The Guide to Open Systems

VOL.6, NO. 2
ISSN: 0887-3054
FEBRUARY 1991

INSIDE

EDITORIAL

NEWS ANALYSIS

PATRICIA SEYBOLD'S

UNIX IN THE OFFICE

Desktop Managers

The Commercial User's Access to Unix

By Laure B. Rowan

IT'S NOT SURPRISING that the commercial evolution of Unix took some 20 years in the making. Unix has a notoriously cryptic system interface. Merely editing a command can be a convoluted process—too much so for commercial end users. That's part of the reason the user interface issue is such a heated one in the Unix marketplace. Since Unix is so obtuse, the need for a more intelligible, standard graphical user interface is especially crucial.

While no single X-based interface technology has emerged as a standard, major Unix *(continued on page 3)*

• E D I T O R I A L •

The Role of The Open Software Foundation

If the U.S. government puts handcuffs on the RFT process, the quest for open systems becomes more complex.

By Judith S. Hurwitz

COMPARED TO previous years when UniForum was the stage for major ideological fights between AT&T, Sun Microsystems, the Open Software Foundation (OSF), and Unix International—to name a few—this year's conference was relatively peaceful. However, one storm cloud did linger over the event. A rumor published in an industry newsletter noted that OSF was being investigated by the Federal Trade Commission (FTC) for restraint of trade because of its Request for Technology (RFT) process. According to the report, a particularly unhappy small independent software vendor (ISV) felt that the RFT process was restricting the ability of all small ISVs to compete fairly in the open market.

I suppose there is some merit to this argument. I can imagine the frustration of a small company that had spent millions developing a technology only to find its technical approach superseded by a suddenly approved "de facto" standard. On the other hand, consider the alternative: to let the marketplace decide which technologies will succeed and which will fail. (In other words, to do what we've always done in this industry.) If we proceeded with that model, then we'd expect to see a technology such as VisiCalc be overpowered by the emergence of a Lotus 1-2-3. Likewise, it stands to reason that a trend-setting word processor such as WordStar would be overshadowed by a new entry such as WordPerfect.

Why not let the law of the marketplace continue to prevail? Why mess with the way technology emerges? Because the need for standards and openness has changed the technology requirements. If users are to be able to implement standards-based infrastructures, the industry must be able to standardize on common programming interfaces and on common base technologies such as protocols and remote procedure calls. This needs to happen in a deliberate and well-planned way. It cannot be left to chance. It cannot be determined by whatever vendor happens to have the best marketing force and the most money to pour into marketing a technology.

Is the RFT process fair? In reality, it is the fairest way to quickly create de facto technology. It is the best way for users to gain access to the technology they need to make open systems work. The alternatives are just not viable in an age where openness is the requirement.

What about the issue of how technology is priced? According to press reports, the FTC inquiries are directly related to whether or not OSF is forcing RFT winners to accept low compensation for their technology. Our understanding is that OSF is not. In fact, if a vendor's technology is licensed to OSF and then is widely implemented, that vendor stands to make a lot more money than it would have in the open marketplace. Therefore, while the threat of the FTC investigation will cause OSF detractors to applaud, it will not impact OSF in the long run.

But what if it does? What if the FTC decides that this RFT process is unfair to ISVs? I believe that such a decision would seriously set the open systems movement back several years. It would, for example, force AT&T Unix Systems Laboratories to rethink the way it doles out pieces of its technology. Hypothetically, AT&T could also be accused of not paying its partners enough for its technology. Is Pyramid Technology paid enough money for its contributions to the new System V.4 kernel? Will this pricing negatively impact Sequent? A negative FTC decision could also stifle cooperation among competitors in the computer industry for fear of retribution. Just as it is dangerous to let outsiders decide on how user interface technology should be copyrighted, so it is not wise to ask a federal agency to determine how we arrive at an open systems infrastructure for the '90s.

PATRICIA SEYBOLD'S UNIX IN THE OFFICE

Editor-in-Chief
Judith S. Hurwitz
MCI: 3538836@mci.com
Internet: hurwitz@dcm1tp.das.net

Analysts and Editors
JUDITH R. DAVIS
DAVID S. MARSHAK
RONNI T. MARSHAK
MICHAEL D. MILLIKIN
LAURE B. ROWAN
JOHN R. RYMER

Managing Editor
DOUGLAS A. FREEMAN

News Editor
DAVID S. MARSHAK

Art Director
LAURINDA PHAKOS

Publisher
PATRICIA B. SEYBOLD

Sales Director
RICHARD ALLSBROOK JR.

Circulation Manager
DEBORAH A. HAY

Customer Service Manager
DONALD K. BAILLARGEON

Patricia Seybold's Office Computing Group 148 State Street, 7th Floor, Boston, Massachusetts 02109
Telephone: (617) 742-5200 FAX: (617) 742-1028 MCI: psocg / 312 2583
Internet: psocg@dcm1sg.das.net TELEX: 6503122583

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, 7th Floor, Boston, MA 02109. Second-class postage permit at Boston, MA and additional mailing offices. POSTMASTER: Send address changes to *Patricia Seybold's UNIX in the Office*, 148 State Street, 7th Floor, Boston, MA 02109.

•DESKTOP MANAGERS•

(continued from page 1) vendors have at least been implementing either OpenLook- or Motif-based environments in an attempt to create an easier, less intimidating system interface. The problem is that neither OpenLook nor Motif alone is a graphical user interface (GUI). They are both GUI programming libraries, and a library does not a graphical user interface make. Rather, libraries are used by application developers to create interfaces. They lend to the system windows, buttons, scrollbars, and the like, but, unfortunately, users are still at the mercy of the system shell for accessing utilities, typing commands, and manipulating files and directories. Only now the shell is in a window, and that doesn't help its interface much.

Unless your system has a desktop manager, that is. Essentially, a desktop manager sits above the window manager and masks such functions behind an intuitive interface. It's a front end to the system—a usable interface. In a graphical desktop manager, objects in the system, such as tools, programs, data files, and directories, are displayed as icons.

Most Unix vendors do indeed include some form of desktop management. But the solutions vary considerably from implementation to implementation. Some vendors consider a desktop manager nothing more than a file browser; others see in desktop management an avenue for application integration and even interapplication communication. Of course, it's the latter perspective that is most intriguing. However, as we'll illustrate later, it would mean the desktop manager adopting object management facilities for exchanging commands and data among applications. Currently, Unix-based user interfaces haven't evolved quite so far, although they are taking on elements of application integration as well as procedural automation capabilities, where desktop operations can be linked together.

Some vendors consider a desktop manager nothing more than a file browser; others see in desktop management an avenue for application integration and even interapplication communication.

Both Motif and OpenLook relegate the desktop management component to individual implementers, leaving room for vendor differentiation. Several vendors have developed their own desktop management component, while others—IBM, Data General, SCO (OpenDesktop), to name a few—have turned to third-party products, namely X.desktop from IXI Limited (Cambridge, England) and Looking Glass from Visix Software (Reston, Virginia). But no one desktop manager stands out as a definitive solution. Although they have much in common, desktop management implementations seem to take different approaches:

- Apple has been developing and perfecting interfaces for years, and its expertise is certainly evident in the Finder tool for A/UX. Too bad it's proprietary. Because of its GUI success and reputation, if Apple ever considered opening up the product and licensing it (highly improbable), Finder could have a huge potential as the de facto desktop manager standard.

- IXI stresses the configurability of its system. IXI wants its customers to make X.desktop exactly what they want it to be, and has developed a robust rule-programming language for configuring the product.

- Hewlett-Packard's basic desktop environment for Unix is VUE (Visual User Environment), which gives the system an iconic representation of the file system, a style manager for customizing the environment, a Help facility, and a workspace manager. Ultimately, HP will couple VUE with NewWave, which will make VUE an object-oriented user environment complete with agents and object management.

- Visix emphasizes the completeness of its system. Looking Glass is suitable for both novices and more sophisticated users—although truly technical users will always prefer the Unix system shell. It also provides a file definition programming language for integrating files and programs to the desktop.

- End-user desktop functionality is just part of Sun's OpenWindows, and much of that functionality comes from the DeskSet tools component of OpenWindows. Interestingly enough, although Sun is offering the source code of most of its OpenWindows components, it's holding onto the DeskSet tools—at least for the time being.

- NCR has put together a true object-oriented desktop system for its new Cooperation environment. The company has wisely adopted the best technology of a number of vendors (including that of Hewlett-Packard and IXI) and has implemented some very creative and practical features, including multiple desktops.

THE NEXT PHASE OF DESKTOP MANAGEMENT. These are the more prominent and interesting desktop management offerings we've been seeing in the Unix market. But they're still in an evolutionary state. Five years from now, we expect desktop managers to be very different from what they are today. Not only will they be more graphical and visual, but they will also better represent the reality of your work environment. Desktop manager developers could actually learn something from video and computer games, because many of them have such a clear sense of purpose and a completely intuitive interface. Recently, for instance, one of our analysts was describing a computer game belonging to her son called "The Playroom." The interface is an animated representation of an actual playroom with a

computer, a bookshelf that holds a bunch of games, a clock, etc.—even a talking parrot. To get a new game, the child goes to the bookshelf and clicks on the game. If the child wants a keyboard-based game, he moves to the computer. To get the parrot to talk, he clicks on the parrot. When finished with one game, the child moves his mouse to the doorway and re-enters the playroom.

If we applied these principles to the desktop, the result would be a clear interface with much smoother navigation between functions than we have today.

