation.

A more elegant solution than terminal emulation, and one designed to exploit the capabilities of the workstation, is to split the application between client and server. In this cooperative processing model, the workstation handles the presentation task while communicating on the back end with the server process.

This differs from the X approach in that all of the application code under X runs on the application machine; the workstation is a network resource (a server) providing the application with display services. (Hence the seeming paradox of the X server running on what would normally be called a client.)

This cooperative processing approach is currently application specific and, thus, is much more specialized than the X-Window solution.

Given the need for uniting all types of equipment into a transparent distributed network, however, vendors will end up providing multiple solutions as the default: terminal emulation along with X-Window, for example, or cooperative processing implementations on top of an X-Window base.

Hardware Independence. X is hardware independent and so supports application portability. Once an application is written using X, it needn't be rewritten to accommodate new display hardware. In other words, users aren't dependent on the application vendors coming up with new display drivers for particular applications. The display vendor would write a new version of the X server, and the applications would run unmodified.

Extensibility. X is extensible. X's protocol extensibility allows vendors to enhance the basic environment. Digital has chosen to do so through the incorporation of Display PostScript and support for three-dimensional graphics.

The trick for the industry will be for vendors to maintain enough basic compatibility among their implementations of X to allow competitive distinctions in the interface, while keeping the need for recompiling applications to a minimum. To do this successfully requires some basic agreements on the widget sets (see "X Toolkit" below).

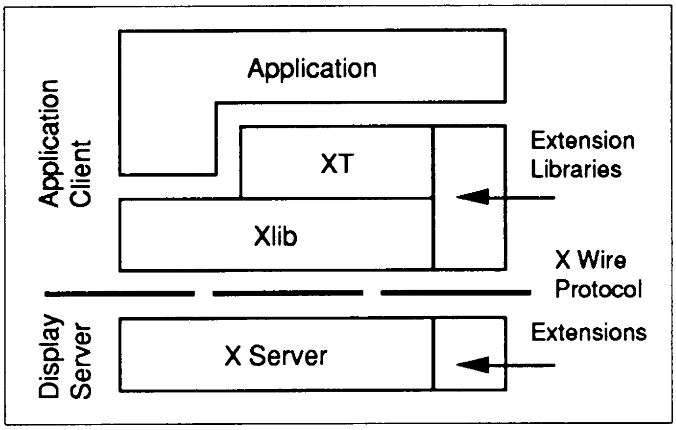
DESIGN ADVANTAGES. In design, however, X supports nonexclusive enhancements and extensions.

Multivendor Support. X is a standard with multivendor support. Some 19 vendors have announced support for this windowing system. Clearly, the momentum is with X. The group responsible now for the continued evolution of X is the MIT X Consortium—a group comprising some two dozen vendors and organizations as primary or associate members. An associated testing consortium is producing test verification suites ensuring the compatibility of X products.

X/Open has adopted X for publication in the X/Open specification documents.

The American National Standards Institute (ANSI) has created a dedicated subcommittee (X3H3.6) to investigate including X in ANSI specifications.

X will be part of the converged Sun/AT&T Unix and will



be part of the Open Software Foundation (OSF) offering that emerges from that group.

Public Property. X is in the public domain. Adoption by standards bodies and work by the MIT X Consortium aside, X had a real good shot at becoming a de facto standard mainly because MIT put X in the public domain.

History

X began life as W at Stanford University. When MIT adopted the W standard for Project Athena (begun in 1984 and jointly funded by Digital Equipment and IBM), W became X. Project Athena's mission was, in essence, to create a generic windowing system to connect dissimilar monochrome and color bitmapped workstations from a variety of vendors. The university needed to create a homogeneous environment on top of heterogeneous equipment with application portability as one aspect.

The first widely demonstrated and adopted version of X-Window was version 10 (X10). Project Athena recognized the need for a serious reworking of some of the original design criteria to make X into a more functional system for a larger class of developers. During the work on the X11 specification, which began in 1986, the developers added:

- A graphics state
- Precise semantics for the output routines
- Generalized input events
- A toolkit for applications developers

Architecture

X-Window consists of four primary components: the X Server, Xlib, the Xtoolkit (XT), and the X wire protocol.

The architectural nomenclature of X-Window generally causes some confusion because the designated roles of client and server seem intuitively backwards. The X Server runs on what most people would generally consider a client: the workstation. The X client, consisting of the toolkit, the Xlib library of primitive graphics functions, and the actual application itself, runs on what most people generally consider a server: a CPU node remote from the workstation that functions as an application server.

THE X SERVER. Basically, the X Server is the program that runs on a workstation creating the basic interface, displaying graphics, and so on.

XLIB. Originally, Xlib was a simple procedural interface to the X protocol. As X evolved, however, Xlib grew to hold common utility routines as well. Now, Xlib is a client-based library of nearly 300 graphics routines. The main function of Xlib and other layered libraries is to translate messages between client and server into X protocol requests.

These routines fall into the following categories:

- Window functions, such as creation and destruction, mapping and unmapping, and configuring.
- Window information functions, such as window attributes and structure of the window tree (root/child windows).
- Graphics resource functions for the creation, destruction, etc., of color maps, pixmaps, or graphics contents.
- Graphics functions such as clear, fill, copy, draw, etc.
- Window manager functions that can change the parent of a window, grab part or all of the server, change event-dispatching and -processing, change the keyboard encoding, change the resident color maps, and modify the list of X clients that have access to the server.
- Events and event-handling functions. Event types include keyboard input, pointer motion, window entry or exit, gain or loss of keyboard to the application, change of window state, and communication from another client.
- Application utility functions. These functions are miscellaneous routines that, although they do not map to X protocol requests, make life easier for applications developers by providing functions such as bit-map manipulation, translation of keyboard events into character codes, and parsing application start-up command lines.

The communication between Xlib and X server rides atop whatever the LAN transport interfaces might be. Neither the client application nor the user display need be concerned with the mechanics of cross-network communications.

The client application passes procedural calls for window management and drawing commands to Xlib for conversion into the network protocol format which the X server then receives and interprets. Xlib then translates the X server messages carrying user-input events (keyboard command, mouse click, etc.) or screen changes (e.g., window change) into return values for the client, all through a procedural interface.

In theory, developers can use the X protocol for direct communications between client and server. In practice, however, the Xlib routines are the lowest level interface that applications should use.

X TOOLKIT. That Xlib interface is, however, fairly low-level, and requires a great deal of programming effort. Possibly the most familiar Unix program around is the simple "Hello World" display:

```
# include <stdio.h>
main ( )
{
 printf("Hello, World\n");
}
```

Those few simple lines would make "Hello World" appear on a terminal screen. Once in the world of bit-map displays, however, you add the additional complexities of window creation, font choice and position, color, repainting the window in case of exposure, recentering the text if the window size changes, possible iconization, and future expansion of the window.

Writing "Hello World" based directly on the Xlib would take about 40 executable statements. Using a toolkit can greatly reduce that number by eliminating the drudgery of dealing with the interface considerations mentioned above.

