GRAPHICS AND THE UNIX SYSTEM

UNIX™/WORLD

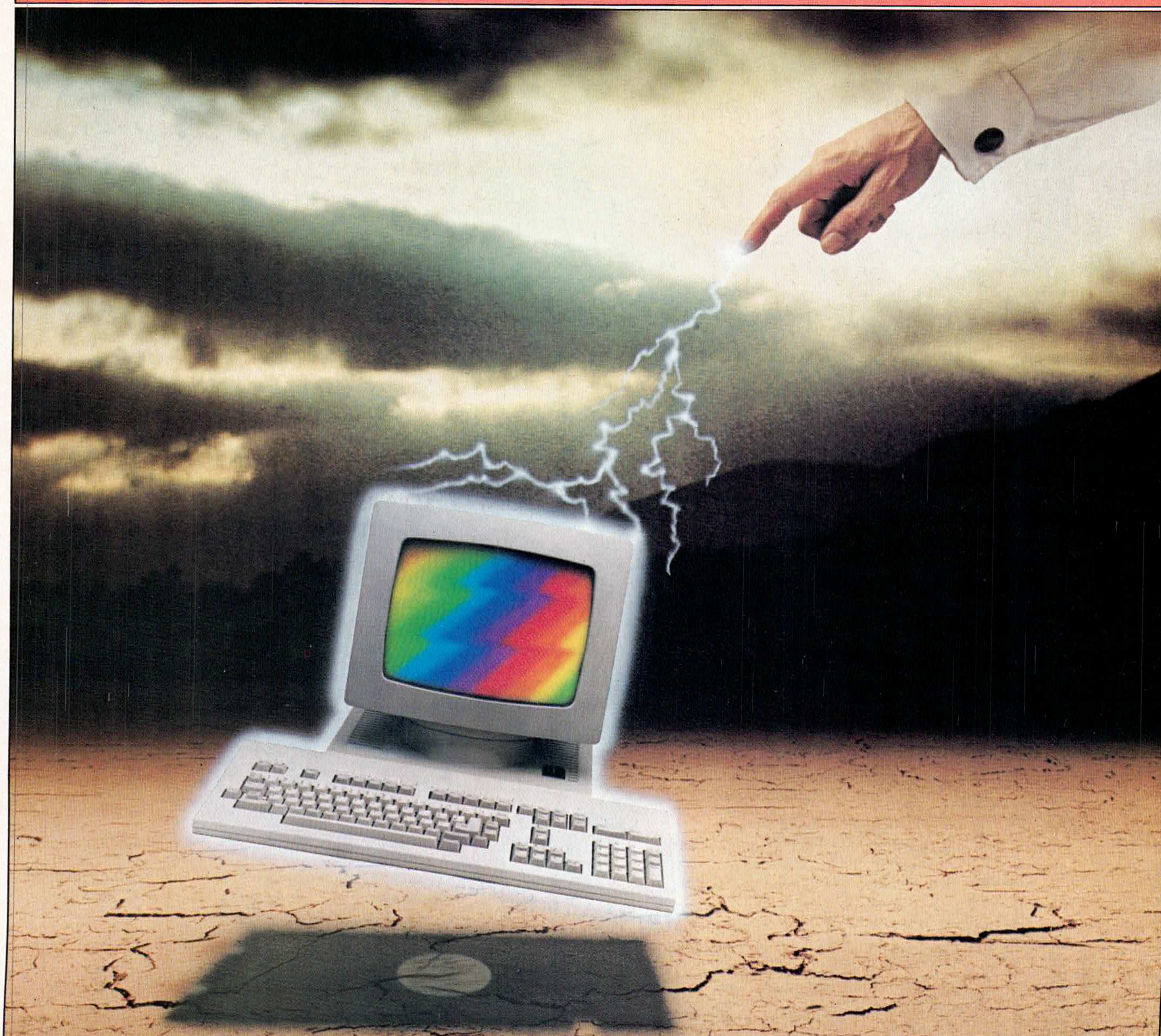
Your Complete Guide to the Frontiers of the Unix System

VOL. I, NO. 5 $3 IN USA A TECH VALLEY PUBLICATION

REVIEWS: ULTRIX 32, SUN, INTERLEAF

REALWORLD ACCOUNTING SOFTWARE REVIEWED

JOURNAL: I/O OPTIMIZING FOR MICROS



BEAT OUR BENCHMARK.