Desktop Management Defined

In a basic sense, a desktop manager is an application program that is always running during a user session and hides Unix system utilities behind an intuitive interface. The desktop manager provides the ability to run programs and carry out file management tasks such as copying, deleting, searching directories, creating new files and directories, printing, and archiving. Desktop management is a necessity for commercial Unix. Take a simple task like copying a directory including its file tree (i.e., including all its modes: file ownership, access rights, permissions, etc.). Using one of the standard Unix shells, the command goes something like this:

```
tar -cpBf - ./directory | (cd /
dest; tar -xpBf -)
```

With a desktop manager, on the other hand, all you have to do is click on the directory icon, select Copy from a menu, position it appropriately, and the directory is copied—file tree, modes, and all. Or, better yet, simply drag the directory where you want to copy it and drop it in.

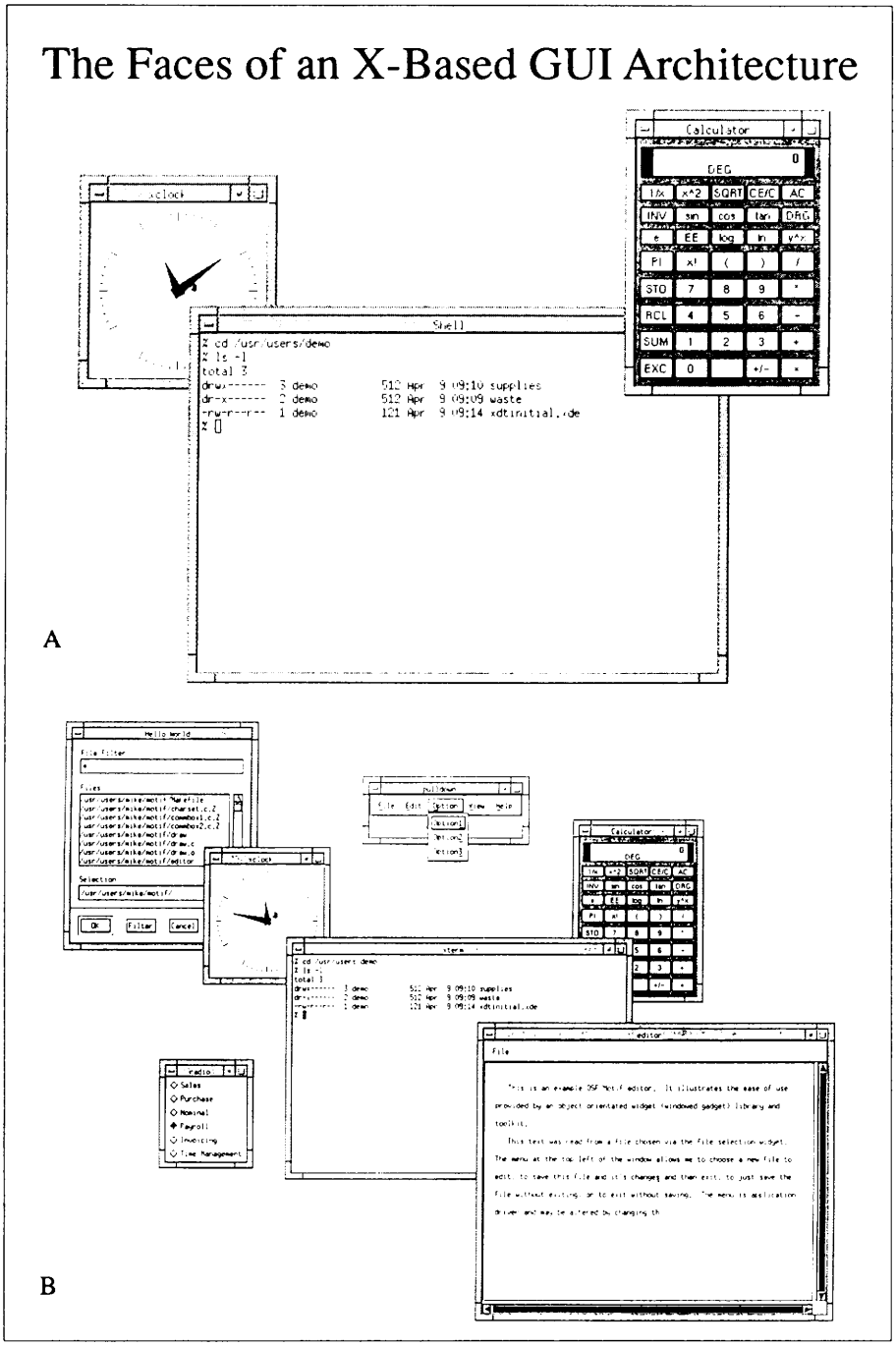


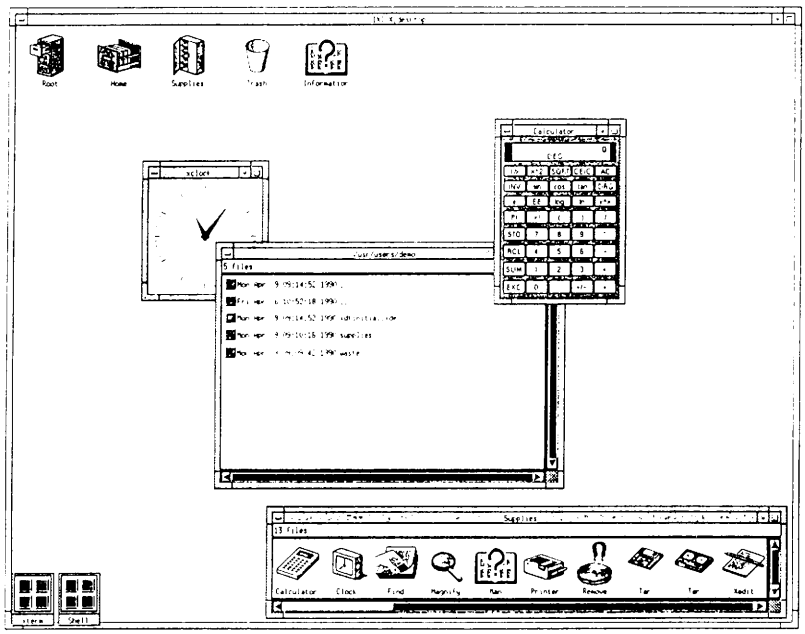
Illustration 1. Screen shot A depicts an X window manager, which is responsible for moving, resizing, and restacking windows. Notice that the user's access to Unix is a simple shell window. Screen B is standard Motif, and its style guide defines the look and feel of the interface. Yet, the user is still stuck in the shell for system interaction.

ARCHITECTURAL PERSPECTIVE. We alluded earlier to some confusion in the industry about precisely where a desktop manager fits in a user interface architecture. Many, for instance, confuse a desktop manager with its underlying window system. Actually, it picks up where the underlying window system leaves off, providing a user environment and a conceptual desktop metaphor of how the system is organized (e.g., via files, file drawers, file cabinets, etc.). (See Illustration 1.)

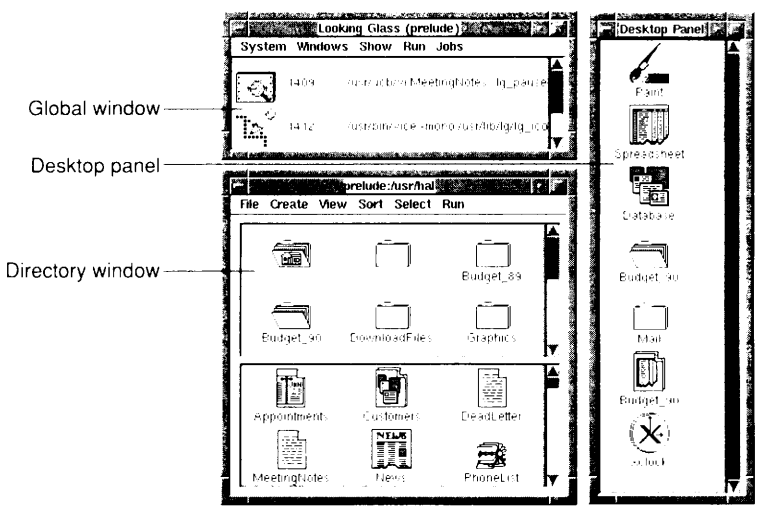
Above the desktop manager is the object model, which determines the behavior of objects on the desktop. Objects in this sense are datafiles as well as the programs that accompany them—task orientation as opposed to tool orientation. The application suite then sits above the object model. (See Illustration 2.)

The Faces of an X-Based GUI Architecture (continued)

C



D



Screens C and D are desktop managers (X.desktop and Looking Glass respectively). These are both based on Motif, but provide the user with a graphical system environment and conceptual desktop metaphor. Source: IXI, Limited and Visix Software

EXTRA PRODUCTIVITY. Desktop managers are beginning to take on more responsibility. Workflow automation is an important direction for desktop management. For example, an IXI customer—a publishing company—has configured X.desktop to automate its managing editor's desktop. Manuscript copy is sent electronically to the editor and appears on the editor's desktop as an icon. After reading and annotating the manuscript, the editor can drop the manuscript into an outbox where the annotated document is automatically forwarded back to the author, drop it into another box to be worked on later, or drop it into the outbox where it is automatically forwarded to the layout department (see Illustration 3).

In the end, the editor's desktop looks nothing like X.desktop's out-of-the-box product. Indeed, the publishing company must have spent a lot of time programming the product to work the way it does. But this is IXI's intent. Rather than trying to figure out what its customers want, IXI urges them to adjust the product to meet their specific needs.