> *In the vacuum created by the absence of a standard, a number of vendors are trying to have their particular approaches to widgets accepted as that standard, Digital among them.*

Several toolkits to date have appeared for X11: The Andrew Toolkit from Carnegie-Mellon and IBM, the Open Dialogue toolkit from Apollo, and the Xtoolkit and widgets from MIT.

The Xtoolkit (XT) consists of two parts: the intrinsics library and the widget set. Intrinsics are low-level, user interface development tools and come standard with X. These are utility routines for implementing code, and must work in conjunction with higher-level, screen-based components called widgets. Together, these create the look and feel of the user interface.

Widgets, basically, are specialized windows with input and output capabilities. They provide interface features such as scroll bars, dialogue boxes, buttons, fields, and so on. Applications can invoke widgets within their own windows, thus freeing the developer from writing user interface code while also creating a consistent look and feel across all supported applications.

The MIT X Consortium has not yet standardized on a particular set of widgets, although work is on-going on these core components. Part of the basic design philosophy of X was that it provide interface mechanisms and tools for window management, without codifying specific styles or policies. Thus, vanilla X comes with a full set of intrinsics but just a few widgets.

However, standardization in this area will probably have to occur, at least for a slightly broader set of widgets. If the widget sets offered by different implementations of X are too different, developers will face a recompile in order to get their applications working under the variety of X environments.

In theory, at least, extensions to the widget set for specialized look and feel of interfaces should be nonexclusive—in other words, they should not preclude the utilization of other superwidgets provided as application-specific extensions within a given environment.

Structurally, widgets are all subclasses of the core widget in a given class. Thus, the additional widgets, or application-specific widgets, should require just the addition of enough code in an application to call the appropriate widget. The level of effort depends on how far down an X implementer decides to go to fiddle around with the core set.

(Or, as Scott McGregor, Digital's DECwindows' architect put it, the answer to how much effort such customization requires is "42." A meaningful answer depends upon the parameters of the question. All of you who haven't read *The Hitchhiker's Guide to the Galaxy* are missing out.)

If a developer decides just to change the interface a bit, then the new environment basically requires only a recompile. If, however, he or she decides to change the application interface, then new source code is needed.

In general, Digital expects application developers to create their own superset widgets for their particular applications. However, because those widgets retain the characteristics of the parent widget class, they will exhibit similar behavior as the foundation widgets (thus retaining basic look and feel.) A change of appearance is more easily addressed within X than is a basic change of behavior.

Part of the potential success of X in the commercial world lies in its ability to attract independent software vendors (ISVs); the fewer times an ISV will have to port an application to a different X implementation, the greater the likelihood that the ISV will embrace the X environment. As the DOS world has shown, success comes from "plug and play" compatibility.

In the vacuum created by the absence of a standard, a number of vendors are trying to have their particular approaches to widgets accepted as that standard, Digital certainly among them. Whatever OSF decides about its user interface environment will certainly influence the shape of the eventual de facto standard.

We expect some sort of accommodation, where the MIT X Consortium releases an enhanced set of core components that all vendors can use, leaving those vendors with some room for enhancement.

X PROTOCOL. The X protocol defines the data structures that transmit requests between applications and display servers over the network. Generally, these structures are created by Xlib or another layered library rather than by the application itself.

The X protocol is independent of operating system, network transport, or application language. All the protocol requires is any error-corrected duplex byte stream with asynchronous operation. (Asynchronicity augments performance because it replaces the process of synchronous handshaking with pipe-lining. Exploiting this pipe-lining does require, however, reliable transmission mechanisms.) Should an application require synchronicity, however (such as a query for cursor position), X can wait for a response from the display server.

Some X requests have return values (such as a state query). Although X does not specify any synchronization requests, any request depending upon the completion of another request will block, pending execution of those other requests.

The X protocol, then, is what allows users within a heterogeneous network to have transparent access to all X applications running over the network. Applications and display devices (X servers) pay attention only to the protocol or to the X routines atop it.

The X protocol also describes connections, windows, events, and graphics routines.

CONNECTIONS. Connections define the communications path between client and server. This path can exist between processes on the same machine or on machines distributed across a network. Generally, a client has one connection to the server.

A client application first opens a connection with the X server (the workstation) using a common transport (such as TCP/IP). The client then passes version and authorization protocol information and a code indicating the client byte order. Knowing the byte order becomes important if the byte order of the client requests differ from the byte order of the server, a condition that can easily happen on a heterogeneous network. For example, the byte order of the Motorola 68000 is different than that of the VAX.

The X protocol is what allows users within a heterogeneous network to have transparent access to all X applications running over the network.

Should the original connection request be successful, then the server sends a reply back to the client. The reply contains information such as display hardware, vendor identification string, display resolution, physical dimensions, and color-handling capabilities.

WINDOWS. Windows, which are familiar to most people by now, are the basic means of interaction with the user. X windows are hierarchical; they allow the creation of windows inside windows to an arbitrary depth.

Windows have a variety of attributes, including:

- Physical location on screen.
- Relationship to other windows.
- Condition of visibility (mapped state).
- Visual type and color map.
- Background pixel and pixmap.
- Border pixel and pixmap.
- Bit gravity and window gravity. (Gravity indicates which pixels are retained when the window is resized or where child windows should go when the parent is resized.)
- Cursor.
- Backing_Store and Save_Under Hints. These optional capabilities provide a little extra efficiency in window manipulation. Backing_store requests space for the window contents, and save_under requests that the server preserve the data for the windows behind the window used. (Release 1 of DECwindows does not implement these capabilities.)

The basic workstation window is called the root window, and it displays background patterns or color. Children of the root window are shell windows, and are associated with a particular application.

Windows can be visible or hidden. Visibility has a variety of stages and processes leading to the actual display event. A requirement for visibility is mapping, or the allocation of resources supporting the eventual display of the window on the screen. A viewable window is one that is mapped and whose ancestors are mapped as well. (In other words, it is ready for display because not only has the server allocated the resources necessary for its display, but it has allocated those resources necessary for the display of its parent windows as well.) A window, then, is visible under the following conditions: first, if it is viewable; second, if it occupies a position in the virtual coordinate system that corresponds to a piece of the physical screen; and, third, if it is not obscured by another window.

EVENTS. Events are indications that something has happened (such as a key press or a mouse entering a region) that are sent to the application from a variety of sources, such as the X server, XT, or other applications.

Events are grouped by type, and X applications can mask out different events.

Important types of events include key actions, button actions, or notification of motion of the cursor device; the entry or departure of the pointing device into or out of a window; becoming the active window or becoming the inactive window; exposing a previously hidden segment of the window; property changes of the windows; and client messages (a client application-to-client application message).

GRAPHICS. X offers about 50 simple graphics routines, including lines, arcs, text, and block pixel transfers.

