Patricia Seybold's

UNIX IN THE OFFICE

VOL. 3, NO. 6
ISSN: 0887-3054
JUNE 1988

INSIDE

PRODUCTS • TRENDS • ISSUES • ANALYSIS

SmartWare from Informix Software

The Quiet Contender

By Judith R. Davis

THE STATURE OF UNIX as a platform for office applications is growing rapidly, and we see a wide-open window of opportunity here. While the competition is starting to intensify, the players are still jockeying for position, and no one has a lock, or even a near lock, on the market. SmartWare from Informix Software, Incorporated, is a potential sleeper, a dark horse that could come from behind in the race for the Unix office environment. *(continued on page 3)*

• E D I T O R I A L •

Working Together: Unix Style

Collaboration is the Key to Making Standards Work.

By Judith S. Hurwitz

IT IS A TIME of confusion and debate for the Unix community. The enthusiasm of a year ago over the possibility of one unified version of Unix has been replaced by dissension—the industry is split into two camps. If you follow the trail of events, it's clear how this happened and why. When AT&T and Sun Microsystems began flexing their combined muscle power, competitors that had planned to go the Unix route became concerned and confused. They wanted a voice in determining the shape of the operating system. They felt shut out. And the rest is history. But now the Open Software Foundation (OSF) has become a reality. What will this mean for Unix and open systems in general?

We challenge those doomsayers who predict the demise of Unix and its role as a standard operating system. Yes, for a while, OSF will cause confusion. But, in the long run, we believe it will be good for the standards movement. As one industry source pointed out, the money pot that OSF is starting out with is equivalent to the R&D budget of a multi-billion dollar corporation. And money means power—even the power to develop standards.

It would be foolish at this juncture to say that the task ahead will be easy. It will not be. Cooperative work can yield astonishing results. Yet, if you ask what it's like to work on task forces, you'll receive mixed reactions. People will often insist that anything outstanding has to be conceived of and designed by a single individual. This is sometimes true because, in reality, team work and the cooperative work process are difficult to achieve. It takes a lot of hard work to develop a team of people who can challenge each other's opinions without becoming judgmental and who can respect each other's strengths. It takes effort to be able to create an atmosphere where each individual can contribute at his or her own level. It's not easy to create a team where no one is quite sure exactly who came up with which piece of an idea—where, in fact, the idea is a group effort for which no one participant can take credit.

However, once a group takes the time to jell and learn to work together, the results can be outstanding and remarkable.

So you might ask, "If team work is so hard to achieve, why bother?" Simply put, the results of the work of a well-functioning team are double, or even triple, anything a single individual can accomplish.

This is a valuable lesson for the standards movement. We are beginning to realize that, in an age of complexity and competition, one vendor alone cannot rule the world—or set the standards. The only way individual companies can survive and thrive is to work together. Coalitions that are springing up all around us attest to this.

Who benefits from cooperative efforts like OSF? Everyone, and most especially users. Users, like their counterparts in the vendor community, cannot afford to go it alone. Users need the reassurance that they are not going to be abandoned by obsolete hardware or esoteric software. Most importantly, users need to be at the center of these cooperative efforts because, in essence, vendors are cooperating in order to satisfy the long-term requirements of users.

X/Open understands this need and has created a user council to provide information and guidance to the vendor community. The new OSF must now take similar action. The challenge will not be technological. With its money, its years of experience, its access to technology from the top vendors and key universities, resources will not be the problem. OSF must learn how to take a group of companies—each with its own needs and goals—and forge a coalition. Such a coalition would have the potential to change the nature of the computer industry for many years to come. We believe that it is in the best interest of the computer industry to rally behind the Open Software Foundation and allow teamwork and technology to flourish.

Patricia Seybold's Office Computing Group

Publisher PATRICIA B. SEYBOLD
Editor-in-Chief JUDITH S. HURWITZ
Managing Editor RONNI T. MARSHAK
Staff Writer LAURE BROWN
Senior Editors MICHAEL D. MILLIKIN, JUDITH R. DAVIS, 33 Summit Road, Lexington, Massachusetts 02173, Telephone: (617) 861-3926
Sales Director RICHARD ALLSBROOK JR.
Circulation Manager DEBORAH A. HAY
Assistant Editor DAVID S. MARSHAK
Contributing Editor GARY J. NUTT, 3450 22nd Street, Boulder, Colorado 80302, Telephone: (303) 444-0341

148 State Street, Suite 612, Boston, Massachusetts 02109 Telephone: (617) 742-5200 FAX: 1-617-742-1028

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 *UNIX in the Office*, 148 State Street, Suite 612, Boston, MA 02109.

•SMARTWARE•

(continued from page 1)

SmartWare is an integrated suite of generic office application software that runs on the DOS, Unix, and Xenix platforms. It includes word processing, a spreadsheet with graphics capabilities, a database management system, personal time management, and communications. SmartWare is also often referred to as the Smart Software System. While Smart does not yet have electronic mail, group scheduling, or access to an SQL-based relational database management system (RDBMS), it is on target with its high level of integration among the Smart modules, and ahead of the game in providing a cross-application macro facility.

Innovative had the misfortune to introduce Smart at the same time the "battle of the giants" took place, when Lotus and Ashton-Tate simultaneously introduced Symphony and Framework.

The recent merger between Innovative Software, the company that originally developed Smart, and Informix Software, the developer of the Informix-SQL family of RDBMS products, will certainly provide Smart with access to a popular RDBMS. And the company is committed to making Smart a fully-featured integrated office product by adding workgroup functionality. The key to SmartWare's success will be how fast the company moves to implement these missing pieces.

Company Background

Innovative Software was founded in 1979 by Michael J. Brown and Mark R. Callegari, then both in their early twenties, as a bootstrap, basement operation to develop software for early predecessors of the IBM PC. Working on Vector Graphics and CP/M machines, their first product was TIM, a database management package for the CP/M operating system. Introduced in 1980, TIM was, in fact, one of the earliest application software programs for the personal computer.

After the IBM PC debuted in 1982 and Lotus hit the market in early 1983, a new type of software approach—integrated packages—began to command interest. Innovative started developing the Smart Series for the IBM PC in 1983, shortly after Lotus came out, and went public to fund the launch of the product. The initial stock offering raised almost $5 million.

Innovative then had the misfortune to introduce Smart at the same time the "battle of the giants" took place. Remember the summer of 1984, when Lotus Development and Ashton-Tate bashed their heads together while simultaneously introducing Symphony and Framework? This situation presented two problems for Innovative. First, the marketing hype and advertising budgets for both Framework and Symphony were overwhelming. Innovative's Smart Series, released in September, had difficulty commanding the attention of either distributors or end users.

Secondly, although Smart took a different approach to integration than either of the other two products (and one that is much closer to what users are looking for today), Smart was perceived as being the same type of product. Innovative was not able to effectively differentiate Smart from the competition in its marketing. Therefore, users comparing Smart to Symphony and Framework found it different and, thus, deemed it not as good as an integrated product.

In spite of these obstacles, sales of Smart gradually grew. A brief sales downturn a year after the product was released was reversed with the introduction of Version 2.0 This time, the company promoted Smart correctly, stressing the power and features of the individual modules as well as the integration aspects. (In actuality, the definition of integration espoused by Smart is the one that has lasted. Notice that Symphony and Framework have pretty much faded from sight, while Smart is now seeing its market share grow.)

Smart began selling well in the corporate environment as users began to appreciate the consistency of interface and functionality across the modules, features which can significantly reduce training requirements and cost.

The Smart Series is now on its third major revision. Version 3.0, introduced in March 1986, included networking capabilities for local area networks (LANs), one of the first PC products to do so. The company maintains that some of its competitors still haven't adjusted to a networked environment. At the time, Innovative provided separate LAN and single-user versions of Smart. Smart now comes as a "network ready" product (Version 3.1 introduced in January 1987). For LANs and Xenix networks, the customer buys additional software licenses and documentation for each workstation on the network.

THE UNIX PLATFORM. While developing Version 3.0 for DOS, Innovative began conversations with potential Unix original equipment manufacturers (OEMs). According to Ron Ferguson, vice president of Marketing for Informix Software, "Unix was an important decision for us. Smart was written in C with the thought of eventually moving it to the Unix environment, even though Unix was primarily used in academia when the company began developing the product." An important factor in the decision was the lack of competition for productivity software in Unix. "Unix represented a marketplace where products with the friendliness and ease of use found in DOS software did not exist."

The first step toward Unix was an agreement with AT&T under which AT&T would OEM a co-labeled Smart product for its 3B1 and Unix PC platforms. Released in the fall of 1986, this was Smart's first multiuser product. While the 3B1 was not wildly successful, Smart is still sold for this machine and is now also available for the 3B2 series.

In January 1987, Smart announced a similar relationship with NCR, co-labeling Smart for the NCR Tower 32 line of

SmartWare Pricing

Product	Introduced	Currently Runs On	Cost
SmartWare, integrated office software that includes: • Smart Word Processor • Smart Spreadsheet with Graphics • Smart Database Manager • Time Management • Communications • Project Processing	1983	DOS	$895
	1986	PC LANS (Smart supports any DOS 3.1 and NETBIOS compatible LAN)	In addition to SmartWare at $895 (for the server), a software license and documentation for each node costs $595.
	1986	Unix: AT&T 3B1, 3B2 NCR Tower Xenix (both 286 and 386) Coming are versions for Microport Unix, Altos 386 (Unix), and OS/2	$1,595 for Xenix (1 to 32 users); the AT&T and NCR Tower versions are sold by the OEM vendor.
The three major modules of SmartWare are also available as standalone products. Each includes the Time Management, Communications, and Project Processing functions as well.			
SmartWord Processor		Same as above	$395 for DOS $199 for each PC LAN node $695 for Xenix (1 to 32 users)
Smart Spreadsheet with Graphics		Same as above	$495 for DOS $199 for each PC LAN node $795 for Xenix (1 to 32 users)
Smart Database Manager		Same as above	$695 for DOS $199 for each PC LAN node $895 for Xenix (1 to 32 users)

Unix computers. This product was released last fall. Both NCR and AT&T not only sell Smart, but also provide their own customer support and maintenance for the product.

OTHER PLATFORMS. Over the past year, Smart has been ported to other Unix platforms, primarily Xenix (both the 286 and 386 versions of Xenix System V) and the Altos 386 (Unix System V/386—although this version of Smart has not yet been released). The company recently demonstrated Smart for OS/2 and Microsoft's LAN Manager and plans to introduce these in the future.

MERGER WITH INFORMIX. A major event for Innovative was its merger with Informix Software, a primarily Unix-based software vendor well known for its family of relational database management products such as Informix-SQL and Informix-4GL. The two companies approached the merger from different but complementary perspectives. Smart's Unix OEMs, NCR and AT&T, also have a lot of customers with Informix databases who want to access the data from the Smart modules. At the same time, Informix was looking for a connection with an office automation software company to provide a front-end user environment for its database products.

The merger was completed last February, and the combined company retains the name Informix Software Incorporated, a wholly-owned subsidiary of Informix Corporation. Revenues for the combined company were $68.5 million for calendar year 1987. Innovative Software contributed $26.8 million of this.

Product Positioning

The Smart Series consists of three of what the company calls "feature modules": the Smart Word Processor, the Smart Spreadsheet with Business Graphics, and the Smart Database

Manager. All three have common commands and screen designs to provide consistency in both interface and functionality, and all three can share data. Each module can be purchased separately. They can function individually as standalone programs or together as an integrated system (as SmartWare). SmartWare provides a menu umbrella for the modules and adds two support modules for time management (calendar and personal scheduling) and communications. In addition, SmartWare provides what it calls a project processing capability. This is essentially a programming language that allows the user to develop automated applications that reach across the Smart modules, a sort of cross-application macro facility.

The company's major direction has always been to develop fully featured standalone products that can compete with the best in their class and then to provide tight integration among the standalone modules. This is different than the approach taken by both Symphony and Framework, which emphasized integration to the detriment of the individual applications.

The company is now poised to enhance Smart in two directions: to improve the capabilities of the individual modules and to add a broader range of office functionality to the product. Both improvements are required for Smart to continue competing with both standalone and integrated products.

DOS. The main competition in the DOS environment comes both from standalone products (primarily Lotus 1-2-3, dBase III, and Word and Word Perfect) and integrated products. In the latter category, Informix sees Symphony, Enable, and Framework. The company feels that it competes very well against the DOS integrated products, none of which offer electronic mail or group scheduling, either.

UNIX. Competition in Unix comes from both directions as well. Standalone competitors mentioned by the company include SCO's products (such as Foxbase—a DBMS; SCO Professional—a spreadsheet; and SCO Lyrics—a word processor), Q-Calc from Quality Software Products, UltraCalc from Olympus, and 20/20 from Access Technologies. The more difficult competition comes from the integrated software from Uniplex II, Q-Office, R-Office, and Alis, since most of these offer a more complete line of office applications than does Smart. The company sees its lack of electronic mail in particular as a bigger disadvantage in Unix than in DOS. On the other side of the coin, the company also sees Unix as offering more of a competitive opportunity than DOS; not as much office software is available in Unix.

Marketing Strategy

Informix Software has four primary channels of distribution for Smart—direct sales, retail channels, OEMs, and value-added resellers (VARs). To date, sales of Smart in the DOS environment have been almost equally divided between direct sales and sales through retail channels. There are also some DOS VARs. For Unix, all of Smart's sales to date have been through its OEMs (AT&T and NCR, who both sell Smart directly) and through VARs. The company also offers sales through all three channels for the recently introduced Xenix version of Smart.

Informix plans to make a strategic shift to a greater emphasis on the VAR and retail channels for distributing Smart. Smart, with its project processing capabilities, is an attractive platform on which VARs can design custom applications. Informix does not offer a run-time version of Smart, an issue that may become more important with increased emphasis on VAR sales. A run-time version is under consideration and is an open issue right now within the company.