Multiple Desktops. Tools for organizing and managing your work are equally important. Consider, for example, NCR's multiple desktops, which allow individual users to set up many individual desktops, each corresponding to a set of tasks or roles. A product manager, for instance, may need to review sales data, marketing plans, R&D progress on new products, and personnel performance information, among other areas.

With NCR Desktop, the product manager could make each one of these four major areas a separate desktop, each with its own mix of applications tools (spreadsheets, document processors, communications), datafiles (reports, forms, letters), logon routines to mainframe databases, distribution lists, a mail inbox, and whatever other facilities, information, and tools are needed to work in that area. The product manager may have separate Personnel, Product Development, Sales, and Marketing desktops.

Using multiple desktops is an attractive way to organize a lot of information and resources. Each desktop can be individually customized. You can even store multiple desktops within a desktop—nested desktops, if you will. The major alternative, folders and file drawers, can result in "desktop clutter," as users stick more and more resources into folders, and more and more folders into other folders. Folders don't give users many visual distinctions between one folder and another, either. Each folder icon looks the same and opens onto the same basic desktop look-and-feel.

NCR isn't alone with this idea of multiple desktops. VUE,

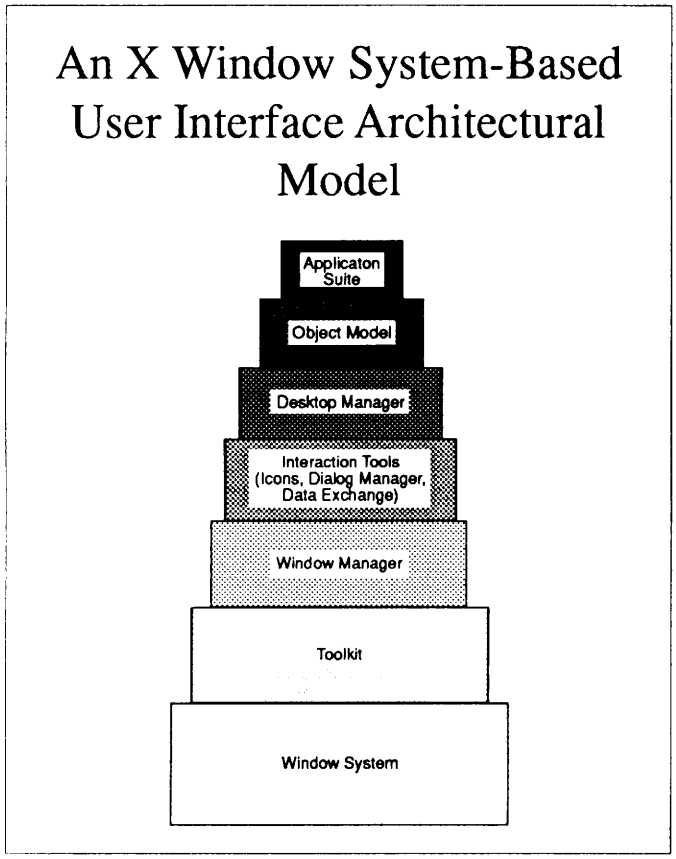


Illustration 2. X itself provides the base window system and development intrinsics. The toolkit and window manager can have various flavors. The X tape comes with the Athena widget set (Xaw) and a standard window manager (twm). But commercially, we're talking about Motif and OpenLook toolkits and their respective window managers at these levels. Motif and OpenLook provide interaction tools such as icons and dialogue management. But the desktop manager determines how easy the system is to learn and use. If the system has an object model, it sits on top of the desktop manager, and the application suite completes the model.

for one, has a similar workspace manager tool that allows you to store up to six color-coded desktops or workspaces, and Sun has a drag-and-drop binder tool that associates user actions and applications into color-coded icons. Looking Glass and X.desktop also let you create multiple desktops. (In fact, much of NCR's desktop functionality for its Unix client is based on X.desktop.) But the NCR implementation is particularly attractive—especially with its notion of nested desktops, where a desktop can be stored within a desktop to reach deeper organizational levels.

Implementation

Typically, desktop managers provide a visual representation of the file system. They handle activities such as creating directories, files, and devices; launching applications and utilities; printing; searching for files and directories; and so on. Yet the reach of commercial desktop managers can be very different. Which leads us to a debate over how much desktop functionality a user really needs. Most users spend their time in applications and don't need to run advanced Unix commands. Not many users know anything about changing environment variables, for example. Nor do many care to. But the point is that some do. Very often, a user is unaware of the power of a system until the proper tools become available. Simply providing access to the system's power is not enough. Ideally, the desktop manager should help the user understand more complicated or cryptic functions. (Environment variables, by the way, hold information about your operating environment—e.g., the executable search path and home directory.) Furthermore, there is a sharp dichotomy between the traditional technical Unix user and the new, commercial Unix user. And as long as Unix is in this transitional state, the dichotomy needs to be dealt with.

ADDRESSING DIFFERENT AUDIENCES. Looking Glass is one desktop manager that offers quite a bit of base functionality. The menus even provide the means for creating hard and symbolic links, terminating jobs, changing environment variables, and changing file and directory properties. Other systems usually leave such operations up to the Unix command line. Yet its sophistication isn't at all daunting for novice users. And to accommodate the other end of the spectrum—the techie—Visix will be selling an additional set of Looking Glass enhancements called Looking Glass Advantage that will focus on higher-level activities. (The product won't be announced until later this year; we'll give you the details when it is.)

Apple's Approach. We think that Apple has probably found the best way to deal with the multiple audience problem. A/UX has a dialogue-driven Unix command builder called Commando—a feature that gives users a logical means of approaching the complexities of Unix (see Illustration 4). An A/UX system folder contains iconized, frequently-used Unix commands, such as apropos, diff, find, grep, and ls. (The icons carry the full name of the command, which is helpful; "chmod" doesn't mean much to a user, but "change permission" does.) After you click on the icon, the Commando dialog box leads you step-by-step through the command, describing its purpose, listing command options, and banging out the appropriate command line code. Commando can be used as a learning vehicle. It doesn't take long before you can add new user activities and commands to your repertoire.

Multiple Environments. IXI's solution isn't bad, either. X.desktop comes preconfigured with three environments: one for the novice user, one for an experienced Unix user, and another for the desktop administrator.

The novice user environment gives the user a simple, rational view of the system. You don't see any dot files or multilevel subdirectories, no system program directories (e.g., /bin, /usr/bin, /usr/ucb, etc.), no access to the system shell.

Rather, it offers basic, single mouse-button functionality, with icons and menus for executing applications and for printing, deleting, and copying files and directories.

The next level environment assumes a working knowledge of Unix. It includes dot files and program directories, access to the system shell, iconized Unix commands, etc. It also has a "Get named directory" function, which lets users type out the directory instead of wading through menu and directory layers. The administrator's environment is as robust as the Unix user's environment, but it also contains bit-map directories, access to the bit-map editor, access to X.desktop rules for configuration, and tools for adding new users, changing passwords, monitoring disk space, etc.

IXI stresses the consistency and configurability among these environments. Each preconfigured desktop has the same interface, so users can graduate to the next level without much difficulty, and additional features can be added to any environment. We do like the tiered-environment strategy. IXI's shortcoming is that it provides no means of graduating to the next environment level without system configuration. For instance, the iconized Unix commands are nice, but if you don't know the command, they're useless. The program has nothing along the lines of Commando that helps you learn what the commands do, nor does it have menus that mask commands and make them more digestible.

HELP. Help really encompasses two issues. One, obviously, is the Help facilities for the desktop manager itself: how to move a file, how to activate a program, how to search a directory, etc. The other issue has to do with system Help facilities, and these—although there are few at the desktop Help level—are dependent on specific hardware platforms and system resources and services. Here, third-party desktop managers will be found lacking. Their best option is to offer extensions for OEMs to include additional applications and hardware- and system-specific Help items, which is

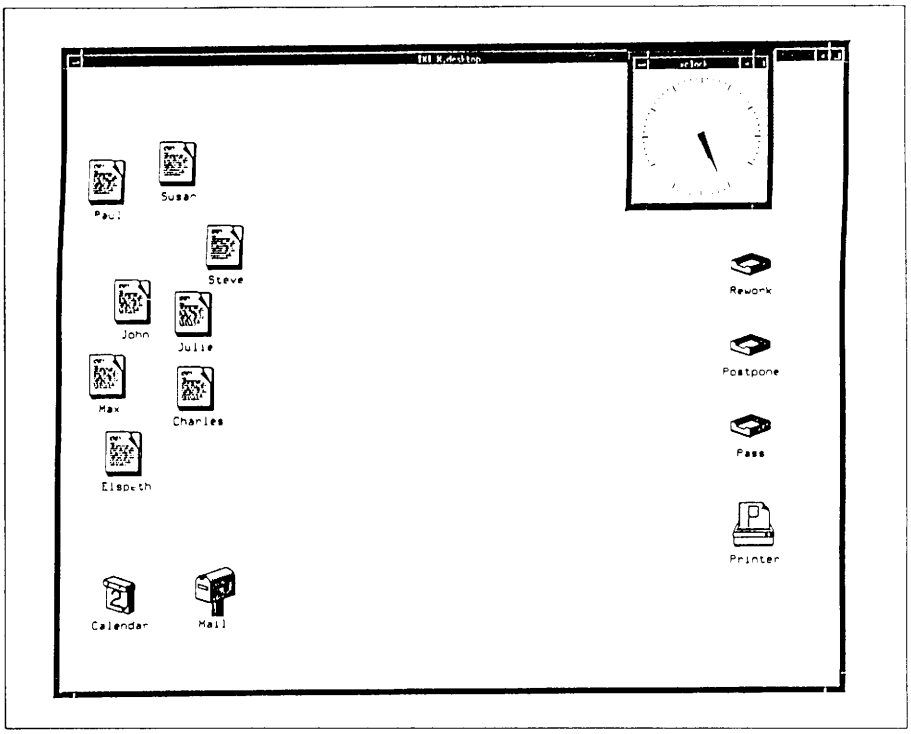


Illustration 3. An editor's desktop created with X.desktop from IXI. Manuscripts are sent to the desktop, from which the editor can drop the manuscript in the Rework outbox, where the annotated document is automatically forwarded back to the author; drop it into the Postpone outbox to be worked on later; or drop it into the Pass outbox, where the document is automatically forwarded to the layout department.