One of the major performance enhancements in X11 over X10 comes from the addition of polyroutines: i.e., the ability to draw multiple lines (PolyLines) rather than one line at a time, PolyRectangles, PolyPolygons, and so on. Such polyroutines reduce the number of conversations required across the network to produce a given set of output.

Another enhancement was the addition of a graphics context that passes basic information (color, for example) along with the graphics routine itself.

Finally, the X11 group worked on the semantics of the graphics routines so that pixels would be defined precisely for each call. This capability addresses predictability more than performance. If you draw a line tangent to a circle, you are guaranteed that line and circle will actually touch.

VENDOR DIFFERENTIATION. Vendors can provide different window-management schemes layered onto the structure. This is one area where vendors can establish interface policies consistent with their approach to the market while retaining some basic compatibility across platforms.

Another area for differentiation is in the server implementation.

DECwindows

DECwindows is an upwardly compatible superset of the basic X-Window design. Digital spent most of its effort in enhancing performance, providing better error-checking, writing a user interface language, and designing a toolkit and widget set.

DECwindows Enhancements to X

X SERVER. Digital has done some clever implementation work with the server, all designed to enhance performance. It has not made any basic functional changes.

Digital has done some clever implementation work with the X Server, all designed to enhance performance. It has not made any basic functional changes.

Digital says that it has attempted to match all X semantics precisely, while adding additional support in areas such as security and font lookups.

However, Digital takes seriously the requirement of exact compatibility with the standard. The MIT X Consortium has a group working on producing validation suites for testing X conformance, and Digital is participating actively within that group.

LOCAL VERSUS NETWORKED PERFORMANCE. When both X client and X server are resident on the same VMS workstation, DECwindows uses a shared memory transport rather than the more cumbersome network transport mechanism. The result is much snappier performance than would have otherwise been possible.

On the Ultrix side, Digital uses a tuned implementation of Unix Pipes to achieve better local performance over the network counterpart.

XLIB. Digital augmented the standard MIT Xlib by providing extended keyboard support for international character sets; parameter checking though the use of a standard synchronization flag; and VMS-specific extensions to make Xlib a shareable library.

EXTENSION LIBRARIES. X is designed to support extensions for the server as well as the client. Digital is exploiting this by incorporating PostScript, three-dimensional graphics (PHIGS Extensions to X, or PEX), and imaging capabilities (X Image Extension Protocol, or XIEP).

Digital is working on the PEX extension in conjunction with Adobe. The imaging capabilities will address the compression and scaling of large, high-resolution scanned images. The X Consortium is dealing with these imaging issues as well as the PEX issue.

Digital's work with PostScript is similar in some ways to Sun's use of PostScript in NeWS; yet, in other ways, such as in the basic philosophy, it is quite different.

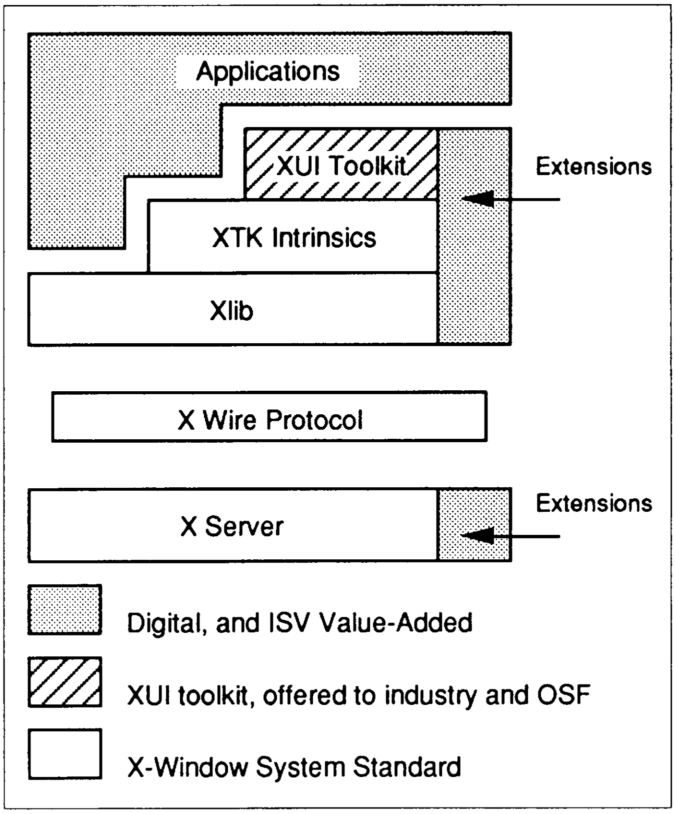
Sun seems to approach PostScript as a good language for applications; Digital, on the other hand, treats Display PostScript much more as a graphics package that provides additional functions (such as rotation and scaling of text) and extensibility.

As NeWS is run-time extensible through the downloading of PostScript code, however, so is DECwindows.

Digital also is working directly with Adobe on its implementation; Sun is not.

These extensions will not be unique to Digital; in fact, the X Consortium is working, with Digital's help, as Digital is a member, to incorporate those same extensions into X. The difference is, as the ad line says, that "Digital has it now."

Although Digital has detailed the specification for such extensions, they are not shipping with Release 1.



INDUSTRY-STANDARD LIBRARIES. DECwindows adds support for industry-standard libraries such as Programmers Hierarchical Interactive Graphics Standard (PHIGS) and Graphics Kernel System (GKS).

ADDITIONAL PROGRAMMING LANGUAGES. While MIT X supports only a C and Common Lisp interface for Xlib, DECwindows adds a complete set of VAX standard language bindings including Fortran, Ada, and Pascal. Consortium efforts are under way to produce Xlib bindings in these areas as well.

QUALITY OF IMPLEMENTATION OF STANDARD X COMPONENTS. Digital believes it has implemented the standard X components in a more efficient manner by using better algorithms, exposing additional interfaces, and offering more flexibility. For example, Digital uses a window-repositioning algorithm that is faster, and that offers international keyboard support and font-caching.

DIFFERENCES WITH X. Digital's main "differences" are in the implementation in certain areas (such as those producing better performance, shared memory, etc.). The bottom line is, however, that DECwindows is 100 percent compatible with the MIT X Consortium standard.

As the X Consortium standard evolves, so will DECwindows. Digital plans to stay on top of the development work. One of the real benefits of X's nonexclusivity is the support for vendor value-added while retaining basic compatibility.

DECwindows Toolkit

Digital basically is providing the standard set of MIT intrinsics with a Digital widget set and a definition of the user interface policy. Technically, the toolkit consists of three components: the Widgets, a resource management facility, and cut-and-paste interfaces.

Digital widgets in the toolkit include:

- Command, a command-line interface
- Dialog, the outer shell in a composite widget for a dialogue box
- File selection (a way of shielding the end user from a variety of operating systems and file management techniques)
- Help
- Label
- List box
- Menus (a set of composite widgets for pull-down, pop-up, and cascading menus)
- Message
- Pushbutton
- Scale
- Scrollbar
- Scroll windows
- Text

Core Applications

With its own implementations of DECwindows, Digital is offering a set of core applications designed to enhance the basic functionality of the interface environment.