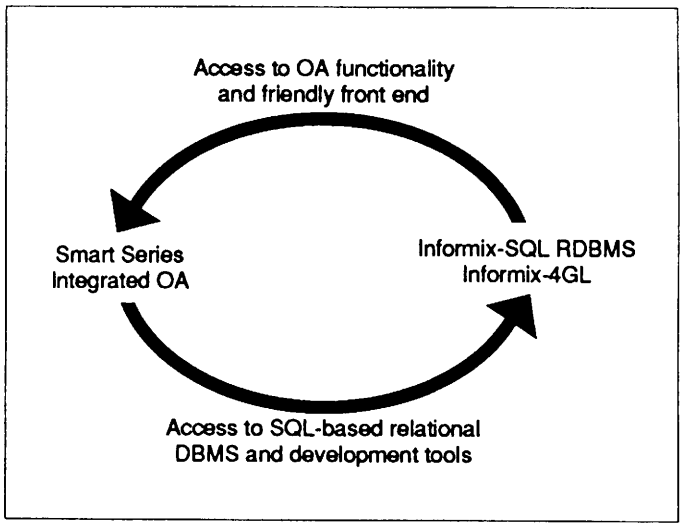
Informix indicates that it has priced Smart to make purchasing the full integrated package more attractive than buying one or more standalone products. Standalone sales account for only about 10 to 12 percent of total revenues. The company also stresses the full SmartWare product in its advertising.

The company states that it has shipped over 160,000 units of Smart worldwide for all platforms. (Smart is available in three languages—English, French, and German, with a Spanish version coming.)

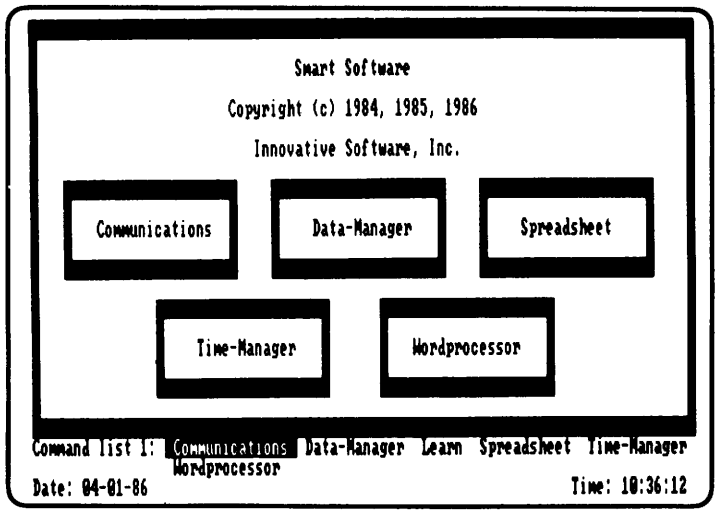
Future Plans

VERSION 4. The next major release of Smart, Version 4.0, will provide many enhancements to the individual modules. Improvements will include streamlining some functions, adding support for a wide variety of printers, and a streamlined user interface for the Database Manager.

INTEGRATION. Today, no direct link exists between Smart and the Informix line of RDBMS products. However, the company has definite plans to bring the two product families together, which was the main reason for the merger in the first place. Innovative Software's first contact with Informix was through requests to access Informix databases through Smart. Both NCR



Informix Software plans to integrate SmartWare with the Informix family of relational DBMS products.



The Smart main menu.

and AT&T have many customers with Smart and Informix installed on the same system.

The first step here will be to enable users to share data between the products using import/export options. Eventually, users will be able to transparently read files directly.

The company recognizes the importance of bringing an SQL-based DBMS to the market with a friendly, windows-based user interface. Informix Software has introduced Wingz, a graphical, windowed interface, for the Apple Macintosh, its first foray into this area. The company also intends to support IBM's Presentation Manager and the Unix-based OpenLook interface. The ability to expand Smart's strong windowing capability to implement multitasking windows will enhance the appeal of the product in the future. The company states that over the next 18 to 24 months, we will see a different look to SmartWare as the company moves in this direction.

Another level of integration will be to bring together the fourth-generation programming language of Informix-4GL and the programming language (4GL) in the Smart Series. The company describes the Smart Data-Manager as an end-user database product (which it will comfortably put up against Ashton-Tate's dBase III), while Informix is a developer's database product with its 4GL and its strong applications development tools. It would be a definite plus to create a language that can manipulate data across an RDBMS as well as office automation (OA) applications.

ENHANCED OA CAPABILITY. The company states that Smart will also take on the look of a more fully-featured OA product with the addition of workgroup-oriented functions. At a minimum, these will include electronic mail and group-scheduling applications.

Informix realizes that it needs to provide an underlying communications mechanism for future group applications. Electronic mail will provide the initial underpinning for this. The next step will be support for networking on different platforms and gateways to other platforms and applications. Digital, for example, has a large base of electronic mail users, as does IBM with Profs and DISOSS. So Informix intends to support wide-area networking (WAN) to connect DOS and Unix as well as to provide gateways to other platforms.

Another major enhancement to Smart will be the handling of compound documents with dynamic links among documents.

VISION. The company's long-term vision includes blending its OA capability with a strong underlying database, giving users the ability to effectively manage and access information among workgroups. The user will be able to get a full response from the database for questions such as, "What information do we have on this subject?" The ability to store any type of data in the database, including graphics, voice, and images, will assist here.

Using Smart

We evaluated SmartWare Version 3.1 (the latest general release version) on an NCR Tower 32 under Unix System V. The program is very easy to install; the installation procedure is set up to use the standard Unix system administration functions.

User Interface

You access Smart simply by entering "smart" at the Unix prompt. (You can also access each of the Smart modules directly from the Unix prompt, e.g., "smartwp" for the word processor.) You can also add codes at the Unix prompt to enter Smart and automatically invoke a process such as: execute a command, execute a project file (stored program), or set the default directory path for datafiles.

Smart works on both DOS and Unix, and the company has chosen to adhere to DOS naming conventions for files, which are more restrictive than Unix, in both environments. In Unix, however, filenames are case-sensitive to conform to the Unix file system.

Smart uses a menu-driven interface with ring menus displayed across the bottom of the screen.

NAVIGATION. There are five different menus in each module. While the way the commands have been allocated among the menus is not intuitively obvious at first, a method does, in fact, exist. Menu 1 includes basic operations in each module (examples: edit, copy, move, print). Menu 2 controls the way information is displayed on the screen (examples: sort, bold, font). Menu 3 includes options such as formatting of information and opening and closing windows. Menu 4 contains file manage-

ment commands, such as load, save, directory, and index. Menu 5 is for more advanced commands, such as setting parameters, creating macros and project files, and miscellaneous commands that don't clearly fit on a previous menu. You can display a specific menu simply by entering the number of the menu, or you can view menus sequentially with the slash key (forward) or backslash key (backward).

Items on each menu are listed alphabetically, and each begins with a unique first letter. Thus, you can highlight an option and execute it in one keystroke by entering the first letter of the command (as in Lotus). Otherwise, you move across the menu with the spacebar or "+" key (to move to the right) or the backspace or "-" key (to move to the left). We had trouble getting used to the fact that the arrow keys didn't work for this. The menus are circular (ring menus).

The entire menu is always displayed even if it requires two lines for all the items. That is a plus in our estimation. We dislike having to press "more" or some equivalent function to see all of the choices on a menu.

One of the things we don't like about the Smart menu system is that you cannot back up through your menu choices one step at a time. If you press Escape, it takes you back to the beginning (the menu item you started on) and you must start over.

Informix will shortly introduce a maintenance release of Smart for the NCR Tower in which the user will be able to put one additional item on the main Smart menu. When this option is selected, Smart will go into the background. When the user exits from the custom option, he will return to Smart. This feature was specifically requested by NCR, and it will be ported to other Smart platforms in the future.

When you exit from a Smart module, you have three options: quit to Unix, return to the main menu, or enter another Smart module.

FUNCTION KEYS. Smart also makes use of function keys on the terminal. Currently in the Unix environment, you use the first four function keys as follows:

- F1 is Escape or Cancel
- F2 is a lead-in to get 10 function keys, each of which is assigned to a particular command—e.g., pressing F2 and then pressing 4 gives you the equivalent of F4; F2 plus 0 is F10.
- F3 is the Control lead-in
- F4 is the Alt lead-in

This is obviously not as nice as just using 10 function keys as labeled, but it is easy to get used to. (We made our own template for the keyboard. Informix, unfortunately, doesn't provide one that fits on the keyboard. You do get a card, though, that summarizes the function key layout.)

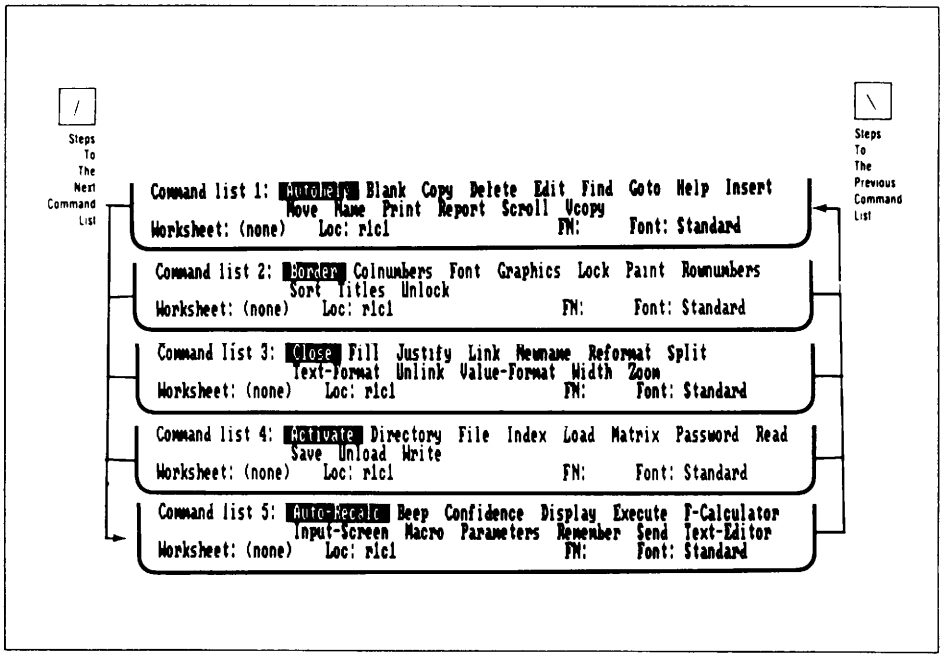
The maintenance release of Smart for the NCR Tower mentioned above will also provide the use of F1 through F10 on terminals that support these keys. F11 and F12 will become the Control and Alt keys. Again, this was requested by NCR, and it will subsequently appear across all Smart platforms.

The function keys are almost totally consistent across the modules. In each module, only two function keys perform somewhat different functions.

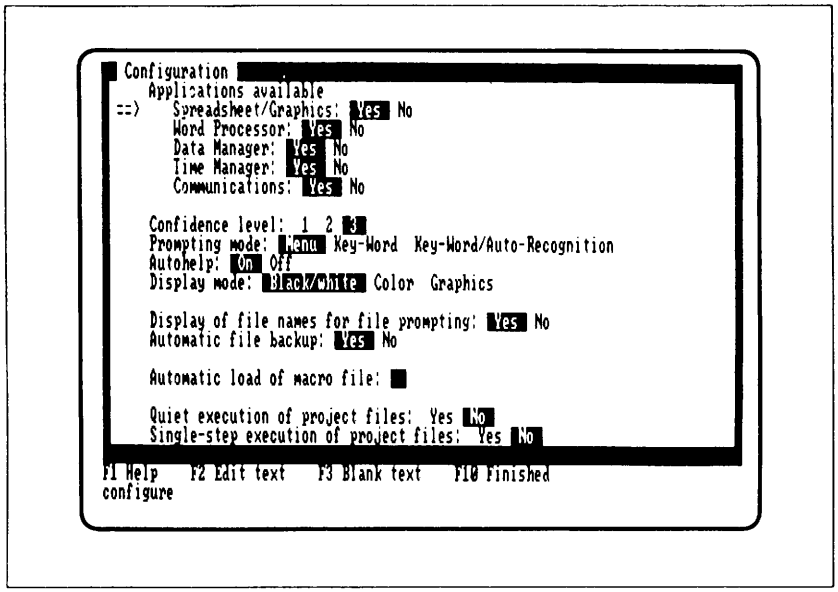
QUICK KEYS. Smart also makes use of a number of "quick keys" that are consistent throughout all modules. These cover frequently used commands such as retrieving and saving files, opening and closing windows on the screen, printing, and viewing a directory. For example, Alt S will save a file without requiring the user to go to the menu.

CONFIGURING THE SYSTEM. The user can establish system defaults, such as date and time format displays, with the "configure" command on the main menu. Module-specific defaults, such as the margins in Word Processing and column widths in the Spreadsheet, are set with the "parameters" command within each module. Smart does not yet support color terminals in Unix, but it does in its DOS version.

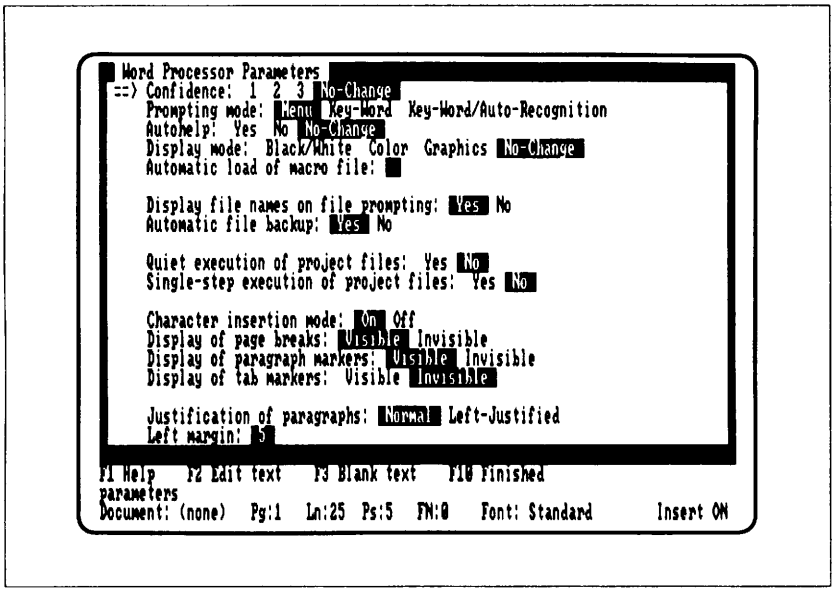
One nice option in Smart is setting the user's confidence level. This ranges from level 1 (only the most basic commands are displayed on each menu) up through levels 4 and 5 (where no menus appear at all and the user is thrown into command mode). The difference between 4 and 5 is that, at level 5, Smart provides automatic command recognition; the user only has to type in



The five command menus for the Smart Spreadsheet module.