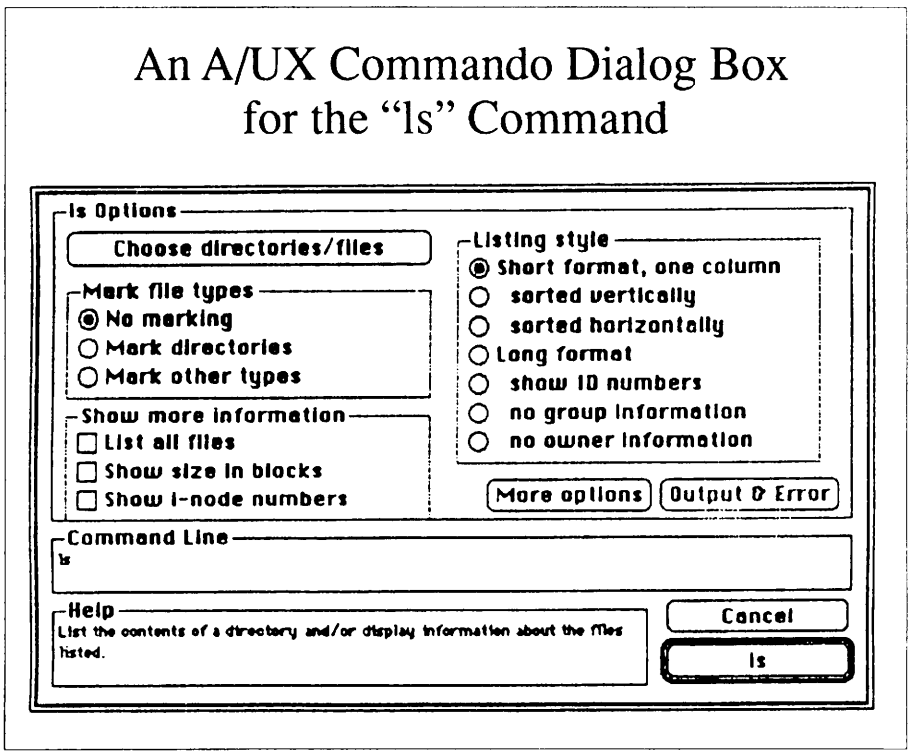


Illustration 4.

A Sample Help Screen from Looking Glass

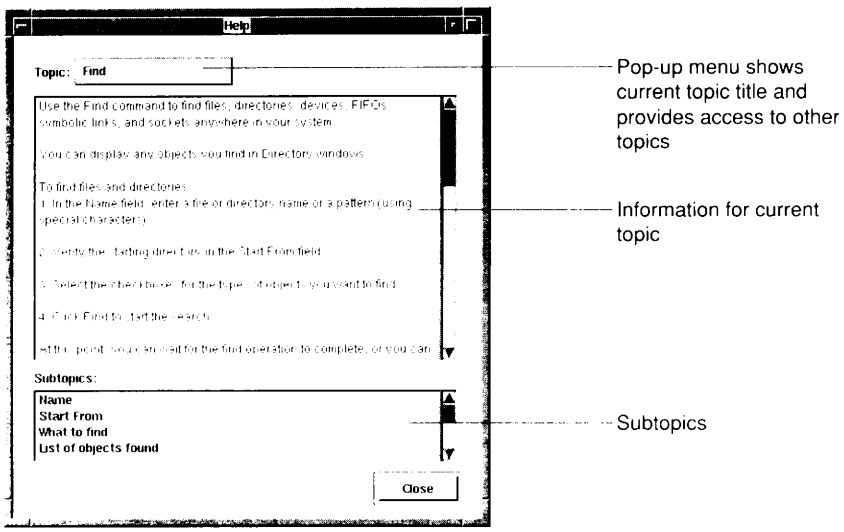


Illustration 5. Looking Glass has a well-designed Help system. Its context-sensitive and hypertext-like implementation is both complete and flexible. The Help window gives you information on a single topic, and then provides a menu of subtopics that are accessible by double-clicking with the mouse. Sometimes the subtopics have subtopics themselves, but, at any point, the user may return from a subtopic to another level (as opposed to just returning to the previous screen).

what IXI does. (The company is actually licensing its drag-and-drop, hypertext-like Help system as a separate product.) So does HP. VUE actually has a really nice Help system, with a context-sensitive interface for information on desktop items and an application programming interface for developers to integrate Help with their applications. You can even "tear off" a Help page and put it on the workspace. This is a small detail, but a useful one. Too often, you find yourself returning to Help for repetitive tasks. VUE's Help also includes an online version of the HP-UX manual pages. (We assume that the IBM, Digital, and Sun versions to come will have the appropriate system's manual pages.)

Visix also has a well-designed Help system. Looking Glass's context-sensitive and hypertext-like implementation is both complete and flexible (see Illustration 5).

Some products don't have Help at all—it's one of limitations of the A/UX Finder tool. On the other hand, the A/UX environment is so intuitive that you don't miss Help much.

ICONS AND ICON EDITING. An icon editor is fairly essential to a graphical desktop manager. If the desktop is to hold all files and applications and give iconic access to these files and applications, you certainly need a means of creating icons for them. The exception here is A/UX, where programs written for the system are pre-iconified. But A/UX isn't exactly your average Unix system. The X Window System itself has a bit-map editor, but it's nothing to write home about. While easy enough to use, it's also fairly limited—not to mention slow. Thus, special icon editors with drawing and editing tools for desktop managers are common.

The potential problem with users designing lots of individual icons is that their systems may lose some consistency. Ideally, the system should have some cohesion in terms of icon display—especially if you're in a networked situation where several users are sharing files and applications. It makes sense, for instance, for all executable files to have similar icons, or for all writable files to be identified uniformly.

CUSTOMIZATION. We haven't bumped into any desktop managers that don't allow for some customization at the user level. A/UX, for example, has a Control Panel, which offers a broad range of dialogue-driven customization options, from specifying the color of icons, background patterns, and alert settings, right down to setting the speed of cursor blinks. Menus are also available for file system viewing options (icon, small icon, name, date, size, or type), arranging your workspace, and selecting network configurations. These kinds of options are typical.

Looking Glass, though, offers a bit more end-user customization than other programs. A few notable Looking Glass preferences include:

- Options for specifying file removal operations: confirmation, confirmation for multiple file removal, confirmation for read-only file removal
- Setting up default access privileges for new files and directories
- Changing default system programs: terminal emulator, system shell, formatting and printing files, and system editors

NAVIGATION. Perhaps the most immediately distinguishable feature of the many desktop management implementations is the way you navigate them. Although most graphical desktop managers offer some combination of icons, menus, and dialog boxes for navigation, some are more pictorial, using a fully implemented drag-and-drop paradigm and relying heavily on icon manipulation. Others rely on drag-and-drop mainly for copying and moving icons, and use menus and dialog boxes for

most other activities. (Looking Glass and X.desktop actually make a nice navigational contrast. See Illustration 2.)

However, judging these different implementations is difficult. Much of it depends on the user's visual orientation. While many users enjoy a lot of cute icons or a wide file-format view, others think such a cluttered display is too distracting. Typically, the more technical users find an over-iconified, over-menued system annoying. They prefer to just bang on the keyboard. On the other hand, less experienced users may find lots of menus and icons inviting.

The bottom line: Desktop managers should allow for some end-user customization in navigation, and most do. One convenient feature some products are beginning to implement is "tree views." The user clicks on "branches" of the tree rather than getting bogged down in too many directory layers. You can also usually modify your file views: by name, by icon, and by wide format (e.g., including file name, permissions, and—in Looking Glass's case—the number of links, file size, and Inode number).

The Drag-and-Drop Debate. Although we think of direct, iconic manipulation—i.e., drag-and-drop—as a more reasonable way to deal with systems, many notable graphical user interface experts deem it too imprecise for full-scale implementation. How does an application know what operation you have in mind when you drag-and-drop? What happens when you drag a printer over to an outbox and drop it in? What if you drop a binary file onto the printer? What about dropping a spreadsheet on top of a text file to make a compound document? Is the spreadsheet incorporated into the text file or vice versa? If so, where? The point is that you need to be explicit about the operations you are performing as well as the objects involved, and drag-and-drop lends itself to fallibility. Unless the drag-and-drop mechanism addresses these ambiguities, you're better off defining operations (such as print, move, copy, etc.) and specifying the objects separately.

Combine this ambiguity with system variables. It's much like the problem we talked about earlier with third-party Help systems. Drag-and-drop is tricky to implement generically (i.e., for third-party GUI vendors) because portable software has no way to address the system resources and services that vary from platform to platform. There's simply too much room for error.