- A Window Manager controls window-positioning, window-decoration, and icons.
- A Session Manager manages user authentication.
- A collection of "out of the box" application utilities offers functions such as a calendar, notepad, clock, cardfile, and calculator. Much like the Desk Accessories in the Macintosh, or the utility programs that arrive with MS Windows, these applications help users practice the skills of their new environment.

DECwindows XUI

Digital is licensing its user interface software (XUI—X User Interface). Licensing XUI will allow ISVs to develop a DECwindows style application on any supported processor. XUI is also Digital's offering to OSF for the standard user interface.

XUI consists of four main parts:

- The XUI Style Guide. The style guide is simply a document describing interface conventions for programmers. In this, it is comparable to IBM's Common User Access (CUA) specifications within Systems Application Architecture (SAA).

- The XUI Toolkit. The toolkit is Digital's own X toolkit, layered atop Xlib and the standard XT intrinsics. The XUI kit contains Digital's widgets.

- The Window Manager.

- The User Interface Language (UIL). UIL separates the interface from the application, allowing developers to create different versions of the interface (either for different application scenarios or for native language implementations) without touching the basic application source code.

XUI does not offer the same set of core applications as are offered within DECwindows. XUI's goal is to support application portability across platforms, not to create a cloned DECwindows environment.

Digital Strategy for DECwindows

DECwindows is not just a product to make some Unix hackers happy. It is a fundamental piece in Digital's strategy, particularly addressing the needs of the desktop. Key points include:

DECwindows is not just a product to make some Unix hackers happy. It is a fundamental piece in Digital's strategy, particularly addressing the needs of the desktop.

1) A universal environment as the front end to Digital's distributed network. Digital is using DECwindows to provide consistency across all its supported workstation and operating system platforms; DECwindows offers a single, bit-mapped graphics user interface for VMS, Ultrix, DOS, and the Macintosh. It will be a bundled, basic component of VMS and Ultrix systems. It's hard to stress this point too much. DECwindows provides a common look and feel across these operating system platforms. Given the increased attention to Unix that all vendors are paying, such consistency is indeed an important capability.

Digital released an early version of DECwindows in December 1987 and put about 250 ISVs and 50 end users through a three-day course teaching DECwindows programming. Some 80 percent of these said they will develop DECwindows applications; of those, more than 100 are working on development now.

Applications development is fairly balanced: 60 percent VMS, 40 percent Ultrix.

As a counterpoint, look at IBM's strategy with AIX and SAA. SAA is the common environment defined for IBM's proprietary systems; AIX is its standards-based Unix implementation. At this point, there is no product that offers a common user interface across SAA and AIX.

2) DECwindows is the design center for all new applications development.

3) DECwindows is the integration vehicle for all desktop devices.

THE DESKTOP STRATEGY. The integration of desktop devices is also critical for Digital. Many observers (analysts—ourselves included—consultants, even some Digital customers) regularly lambaste Digital for its lack of a desktop strategy. Much of this criticism stems from Digital's track record with PCs.

Now isn't really the time to delve into the strengths and weaknesses of Digital's past desktop approach. Rather, we'll just sketch the role DECwindows will play in the emerging strategy.

Basically, Digital has two primary types of market to attack with its DECwindows-based approach: loyal Digital clientele and everybody else. The advantages of DECwindows for Digital accounts are evident and need no further explanation.

With the others, however, Digital has two types of opportunities.

First, it can integrate other PC platforms into its network and provide those machines with X display server capability. Such an implementation will allow any DOS PC or Macintosh to display the remote applications running either on VMS or Ultrix within the network.

Within that scenario, there are two possible ways of implementing the X server: as a layered product directly on top of the operating system or as a direct application running in an environment such as MS Windows. Variables include, for example, the amount of memory required for a specific type of implementation. In other words, you will be able either to run an X display server application from within another environment, such as MS Windows, or just run an X Server application that turns the entire PC into an X display terminal. In the first instance, you would have access to other MS Windows applications (given memory availability) concurrent with the X display application. In the latter case, the X server would be the only application running.

Such configurations are indeed part of Digital's plans, and they fall within the category of Network Application Support (NAS).

The other market opportunity arises from the ability of Digital to actually capture new desktops with its long-awaited, rumored, etc., etc. Personal VAX PC (PVAX). Should Digital actually come out with a potent $5,000 workstation, then it could, at least in theory, have a shot at those desktops in transition—the desktops that are exploring the possibility of upgrading hardware and software to move to OS/2.

Digital has two primary advantages. First, it appears to have done a good job. Second, it has done the job of implementing X across both its strategic platforms.

From a systems-oriented point of view, the argument makes some sense. Digital would be offering its consistency, its networking, its compound document architecture, and DECwindows as platforms for general purpose workstations in an enterprise environment.

(The argument makes even more sense now that PC clones vendors are sparring with IBM over the design of a 32-bit bus architecture, introducing yet more uncertainty into the market.)

The cost for a new Digital desktop would be approximately the same as for an OS/2 machine that would perform the same type of function, and Digital has many of the pieces in place that the OS/2-based approach does not.

One critical factor in this approach is the application strategy. Digital needs to convince enough major ISVs to write for VMS or Ultrix to be able to offer an application library comparable to that available in the PC world (including some of the same titles). We think that making the argument for an Ultrix port would be easier than making one for a VMS port. Many of these PC ISVs are looking at Unix ports anyway.

Digital, accordingly, is making a very strong effort to entice ISVs into the DECwindows camp. Applications were the factor that drove PCs to popularity, after all.

This latter scenario is not one for global domination. However, it could give Digital a much more sizable share of the desktops, particularly in larger businesses.

OS/2 CLIENTS? Although it certainly is technically possible, Digital is not planning to implement client-side services on OS/2 machines. Offering an X server would be within the bounds of NAS. Offering X client applications would put Digital in the business of regarding OS/2 as one of its primary operating systems (along with VMS and Ultrix.) It would also open up OS/2-based competition for those desktops.

DOS CLIENTS? Another intriguing technical possibility would be to provide a DOS client implementation through DOS emulation on a VAX. Digital has said nothing about actually providing such a beast. However, if done, that would provide X-based access to DOS applications from any X server in the network. Interesting possibilities.

FUTURES. Digital has a long list of next steps, among them:

- The completion of international applications versions
- The complete transformation of existing applications and the design of new applications
- Adding in the specified extensions
- Support for new hardware

SUMMARY. In many ways, any vendor opting to build a strategic approach atop X-Window is going to have a story that sounds much like its competition's. Unless you decide to defeat the purpose of building on a standard platform, then you, as a vendor, will be using the X Consortium foundation upon which to build differentiation in certain areas. Competitive distinction then comes in when those differentiators are combined with the other elements of the total systems offering: compound document architecture, object orientation, distributed networking, etc.

Digital is accomplishing two things with its ushering of DECwindows into the market in such a timely fashion. First, it is addressing the current and future needs of its customers. Second, it is luring ISVs, all the better to bolster its desktop strategy.