This is the menu for configuring the Smart system.



This screen illustrates some of the parameters the user can set in the Word Processor.

enough characters of a command to make it unique before Smart executes the command.

So Smart has the flexibility to cover novice users up through experts. The power user can choose to zip through commands rather than see menus displayed. Confidence level 3 fully displays all of the menu items, and that is where we set our confidence level. The tutorials, described below, cover each module at confidence level 1.

DOCUMENTATION. Smart documentation includes a User's Startup Guide, a System Manual, and a separate manual for each of the major application modules—Word Processing, Spreadsheet/Graphics, and Data Manager. The System Manual explains the communications, time management, project processing, and the programming language, and contains a single index covering all of the documentation.

The manual for each module has two sections, a Reference Guide and a User's Guide. The Reference Guide contains an explanation of each module command as well as a general description of the Smart environment. The documentation is generally very clear and takes you step-by-step through most operations. We would have liked more upfront discussion of how some commands are actually used and more extensive definitions of Smart terms. We also would have like more illustrations, although there are a good number of sample screens in the manuals.

One problem we had with the User's Startup Guide was the use of terms that were not explained, such as "confidence level 1." The manual also referred to displays like "the main menu" without providing a screen illustration for reference or explaining how it works. Since this is the first manual we reviewed, we expected more assistance.

Tutorial. Smart has a good online tutorial, accessed by selecting the "Learn" option on the main menu. You then can choose to learn about word processing, spreadsheet, or database management. The tutorial is interactive; the system performs some steps and asks you to perform others. If you make a mistake, you are given the opportunity to correct it, and, if you do it right, you get a pat on the back. The tutorial is also designed to accommodate more experienced users. It asks questions like "have you ever used a spreadsheet [database/word processor] before?" If you answer "yes," it skips over the explanation of the basic benefits and concepts and goes right into the exercises.

HELP. The online "help" function is context sensitive and provides several levels of assistance. When you first enter help (with F1), the help text is displayed in a small window of four lines on the screen, thus preserving context with the file in which you are currently working. You can press F1 a second time for what is called "large help," and the help text will take up the entire screen. There are also function keys to print the help text and to display the help index (from which you can select another topic). In addition, Smart lists related topics in a ring-menu format. You can select one with the standard menu navigation tools.

There is no "macro" level of help, such as a general explanation of each of the main menu options or of ways to move

around on menus. Therefore, help does not provide a complete substitute for using the reference manual.

Smart also provides an "autohelp" function, in which a one-line explanation of the highlighted item appears below the menu. Unfortunately, autohelp doesn't display on terminals with less than 25 lines (which includes most of the ones we know and use). We would like the option, as new users, to see less of the spreadsheet/database/document and see the autohelp message. Once we learn the system well, we could then turn it off.

MULTIUSER FEATURES. Smart provides several important multiuser capabilities, allowing users to share access to files. In the Data-Manager, Smart provides concurrency control with locks at the file (table) level for operations such as purging the file of deleted records and repairing the file. It places locks at the record level when records are being added or updated. In both the word processing and spreadsheet modules, only one user (the first one) using a document/spreadsheet can save it under its current name. All other users get a message that the document/spreadsheet is in use and that additional users can access it but must save it under a different name. In communications, only one person is allowed to access a port.

Unfortunately, Smart does not have its own options for changing the permissions on a file (i.e., to allow specific users specific levels of access to a file). This must be done at the Unix level with the "unmask" command to share all files, or the "chmod" command to change individual file permissions. We would like to see Smart implement menus for file access operations, masking the user from Unix.

The user has the ability to password-protect any document, spreadsheet, or database file created with Smart. The password is four characters, and the user has three chances to enter the password correctly.

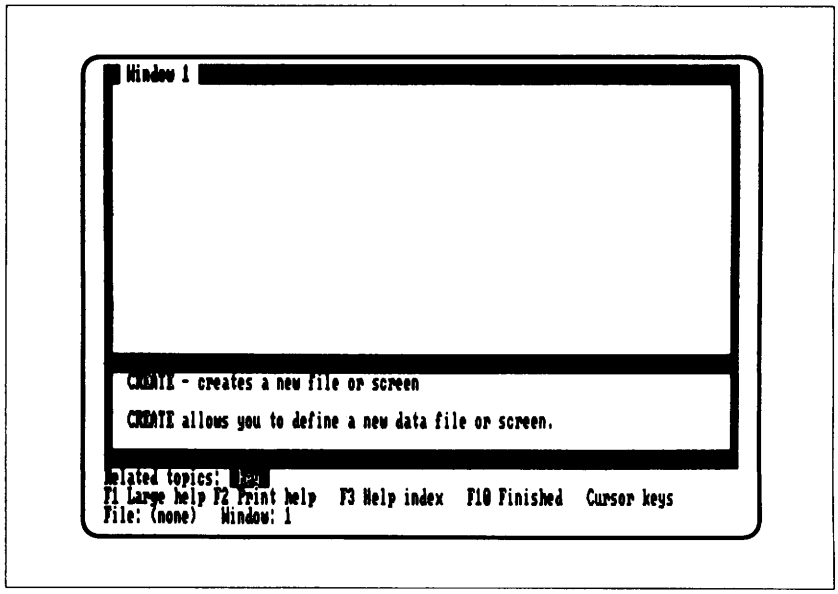
Smart does not yet provide co-authoring facilities (e.g., redlining) for documents.

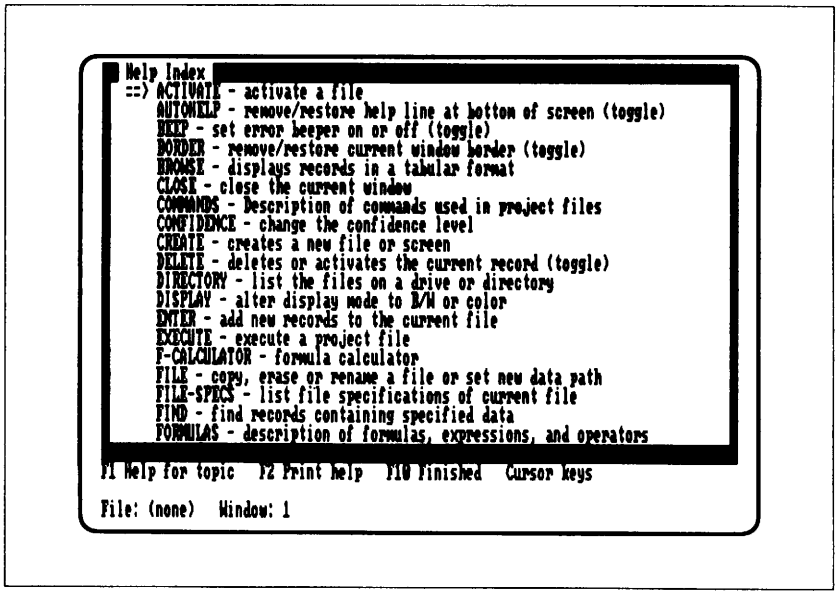
OTHER GENERAL OPERATIONS. A number of other features apply across all three of the primary Smart modules.

Entering a Module. When you enter any of the Smart modules, the screen displays are relatively consistent. Thus, once you have worked in one module, the rest all fit a general pattern. This makes Smart easier to learn and use than some other integrated products.

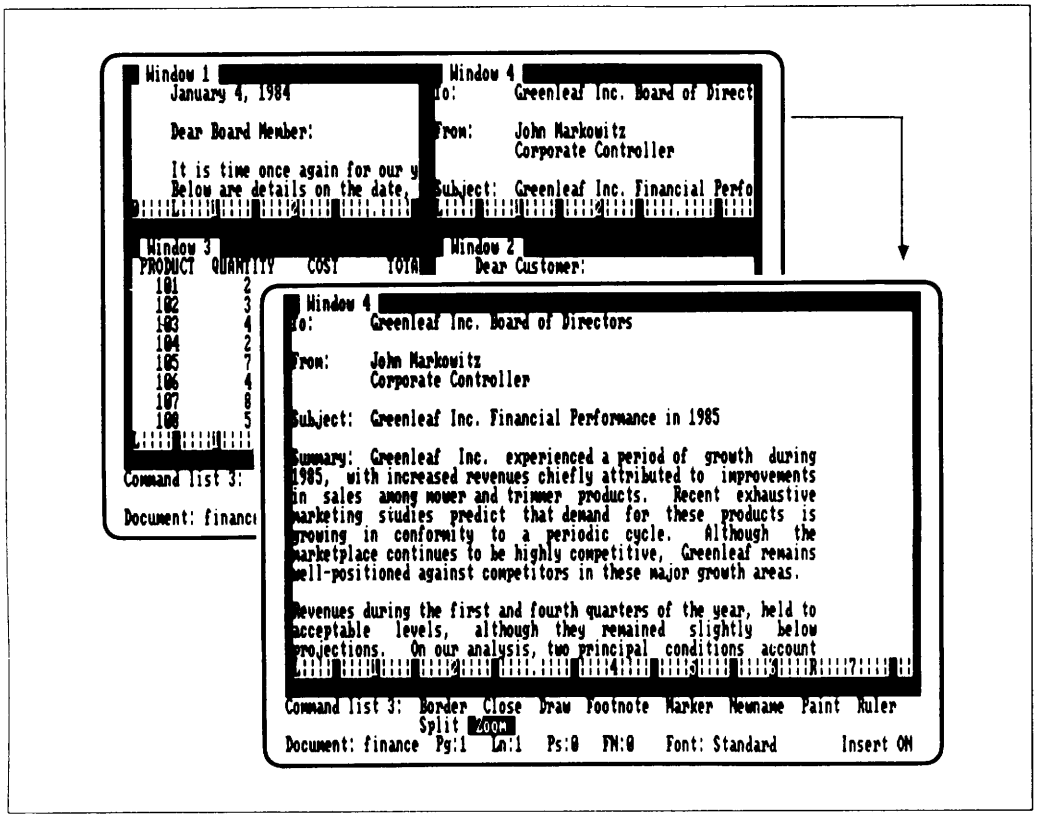
The screen display is divided into three sections. The top 20 or 21 lines make up a window for displaying the current object you are working on: a document, spreadsheet, database, or calendar. Below this window are two lines (called the Control Area) where the current menu or list of function keys is displayed. The bottom line on the screen consists of a status line. This includes the name of the current file and other status information relevant to the module (for example, in word processing, you see the page, line, and column number of the cursor location as well as font information and whether "insert" is on or not).

In Smart, you toggle between entering data/text in the window and the command line with the Escape/Cancel key (F1 on our system). The only real difference we found among the modules was that, when you enter the Database Manager, the first command menu is displayed in the Control Area and you are in command mode. When you enter the Spreadsheet or Word Processor, the 10 standard function keys are displayed and you are in data entry mode. Therefore, you can immediately start entering text into a new document or spreadsheet.





These screens show the initial help screen and the help index.



This sample screen split into four windows shows what happens if window 4 is "zoomed" to fill the screen.

Fonts. Smart provides multiple-font support for up to 13 fonts. Eleven are defined by Smart and two more can be custom-designed (with a graphics terminal) with the "font-design" command on the main menu. Generally, you can change the font of any text you enter in two ways: You can specify a current font, which applies only to new text entered, and you can change the font of a particular block of text at any time. This operation does not change the current font. Font control is available in the Word Processor and the Spreadsheet modules, but not in the Database Manager. (To include font control on a database report, for example, would require writing the report to a file and editing the file with the Word Processor.)

Windows. You can define up to 50 windows on a single screen in any Smart module. You use the "split" command to split the current window either horizontally or vertically at a location you choose by placing the cursor. Each window can contain a different file or a different view of the same file. Each window is given a number, and you move from window to window either with the "goto" command or with a function key to move backward or forward through the windows sequentially. The status line will display the status of the current window.

You can use the "zoom" feature (this is a toggle key) to enlarge any window to fill the entire screen. This is particularly helpful if you want to have more than two windows defined; the more windows you have, the smaller each one is and the more confusing it gets to remember which is which.

One nice option in Smart is to use a window to view the footnotes referenced in the text of a document. There is a specific command for this.

To access information in one file by another file, the first file must be loaded into memory (activated). This is done with the "load" command, which places the file in a window on the screen, or the "activate" command, which places the file in memory but not necessarily in a window. The "index" command provides a list of all currently active files. (It does not, however, tell you what file is in what window.)

Macros. Smart allows you to store up to 160 characters in a macro file and map the macro to any key or combination of keys on the keyboard. Selecting the "macro" command leads you through the following steps:

- Enter the key(s) that will invoke the macro.
- Enter the keystrokes to be stored in the macro.
- Save the macro.

Thus, macros are created with a "learn" mode, not a script. They can be edited, loaded automatically or manually when you enter Smart, and deleted.

Integration. Each Smart module can translate its own file format into one that can be read by another Smart module. Thus, you can share data among the Word Processor, the Spreadsheet, the Database Manager, and the Communications modules. You do this in two ways. First, you can "send" data to another module. When you do this, you actually follow the data into the receiving module. Sending a section of a spreadsheet into the Word Processor will activate the Word Processor. You also have the option to indicate the project file that you want executed after the data is sent to the new module.

The second method of sharing data is to "write" a file in one module in a specific file format. This file can then be "read" into another module. In this case, you do not follow the data; it is simply stored to the disk in a file.