At the moment, there are no standard drag-and-drop implementation guidelines. The X Window's InterClient Communication Convention Manual (ICCCM) does address a minimal layer of drag-and-drop support: the visual effect of dragging and dropping and data transfer (i.e., interaction among interface objects). And OpenLook 2.0 has its own drag-and-drop protocol—a fact that Sun is touting in its OpenWindows marketing campaign. Sun and AT&T have done some interesting things with the OpenLook file manager. For example, it lets you launch an application simply by dragging the icon onto the workspace. What actually happens is that in moving an icon outside the border of the file manager base window, you create an operating system link to the application. When you release the Select button, the application is launched. To load a file into an application, you can drag the data file from the File Manager and drop it onto an application window or icon. OpenWindows supports drag-and-drop for copying data into other applications only between two XView applications or between two OpenLook Intrinsics Toolkit (OLIT)—pure X Window—applications. In the future, Sun plans to provide drag-and-drop integration among the three OpenLook toolkits.

Meanwhile, the major vendors and standards bodies are working to see that the next release of the X Window system and its various flavors include a single drag-and-drop standard. And, in keeping with these efforts, there is a proposal before the X Consortium for a standard X "drop object" protocol. Essentially, the proposal advocates that the initiating client remain in

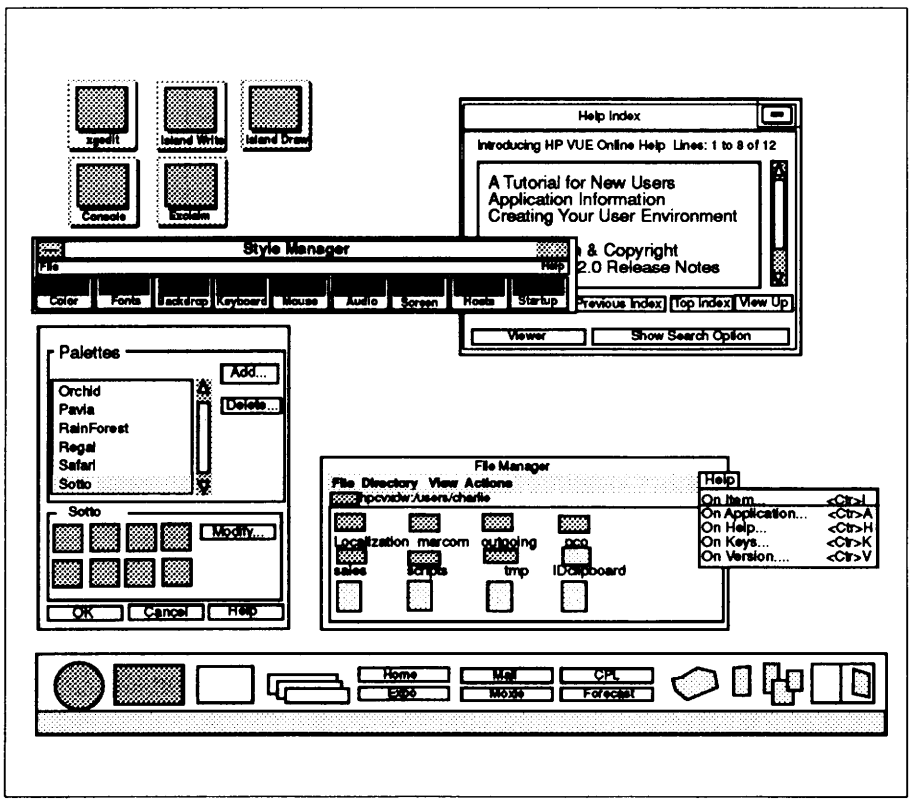


Illustration 6. HP's Visual User Environment. VUE includes a visual file manager, a style manager for customizing your environment, a context-sensitive Help facility, and a color-coded workspace manager that lets you create multiple desktops.

control of the drop action. In other words, if you drop a file onto the printer, the file transfers its data to the printer, not vice versa. That means that, if you drop a binary file onto a printer, the file has to know that it is not printable and must display an error message stating as much. System objects would need an incredible amount of on-going configuration (i.e., to make allowances for added desktop operations and objects). Still, the drop object protocol would do much to alleviate the confusion we described above. And the drag-and-drop implementations we've seen are compatible with it.

But the drop object protocol wouldn't help the system variables problem. There's no easy way around this one. Implementers simply have to set these up individually. And, actually, a menu-driven interface doesn't help these administration problems, either.

Application Integration

We've talked a bit about customizing your desktop with icon editing, and you can usually adjust display settings (e.g., for color, background color, mouse-button speed, etc.).

But a deeper level of customization comes in application integration. The desktop manager needs APIs so that applications can be included in the environment. Looking Glass, for instance, uses file-type definitions for integrating programs and files to the desktop. File definitions give the system information about the files it contains and how those files should appear and behave on the desktop—the icon for the file as well as the correct action to take when you double-click on it. A file-type language exists to let programmers add or modify specific, customized definitions. If, for instance, you want to run some home-grown 4GL application on a datafile when you double-click on it, the desktop can be configured to do that.

A deeper level of customization comes in application integration. The desktop manager needs APIs so that applications can be included in the environment.

PROCEDURAL AUTOMATION. Ultimately, though, users are going to demand more productivity at the desktop. And that's where products like NewWave and X.desktop come in. These let users link and nest desktop objects to facilitate workflow automation.

NewWave Agents. In the HP environment, VUE provides the basic desktop functionality: a visual file manager, a style manager for customizing your environment, a Help facility, and a workspace manager (see Illustration 6). The style manager is typical of other user configuration tools (see "Customization," page 7). But, as we've mentioned, the Help facility is very well-designed, and the workspace manager lets users create multiple desktops (up to six).

On top of that environment, NewWave will provide objects (see "Object Management," page 10) and tools. Among those tools is an Agent, a powerful procedural automation or cross-application macro facility. It can automate anything you can do within the NewWave environment. An agent task is created by simply recording the steps you want it to perform when it is activated. You can modify the resulting script using the agent task language (which is much like using a 4GL). Context-sensitive Help is available, as is a full dialog box capability within agent tasks. Agents can also be nested.

Unfortunately, NewWave doesn't yet support X, but Hewlett-Packard is planning an X version of the product, and, when it is released, VUE coupled with NewWave may turn out to be quite an attractive solution. Of course, NewWave coupled with anything is quite an attractive solution. When the X version is released, NewWave will be usable with any X-based desktop manager. Furthermore, VUE will soon be available for IBM, Sun, and Digital platforms.

IXI's Rule Files. In the meantime, IXI is quick to point out its desktop object manager, which links the X.desktop rule files. These are similar to the Looking Glass file-type definitions we described above. The rule files determine more than just icon appearance and behavior; menus can be created or modified with the rule files to hold more information or perform new operations, and messages can be tailored to specific applications. Best of all, objects can be linked, repeated, or combined to provide a procedural automation facility. For example, a directory rule file can be designed to pass into five other directories any file that is dropped into it. Administrators can build both composite objects that set off several actions at once, or a series of actions that are to be performed in sequence.

The procedural capabilities of X.desktop's rule files are certainly an advantage for IXI, but it's not quite the macro facility that the NewWave Agent is. The rule file language is a programming language; there are no recording capabilities. Ideally, users want to combine and automate their own tasks. A simpler, more visual way to write the rules would be an even greater benefit to the product. IXI is working on a product that will let users build new objects, encapsulate utilities, and change system behavior without editing the rule files. We look forward to its release later this year.

OBJECT MANAGEMENT. NewWave goes further than procedural automation. It's an object-oriented desktop model—a foundation for creating compound documents and application communication. To detail NewWave is beyond our scope here, but we will point out the highlights. (Complete analysis of NewWave Office for Unix appears in Vol. 5, No. 4.) Objects in NewWave are applications that create and manage datafiles. At

the heart of the system is the Object Management Facility (OMF), which keeps track of the status and relationships of all local objects on the desktop. Among its responsibilities are:

- Binding data and applications into objects. When you create a WordPerfect document, the OMF links the document to the WordPerfect application.
- Managing information links between objects. If a compound document contains a range of cells from a spreadsheet, and you double-click on that range of cells, the OMF knows to activate the correct program and open the spreadsheet.
- Providing horizontal integration. Every application written to the NewWave APIs gets "instant integration" with all other NewWave-compliant applications. The OMF then takes care of all the links and communication between objects. Thus, a developer can use NewWave applications as interchangeable parts in building more complex or specialized applications.

The drawback to NewWave is that it deals only with large-grain objects—entire documents, not sections or paragraphs, and whole spreadsheets, not individual columns or cells. Thus, NewWave won't let you include a range of cells in a text document. And you need this level of granularity for true compound documents.

Conclusion

STANDARDS. The gulf that separates graphical user interfaces also separates the products we've been discussing. Sun's OpenWindows is obviously based on OpenLook, and VUE, IXI, and NCR (under Unix) are Motif implementations. Looking Glass alone bridges both camps. The Finder tool for A/UX is, of course, a law unto itself and has nothing to do with X at all.

Visix has its own toolkit, and that fact has drawn some criticism from its competitors. Looking Glass is built with the Visix toolkit, which has enabled the company to churn out both Motif and OpenLook versions of the product. Each is completely compliant with its respective style guide in terms of behavior and appearance. The problem, these critics contend, is that Visix's toolkit is not compliant with X intrinsics, and several standards bodies—including the X Consortium and the National Institute of Standards and Technology (NIST)—are zeroing in on X intrinsics.