Digital is not out to turn every vendor's environment into a clone of DECwindows. Rather, it is out to plant the real thing—VMS-based or Ultrix-based—on as much territory as it can occupy.

Digital also now has two primary advantages. First, it appears to have done a good job (not an unimportant consideration). Second, it has done the job of implementing X across both its strategic platforms. Being first with such an environment counts.

Unaddressed is the issue of OS/2 as competitive platforms. Hypothesizing that someone will do a Presentation Manager (PM) version of Unix, a user interface atop X that used PM as its model would provide a consistent user interface that embraced the PM-OS/2 world as well. Given the likely penetration of PM machines, such a consideration is not trivial.

Even granting the presence of such products, however, Digital has pretty much of a clear field between now and whenever all the PM pieces are in place.

The Sixth Annual Seybold Executive Forum:

TEAMWORK, TECHNOLOGY, & ORGANIZATIONAL PERFORMANCE

October 25 – 27, 1988

Royal Sonesta Hotel

Cambridge, MA

This year's topic is *"Teamwork, Technology & Organizational Performance."* Join us as we explore these issues with the computer industry's leading executives. Hear first-hand what their visions are for future technology directions. Hear their response to the challenges today's business and government organizations are facing. Learn why they are banding together with their competitors in often-surprising alliances to address customer needs. And interact with them in a relaxed, informal dialogue—no canned speeches, just straight talk.

DAY 1: TUESDAY, OCTOBER 25, 1988

In Search of a New Paradigm
The Office Computing Group
Michael D. Millikin, Vice President, Chief Technologist

The Dawning of a New Age in Computing
Sun Microsystems
William Joy, Vice President, R&D

Digital Equipment Corporation
William Strecker, Vice President, Architecture & Product Strategy

The "New" Organization Through Information Technologies
International Center for Information Technologies
Dr. Peter G. W. Keen, Executive Director

Cooperative Computing for Competitive Advantage
Hewlett-Packard
John Young, President

Measuring the Productivity Payoff
Computer Aided Manufacturing-Int'l.
Thomas E. Pryor, Program Manager

Chevron Chemical Co.
Don W. Matto, MIS Manager

DAY 2: WEDNESDAY, OCTOBER 26, 1988

Restructuring American Business
IBM
George Conrades, General Manager, IBM-U.S., Marketing & Services

Managing for Multiple Constituencies: Stakeholder Management
NCR
Charles E. Exley, Jr., Chairman, CEO

The Payoffs from Computer-Supported Cooperative Work (CSCW)
Institute for the Future
Robert Johansen, Director, New Technologies Program

Groupware User Panel

Approaches to CSCW
Coordination Technologies
Anatol Holt, Chief Technical Officer

Lotus Development Corporation
James P. Manzi, President, CEO

MIT—Sloan School of Management
Thomas W. Malone, Professor of Information Technology & Management

Moving Toward "Real-Time" Businesses
Stan Davis, Author, *Future Perfect*

Managing Change
Index Systems
Tom Gerrity, Chairman, CEO

Xerox Corporation
Paul A. Allaire, President

DAY 3: THURSDAY, OCTOBER 27, 1988

Technology: The Trojan Horse for Organizational Change
The Office Computing Group
Patricia B. Seybold, President

On the Road to "Real-Time" Systems: Monitoring the Business Environment
Salomon Brothers Inc
Edward Boyhan, Vice President & Manager of Advanced Technology Group

Information Systems as Change Agent
Nolan-Norton
David Norton, Principal

Partnering for Progress
McKinsey & Company
Ian Somerville, Partner

User Panel

Planning for the Future Today
ON Technology
Mitch Kapor, Chairman and CEO

Concurrent Sessions:

Technical Sessions:

Object Oriented Environments
The Office Computing Group—Michael Millikin, Vice President, Chief Technologist

Data General—Chris Stone, Manager, Office Systems Software

Metaphor—Dave Liddle, Chairman

Wang—Jean Friedman, Department Manager, Wang Core Applications

Hewlett-Packard—William Crow, NewWave R&D Manager, Personal Software Division

Moving Toward EDI
Soft•Switch—Michael Zisman, Chairman

The Role of Standards
The Office Computing Group—Judith Hurwitz, Senior Consultant

The National Bureau of Standards—James H. Burrows, Director, Institute for Computer Science & Technology

X/Open—Robert Ackerman, Chief Marketing Officer

Open Software Foundation—Alex Morrow, Director of Strategic Relations

Leveraging ISDN
Northeast Consulting Resources, Inc.—Jim Herman, Telecommunications Consultant

Teamwork & Technology Workshops:

Simulating Organizational Dynamics
Digital Equipment Corporation—Skip Walter

Coordinating Actions
Action Technologies—Russ Redenbaugh

Inventing The Future
Northeast Consulting—David Mason

Building Consensus
Wilson Learning—Kalen Hamman

The Natural Planning Model
Insight Consulting—David Allen

CALL TODAY!

(617)742-5200

SPACE IS LIMITED TO 300 PARTICIPANTS.

☑ PLEASE REGISTER ME!

Discounts are available for two or more attendees from the same company. For information, call Deborah Hay at (617) 742-5200.

Send registration form and payment to:

The Seybold Executive Forum
Patricia Seybold's
Office Computing Group
148 State Street, Suite 612
Boston, MA 02109
(617) 742-5200
Fax #: 1-617-742-1028

☐ My check is enclosed for $1095 (Please make check payable to Patricia Seybold's Office Computing Group)

☐ Charge to my:
MasterCard/VISA/American Express (circle one)

☐ My purchase order number: ____________________

Card Number ____________________ Expiration Date ____________________

Signature ____________________

Name ____________________ Title ____________________ Company ____________________

Address ____________________ City, State, Zip ____________________

Telephone Number ____________________

Cancellation Policy:
Should a registrant be unable to attend the Forum, the Forum Office will refund the full registration if notified before September 27. Cancellations from Sept. 28 to Oct. 11 are subject to a $50 service charge. There will be no refunds as of Oct. 12. Substitutions may be made at any time.

NEWS ANALYSIS

PRODUCTS • TRENDS • ISSUES • ANALYSIS

• INSIDE •

• APPLE •

Apple Xtends Mac II

Ever since the Macintosh II was introduced, Apple has viewed it as the company's entry into the hot technical workstation market. Along with its commitment to develop a Unix-based operating system for this product, Apple has made the Mac II the core of its strategy to penetrate large corporations, universities, and the government.

Apple, until now, has been less than successful in delivering a full-blown version of its Unix—A/UX. However, the company still maintains its commitment to A/UX and its goal of penetrating the technical workstation market. To this end, Apple has announced a new version of the Macintosh II, one which has greater multitasking and connectivity abilities.