When sending data from the Word Processor (WP) to the Database Manager, each line of the WP file is assumed to be a record in the datafile. Each line must have a carriage return at the end, fields must be separated by spaces or commas, and text fields must be enclosed in double quotes. In addition, the first record in the WP file determines the file structure for the database file. If the first field is text, the field will be defined as alphanumeric. The number of characters in the first field will determine the length of the field. You can create a new file or append records to an existing file.

When sending data from the Word Processor to the Spreadsheet, the data must be in tabular format. Text for a single cell must be enclosed in double quotes. You can also send a document or text file to the Communications module.

When sending data from the Database Manager, you have the option to send an entire file or only selected fields. You can select to send summarized information, and you can also indicate whether the file format is data (formatted so that it can be used in a WP merge function or in a spreadsheet) or text (for incorporation in a WP document).

When sending data from the Spreadsheet, you can send an entire spreadsheet, a portion of a spreadsheet, or a graph. You can send it in the following formats (depending on where you are sending it): document (retains font information), text (ASCII), graphics, or Smart.

Access to Unix. You can access the operating system from within any of the Smart modules. The module is suspended while you work in Unix. You return to Smart by entering "exit" at the Unix prompt. In addition, some Unix file commands are accessible from within Smart (e.g., copy, erase, rename files). You can also see a directory listing of files on the disk in each module, with handy options to sort the listing by filename, date and time, or size, or to print the listing.

The Smart Word Processor

As we mentioned, when you first enter the Word Processor module, you are in text entry mode. In addition to Smart documents, the Word Processor can also create standard ASCII files, called text files. (You can toggle between the two file types.) Smart recognizes the terms character, word, line, sentence, paragraph, document, remainder, and block for performing operations on text. Text attributes include font, underscore, and bold. Justification options include left, right, both, and center. Smart supports two types of tabs: left-justified and decimal.

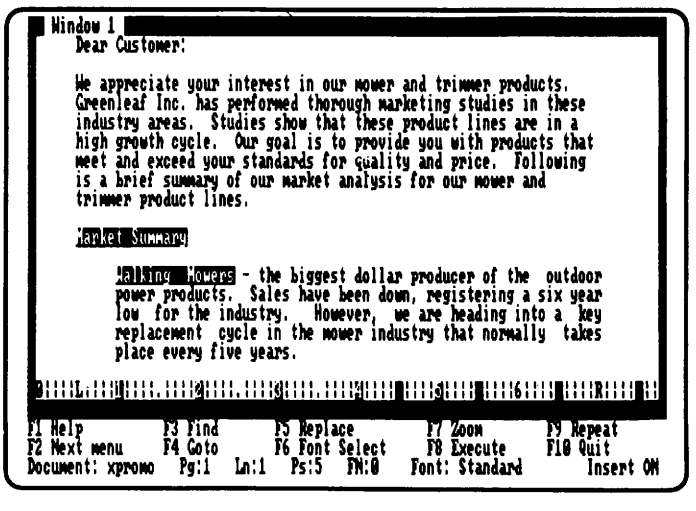
FORMATTING. You change the default document format settings with the "parameters" command. Individual commands change specific settings for the current document (e.g., margin, justify, indent, and spacing). These commands affect new text only. To change the format of existing text, you use the "reformat" command. Any formatting commands affect text at the document or paragraph level.

The indent function in Smart is equivalent to a paragraph indent, affecting the first line of a paragraph only.

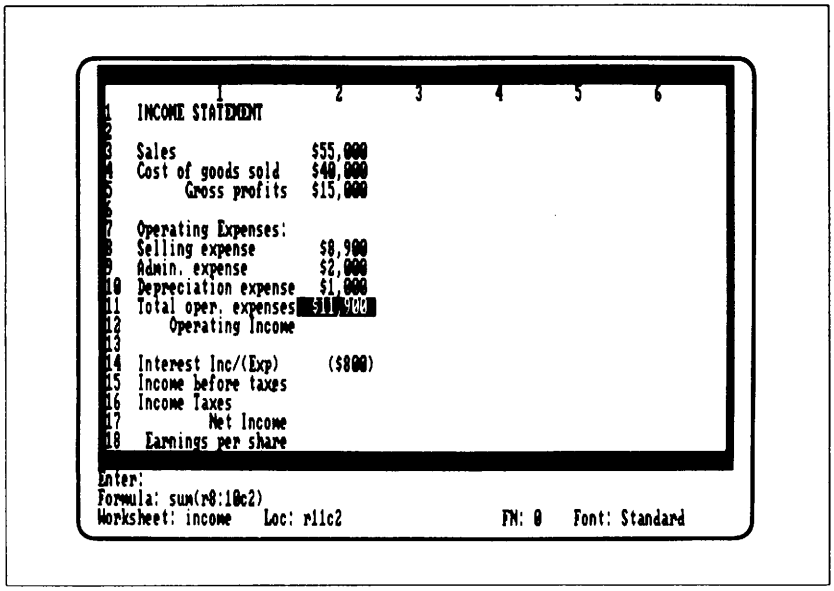
When you edit text, the Smart Word Processor does not automatically reformat it to conform to the margins until you move past the edited text. We would rather see this happen without intervention or thought from the user.

Other Word Processor functions are:

- Search and replace. (Smart recognizes capitalization but not text attributes when replacing text.)
- An "undelete" command to undo the last deletion.
- Placemarks. Marks are given names and can be viewed within the document or listed with the "directory" command.
- The ability to merge data from the Smart Data-Manager, a Smart text file, or a properly formatted ASCII file with a standard word processing document, such as a form letter. Each record in the datafile is limited to 20 fields and 1,000 characters.
- The ability to draw lines and boxes in a document. Smart gives you single, double, or heavy as the choices for a line or a box, and even gives you a choice of corners for a box: square or



Here is how the Word Processor looks after a file has been loaded into the document window. The user is in text input mode, and the 10 function keys are displayed at the bottom of the screen.



A sample spreadsheet display in Smart.

rounded. You can save a drawing in a file and recall it into a document.

- The ability to build simple formulas with arithmetic operators (+ - * /) and a "sum" command.

- Footnotes. Footnotes are automatically numbered by Smart according to where they will appear. If they are on each page, the numbering begins with 1 on each page. If they appear at the end of the document, the numbering is sequential through the document. Footnotes can be up to three lines (70 characters each) long. When you create a footnote, a three-line window opens at the bottom of the screen for the footnote text. Unfortunately, text does not wrap in the window, forcing you to hit return at the end of a line.

- The ability to sort text in a column in ascending or descending order (up to 150 lines).

PRINTING. Like formatting specifications, default print specifications can be set for all documents and changed for an individual document. Print options include, in addition to standard page layout specs such as lines per page, characters per line, and top and bottom margins:

- Headers and footers. You can indicate header and/or footer text of up to five lines each, the number of lines after the heading or before the footer, and odd/even headers/footers.

- Date style.

- Page-number style and starting number.

- Position of footnotes (end of page or end of document).

SPELLING CHECKER. The Smart Spellchecker gives you a spelling check function as well as automatic hyphenation for words that are contained in the dictionary. Smart also flags any word appearing twice in a row, or possible misuse of capitalization or hyphenation. The user can create a custom dictionary containing specialized words not found in the standard Smart dictionary. Smart supports American, English, and French dictionaries.

GRAPHS. You can insert graphs created by the Smart Spreadsheet into a document. You can size the graph as large (the entire width of the page), medium (one-half the page width), or small (one-third the page width). For medium and small sizes, you specify whether the graph is left- or right-justified; Smart automatically adjusts the text to flow around the graph. The graph is not displayed in the document, but Smart inserts a shaded block containing the pathname of the graph where it will appear. You can also view the graph in the document, although this was not enabled on our system. The link between the graph in the document and the actual graph file is not dynamic.

TRANSLATIONS. Smart has a translation utility to convert Smart documents to and from IBM's Document Content Architecture (DCA): Revisable Form Text (RFT). Smart has chosen this as its gateway, or intermediary, to get Smart documents to and from other word processing formats. There is also an option in the conversion utility that permits you to indicate font translations between Smart and DCA.

WHAT'S MISSING. While the Smart Word Processor module has a good level of functionality and ease of use, it is lacking some features that are required for the product to effectively compete with some of the standalone WP software available today. These missing features include: an uppercase/lowercase conversion, additional tab support, a complete indent facility, support for multiple columns (both parallel and snaking, with column-editing functions), support for outline formats and paragraph-numbering, additional text attributes (e.g., double underscore), widow and orphan control, block-protect facility, the ability to generate a table of contents and an index, and the ability to embed nonprinting notes in a document.

Beyond these functions, we see some WP vendors moving toward implementing other advanced WP features such as a compound document facility and the ability to generate highly structured documents (à la Lotus Development's Manuscript).

As we have mentioned, Smart does not yet support concurrent editing by multiple users. This will become more and more important as the user community begins to give more attention to workgroup computing and group productivity. And, since Informix Software has already chosen to move Smart to multi-

user platforms with its DOS networking and Unix versions, this need cannot have escaped observation.

The Smart Spreadsheet with Graphics

In the Spreadsheet module, the screen display includes the spreadsheet window (top 20 lines of the screen) with the data entry line, command display, and the status line at the bottom of the screen. The status line shows the name of the worksheet, the location of the cursor, the font number of the current cell, and the font number used for any new input.

USING THE SPREADSHEET. Smart uses numbers for both rows and columns, and designates a cell as "r4c2" for row 4, column 2. In this respect, it looks more like Multiplan than like Lotus 1-2-3. We much prefer the way Lotus labels cells, using a letter for the column and a number for the row (e.g., B4). It makes entering and recognizing cell addresses much easier for the user (and takes fewer keystrokes!).

This also makes range notation more complex. For example, a range of cells designated as r1:4c2 (i.e., the first four cells in the second column of the spreadsheet) in Smart would simply be stated as B1..B4 in Lotus. Another example: A block of cells in Smart labeled r1:4c1:4 would be A1..D4 in Lotus. We find the Lotus notation much more helpful in actually picturing the range.

When we asked Informix about this, the answer was that the spreadsheet in Smart is so large (9,999 rows by 999 columns) that numbers make more sense than letters for column labels. (In Version 4 of Smart for the NCR Tower, at NCR's request, the user will have the option of using letters to specify columns. This feature will eventually be incorporated for all Smart platforms.)

Smart gives you more flexibility than Lotus in sorting data in a spreadsheet.

Absolute references are handled the same way in Smart as they are in Lotus, again with only the designation being different—r[2]c[5]. The brackets indicate an absolute reference. (Lotus would use E2 for the same cell.)

Smart uses a sparse matrix for saving spreadsheets (blank cells don't take up space when the file is saved to disk), a concept that Informix claims it pioneered. (Lotus now does this as well.)

Formulas are built as in Lotus, including pointing to cells and ranges of cells, although the syntax is slightly different.

Other functions in the Smart spreadsheet include:

- The ability to reference data in external spreadsheet files.
- Automatic or manual recalculation.
- A "find" command to locate a specific entry or an error on the spreadsheet.
- The ability to lock cells so they cannot be edited. You can also prevent users from viewing the formulas in a spreadsheet. (This action cannot be reversed, however; the formulas are permanently protected.)
- The ability to print the formulas in a spreadsheet.

PLUSES. Some of the operations Smart can do that you will not find in Lotus include:

- The ability to build macros with a "Learn" mode. Macros in Smart are different than, and an addition to, project processing macros which can be up to 160 characters.
- The ability to print information from multiple spreadsheets on a single report without having to physically combine the information in one spreadsheet.
- The ability to display more than two windows/spreadsheets on the screen. Each window can contain a different file, not just other views of the same spreadsheet.
- The ability to repeat the last command with a function key.
- Password protection for spreadsheets.
- Project processing function that permits cross-application macros.

Smart gives you more flexibility than Lotus in sorting data in a spreadsheet. It allows you to sort the contents of a spreadsheet on up to 15 sort keys, and you can perform the sort by columns or by rows.

Another command in Smart, for those of you who are into advanced mathematical operations, is the "matrix" command. This performs a variety of sophisticated calculations on a matrix in a spreadsheet including: calculation of the determinant, rank, and power of a matrix; calculations on diagonals; calculation of eigenvalues or eigenvectors; inverting a matrix; transposition of a matrix; multiplication of two matrices; the ability to perform calculations on an element-by-element basis on two parallel matrices; and the ability to perform a multiple linear regression on values in a matrix.

SOME LITTLE NITS. Other than our problems with the cell notation in Smart, there isn't much you can't do with the Smart spreadsheet that you can do with Lotus. When you save a file under the same name, Smart does not ask you if you want to

overwrite the existing file or not. We would like to see this safety feature just in case we make a mistake and don't, in fact, want to overwrite the original file. Smart also does not ask if you are sure you want to erase the current worksheet.

One of the difficult things to get used to in Smart is the fact that the menu doesn't go away after you have completed a command; to return to the spreadsheet, you must press Cancel (Escape). This is consistent with the overall Smart interface, but we often forgot to do this since, in Lotus, the menu disappears when you are finished with certain operations (like copy, move, format, etc.). One nice thing about Smart is that it remembers your last selection on a menu and highlights that the next time you access the menu.

IMPORT/EXPORT. Smart can read and write several spreadsheet formats: DIF, SYLK, text (ASCII), and Lotus .WKS and .WK1. In addition, it can write a spreadsheet into a Smart database format and a Smart document format (including the fonts).

GRAPHICS. The Smart Spreadsheet module has an extensive graphics function for creating business graphics. You can create several types of graphs, including bar, stacked bar, line, step chart, pie chart, high-low, layer (a line chart with the area under the lines filled), histogram, and xy-graph from data on a spreadsheet.

The options provide a great deal of flexibility:

- Bars can be horizontal or vertical, lines, or three-dimensional, with values displayed on top of the bars (either vertically or horizontally oriented).
- Pie charts can be three-dimensional. They can have sectors that are exploded or left out, and the values or percentages can be displayed; you can specify the color and pattern for each sector, and the sectors can be sorted.
- Graphs can have a title and footer (three lines each, three sizes for each line of the title—large, medium, small—six fonts, 15 colors), a border around the graph and/or the entire page, a legend (you can specify where it will appear on the graph), x- and y-axis titles (color and font), and a grid—horizontal, vertical, or both, in six styles.
- The y-axis can be scaled three ways.
- You can specify fill patterns for bars and lines.