Though this could be problematic for the company, we think Visix's approach has merit. The toolkit supports multiple window systems—including X and SunView. The interoperability gained by a window system-independent toolkit is a credit to Looking Glass. Too many application developers are being stymied by the current user interface battle. The cost of writing to multiple window systems is simply too expensive. Not only does the Looking Glass toolkit spare the company the expense of writing to multiple interface libraries, but Visix also claims that it improves the performance of the system because processes don't need to carry the Motif or OpenLook toolkit layer. In a networked situation, speed can be a problem for X-based applications. Furthermore, from the user's perspective, the Looking Glass versions are completely consistent with Motif and OpenLook.

COMMENTS. As we said earlier, most major OEMs offer some form of desktop management. Digital, it seems, is the one major Unix hardware vendor that is still on the bench. Digital is selling Looking Glass in the United States, although the Digital 386/486 boxes bundle X.desktop—and that doesn't count the internal development work going on at Digital. Actually, several important OEMs have turned to Visix and IXI for their desktop managers. X.desktop has received recognition especially from its OEM relationships with IBM and SCO. X.desktop is the desktop manager component of SCO's OpenDesktop platform. And at least one of IXI's large user customers opted for X.desktop for its Unix workstations largely because it's the IBM standard. But Looking Glass has its share of big OEMs as well, among them Data General and Motorola. Each company also has a bunch of end-user customers. Hewlett-Packard has VUE and is licensing it. And Sun has OpenWindows. Sun, in its quest to become standards-maker, is offering the application development environment source code of OpenWindows free of charge, and it includes X11/NeWS, OpenLook, and XView, as well as its font technology. But, again, it's not offering the OpenLook DeskSet tools at all, and therein lies a good deal of Sun's desktop management functionality, including: the visual file manager, drag-and-drop print tool, icon editor, and the binder tool we described earlier. Apparently, Sun is saving a little value-added functionality for its own platforms. (For a detailed review of OpenWindows Version 2, see Vol. 5, No. 10.)

State-of-the-art desktop managers are very good at masking the intricacies of Unix and are starting to provide functionality to help users organize their work and manage their workflow.

State-of-the-art desktop managers are very good at masking the intricacies of Unix and are starting to provide functionality to help users organize their work and manage their workflow. We hope that they will continue in the direction of application integration and interapplication communication. Ultimately, the desktop manager must take on the responsibilities of an object manager to exchange commands and data among desktop applications and must perform agent facilities. Of course, standards will loom large in this evolution of desktop management. Applications need to conform to an object management facility before they can fully exploit such an

environment. Perhaps the Object Management Group will help here, since its primary goal is to create an object-oriented, application-to-application communication standard.

We see no reason why current desktop management solutions can't evolve to equal the simple elegance of the interface of "The Playroom" that we described in the beginning of this article. The technology is there. Developers are adding functionality and graphics extensions to the X Window system all the time. Unix may actually have an edge over other platforms because Unix machines are already perceived as graphical workstations, and so many innovative developments seem to come out of the Unix community. But, in the end, it's the user who will want to define the way he interacts with his computer. There's a real need for desktop mangers to include user tools for customization. And we don't mean merely adjusting screen colors and mouse-click intervals. Perhaps it makes more sense for the desktop of a sales manager to be made up of sales districts to click on, rather than files and directories. Tools for specifying the desktop this way are really essential. An easier interface to Unix is certainly worthwhile, but the real promise of desktop management is to put more user-prescribed productivity at the desktop.

The title of next month's Unix in the Office *is* ***Ingres: A Database Vendor in Transition.***
For reprint information on articles appearing in this issue, please contact Richard Allsbrook at (617) 742-5200.

NEWS ANALYSIS

PRODUCTS • TRENDS • ISSUES • ANALYSIS

• INSIDE •

• TRANSARC •

Future Shot: IBM, Sybase, Others Adopt Transarc's Distributed, Open OLTP Tools

Technology announced last month by Transarc Corporation, a small Pittsburgh-based company, gives the industry important new tools for developing online transaction-processing (OLTP) applications that operate across distributed networks. Transarc's technology will become available to users in products from IBM, Hewlett-Packard, Stratus Computer, Sybase, and Informix—and we expect other adopters as well—in 1992 and 1993.

Open, distributed computing architectures are clearly the future of OLTP. Many big OLTP users, such as banks and brokerages, and smaller users, such as manufacturers and distributors, are moving to Unix-based OLTP to gain openness. The reasons users want openness: Unix systems are generally less expensive than proprietary systems, and applications written to open interfaces can be rehosted if necessary.

As users move toward open OLTP, they are also hoping that emerging standards in transaction and database management will allow them to distribute their OLTP applications as well. The reasons for the interest in distribution: An increasing percentage of transactions take place over time and distance, and distribution allows for graceful capacity upgrades to support growing applications.

Distributed transactions are usually complicated. Typically, two or more parties must agree to complete separate parts of a single distributed transaction at roughly the same time. Take an equipment sale that includes a service contract, for example. The transaction must be booked in the corporate equipment-sales database *and* in the corporate service-contracts database. If any one of these bookings of new information can't be completed—for whatever reason—the transaction processing system must be able to roll back both databases to their prior states.

Many large OLTP users use distributed applications today—automatic teller machine networks are an example—but at a very high cost. For most users, distributed OLTP will be impractical until more development and management tools are available.

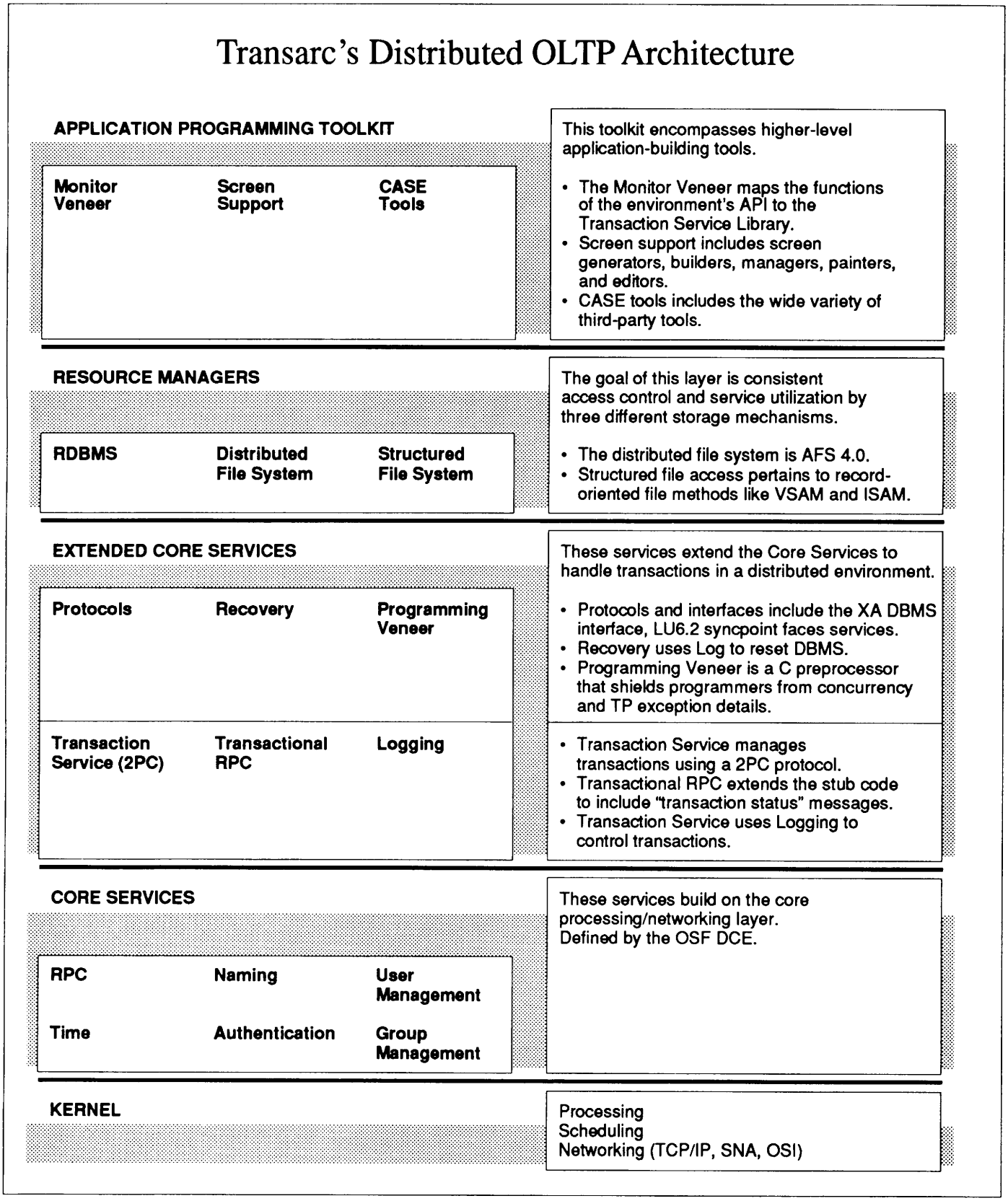
TRANSARC'S OLTP TOOLKIT. Into this fray comes Transarc, a company devoted to developing technologies to enable distributed computing. Transarc's OLTP toolkit rides atop the Open Software Foundation's (OSF's) Distributed Computing Environment (DCE), an operating system-independent, modular set of distributed services. (Transarc is the contributor of the Andrew File System—the distributed file system technology—to DCE.) Its OLTP Toolkit adds modules that define, track, and log transactions to DCE's basic distributed services. (See illustration, page 14.)