Apple's first upgrade of its Mac II, the Macintosh IIx, features a 68030 CPU and a 1.44MB floppy disk drive that can handle Mac, DOS, OS/2, and Apple II files. The long-rumored 68030-based machine is aimed at advanced users, particularly those requiring multitasking operating systems, initially A/UX. (We expect that Apple will eventually follow by also implementing a true multitasking Macintosh Operating System.)

NEW PROCESSOR AND MEMORY MANAGEMENT. The heart of the Mac IIx is the Motorola 68030 32-bit microprocessor running at 15.7 MHz. The 32-bit NuBus address bus provides a total addressable space of 4 gigabytes.

However, the major benefit of the 68030 does not lie in its speed, as it is only about 15 percent faster than the 68020. Rather, the 68030 provides a built-in Paged Memory Management Unit (PMMU) which supports virtual, shared, and protected memory. This is necessary to run multitasking operating systems, such as A/UX. With the current Mac II, a separate PMMU chip set had to be added to the motherboard in order to run A/UX. Thus, the 68030 can provide significant cost savings over the 68020.

SUPERDRIVE. While possibly less sexy than the use of the 68030 processor, the Mac IIx's new floppy disk drive, the FDHD—Floppy-Disk, High-Density, is ultimately more exciting. Dubbed SuperDrive, the 1.44MB drive increases floppy capacity by 80 percent, equalling that of the IBM PS/2 line. More importantly, FDHD uses a new proprietary controller chip, the SWIM (Super Wozniak Integrated Machine) chip—a superset of the IWM (Integrated Wozniak Machine), which allows the drive to read, write, and format all Mac, MS-DOS, OS/2, and PRODOS (Apple II) disk formats. This means that a Mac application that can read DOS files can read them directly from a DOS diskette.

The SWIM controller, which maintains current throughput in all formats, does not increase disk performance.

OTHER FEATURES. The Mac IIx also features 256K of Single In-line Mounting Module (SIMM) ROM, which can easily be removed and replaced for more convenient configuration, servicing, and upgrading. A new floating-point coprocessor—the 68882—offering up to twice the speed of the 68881 while maintaining applications compatibility with it, has been included.

PRICING AND AVAILABILITY. The Macintosh IIx will be delivered in two configurations. The basic system includes the 68030 CPU, 68882 coprocessor, 4MB of RAM, the 1.4MB floppy disk drive, and a mouse. It is priced at $7,769. The price with an internal 80MB hard disk drive is $9,369.

(In comparison, Apple just announced new pricing for the 68020-based Mac II—a Mac II with 4MB of RAM and a 40MB internal hard drive now lists for $7,269.) Neither price includes monitor, keyboard, or other peripherals. Apple plans to have both Mac IIx configurations available immediately.

Apple foresees a great demand from current Mac II users to upgrade to IIx capabilities. The logic board upgrade (68030 CPU) will be $2,199. The FDHD drive upgrade kit (including SWIM chip and drive) will cost $599. Both will have to be installed by a dealer. Due to the scarcity of DRAM and the anticipated high initial demand for the Mac IIx itself, the upgrade kits will probably not be available before November.

WORKSTATION WARS. Apple has succeeded in beating companies such as Sun and NeXT (but not Apollo or HP) to the 68030 punch. The question for Apple will be twofold. First, can the IIx and its future brothers stack up to the Suns, Apollos, and NeXTs of the world? The first Mac IIx seems to fall a little short. For instance, Apollo has used 25 MHz versions of the 68030 processor in its DN3500 and DN4500 personal workstations, which are priced competitively with the Mac IIx. At $7,990, the low-end 4 MIPS DN 3500, like the Mac IIx, includes 4MB RAM and a 68882 floating-point coprocessor (though Apollo's runs at 25 MHz). The under-$8,000 configuration is diskless, but includes a 15 inch display, keyboard, and Ethernet or Token-Ring adapter. (At press time, the prices and configurations of NeXT's workstations had not yet been announced.)

Equally important to an MHz-by-MIPS-by-dollars comparison is the question of whether these comparisons will ever be made by potential purchasers. Apple's job will be to make users perceive these new Macs as competitors to their current favorite technical workstations. This may be far more difficult than producing another hot box.

—D. Marshak

•RISC TECHNOLOGY•

A Smart RISC

We've been hearing a lot about what Texas Instrument's (TI's) licensing of Sun's Scalable Processor Architecture (SPARC) will do for Sun's credibility in the Reduced Instruction Set Computer (RISC) market. And, yes, the addition of TI to the SPARC camp should give Sun some pull in its quest to become standards-maker of the workstation world. But, more importantly, the long-term licensing agreement should give the SPARC chip a real shot in the arm. Sun and TI will work together to develop the next-generation SPARC microprocessor, which means that future SPARC machines will not only be powerful, but smarter, too.

TI's semiconductor division will co-design and manufacture SPARC chips, and its computer division will use SPARC as the core of a line of artificial intelligence-enhanced, high-end, commercial AT&T Unix System V workstations.

TI also made a five-year, second-source agreement with Cypress Semiconductor for SPARC components. In exchange, TI has given Cypress the resale rights to its 8847 Floating Point Unit. The agreement lets both companies manufacture complete SPARC chip sets. And Cypress, in turn, will get to second-source the next-generation TI/Sun SPARC processor when it's available (although TI SPARC workstations aren't expected for two years).

Sun's open architecture was its major draw. Sun hardly has the RISC market all sewn up, but it carries support from some well-known companies, such as Unisys, Xerox, and Fujitsu. TI can capitalize on that. The two companies hope that TI's support will give SPARC even more support; thus, TI would reach an even broader spectrum of platforms.

TI's plan is to use SPARC to merge artificial intelligence and conventional processing technologies on a Unix-based multiuser system at a low cost—$1,000 to $2,000 per user.

TI's agreement with Sun also includes the SunOS operating system, C language and Fortran compilers, X11/NeWS windowing system, and the Sun Network File System (NFS). We've also heard that Sun has committed to buy a wide variety of semiconductors from TI.

So, while the TI/Sun licensing agreement adds another dimension to the recent hullabaloo surrounding RISC, in the grand scheme of RISC architecture, the agreement may prove to be a leap in technology. The combination of an open RISC architecture with sophisticated artificial intelligence capabilities will be an interesting technology, not to mention an interesting marketing opportunity for TI.