You can save a graph to a disk file, send it to the Word Processor, display it on the screen, or print it on a graphics printer or plotter. When printing to a plotter, you can size the graph as a full page or a specific quadrant of the page (up to four graphs per page). Some versions of the Smart graphics function have a slideshow option for showing a series of graphs in sequence.

The Smart Database Manager

As in the other three modules, Smart divides the screen display into the Data Window (where the contents of database files are displayed and accessed), the command list, and the status line (where the file name, the number of the current window, the number of the key field, the current record number, and the status of the current record—active or not—are shown).

DATABASE PARAMETERS. Database files are essentially unlimited in size (up to one million records). Each record can have a maximum of 255 fields, up to a total of 4,096 characters per record. A record can also have up to 15 key fields defined. Each field can be 1,000 characters. (Note that, if the user has selected confidence level 1 in the Database Manager, the database parameters are much more limited.)

In variable-length records, Smart only stores the data that is actually entered in a field. This can save disk space. On the other hand, using fixed-length fields can mean faster access to the data.

CREATING A DATABASE. Smart refers to a database as a file, and clearly describes the three steps involved in creating one: Create the file structure by defining the fields, create a data entry screen (Smart provides a default form), and then enter the data.

You choose the "create" command and give the database a file name. Then you select whether you want the file to contain fixed-length or variable-length records. (In variable-length records, Smart only stores the data that is actually entered in a field. This can save disk space for fields that are empty or that have short entries. On the other hand, using fixed-length fields can mean faster access to the data.)

Then you choose whether you want to password-protect the file. If yes, you enter the password, and all users will now be asked for this before they can access the file.

One of the helpful features of Smart is that it allows you to select a file structure template. Here, you can choose new, to create a file structure from scratch; matching, to use the structure of an existing file (Smart also creates matching screen forms for the new file based on those defined for the referenced file); or similar, to modify an existing file structure.

Defining Fields. For each field, you enter a title (field name up to 16 characters), the field type (this is a single character designating the data type, such as "D" for date or "A" for alphanumeric), and the length of the field, if required. For any numeric field, you also indicate the number of decimals (precision) and whether or not you want a running total maintained of the values entered in the field.

Fields can be any one of eight types:

- Alphanumeric, which contains character fields up to 1,000 characters (only the first 100 are used for sorting).
- Inverted name, which is an alphanumeric field where sorts are done on the last word in the field. For example, if the field contains "Michael Smith," the word Smith is used to sort on the field. In multiword entries, the user also has the option to specify which word to sort on. An example here is "The Smith Company," where you might want to sort on Smith, not Company.
- Numeric, which contains 2 to 18 characters, including the decimal point. The user can define the precision, or number of decimals, for a numeric field.
- Date, which has a default length of eight. The user does not have to enter slashes in dates. The format can be one of three choices.
- Time, which can be a 12-hour or 24-hour format.
- Social security number, which is formatted as xxx-xx-xxxx.
- Phone number, which is formatted as (xxx) xxx-xxxx.
- Sequential count, which automatically assigns a sequential number to each record. The default is to begin with 1, but you can change this or renumber an existing file.

You can also define calculated fields that can reference other fields in the record, and can perform numeric, text, date, and time calculations. And you can embed if/then/else logic in the definition of a calculated field. A calculated field can be any field type except a count field, and you indicate it by adding "c" to the field type.

Modifying File Structure. To modify the structure of a file, you first create a new file using the "similar" option. Then you make the desired changes and transfer the data from the old file into the new file. We would like to see this process done automatically.

INDEX KEYS. Smart creates indexes on a file by means of key fields. You can define up to 15 key fields (or indexes) for one file; each key can be based on multiple fields up to a total of 500 characters. Each field in a key can be indexed in ascending or descending order, and you can choose to index on only a specified number of characters in the field rather than on the entire field.

These keys are not dynamically maintained as you enter and update records in the file. However, when you are finished entering/updating, Smart asks if you want to update the key indexes now. Keys must be updated to reflect changes to the file since the last key update.

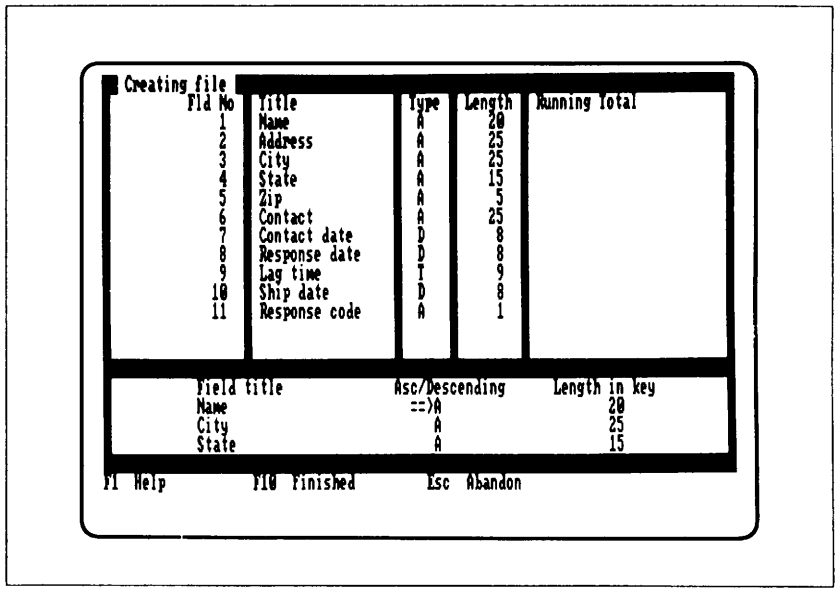
Once you create one or more keys for the file, you can view the file in order by pressing a key with the "order" command. This stays in effect until you change it or exit from the Database Manager.

Sort. The "sort" command performs the same operation as creating keys. However, the results cannot be updated based on changes to the original file; the sort must be redone to reflect changes. Smart creates a little confusion in this area for the user experienced in working with sophisticated DBMSs. These products generally use the term "index" to mean the same thing as "keys" do in Smart. Smart, on the other hand, uses the term "index," but uses it in conjunction with sorts and queries, so that it means something different than the standard definition of index.

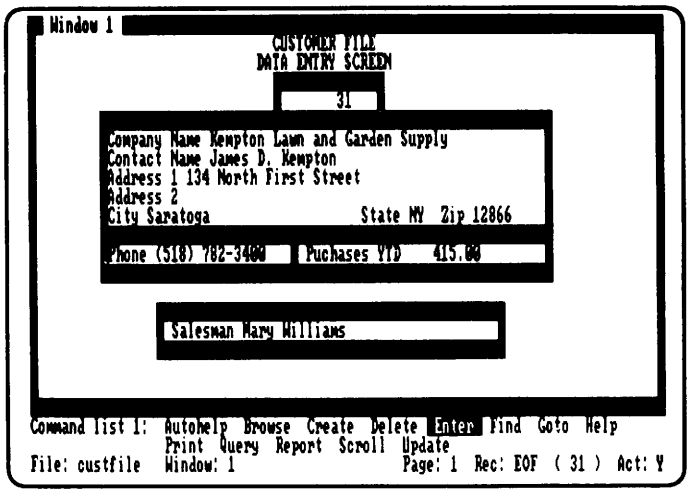
One thing that Smart does is allow the use of field numbers when defining a key or a sort. When a key is active, the field number on which it is based is displayed on the status line. We are not used to thinking of fields by number, and would rather have access to the field name(s) on which the key is built.

CREATING SCREEN FORMS. The Database Manager provides a standard default data-entry screen with all of the fields listed vertically in the order that they were defined. You can also create up to nine custom screens for each database. You again use the "create" command, choosing the screen option, and give it a name. As when designing the database itself, you can password-protect a screen and can either design the screen from scratch or choose to start with another existing screen and modify it.

Setting up a screen form is quite easy. Smart gives you the



On this screen, you define the fields in a database file and key fields.



A sample custom screen form designed in the Database Manager.

name of the field, you place the cursor where you want the field title (label) to start, and press a function key to insert the field. You can also insert a field without the field name and create a custom label for the field, and draw boxes and lines on the screen. Another option is to change the order in which data is entered into fields. Each custom form can consist of up to 15 pages, or screens.

Validation Criteria. You can specify restrictions on data entry when designing a form. The available criteria include mandatory entry, range verification for both character and numeric fields, and read only. However, the criteria defined on a form affect only data entered on that form. They do not extend to other forms, including the standard form. We would rather have the ability to define validation criteria once and have them apply to all data entered in the file, no matter what the source. We would also like to have additional options available, such as case conversion, default values (which can be defined with the use of a calculated field in Smart), customized error messages/prompts/help for a field, and more formatting options (e.g., currency).

ENTERING AND EDITING DATA. Entering and editing data is done on the active screen form for a database file. (This is the screen that is loaded with the database file.) Smart includes a repeat function that carries forward the entry made in the current field in the previous record to eliminate rekeying.

Deleting records from the database is a two-step function (handled the same way as it is in dBase III, for those of you familiar with dBase). The "delete" command is a toggle, essentially marking a record for deletion, or unmarking it if it is already marked. When a record is displayed, the status line shows whether the record is active or not (if not, the record has been marked for deletion). To physically remove deleted records, you use the "purge" command.

RELATING FILES. The Smart Database Manager has relational capabilities allowing the user to work with multiple datafiles. Relating two database files is done with the "link" command. Each file must be loaded into a window, and the two files must have a field in common (defined with the link command). (There are other restrictions to linking databases as well.) Once you have defined a link between two files, you can perform operations such as looking up information in one file and transferring it automatically to another. An example here would be entering a customer number into an invoice file and automatically retrieving the customer's billing and shipping addresses from the customer file. Another operation, called "transactions," allows you to update fields in one file based on a field in another file.

Smart also has a "relate" command used to create a new file from two existing files. The new file results from the union, intersection, nonintersection, or subtraction of the records in the two files.

As with most DBMSs with relational capabilities, you need some time to understand how to use the link and the relate commands. This includes not only the terminology and syntax, but what you are trying to accomplish with the command. And, if you are used to working with a sophisticated RDBMS such as Informix-SQL, you will have to translate some of the Smart commands into their equivalents in order to understand them (as you do with indexing and sorting mentioned above). For example, Smart uses "link" instead of "join," and Smart "transactions" are very different from transactions in Informix-SQL. These differences could present some problems as the company begins to integrate the two products.

QUERIES. Queries can be done at two levels. The "find" command provides a simple query facility; the operators available are equal to, greater than, less than, and partial. The scope of the find function can also be qualified. The default is to search forward from where you are, with other options for searching the entire file, searching backward, ignoring upper/lower case, etc.

The "query" command allows you to define more complex query criteria, store the definition, and create a subset of a database file from the results of the query. Within the query command, you can find the highest and/or lowest values in a field, use relational operators, link criteria with "and" or "or," and use parentheses to indicate the order of evaluation. Query also includes a function for replacing existing field entries for the selected records.

We found the query command somewhat confusing because of its many options, its syntax, and its use of the term "index." The manual could be more helpful in describing how to perform queries.

Smart is missing a query-by-example facility, a feature that would appeal to nonexpert users and that can be valuable for quickly defining less complex queries.

GENERATING REPORTS. Like queries, reports can be produced at two levels. With the "print" command, you can print the entire file, only the current page (what is displayed in the data window), or a single record. You can print the data only, or everything, including graphics and field titles.

You can choose one of two formats for the print command: a list (prints each record vertically with each selected field on a new line) or a report (prints a columnar report format).

Custom Reports. To customize the report and get fancier report specifications, you use the "report" command. This provides the ability to print only selected fields, some page layout specs (lines per page, line spacing, characters per line, pitch, the ability to print a page number), and the option to print records in a columnar or form layout. You can combine a table and a form on the same page of the report, and each can contain information from two different files if the files are linked. A table can also include data (columns) from two different files.

The user has many options when designing either a form or a table. Tables include column definitions (contents, heading, width), breakpoints, totals, and a report heading and footer. When defining a column, you can also specify the justification; how to handle text entries that are wider than the column; and, for numeric fields, the precision and formatting instructions (commas, currency sign, percent sign, negative number format).

Smart has implemented an impressive range of flexibility for the user in designing and formatting customized reports that will "look good" when they are printed. Numbers can be formatted, breakpoints and totals can be formatted and placed on the report, duplicate field entries in a column can be suppressed, record counts can be displayed, and the report can be enclosed with lines (to separate the heading and footer). You can print the report to a disk file, the printer, or the screen.

IMPORT/EXPORT. The Smart Database Manager can read the following file formats: ASCII delimited and fixed-length field formats (properly formatted), data sent from other Smart modules, and dBase II and III. The Database Manager can write ASCII delimited and fixed-length, DIF, SYLK, dBase III, and Smart formats.

Time Management

The Smart Time Manager, while it does not provide any group functions such as scheduling meetings (offering strictly a personal calendar for the user), is very well implemented. It is easy to understand and use, and it displays calendar information very clearly.

USING THE TIME MANAGER. The default calendar is the daily one, and, when you enter the Time Manager, today's calendar is displayed. It is quite simple to view different days with the cursor keys or with the menu system. You can also display a weekly or a monthly calendar.