The components of Transarc's toolkit, including the DCE, are as follows:

Core Services. This is what Transarc calls the threading services, remote procedure call (RPC), distributed time service, naming services, security services, and directory services of the OSF's DCE.

Commit Coordination Service. The Commit Coordination Service is a transaction-monitoring service. It ensures that parts of a single transaction executed in different places on a network are all completed before recording the transaction. Transarc's service also supports nested transactions, a way of structuring distributed transactions that can improve performance and server availability.

Transarc's Distributed OLTP Architecture



Transarc's architecture for open, distributed OLTP assumes the presence of the Open Software Foundation's Distributed Computing Environment (OSF's DCE) to provide core systems services. Transarc adds Transaction Processing Services, OLTP-oriented RPC extensions, Logging, and Recovery to this base. Transarc hopes its C language macros and transaction services programming "veneer" will obviate the need for application developers to learn the intricacies of OLTP.

Transactional RPC. Transarc's Transactional RPC extends the interface definition language of OSF's RPC (Hewlett-Packard/Apollo's NCS 2.0) with additional semantics to identify, start, and stop transactions.

Common Log. Transarc's Common Log is a recovery and synchronization facility that operates across multiple DBMSs or file systems. Transarc adopted technology developed by IBM for this part of its environment.

Security Services. Security Services allow transactions to be locked across multiple data management facilities.

Transactional C. Transarc is providing C procedures and macros to shield developers from the details of transaction management as they build their applications.

Protocol and Interface Translators. The protocol and interface translators allow interoperability of OLTP systems using different protocols (LU6.2, etc.) and interfaces (XA, etc.).

XA Interface. The Transarc Toolkit incorporates the XA data management interface developed by X/Open's Distributed Transaction Processing working group. XA gives a transaction management facility, such as Transarc's, a single interface with which to give instructions and receive feedback from databases and file systems.

WHAT'S MISSING FROM THIS PICTURE? Transarc's toolkit looks like compelling technology. However, we don't have any direct experience with products based on the technology, and this makes it difficult to pass judgment on its efficiency and performance.

We can say, however, that Transarc's toolkit is not a complete solution for building and running distributed OLTP applications. It has two major holes: management tools and development tools.

Management Tools. Transarc's toolkit incorporates basic management utilities but is not a complete distributed management solution. Some implementers will provide their own tools—Stratus is incorporating its own management tools into its implementation, for example. For a more general solution, however, Transarc is relying on OSF's Distributed Management Environment (DME) technology identification and selection effort. OSF hopes to complete this selection process this year.

Development Tools. Transarc's toolkit includes "transactional" facilities for C. There are no screen painters or higher-level tools. Transarc is currently seeking partners to deliver such tools by the time the core technology becomes available from IBM and others.

IBM AND THE EARLY ADOPTERS. IBM, Hewlett-Packard, Stratus, Sybase, and Informix Software are planning to adopt Transarc's distributed OLTP technology. In addition, Digital Equipment Corporation is apparently leaning toward adopting it.

This list includes many of the most important vendors in OLTP, which bodes well for the technology's acceptance. However, some important parties are absent, most notably, Tandem Computers Incorporated, the leading vendor of fault-tolerant systems. Nor have NCR and Unisys adopted Transarc's strategy. NCR just announced its own Top End large-scale Unix-based OLTP system. Unisys is using AT&T's Tuxedo transaction monitor, which has some distributed features, as a base for a large-scale Unix OLTP system called Open/OLTP.

NCR and Unisys appear ready to fight it out for large-scale OLTP applications on Unix. Transarc will focus on distributed OLTP applications on heterogeneous platforms, not just Unix. HP, for example, plans to implement the Transarc toolkit on its MPE operating system, as well as its HP-UX Unix. Stratus will incorporate Transarc's toolkit into its VOS operating system, as well as its FTX Unix variant.

WHY SHOULD USERS CARE? With IBM and the other adopters not planning to introduce Transarc's technology for more than a year, why should anyone care about this announcement? We believe it is important to the future of OLTP applications, which will be defined by two trends: openness and distribution.

Transarc's toolkit technology fills a gap in the OSF's DCE. DCE, by itself, doesn't include transaction management services. IBM, Digital, HP, Stratus, and others plan to offer DCE as part of their systems software beginning in late 1991. Transarc's OLTP toolkit will allow these vendors to use DCE-based systems to support the most important group of applications today: OLTP.

However, all of this will be a long time coming for most users. IBM, Digital, HP, and other DCE adopters will first offer basic DCE services, and then extensions such as Transarc's toolkit.

—J. Rymer

•IXI•

Practical Developments in Desktop Management

We refer often to X.desktop in this month's feature on desktop management, and those are not arbitrary references. The latest release of X.desktop, announced at UniForum, has more than a few notable features. Particularly interesting are the level of customization it allows and the way it addresses the diverse Unix audiences. X.desktop has always provided a visual file system and an iconic, drag-and-drop paradigm for navigating the system. It's also always included rules for adding files and

applications and for customizing the desktop environment. This release, 3.0, has been extended to include greater application integration capabilities, a hypertext Help system, a special icon editor, and preconfigured environments that are more in sync with the expertise of the user.

PRECONFIGURED ENVIRONMENTS. As we discussed in the feature, X.desktop's preconfigured environments are a workable way to bridge the gulf between the traditional technical Unix user and the new commercial Unix user. The new release comes with three different user environments: one for the novice user, one for experienced Unix user, and another for desktop and some system administration.

The novice environment is very simple, so that new users can get accustomed to using Unix without being intimidated. Therefore, it doesn't include dot files or multilevel subdirectories or system program directories that might be confusing to new users. Rather, it provides a basic, iconic representation of the file system and single-mouse button functionality for executing applications, accessing Help, and printing, deleting, and copying files and directories. IXI also points out that the novice environment is pretty goof proof. For instance, it won't allow users to kill an executing job, reasoning that the users might not realize that data could be lost if they try to do so.

Once the novice user gains some confidence with the system, he can move up to the next level, which includes dot files and program directories, access to the system shell, iconized Unix commands, etc. The administrator's environment is just as robust, but it also contains bit-map directories, access to the bit-map editor, and access to X.desktop rules for configuration, as well as tools for adding new users, changing passwords, and monitoring disk space, etc.

HYPERTEXT HELP. We were pleased that X.desktop finally has a Help system at all. Previously, when you invoked Help, the system dropped you into the Unix manual pages. Pretty nasty. Apparently, IXI assumed that individual implementers would build Help to address specific platforms. Philosophically, we agree. Help systems should be designed to handle specific hardware platforms and system resources and services. The problem was that too many X.desktop OEMs merely bundled the product as is, which made for no Help at all (a common problem with customizable, third-party products).

Therefore, IXI built a new, very well-designed Help facility. X.desktop has both context-sensitive Help and an online directory that looks a lot like its user manual—complete with bit-map drawings and descriptions of how to use the system. If you drop a desktop object on the Help icon, Help brings you to the correct section of the online directory. You can look at associated information by clicking on the highlighted text of particular topics.

IXI has left Help extensible so customers and OEMs can add to or change it. Furthermore, IXI plans to market Help separately.

ICONS AND ICON EDITING. Another weakness of X.desktop's previous release was its icon editor. Until this release, the product depended only on the X Window bit-map editor, which is limiting and slow. X.desktop's new icon editor is definitely an improvement, with the drawing tools, support for color, and support for different icon sizes and shapes.

In addition, X.desktop lets you create "transparent" icons, which let you view an icon that has been overlapped by another. Another nice feature is that you can import icons from Microsoft Windows—useful from a systems integration point of view. It also supports animated icons, so, for example, you can watch a piece of paper zip through a printer icon.

APPLICATION INTEGRATION. X.desktop's application integration capabilities make it a very flexible product. As we pointed out in the feature, using the X.desktop rule files, administrators can link desktop objects (i.e., iconified files, directories, and applications) to automate your workflow. Administrators can build both composite objects that set off several actions at once, or a series of actions that are to be performed in sequence.

This is a very powerful facility. Too bad the rule files are for programmers. Users could use this kind of capability to string together their own personal procedures. IXI states that it is indeed working on a more user-oriented tool for building new desktop objects, encapsulating old utilities, and changing desktop behavior without editing the rule files. It should be released later this year.

CONCLUSION. IXI has done a lot to make its X.desktop a more productive and well-targeted system. We especially like its preconfigured desktop environments, which meet the needs of different Unix audiences. And we like the fact that even these can be configured to suit the precise whims of the user. The addition of an icon editor and a Help system make it a more competitive product.

IXI has OEM relationships with a number of important Unix hardware vendors, including: IBM, Unisys, NCR, SCO, NEC, Tektronix, Panasonic, Dell, Omron. The product also runs on a bunch of Unix platforms. Licenses go for approximately $795. —*L. Rowan*

•BBN•

An Update of Slate

It's been over two years since we last looked at Slate, a compound document processor and electronic mail and conferencing system from BBN Software Products (Cambridge, Massachusetts). Slate first caught our eye because it was one of the first compound

document architectures (CDAs) under Unix, and, although other more prominent CDAs have emerged, Slate still has some features—especially a few new ones—that set it apart. We're particularly interested in its new extension language that lets administrators set up Slate as an integrating environment.