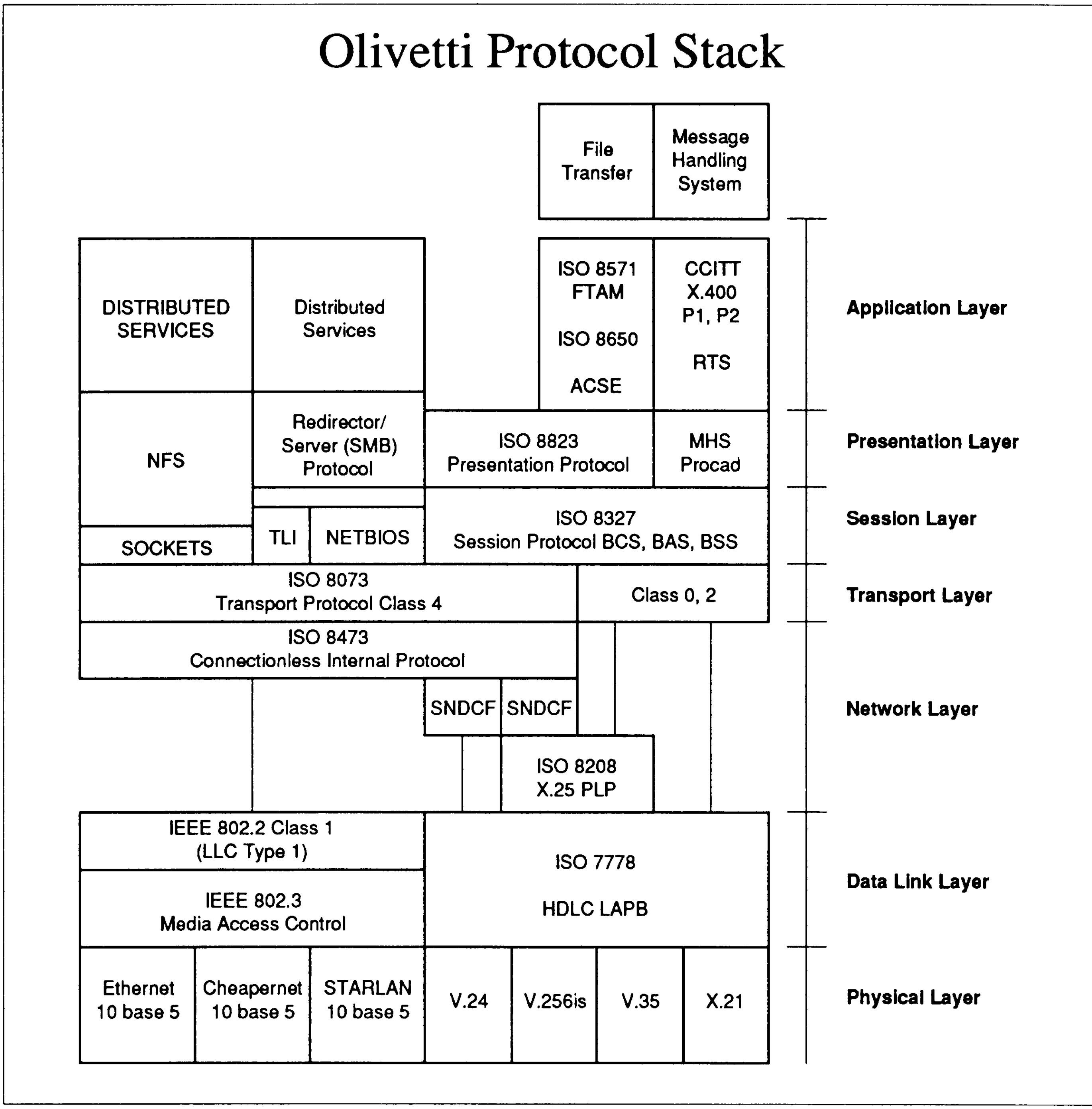
—L. Brown

•OLIVETTI•

Olivetti Defines a Leading-Edge Architecture

It seems we underestimated Olivetti's Open System Architecture (OSA) when we first wrote about it in June. Herewith, an update and an expansion on an architectural and application approach that is a leading-edge solution in implementing a distributed network computing architecture built atop a mixture of standard and proprietary platforms.

DESIGN AND INTENT. OSA is indeed a definition of interfaces and protocols. However, as it also associates those interfaces, protocols, and conventions with specific hardware, OSA becomes a very practical way to generate what Olivetti calls a "Certified Integrated System." OSA specifies a set of software functionality associated with specific hardware components and interfaces to guarantee application inde-



pendence from the current implementation of the products.

Approached from this perspective, OSA thus becomes a much more effective tool for defining an architecture (embracing both the conceptual and the application aspects thereof) than does IBM's SAA, which, after all, merely covers the application aspect of the architectural specification.

DISTRIBUTED NETWORK COMPUTING AND THE CLIENT/SERVER ARCHITECTURE. Olivetti has gone much further in moving toward distributed network computing based upon the client/server model than we first reported.

OSA's model supports the distribution of computational management across the network. In other words, a client/server application can run either locally or distributed across the network (client PC, Unix server). Olivetti regards these client/server environments as the migration units of the application across the LAN/WAN network.

Olivetti's remote procedure call functions across PCs running DOS and the LSX Unix mini, and PCs with the

MTX Unix transaction handler within the OLTP environment. The Commit Manager supports the consistency of the entire distributed application.

This support for distributed services is not based upon the Named Pipes facilities in OS/2 LAN Manager or in LM/X, and for good reason: Those products are not yet out in the marketplace. Olivetti has gone ahead and implemented its own solution that bridges the DOS world with the Unix and MOS environment. Currently, it is also at work on integrating OS/2, using its currently available solution.

At such time when incorporating Named Pipes is feasible and necessary, given availability of third-party applications based upon it, we would hope and expect Olivetti, with its open systems approach, to do so. Olivetti currently is a licensee of the OS/2 LAN Manager.

OLIVETTI'S OFFICE INFORMATION SYSTEM. OIS is Olivetti's strategic architectural framework for the creation, manipulation, and management of information, and is the alternative to "generic" office automation, which is represented, within Olivetti's product line, by products such as Q-Office.

Olivetti sees a transition in the office toward providing generalized, transparent access—that is, manipulation and distribution of information from any place on the network to any user on the network (LAN/WAN). Migration to this distributed network architecture requires a reliance upon networking and communications, adherence to standards, a different set of tools for managers and professionals, and, from the vendor, emphasis on service and support. We concur heartily.

Olivetti's OIS is modular, thus accommodating future growth. However, it also currently integrates the following:

- X_MANAGER, the environment that runs with the same interface on PC/MS-DOS, TTY, and a PC with graphics capabilities (with windowing, icon, and mouse support). X_MANAGER includes a set of routines (X_TOOL) to handle the interaction between the OPT and the required OCS. X_MANAGER also has a keystroke recording facility for all supported components.

- Office Productivity Tools (OPT), a set of end-user applications, including word processing, spell-checking, a spreadsheet, and a database manager, all integrated within X_MANAGER.

- Office Cooperation Services (OCS) for file-sharing, distribution of data objects, and electronic mail in a local environment. OCS handles the objects produced by the PCs using OPT. The OCS are: an X.400 conformant E-mail offering; Index, a file/retrieval system; and Object Distribution Services (ODS). ODS consists of Object Distribution Services, DISOSS-compatible mail packages in PC-Host and LAN-Host configurations.

- A DP environment for professionals that includes software products designed to optimize cooperation between the DOS and Unix environments, and databases such as Oracle and Informix.

- Communications and networking support.