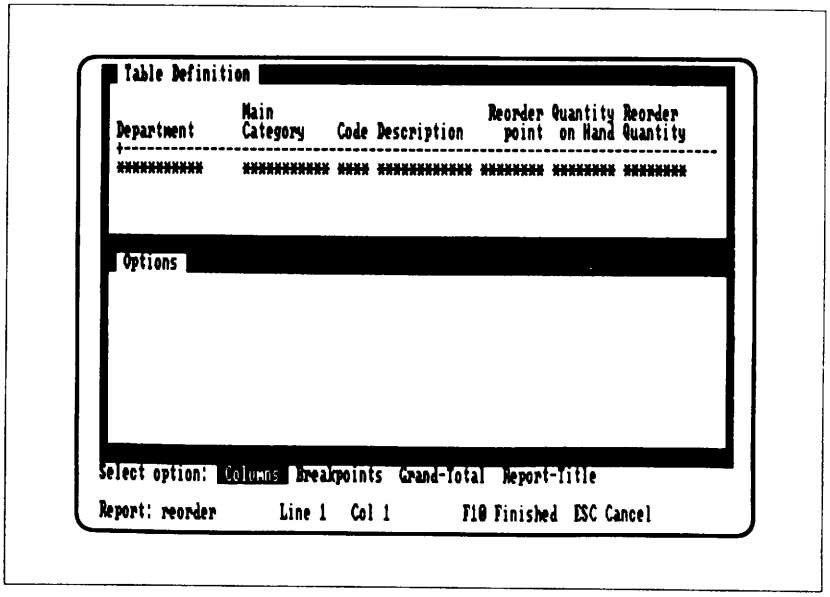
The Time Manager has only one menu. The primary calendar functions are: Insert a new item (you choose whether it is a meeting or a task), edit an existing item, delete an item, change the calendar display (day/week/month), find a particular day, and output your calendar in hard copy.

Entering a new item is very easy. You merely move to the appropriate day, choose "insert," then choose "meeting" or "task." At this point, Smart asks you to fill in the day, time (for a meeting), or priority (for a task), a short description that will display on the calendar next to the time/priority, and a long description that will display when that item is highlighted on the calendar. Smart displays that day's date as the default, but you can change this to schedule something on a different day.

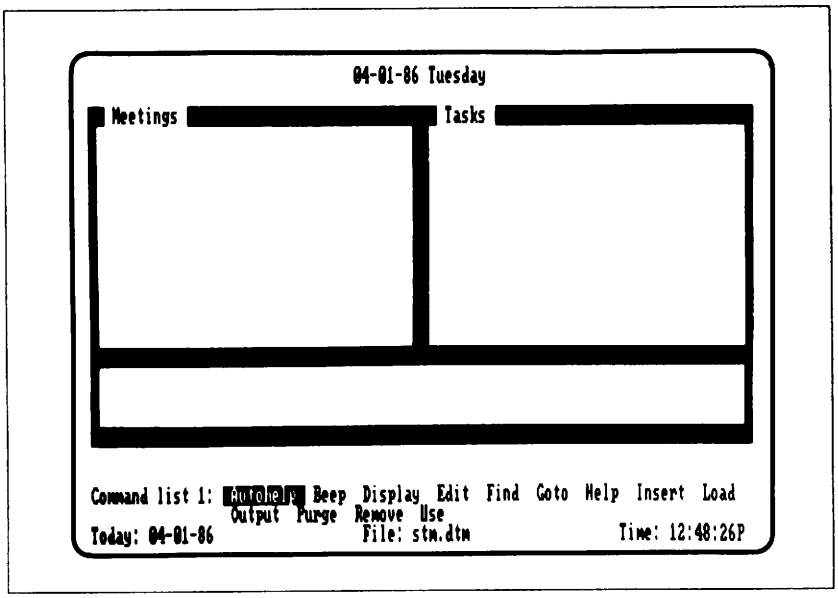
You can edit or delete an item simply by highlighting the entry with an arrow key and choosing the menu option. Another nice feature is the "use" command. This allows you to use an existing meeting or task as a template for creating another one. Smart fills in the fields for you and you edit them appropriately. This would be helpful for setting up recurring activities, since Smart has no built-in function for this.

Output. You can print out a copy of your calendar in two different formats. The short format lists tasks first and then meetings for the day, showing the short description only. The long format also inserts any long description after the item.

SOME LOOSE ENDS. One thing we don't like is the fact that, when you enter the text of a long description (which can be up to 255 characters), it doesn't wrap within the entry window. You must hit the return key at the end of the line (signaled by a beep) and continue on the next line. In addition, when you delete calendar items, Smart doesn't ask if you are sure you want to do



This screen shows the layout of a table in a custom report format.



The daily calendar format. The display consists of three components: a list of meetings or appointments in order by time scheduled next to a list of tasks in order by priority, and a "notes" section across the bottom. The notes section provides a long description of whatever meeting or task is highlighted. As the highlight moves to another item, the description changes accordingly.

this (or offer the option to turn on such a feature). You select "remove," and it's gone.

Smart asks you only for a starting time for meetings, not an ending time. Even if you wanted to fill in the time as "1:00-3:30," you can't; it won't accept more than one time in the field. (When we tried to do this, Smart beeped at us and wouldn't let us proceed, but did not display any type of error message to explain the problem.) We would like to see the addition of an ending time or duration field to give a more complete calendar for the user. This would also enable Smart to provide more effective conflict notification, since it cannot now discern two meetings that overlap but do not start at the same time. Smart does give you a message if you schedule a meeting at exactly the same time as an existing one, or a task with the same priority as one on the list, but it will go ahead and put it on the calendar anyway. It does not facilitate conflict resolution.

MISSING PIECES. As we have mentioned, the Time Manager is for personal use only. The ability to access other calendars and to schedule meetings across a workgroup would be a welcome addition. We would also like to see an active reminder function and the ability to more easily schedule recurring appointments or meetings.

Project Processing

One of the most powerful features in Smart is project processing, the programming language that underlies all of the modules. For example, you could set up a "project" that, when executed, goes into a database of sales numbers, summarizes a set of data, sends the summarized data to a spreadsheet, creates a graph, and then pops the graph into a word processing document along with a table of data.

Projects can be created with the "remember" command, which captures the user's keystrokes and stores them in a project file. The user can also create projects with the programming language. In fact, a script generated with the remember command can be modified with the programming language if appropriate. When the project is executed, the keystrokes are played back automatically. Projects can be edited, and the language is clear enough for the user to read and understand what the project is designed to do. Projects can also be mapped to keys as keyboard macros.

USING PROJECTS. Projects can contain two types of commands: module commands, which are accessible within the Smart modules, and project commands, which cannot be selected from the module command menus. The "remember" process can only be used to pick up module commands. Project commands must be added later by editing the project file. They include:

- The ability to prompt the user for input at run-time. This input is stored in variables. You can also create custom input screens for entering variable information, displaying menus and prompts.
- If/then/else logic.
- The ability to go to a specific location in the program depending on the results of an "if" statement.
- The ability to create a custom menu.
- The ability to define and call subroutines.

When editing a project, the user is put into a separate project editor. We would prefer the ability to use the Smart Word Processor for project files as well as for documents. This would provide greater functionality and eliminate the need to learn another set of editing commands.

The project language is similar to a fourth-generation programming language (4GL), requiring the user to understand both the commands and how to appropriately structure a program. Therefore, it is aimed at the professional developer or the experienced user who is willing to spend the time and effort to learn a programming language. The real benefit of Smart project processing is its ability to create cross-application procedures; it is not restricted to a single Smart module. Thus, Smart comes closer than many products to achieving a procedural automation facility. However, two major pieces are still missing: a multiuser (workgroup) connection for projects, and access to an electronic mail facility.

Because of the project processing capability, many users and professional developers have put together customized applications within Smart, enough so that the company publishes an applications manual listing all of these. In fact, Informix Software has implemented its own lead tracking system and its support call logging system in Smart. Informix does not, however, sell a run-time-only version of Smart.

Smart Management Utilities

THE SMART MANAGER. The Smart Database module lets the user/system manager define peripherals for Smart using a menu system rather than the Unix shell. This is necessary in order to use communications (defining the serial port for the modem), a printer, plotter, etc.

COMMUNICATIONS. The Communications module provides commands to establish a connection between two computers to transfer files (both send and receive). The macro and project processing facilities can be used in Communications as well as in other Smart modules to automate procedures.

Summary

We were impressed with several aspects of Smart:

- The consistency maintained for the user across the modules
- The functionality within each module
- The integration among the modules
- The ability to define programs (projects) that can automate procedures across applications

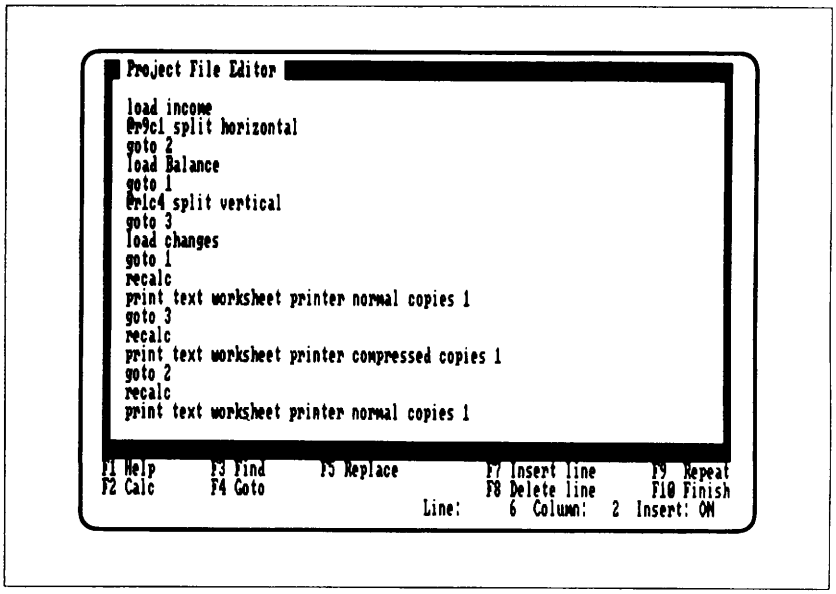
In addition, Smart is generally very easy to use, and we like the fact that the user can tailor the interface and functionality to his or her own experience level.

We have noted those areas where we would like to see improvements in Smart. These include more extensive online help and reference material, the ability to restrict access to files from within Smart (beyond just assigning a password), additional word processing and calendar features, some streamlining in the spreadsheet, and simplified query and relational functions in the database manager.

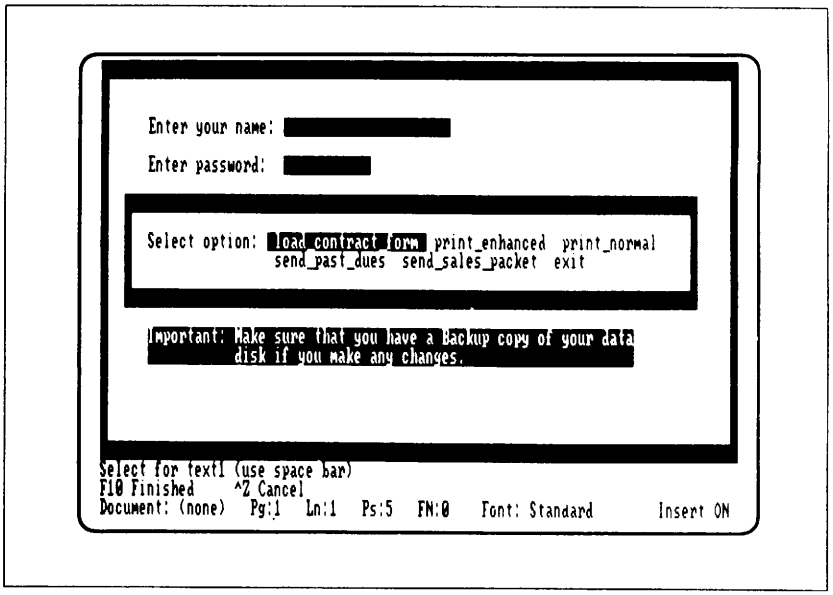
The addition of the following enhancements will improve the overall product's appeal and ability to compete across the board with other integrated software platforms:

- Workgroup-oriented applications such as electronic mail and meeting scheduling
- A more sophisticated windows-based user interface, including multitasking, overlapping windows
- A compound document editor
- Access to an SQL-based RDBMS
- More extensive networking support

The company states that all of this is coming in the future.



This illustrates the project file editor and a simple project file definition.



In the project processing facility, you can design custom input screens. This screen includes input fields, a menu, and a text message.

Conclusions

Smart is, to some extent, caught between a rock and a hard place. Because it lacks workgroup-level functions such as electronic mail and group scheduling, it cannot compete with fully functional office automation products like Uniplex, Q-Office, Alis, or CCI's OfficePower in the Unix environment. That leaves Smart to compete with standalone products—Lotus 1-2-3 for spreadsheets, WordPerfect and Microsoft Word for word processing, and dBase, Paradox, et al. for database management. Not an easy task. These software products have each attracted an extensive following of devoted enthusiasts, and we all know how difficult it is to get users to switch software products.

Smart does offer advantages over standalone products in its integration between modules, but the user must first be persuaded that the individual modules are worth switching for. Our recommendation is that Informix Software add the group-level functionality as fast as possible, moving Smart to another competitive arena.

Ideally, users would like to choose whatever software they like, and then get the integration on top of that. This is another potential avenue for Informix Software, but not one that we will expect to see soon. Smart does provide gateways to non-Smart software, with its ability to read and write various file formats.

Informix Software is actively working on integrating its products and on broadening the scope of Smart as an office automation product. In the near future, the company plans to communicate its plans more clearly. The combination of strong OA tools, a proven RDBMS, and the ability to run across multiple operating environments could give Informix a leg up on all of its competitors. Informix also has obtained, with Innovative's Wingz, valuable expertise in designing a graphical user interface for the Macintosh. Currently, not many other vendors command a set of software resources that can match the combination of the Informix RDBMS family and the Smart Software System. It remains to be seen if Informix can make the most of this advantage. At the very least, any level of integration and enhancement will be a real plus for the Unix environment.

•TUTORIAL•

Introduction to the Shell

The role of the CLI, or shell, is to interpret your commands so that the system understands.

By Gary J. Nutt

Every interactive computer system provides a command line interpreter, or CLI, with which the user can "converse" with the operating system. The system employs the CLI to inform the user of relevant internal conditions, and the user gives commands to the operating system via the CLI. The commands are typically directives to the operating system, the most common being "run this program."