SLATE FUNCTIONALITY. Slate compound document functionality hasn't changed much since we last looked at it. Slate compound documents can comprise up to six different media elements: text, graphics, scanned images, speech, spreadsheets, and charts generated by spreadsheets. The Slate compound document editor supports all of these elements. Actually, it's a meta-editor; the graphics, image, text, spreadsheet, and voice modules are complex editors in their own right, and the compound document editor allows them to interact in a single document. Thus, switching among various media elements is seamless. You don't have to open a separate window to edit an element. Each is displayed and edited within a single display surface within a single process. And you don't have to tell Slate which media element you're editing; the pointer position provides it with that information, and Slate gives you the appropriate editing tool.

Slate currently has its own X-based interface, so it's neither OpenLook nor Motif compliant. Until recently, BBN was focusing on government and engineering sites as probable customers, but now the company is trying to make its way to commercial organizations. And Slate's lack of support for either of the two major user interface toolkits might hinder its acceptance.

MULTIMEDIA MAIL AND CONFERENCING. One of the most noteworthy aspects of Slate has always been its support for electronic mail and conferencing. Slate actually originated as a university project that sought to improve the quality and exchange of electronically submitted documents, so the product has evolved to be very strong in its E-mail support. Documents are exchanged via either Standard Mail Transport (SMT) or X.400. You can send even complex Slate documents electronically, and its CDA is maintained all the while. The person receiving the document has the same editing options that the sender had—as long as he's a Slate user, that is. You can indeed send Slate documents to non-Slate users, but the document may lose some of its elements as it's translated to different environments.

BBN takes communications a step further than normal electronic mail by offering what it calls "multimedia conferencing." Slate's electronic conferencing is different from the conventional concept of electronic conferencing, where electronic mail messages are tracked and stored as conversations. BBN's conferencing lets you collaborate electronically in real time. Several users can edit the same document simultaneously with the changes appearing on everybody's screen. Usually Slate's conferencing is accompanied by a telephone conversation, so you can talk about what you're editing. It's a very useful tool for workgroup computing.

SLATE AS AN INTEGRATING ENVIRONMENT. As we mentioned earlier, BBN has recently added an extension language for Slate called SEL (Slate Extension Language) that turns the product into an integrating environment. SEL is essentially a scripting language—for programmers, not end users—that makes the system more configurable. You can customize menus, add applications, create new menu choices, and even automate repetitive tasks and pull information from non-Slate applications into Slate documents (pulling information from a database into a spreadsheet, for example).

This is a very powerful addition to Slate, and we were duly impressed with its capabilities. Our only disappointment is the interface. Again, this is a programmer's tool. Yes, you can link or repeat user operations to automate a routine task, but Slate doesn't have any keystroke capture mechanism to make the creation of that automated task easier. Also, the interface of enclosure mechanism that BBN uses for including non-Slate application data into Slate documents could be a little smoother. The first few steps are easy: A dialog box pops up that asks you to specify the name of the source file as well as its editor. But in the end, you still wind up with some Unix-ish procedures.

COMMENTS. Despite the fact that Slate's extension language and enclosure mechanism are a little too technical for commercial end users, providing the ability for organizations and administrators to customize the product as well the ability to integrate third-party and existing applications is a wise move for BBN. It gives the product a good deal of flexibility. SEL coupled with Slate's original compound document and electronic mail functionality make the system quite compelling. These are different features from those you'll find in most other compound document products.

Slate runs on Digital DECstations and VAXstations running Ultrix, IBM's RS/6000, and Sun workstations. BBN has lowered the original price of the product to $995 per license.

—L. Rowan

·X WINDOW·

Making X Terminals a Better Option

Implementing an X terminal environment is no small decision. There are many pros and cons involved. X terminal critics are quick to point out their drain on network resources. However, some recent developments in X terminal technology—especially in networking—are making them a more viable option for organizations ready to plunge wholeheartedly into X.

FIRST, THE CONS. Despite the fact that X terminals are relatively inexpensive (you can buy a decent black-and-white model for less than $2,000), they can exhaust a network. Typically, the host needs to keep a copy of the window manager for each X terminal it serves. Window managers use about 225K of memory—and that can really add up. (Consider that an average X environment can include about 100 terminals.) Of course, the host can use shared libraries to cut down on memory, but there's still a network congestion problem because the terminals need to access the client each time a user merely moves a window or pushes a button. Furthermore, if the terminals are not configured with EPROMS (a memory chip version of the X server), the host must download the entire X server each time an X terminal starts up.

Therefore, many organizations are planning to rely on workstations and PCs instead.

THE PROS. But recent X terminal technology might change these plans. NCD, the largest X terminal vendor, has effectively solved the problem of tying up memory for multiple copies of the window manager. The company recently developed a window manager (NCDwm) that resides on the X terminal rather than the host. NCDwm is part of the server—either on the server binary file that is downloaded from the host or included in the EPROM configuration. Thus, the X terminal doesn't rely on the host for window operations, thereby improving performance and cutting down substantially on network cost. NCDwm has the behavior and appearance of the Motif window manager, but NCD claims that it's smaller, so it takes up less room—again, saving memory. The bottom line is that the local window manager can save up to 1MB of memory per X terminal. Another big plus is that you can use NCD X terminals to access hosts that don't run the X Window system. Since these terminals carry both an emulator and window manager, they don't depend on the host for window management. You can even run PC applications on these machines.

NCD has done some impressive work here. The only shortcoming is that NCDwm has no window manager config file, which somewhat limits its configurability. But you should still be able to change all the typical window attributes, such as window borders, color, menus, buttons, etc.

NCDwm is part of NCD's latest release of its X server software. In addition to the local window manager, NCDWare 2.3 also features:

- A remote reset, which lets administrators restart NCD servers from a remote site
- Support for MIT's X11R4 authentication scheme
- Support for up to eight local X clients
- Compatibility with Ultrix 4.0

Network Performance. NCD seems to be focusing particularly on taking the load off the network and putting it on the terminal to speed up performance. On the other hand, its nearest competitor, Visual Technology, is zeroing in on enhancing network performance. Visual, for instance, has rewritten TCP/IP to optimize it for the network performance of its X terminals. Its server software also includes Network Services set up enhancements so that network administrators can configure the Visual machines to the network environment. Another nice feature is Visual's memory failure protection. With it, X terminals don't lock up when the client has run out of memory. Instead, the client is notified that the terminal is out of memory and warned when memory starts getting low.

Administration. Furthermore, X terminals generally offer a big administrative advantage. There is a huge difference between maintaining a large number of workstations and a large number of X terminals. Terminals require no application or operating system installation, no backup, no Unix overhead, no software updates, no network management chores. They are also more secure than workstations (there's no "root" on an X terminal). Performance is better because all the processing power is devoted to X Window operations. Moreover, X terminals are easily adaptable to multiple hosts and multiple applications (at least, the third-party X terminals are; Digital and IBM X terminals must be hooked to Digital or IBM host counterparts—very disconcerting).

Most importantly, X terminal technology has evolved quite a bit lately to take care of some of the problems outlined above.

COMMENTS. Other X terminal vendors besides Visual have adopted network optimization techniques to improve X performance. NCD is no exception. It even provides compression algorithms for serial connections. Frankly, NCD's network optimization techniques coupled with its server developments give the company a considerable technical advantage in the X terminal marketplace. *—L. Rowan*

THE EXECUTIVE UNIFORUM SYMPOSIUM

▲ UNIX® AND OPEN SYSTEMS

Applications, Tools, and Solutions for the 1990s

May 21-23, 1991
Four Seasons Biltmore Resort Hotel, Santa Barbara, California

Registration Fee:	For orders paid by February 22, 1991	$995
	For orders paid after February 22, 1991	$1,095

▲ Sponsored by:

UniForum, X/Open & Patricia Seybold's Office Computing Group

▲ Overview

The Third Annual Executive UniForum Symposium will spotlight the realities of implementing UNIX by exploring the applications and tools necessary to move UNIX and Open Systems into a traditional MIS shop.

Tools, applications, and strategies will be the major focuses of this conference. We will explore strategies for integrating UNIX with existing systems, tools for CASE and 4GL development, and critical applications for Open Systems migration and implementation. In addition, we will look at database interoperability, systems integration, and user requirements in the age of Open Systems.

This three-day conference will present views of respected industry leaders—including hardware, software, and networking vendors—and influential commercial users. More than half of the conference speakers are corporate information systems executives. We will also discuss and explore next-generation UNIX applications and platforms, such as distributed computing applications and object orientation.

▲ Registration and Speaker Information

Call **(800) 826-2424** for a conference schedule and registration information. Massachusetts and international callers dial **(617) 742-5200** or fax **(617) 742-1028**.

Why You Need to Attend

If knowing the direction of technology is critical to your success and to the success of your corporation...

If you're currently migrating to UNIX and Open Systems from a traditional Information Systems environment...

If you're vitally interested in the evolution of commercial UNIX and open systems...

If you're concerned about planning, implementation, and migration issues...

Then you need to attend this conference!

Final Registration Notice!

Objects and Networks: Creating Object-Oriented Applications in the Distributed Environment

April 9, 10, & 11, 1991 **Registration Fee: $ 895**
Marriott Hotel, Cambridge, Massachusetts

Overview:

Object-oriented architectures and distributed computing environments promise to end the reign of systems that are inflexible to use and unable to evolve. These key technologies will become the foundation for information systems that support the "adaptable enterprises" that will succeed in the tough competitive climate of the '90s. "Objects and Networks" gives corporate information-systems planners what they need to understand how to put these powerful technology enablers to work today.

Call 1-800-826-2424 (or 617-742-5200) for a speaker schedule and updated conference information.