OIS specific products include:

- PROCEDURE, which creates and manages individual procedures in the X/OS and DOS environments. PROCEDURE consists of a run-time interpreter under X_MANAGER, panels created through the ToolKit component, activation under DOS and/or Unix (X/OS), integration with the basic functions provided by the different OIS components (send, receive, search, insert, etc.), and integration with the Shell language.

- X_DATA VIEW, which initiates queries on the LSX DBMS by PC users with a spreadsheet style interface.

- X_DATA MERGE, which controls a batch procedure to merge OLISOFT WP documents with data coming from the DP environments.

- X_PRINT, an interface to the Unix spooler.

- X_NOTE, a notepad.

- X_ADMIN, OIS administration.

- X_PM Windows (not Presentation Manager, but a character-oriented windowing environment).

NETWORK ARCHITECTURE. Olivetti is supporting standards at all levels of its network stack, implementing a full range of the OSI protocols in both LAN and WAN configurations. On the LAN side, the company has added support for TLI, NETBIOS and Sockets on top of the Transport layer. Its support in the application layer includes FTAM, X.400, ACSE, and RTS.

In short, Olivetti offers a complete OSI-based network running on a variety of platforms (Ethernet, StarLAN, and Token-Ring) and extended all the way down to the PC-DOS level. The flexibility of these configurations goes beyond what other vendors currently offer, and supports Olivetti's contention that OSA is a flexible and open architectural framework that can meet a variety of specific needs with specific products.

CONCLUSION. Olivetti has done a detailed and thorough job in creating a distributed-network computing environment that supports DOS clients and standard and proprietary servers (Unix and MOS). Although other vendors have produced client/server architectures that integrate PCs (HP's Cooperative Services for one example), Olivetti has gone beyond that by extending that client/server support to include Unix products within the same architecture.

Given its commitment to standards, we are confident that Olivetti will continue to enhance its OSA offerings to keep pace with developments in other areas of the market—graphic user interfaces, for example, and support for X-Window. —*M. Millikin*

•PROGRESS•

More Progress: Unix/DOS Networking

Progress Software Corporation is beginning to make good on the future plans it described when we reviewed the Progress fourth-generation language (4GL) and relational database management system (RDBMS) earlier this year (see Volume 3, Number 2, February 1988). The company has just introduced its first support of distributed processing among heterogeneous platforms, allowing versions of Progress running on Unix, Xenix, and DOS to interact on a single network.

Progress already runs on DOS- and Unix-based (Berkeley 4.2) LANs with a separate front end and back end. However, these versions are restricted to LANs where the hardware and operating system are homogeneous (e.g., all PCs running DOS, all NCR Towers running Unix, etc.).

Now, the customer can configure a network where the back-end database server runs on either a Unix or Xenix platform and the front-end client process can run on DOS, Xenix, and/or Unix platforms. The initial protocol supported for heterogeneous networking is TCP/IP on Ethernet.

Thus, Progress will support customers who want to connect Unix, Xenix, and DOS users of Progress in a single network and allow them to share both data and applications.

PACKAGING. On multiuser Unix and Xenix platforms, Progress Software will offer heterogeneous networking as a standard feature in version 4.3 of Progress.

For single-user workstations, such as DOS, Progress is unbundling the client software and will offer this with networking support as a separate product. The same will be done for other workstations, such as Sun, Apollo, and the IBM RT, for customers who wish to use the workstation only as a single-user front end. The cost for the front-end software will be between $1,000 (on DOS) and $3,000 per workstation. (The company does not plan to build the heterogeneous networking feature into full Progress for DOS.)

Version 4.3 with heterogeneous networking will be available in November on the following multiuser platforms: Pyramid, Plexus, and Sun 3, 4, and 386i. There is no change in price over Version 4.2, and current customers on maintenance will receive 4.3 at no charge. The front-end software will also be available for DOS workstations. Over time, the company plans to implement Version 4.3 on all of its Berkeley 4.2-based platforms and on others where there is TCP/IP support (e.g., an Excelan or comparable solution is available).

MORE COMING. The next step for Progress Software is the introduction of Fast Track, a set of menu-driven application development tools for both professional developers and end users. Fast Track, which uses a WYSIWYG approach, includes a report writer, screen painter, menu editor, and query-by-forms generator. We will cover Fast Track in next month's news. Then, in early 1989, Version 5 of Progress will debut, sporting ANSI Level 1 SQL compatibility as well as performance enhancements. —*J. Davis*

Topics covered in Patricia Seybold's Computer Industry Reports in 1987/1988:

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

Office Computing Report

1987—Volume 10

#	Date	Title
7	July	End-User Computing—Tools for Visionary Users
8	Aug.	Will Wang Make it? Can Fred's Leadership Turn the Tide
9	Sep.	Voice Messaging Systems—User Perspectives
10	Oct.	IBM Office Software—Implementing A Cohesive Vision
11	Nov.	User Developed Applications—They Know What They Need
12	Dec.	Hewlett-Packard's New Wave

1988—Volume 11

#	Date	Title
1	Jan.	Office Computing: 1988
2	Feb.	Mac Attack
3	Mar.	Electronic Data Interchange—The New Fontier of Electronic Mail
4	Apr.	PC LANs Revisited
5	May	Remission or Recovery—Will NBI's New Direction Bring Renewed Life?
6	June	Service And Support *Second Feature:* LAN Support

UNIX in the Office

1987—Volume 2

#	Date	Title
7	July	Texas Instruments—Blending AI with Commercial Systems
8	Aug.	Banyan Systems—A Unix Solution to the PC LAN Conundrum
9	Sep.	AFCAC 251—Defining the Future Unix Office
10	Oct.	The Oracle Relational DBMS—The Leader of the Pack
11	Nov.	Expres Initiative—Is there an Office Use for University Communications
12	Dec.	Unix Roundup—Unix-Based Integrated Office Systems

1988—Volume 3

#	Date	Title
1	Jan.	Beyond Unix—Creating the Standard Operating Environment *Second Feature:* X/Open Champions Open Market
2	Feb.	Progress
3	Mar.	Pyramid Technology
4	Apr.	IBM's Unix
5	May	AI and Unix
6	June	SmartWare from Informix Software—The Quiet Contender

Network Monitor

1987—Volume 2

#	Date	Title
7	July	Apollo's New Thrust—Shaping Computing's New Generation
8	Aug.	Proteon Inc.—A Better Mousetrap Isn't Enough
9	Sept.	Digital's DecNet
10	Oct.	Data General—Better Late Than Never
11	Nov.	IBM and SNA—Meeting the Challenge?
12	Dec.	Micom Interlan Inc.—Learning From Past Mistakes

1988—Volume 3

#	Date	Title
1	Jan.	Taking Stock, Looking Ahead *Second Feature:* Mac to PC: Making the Connection
2	Feb.	Excelan—Riding the Crest with TCP/IP *Second Feature:* Update on Banyan
3	Mar.	3Com—Opening a New Window
4	Apr.	OS/2 Networking
5	May	Hewlett-Packard—Riding with the Wind
6	June	Exploring Distributed Network Computing