GET THIS WATCH.

Take the Plexus Challenge.

Everybody in the UNIX-based supermicro business talks performance.

Only Plexus dares to prove it...with the Plexus Challenge:

Show us a UNIX-based supermicro that can beat Plexus—running any recognized benchmark (or even your own application code*)—and we'll present you with this elegant Heuer chronograph valued at $485.00.

How do we dare make this offer? Simple: our multiprocessor architecture and highly-tuned version of UNIX leave the single-CPU competition in the dust. Multibus architecture and intelligent I/O processors let you expand modularly from 5 to 40 users—without changing a single byte of object code.

Are you thinking "Fine, but I really don't need all that performance"? Well, some of our customers tried other hardware first. They used to think the same thing.

Challenge details: This offer is extended to bona fide OEM's, value-added resellers, and volume end-users in the multiuser commercial UNIX systems market. Limit one award per company. Offer expires Jan. 31, 1985 unless withdrawn earlier. For complete rules and participation information, call (800) 556-1234 Ext. 560 (in Calif. (800) 441-2345 Ext. 560)

PLEXUS

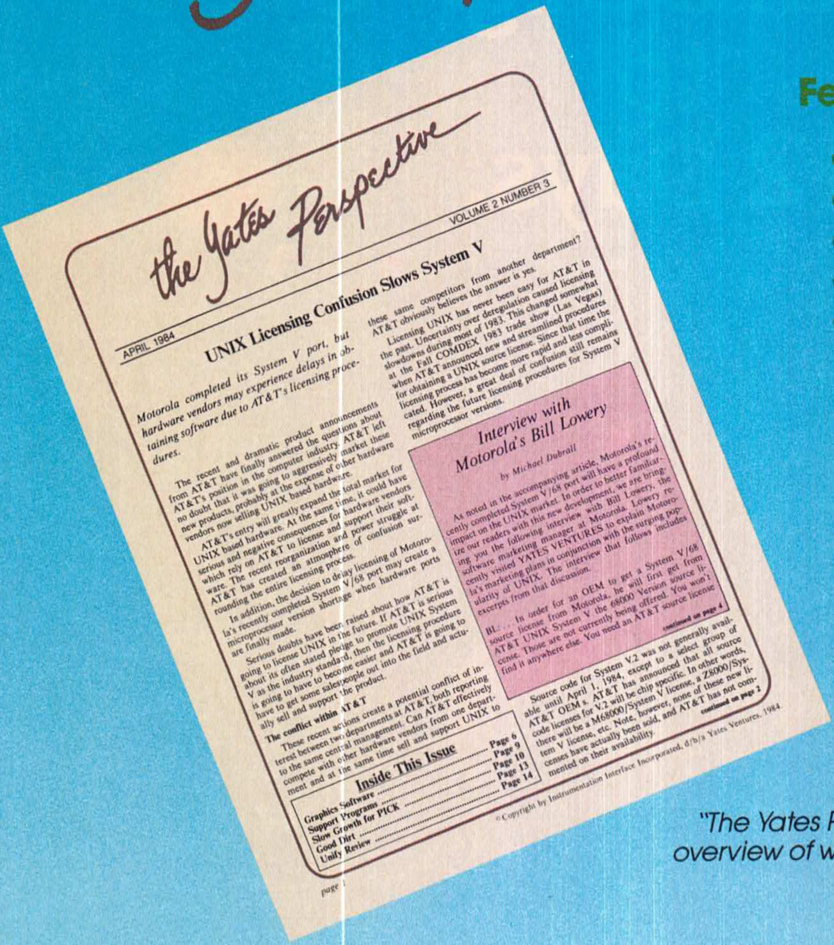
Supermicros built for speed.

Plexus Computers Inc., 3833 North First Street, San Jose, CA 95134

*Subject to prior approval © 1984. UNIX is a trademark of AT&T Bell Laboratories. Multibus is a trademark of Intel Corp.

You can receive an informative, in-depth analysis of leading-edge software industry trends and the UNIX* System Market with

the Yates Perspective



Featuring:

- **Competitive Strategies**
- **Market Analysis**
- **New Product Reviews**
- **Interviews**
- **Benchmarks**

"The Yates Perspective contains provocative and highly original research with no me-too analysis."