The shell is the generic name of the CLI in Unix, a powerful and flexible interface program. In this column, we will describe the structure of the shell so that you can begin to see how one can take advantage of its power. In the next few columns, we will explain how you can use the shell to customize your own Unix user interface, and to explain how the shell can be used to extend the command language of the system. First, consider some other interfaces for controlling the computer.

The DOS CLI is initiated whenever the system is started (the CLI is called the command.com file in DOS). Users know that they can send commands to the system when the CLI prints a prompt on the screen. In DOS, every command is stored in some file. Thus, to execute the command, you must direct the CLI to find the file, load it, and execute it. A shorthand method for doing this is to simply type the name of the file containing the desired command; the CLI responds by finding the file, loading it, and executing it.

Graphical, mouse-oriented interfaces, such as the Macintosh and MS Windows, also have a "command line interpreter," although it is not designed to operate with a keyboard interface—so it is really a point-and-select command interpreter. The same paradigm is followed in the graphical systems as in the DOS CLI, except that the syntax to cause the command interpreter to "find, load, and execute" a file is changed to a series of mouse-clicks and/or menu selections. The end result is the same.

The main purpose of the shell is to provide a mechanism by which the user can direct the operating system to find, load, and execute a file. In addition, the shell environment also provides a rich programming environment for customizing the user's interface to the machine and for programming groups of operating systems commands; these features are more general than the auto.bat files found in DOS. While the shell is a simple CLI on the surface, it includes a number of design features that make it an elegant tool for controlling the system.

The shell is an application program that the operating system automatically invokes whenever a user logs into the system. Thus, you can arrange to run any CLI whenever you log in (by having the Unix system administrator edit the /etc/passwd file so that it specifies the particular CLI to be used—see the March 1987 tutorial). In System V Unix, the default shell is called the Bourne shell, since it was created by Stephen Bourne when he worked for AT&T. The BSD Unix default shell is called the C shell. Either shell is typically available on either version of Unix. The C shell is a refinement of the Bourne shell, basically intended to provide additional programming facilities and an audit mechanism for any login session. Our discussion applies to either shell.

The shell begins execution by initializing the environment, executing a file from the user's home directory called the .profile file, then indicating that it is ready to receive directives from the user by typing a prompt character on the screen. The prompt character is customizable; the default Bourne shell prompt is a "$" character, and the default C shell prompt is "%".

The user types a command line of the form:

command <optional arguments separated by blanks>

terminated by a carriage return (or Enter) character. The shell interprets "command" as a filename and assumes that the remainder of the characters on the command line are parameters meaningful to

the program that will interpret the command (and are of no meaning to the shell itself). The shell now needs to find, load, and execute the file that contains the program associated with the command.

The command is usually a Unix pathname for a file. The pathname may be absolute, i.e., specified as a path through various directories starting from the system root directory; or it may be a relative pathname, i.e., a specification of the file's location relative to the current directory where the user is executing the command. (There are also default directories that the shell will search for relative pathname matches, specified by a command in the .profile; this will be discussed in a later column.) If the shell is unable to locate the file from the pathname, then it will inform that user that the file cannot be found.

There is a subtlety in the way that the shell executes commands: It does not call the command as a procedure. Instead, it creates a new process and causes it to execute the command.

Since a command is simply a file, it is easy to extend the set of commands that the shell recognizes: Simply write a program, compile it, and store it in a file. Whenever a user enters the file name as the first string on a command line, the shell will find the file, load it, and execute it.

The shell will create a new process to execute the file containing the command. When it causes the new process to execute the file, it will pass the optional parameters to the new process. There is a standard format for passing parameters to the command program, where flag parameters are prefaced by a "-" symbol and additional filenames simply appear as strings. Each command expects a different set of parameters and filenames; thus, the form of the parameters is not standard from one command to the next.

For example, "nroff" is a command that reads a file containing text characters and formatting commands and produces a file containing formatted output for printing (with hyphenation, fonts, etc.). The nroff command has several options, including various special packages for processing the document. The command

```
nroff -me ftext
```

will cause nroff to be executed with the "me" parameter set on the input file named ftext.

The shell relies on all command programs acting as filters. That is, they read all required information from a standard input file—the keyboard is treated like a file in Unix—and write all normal output onto a standard output file—normally, the screen. These files are named stdin and stdout, respectively. (Error output is written to a different output file, called stderr, which is also displayed on the screen.) A filter reads input from stdin, processes it, and then writes the output to stdout. If we had entered our nroff command line as

```
nroff -me
```

then nroff would have expected the file containing text and formatting characters to come from the keyboard; the result would be printed to the screen in either case.

The user may wish to explicitly define the file names for stdin and stdout for a particular execution of a command. This is done by preceding the file that is to be treated as stdin by a "<" character, and the file that is to be treated as stdout by a ">" character. Thus, if we wished to use nroff to read an unformatted file, ftext, and write the formatted result to another file named result, we could use the command line

```
nroff -me <ftext >result
```

or, since nroff will treat a named file as stdin, we could simply write

```
nroff -me ftest >result
```

There is a subtlety in the way that the shell executes commands: It does not call the command as a procedure. Instead, it creates a new process and causes that independent process to execute the command. While the command is being executed, the shell executes the system call "wait" to block until the child process completes the command execution. This strategy, while somewhat complex, allows the user to direct the shell to run two or more commands simultaneously. If the user terminates the command line with the "&" character, it is an indicator to the shell to create a new process and to have it execute the file, but not to wait for the execution to complete! Instead, the shell immediately prompts the user for another command. The user can invoke another command while the first is still in execution. The bad news is that because both of the commands that are in exe-

cution have the same default stdout file, they will both be enabled to write output to the screen (unless one of the commands has redirected stdout to another file), potentially causing interleaved output to appear—output from one of the commands, then output from the other.

Since nroff is typically a slow, time-consuming command to execute, we may wish to execute it "in the background" while we go about other business. This can be accomplished by the command line

```
nroff -me ftest >result&
```

The shell also allows the user to start two commands in execution in parallel, with the output from one going to the input of the other. Conceptually, the output from the first command is being "pipelined" into the second command. Thus, the shell construct that provides this ability is called a pipe.

If the user types a command line of the form

```
command <args> | command <args>
```

then the shell creates two new processes, the first executing the first command, and the second executing the second command. The stdout file on the first command is made to be the same as the stdin on the second command, so that, as the first creates some output, it is read by the second.

In our nroff example, the output is likely to be many screens full of information; the information will be written to the screen (if we have not redirected the output to a file) as rapidly as nroff produces finished text. Some versions of Unix have a command called more; the purpose of the "more" command is to write one screen full of information to the screen, then to wait until the user reads the information and explicitly tells the command to put the next screenful of information on the screen. Thus, more allows us to browse through a file at our leisure. The two commands can be composed using the pipe:

```
nroff -me ftext | more
```

The nroff command may "run ahead" of more (the amount is limited by the operating system's constraints on the amount of data that can reside in a pipe), but the user can browse through the output at his own pace.

Now we have the basics of the operation of the shell. This facility can be used to customize a user's environment by providing specialized commands; by constructing the special .profile file with a number of commands to be executed each time the user logs into the machine, and by writing files of shell commands and then having the shell execute these files. In the next two tutorials, we will address the issues of environment customization—setting up the .profile file—and of combining shell commands into larger, more complex commands.

A N N O U N C I N G

The Sixth Annual Seybold Executive Forum:

TEAMWORK, TECHNOLOGY, & ORGANIZATIONAL PERFORMANCE

October 25 – 27, 1988

Royal Sonesta Hotel,

Cambridge, MA

The Sixth Annual Seybold Executive Forum will highlight the need for teamwork between business executives and technical executives in deploying technology to boost productivity. We will focus on the urgency of evolving our organizations in cooperation with technology. As technology has evolved, organizations have had to evolve to meet the challenge. Now, technology is no longer a cause for change, but rather a tool to facilitate change.

So join us as we ...

- ❍ Examine the impact of real-time systems
- ❍ Explore the relationship between technology and organizational performance
- ❍ Experience the catalyzing role technology can play
- ❍ Demonstrate how technology can be used to support teamwork
- ❍ Learn basic principles, practices, and techniques that sustain group cooperation and synergy

☑ **PLEASE REGISTER ME!**

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

☐ 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: _____

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

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. 8 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.

THE SIXTH ANNUAL SEYBOLD EXECUTIVE FORUM

TEAMWORK, TECHNOLOGY, & ORGANIZATIONAL PERFORMANCE

Royal Sonesta Hotel, Cambridge, MA

Oct. 25, 1988 — STALKING THE PRODUCTIVITY PAYOFF

Vendor Panel:
- Conversations with Top Executives
  IBM, Digital Equipment Corp., Hewlett-Packard, Apple

Noted Authorities:
- Teaming Up for Global Competiveness
- Aligning Informative Systems & Business Strategies
- Creating the Corporate Future

Hands-On:
- Workshops in Teamwork & Technology
  Action Technologies, Digital Equipment Corp., Insight Consulting Group

Oct. 26, 1988 — BUILDING "REAL-TIME" INFORMATION SYSTEMS

Customer Case Studies:
- Capturing Information & Modeling the Environment
- Executive Information Systems: ***Taking the Pulse of the Organization***

Noted Authorities:
- Real-Time Organizations: ***Dream or Reality?***
- Can We Get There From Here? ***Information Systems as Change Agent***

Vendor Panels:
- Object Orientation: ***Building Block Approach to Application Design***
  Metaphor, IBM, Microsoft, Wang, Data General
- Partnering for Progress: ***Industry-Specific Solutions***
  Tandem, Unisys, NCR, Prime

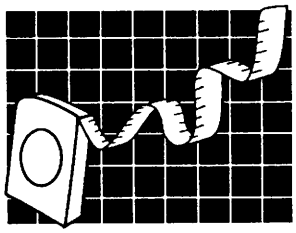
Standards Organizations:
- Teaming Up with Standards
  X/Open, Open Software Foundation

Hands-On:
- Workshops in Teamwork & Technology
  Action Technologies, Digital Equipment Corp., Insight Consulting Group

Oct. 27, 1988 — PAYOFFS FROM TEAMWORK

Panel Discussions:
- Directions in CSCW
  Lotus, Coordination Technologies, MIT, Xerox Collab
- Stategic Applications of CSCW
  User Organizations
- The Payoffs from EDI
  User Organizations
- Building the Communications Infrastructure: ***Planning for ISDN***
  NYNEX, Northern Telecom, Nippon T & T, Siemens
- Platforms for the Future
  Sun Microsystems, Next Computer, On Technology

•STANDARDS•



Standards, Unite!

You Have Nothing To Lose But Your Incompatibility

By Judith S. Hurwitz

The formation of the Open Software Foundation (OSF) is symptomatic of the need for an independent computing environment. Indeed, the Portable Operating System Interface for Computer Environments (Posix) Committee has been the one area where all warring factions have been willing to unite. Therefore, it is not surprising that the scope of the Posix effort would expand. One of the major players leading the charge is the U.S. National Bureau of Standards (NBS).

P1003.0: Defining the Total Environment

Initially, the founders of Posix intended to create a series of interfaces that would sit between applications and the Unix operating system (see Vol. 3, No. 1). Areas such as user interface, graphics, communications, windowing, database access, and programming languages were originally outside the scope of the Posix Committee's focus. However, this has changed with the formation of the P1003.0 Committee. The reason for the creation of the new Committee is, in large part, related to the need to define a complete standards-based computing environment.

The NBS has been impatient with the progress of the standards process. It desperately needed clearly defined criteria for federal Requests for Proposals (RFPs). Not willing to wait, it proposed the Application Portability Profile (APP) as an interim Federal Information Processing Standard (FIPS) for Posix. While, in the long run, the NBS intends to use the completed version of Posix, it cannot afford to wait until the standard is finally set. (Standards do not get completed quickly.) The P1003.0 Committee has decided to use the APP as the basis for its forthcoming standard.

APPLICATION PORTABILITY PROFILE. Since the APP will be critical for future government RFPs and will be the basis of an IEEE standard, it is interesting to look at the initial specification.

The interim standard being developed by the P1003.0 Committee will be divided into two sections. The first includes information about applicability, implementation, and maintenance of the standard, and an appendix that provides an initial outline for the components of an API. This profile will be developed jointly by computer vendors and users. The first part of the Posix effort, the P1003.1, defines the C language source interface to an operating environment. It has already been incorporated into the APP and is the first of the interface standards being developed by the IEEE. (The NBS indicated in March that it would use the 1003.1 Posix Draft 12 as a basis for RFPs.) The proposed APP is intended as the first step towards a systems design standard, an index of all the components needed to ensure portability of software across computer hardware environments.

Role of Federal Information Processing Standard

The Applications Portability Profile is a linchpin in the Federal Information Processing Standard (FIPS). The objectives of the FIPS is to provide the government with better control over the production, management, and use of governmental information resources. The objectives include:

- To promote applications portability at the source code level
- To simplify computer program documentation by using a standard portable system interface design
- To reduce staff hours in porting computer programs to different vendor systems and architectures
- To reduce personnel training costs
- To maximize the return on investment of computers by ensuring operating system compatibility

One of the biggest concerns of the federal government is that, even when all the phases of Posix are completed, the standard will still not include everything necessary to ensure portability for all applications. NBS wants to see Posix expanded to include four additional areas not currently part of the standardization effort:

- Shell and Tools, functions such as listing the files in a directory, which provide an interactive interface for users to control processing.

- Advanced Utilities, which provide additional capabilities and specialized functions, such as full-screen editing, that make users and programmers more productive.

- System Administration functions such as mounting a file system, which are required to operate the system.

- Terminal Interface Extensions, which are called by applications programs. These functions enable programs to perform interactive terminal operations in a way that is independent of the types of terminals being used. An example is the ability to turn on attributes such as blinking characters or reverse video.

Beyond Posix

NBS perceives that, even with these additions, Posix will not be sufficient to incorporate all the functionality required to achieve portability of all applications. For example, language portability is a particular problem. It perceives that applications software portability will depend on additional factors, including:

- Characteristics of the underlying hardware and software, such as word length, input/output (I/O) architecture, processor, and operating system

- Portability of software utilities used by the application, such as database management, graphics, operating system functions, and communications

- Data form, format, and representation that may need to be transported with the software

- Language implementation, such as compiler, interpreter, and processor, including specific limits or subsets of the language used in programming

The APP is intended to provide sufficient functionality to accommodate a broad range of application requirements. The functional components of the APP constitute a "toolbox" of standard elements that can be used to develop and maintain portable applications. Not surprisingly, open systems architecture based on a nonproprietary standard is key.

DATABASE MANAGEMENT. A database management component is a fundamental aspect of applications portability. Therefore, SQL and a standard information resource dictionary system are two pieces of the database management requirement. In addition, the data itself has to be portable.

DATA INTERCHANGE. Within the APP, data is defined as being in three formats: business graphics, product data, and document processing. To ensure that these data types are compatible, data interchange is critical. Therefore, for graphics requirements, APP will initially include Graphical Kernel System (GKS) and Computer Graphics Metafile (CGM). To exchange product information, the International Graphics Exchange Specification (IGES) has been adopted. To allow for data interchange, standards including Standard Generalized Markup Language (SGML) and Office Document Architecture/Office Document Interchange Format (ODA/ODIF) are the first to be required.

NETWORK SERVICES. While the NBS recognizes that file management is a requirement, its focus has shifted to the need to share and access data across a heterogeneous environment of hardware. Therefore, it has selected Sun Microsystems' Network File System (NFS) as its initial component to meet the need for file management. However, this is just the first step. The next step will be compliance with the NBS's Government Open Systems Interconnection Profile (commonly known as GOSIP). GOSIP will include the tested OSI protocols and transition components that conform to the OSI proto-

Applications Portability Profile

Function	Element	Specification
Operating system	Posix	IEEE P1003.1
Database Management	SQL IRDS	IEEE P1003.2 FIPS 127 X.3.138 (proposed FIPS)
Data Interchange:		
Business Graphics	GKS & CGM	FIPS 120, 128
Product Data	IGES	NBSIR 86-3359
Document Processing	SGML ODA/ODIF	ISO 8879-1986 ISO/OIS 8813
Network Services:		
Data Communications	OSI	GOSIP
File Management	NFS	IEEE P1003.X
User Interface	X-Window System	X3H3.6
Languages	C	X3J11 draft X3.159
	Cobol	FIPS 021-2
	Fortran	FIPS 069-1
	Ada	FIPS 119
	Pascal	FIPS 109

cols. The government intends to migrate from the de facto standard Transmission Control Protocol/Internet Protocol (TCP/IP) to OSI.

USER INTERFACE. NBS is looking to use MIT's X-Window as its initial user interface standard.

PROGRAMMING LANGUAGES. The initial programming language binding that will be incorporated into APP will be C. Additional bindings will be developed for Fortran, Cobol, Ada, and Pascal.

The Committee Process: P1003.1

With such a tall order, the NBS could not possibly accomplish its goals alone. Therefore, it has sought cooperation from organizations with similar goals, including the IEEE Posix committees, X/Open, and the newly formed Open Software Foundation (OSF).

According to P1003.0 co-chair Kevin Lewis of Digital Equipment, "We accomplished a lot more at the first meeting than we thought we would. We even got to the point of having a proposal that was put on the table." He explains that the goal of the 1003.0 committee is to develop a "guide that shows the user what the relationship is between Posix and other standards." As a result, the committee hopes to be able to define "the open systems environment."

How do the goals of P1003.0 differ from X/Open and OSF? Lewis agrees that there is similarity. "X/Open uses de facto standards and proprietary interfaces and puts these into their Portability Guide [see Vol. 3, No. 1]. They take a look at what exists and make sure that it can be used accurately." In fact, the goal of the P1003.0 committee is to have its forthcoming guide completely in synch with the X/Open Portability Guide. Lewis views OSF, on the other hand, as a body that intends to actually implement standards that conform to Posix. A key differentiation, in his view, is that OSF will implement extensions. In addition to cooperation with these two bodies, the committee has a strong working relationship with AT&T.

Timetable

The National Bureau of Standards has an ambitious timetable for approval of APP. It intends to have a majority of its standards completed by the fourth quarter of 1988. Included in this time frame are Shells and Tools, Advanced Utilities, System Administration, and Terminal Interface Extensions. By the first quarter of 1989, its standards for X-Window and NFS should be approved.

Conclusion

It makes sense for the heaviest user of computer systems, the U.S. government, to take the lead in moving towards a full-function standards environment. The newly created P1003.0 Committee is an excellent and sorely needed group. It should provide an excellent foundation upon which vendors, users, and consortia can unite.

NEWS ANALYSIS

PRODUCTS • TRENDS • ISSUES • ANALYSIS

• INSIDE •

• OPEN SOFTWARE FOUNDATION •

Interactive Systems

In the wake of the inauguration of the Open Software Foundation (OSF), Interactive Systems is establishing the Open Systems Technology Center. In fact, the center is a direct result of OSF's formation. Interactive intends to use its newly-formed center to provide support services for system vendors as well as users "who are working to comprehend the ramifications of the recent launch of OSF." In essence, Interactive hopes to capitalize on both its knowledge of System V (via its work with AT&T on the merged Xenix/Unix for the Intel 386) and its initial work for IBM on the AIX operating system. In addition, Interactive becomes one of the earliest vendors to become a member of OSF.

Interactive, which was recently purchased by Eastman Kodak Company, hopes that by joining the new consortium and opening a technology center, it will propel itself into a position of prominence. To accomplish this, Interactive intends to write compatibility suites so that users with an installed base of System V applications can maintain applications written in both operating systems, if they also install systems based on the OSF operating system. Other products that Interactive is considering include libraries and other software facilities that it would license to users and software developers.

How does Interactive view recent events? According to company president Ronald Fisher, "both operating systems will end up with equivalent functionality. The key will be how the functionality is implemented and how this implementation will affect porting." Fisher admits that how easy porting will be will remain a mystery until vendors and users have an opportunity to view the new AIX code. However, he is optimistic. "Given the nature of Unix, portability should not be a significant issue."

The Technology Center, which will begin operations July 1, will be based at Interactive's Boston development facility. Initial services that will be provided by the center include:

- Assessments of AIX, Unix V.3, and X/Open systems (including their interrelationships)
- Seminars on each system, including updates on new release specifications
- Compatibility-planning services
- Development and licensing of Unix V.3 and OSF compatibility modules
- Detailed design studies for system implementations
- Porting services

CONCLUSION. Interactive has taken advantage of an obvious need at a time of great confusion. If it can establish a technology center that helps software vendors and users understand how to implement technology both for System V and for the forthcoming OSF operating system, it will serve a valuable function. If this is done well, Interactive could propel itself into the spotlight. If the company's undertaking is successful, it will also provide a valuable service to the Open Software Foundation.

However, Interactive will have to prove that it can provide a high level of service to an industry in transition. Although the company has accomplished much over the past 11 years, its relationships with some key vendors have been less than pristine. For example, IBM developers contend that very little of the current version of AIX was developed by Interactive. To succeed, Interactive will have to prove its mettle.

—J. Hurwitz

•RESEARCH & DEVELOPMENT•

Mach Wanders into the Spotlight

The Unix industry is curiously anticipating just what Steven Jobs's new venture, NeXT, will bring forth next month. One thing known for sure is that Jobs plans to implement a new operating system for his workstation. Mach is an operating system developed at Carnegie Mellon by Richard Rashid with funding from DARPA and IBM.

Why yet another operating system? On one hand, Mach is not entirely new since it is a derivative of Unix. On the other hand, the concept behind Mach's development was to come up with an operating system that would not be tied to the limitations of Unix. Unix is a ten-year-old operating system and therefore includes old technology. When Unix was designed, no one envisioned that it would be used to run complex distributed network computing systems. In this regard, Mach represented the clean slate. In essence, researchers were able to disregard history and begin anew. This is something that most commercial operating system developers only dream of. It is also interesting to note that DARPA also funded the work that went into the development at UCLA at Berkeley of the Unix BDS operating system. In addition, DARPA funded development of the de facto TCP/IP communications standard considered a precursor to the transport level of the ISO communications standard.

So, what's in Mach? Basically, it includes a kernel that is very distinct from existing Unix kernels. However, it does support Unix system calls (in theory, you could take a program written in System V and recompile and run in Mach). Like any research project, there are problems in translating from the theory to the reality. For example, currently Mach is slower than Unix. It also only runs on a few machines including the VAX and the RT.

When Rashid first started working on Mach his goal was "to target a multiprocessor environment." While Unix has a lot of facilities that are important for the database environment, it was written to operate on a uniprocessor. Vendors who have created multiprocessor versions of Unix have had to create complex extensions to deal with Unix's architecture. What Rashid has done is to create an operating system that includes the concept of multiple threads (Unix employs a single thread concept). The thread facility in Mach allows multiple threads to share all physical resources, including files and resource memory of a process.

Another strength of Mach is in the area of memory management. For example, it includes a sophisticated mechanism that allows large amounts of memory to be passed from one program to another. Also, Mach includes tools that allow programs to inherit portions of address space from child programs. While Unix allows for memory sharing, it tends not to be very dynamic. Sharing in BSD 4.3, for example, can only be accomplished by sharing a copy of a program. In System V, shared memory constructs are limited to sharing between two programs. In contrast, with Mach, sharing can be done at the page level that allows copying to take place at write time. In this way, rather than having to copy data into memory, information can be logically copied through the use of pointers. Simply put, this becomes virtual memory for multiprocessors. Only pages that are changed are copied. Likewise, data is transferred to another location only when it is specifically called. Rashid says that this helps to optimize the operating system. As a result of this memory management construction, a lot of memory and time are saved.

One of the most promising areas for Mach is in the on-line transaction database arena. Mach's architecture enables it to provide for transaction processing at competitive speeds, according to Rashid. This is possible because, unlike Unix, Mach was created to deal with the characteristics of hardware. In contrast, Rashid points out that with Unix, developers have had to add extensions on top of the base operating system to handle virtual networks, communications, and hardware. In his view, these have not related to each other very well.

CONCLUSION. Mach is interesting for a number of reasons. First, Carnegie-Mellon University intends to put Mach into the public domain. Therefore, we expect that many companies looking at the standards environment will take a look at this possible successor to Unix. Second, Rashid intends to make Mach Posix compliant in the long run. Third, the Open Software Foundation (OSF) intends to work closely with the University research community. Therefore, we expect that Mach will begin to become a strategic operating system in the future—either in its own right or as part of the new operating system based on IBM's AIX.

We believe that the industry should take a close look at Mach as one of the foundations for the future. ◐

—J. Hurwitz

•TOLERANT SYSTEMS•

Tackling On-Line Transaction Processing

Unix has never been strong in its handling of on-line transaction processing (OLTP). Standard Unix systems lack the essentials needed for OLTP and related applications, namely: efficient handling of large, structured data files; transaction data integrity and concurrency control; and speed. To handle OLTP, vendors such as Tandem and Sybase have had to make changes to the operating system. Another company making inroads into OLTP on Unix is Tolerant.

Tolerant Systems Incorporated, a six-year-old company based in San Jose, California, has been addressing these shortcomings. The company has modified the Unix kernel and developed a new machine architecture for its Eternity Series, a line of medium-to-large-scale Unix systems.

Since standard Unix file systems hamper Unix from handling large data files, Tolerant redesigned it. Tolerant's file system allocates disk space in large, mainframe-style blocks called "extents" and distributes files across multiple disks. The company also added concurrency control via integral file and data block locking, with automatic deadlock detection and resolution.

Like its competitors, Tolerant protects data integrity, enabling application developers to use system calls to signify the precise begin, commit, and abort points for each application. Copies of rewritten data records are logged for each transaction just before rewriting. Should the hardware fail before you complete a transaction (i.e., before it reaches the commit point), the log can be used to undo all rewrites and return the record to its previous state. So data integrity stays intact on a transaction-by-transaction basis.

Tolerant's approach to fault tolerance is software-based. It has developed a fault-tolerant system by transparently distributing transaction control across two or more CPUs. Consequently, transactions are not only "rolled back," but they can also be automatically restarted for continuous processing.

The Eternity Series architecture distributes Tolerant's version of Unix—Transaction Executive (TX)—among functionally-dedicated VLSI microprocessors in each CPU, and across loosely-coupled CPUs in a single system. Thus, the system balances heavy volumes of applications better. The loosely-coupled system resists bottlenecks: Instead of waiting in line, each transaction is processed by the next available resource.

We welcome Tolerant's progress. If Unix is to evolve as a worldwide standard, it can't be stumbling over OLTP or, for that matter, any significant commercial application. ◐

—L. Brown

Patricia Seybold's Computer Industry Reports

ORDER FORM

Please start my subscription to:

		U.S.A.	Canada	Foreign
☐ *Patricia Seybold's Office Computing Report*	12 issues per year	$385	$397	$409
☐ *Patricia Seybold's UNIX in the Office*	12 issues per year	$495	$507	$519
☐ *Patricia Seybold's Network Monitor*	12 issues per year	$595	$607	$619

Please send me a sample issue of:

☐ *Office Computing Report* ☐ *UNIX in the Office* ☐ *Network Monitor*

☐ **My check for $_____ is enclosed.** ☐ **Please bill me.** ☐ **Please charge my subscription to: Mastercard/Visa/American Express** (circle one)

Name: _____ Title: _____

Company Name: _____ Dept.: _____

Address: _____

City, State, Zip code: _____

Country: _____ Bus. Tel. No.: _____

Card #: _____

Exp. Date: _____

Signature: _____

Checks from Canada and elsewhere outside the United States should be made payable in U.S. dollars. You may transfer funds directly to our bank: Shawmut Bank of Boston, State Street Branch, Boston, MA 02109, into the account of Patricia Seybold's Office Computing Group, account number 093-118-7. Please be sure to identify the name of the subscriber and nature of the order if funds are transferred bank-to-bank.

IU-0688

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

Topics covered in Patricia Seybold's Computer Industry Reports in 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