Otis Wilson, Manager of Software Sales and Marketing AT&T Technology Systems

"The Yates Perspective offers the best high-level overview of what's happening in the UNIX industry."

Karan Kauppila, President Handle Corporation

Published by Yates Ventures, *The Yates Perspective* is a monthly newsletter on the software market. With this SPECIAL OFFER you can subscribe to *The Yates Perspective* for a reduced rate of $349.00 per year. That's $101.00 off our regular subscription price.

*UNIX is a trademark of AT&T Bell Laboratories

☐ Yes! I want to subscribe to *The Yates Perspective* for a reduced rate of $349.00.

Name ______________________

Title ______________________

Company ______________________

Street ______________________ M/S __________

City __________ State __________ Zip __________

Phone ______________________

☐ My check or money order is enclosed.

☐ Please bill my company. My purchase order number is __________.

☐ Please charge my (circle one) VISA/ Mastercard/American Express

Number __________

Signature __________

Exp. Date __________

Address all correspondence to: Yates Ventures, 3350 West Bayshore Road, Palo Alto, CA 94303

Please circle Ad No. 53 on inquiry card.

UNIX™/WORLD

VOLUME I, NUMBER 5

CONTENTS

A TECH VALLEY PUBLICATION

THEME 32

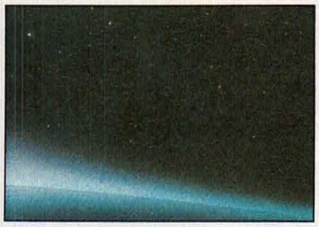
MAINFRAME GRAPHICS HEADS THIS WAY *by Jim Warner and Geoffrey Scott*

Two noted purveyors of graphics software forecast a brave, and colorful, new world of capabilities for the Unix system user.

38

ICONIC DESIGN AND CORPORATE IDENTITY *by William Elmore*

What are icons, and how do they do what they do anyway? A graphic designer tells you about the theories and practices behind the boom.



50

UNIX AND GKS IN A NEW AGE *by Aaron Marcus*

A partisan look at the Graphical Kernel System and its chances for setting *the* graphics standard for Unix systems.

59

THE UNIX SYSTEM AND INTERACTIVE GRAPHICS *by Mark S. Callow*

Low-cost hardware and software are prodding experienced Unix system programmers to push the graphics frontiers. Here's a technical look at the issues.

REVIEWS 66

DEC'S ULTRIX 32 *by Gene Dronek*

DEC looks to the Berkeley doctors for its first officially blessed and supported Unix system port for the mighty VAX.

78

THE REALWORLD ACCOUNTING SYSTEM *by Nicholas Feakins*

Our integrated accounting software series continues this month with the RealWorld suite, one of the most popular for the Unix system.

84

THE SUN-2 WORKSTATION *by Bruce Mackinley*

Seer Bill Joy and company have created the Xerox Star of scientific workstations, the powerful network-based, single-user Sun 2.

CONTENTS CONTINUED ▶

Gould...Innovation and Quality in UNIX-Based Systems

Now Gould offers the widest range of UNIX-based systems in the world.

Gould's PowerSeries™ computers span the performance range. From professional desk-top computers and high-resolution graphics workstations to shared resource and back-end processors with up to twice the performance of a VAX 11/780.™ And, all with the UNIX* operating system to assure interconnectability and software compatibility.

In addition, there's utility. Gould PowerSeries computers are designed for the way you work. Users have all the advantages of a dedicated system plus the lower-cost-per-user option of sharing resources with Gould's offering of standard networking strategies including Ethernet.™

Computer users also need compatibility. Gould's years of expertise with UNIX software have produced the "Compatibility Suite" of application software packages that is compatible and consistent across the entire PowerSeries product line under our Universal Timesharing Executive, UTX,™ a unique combination of Berkeley 4.2 BSD with selected features from AT&T's System V.

And, for your service and support needs, you can rely on Gould's worldwide customer support network.

Put Gould PowerSeries Computers to work for you now.

Gould Inc., Computer Systems Division
Distributed Systems Operation
6901 West Sunrise Boulevard
Fort Lauderdale, FL 33313
1-800-327-9716

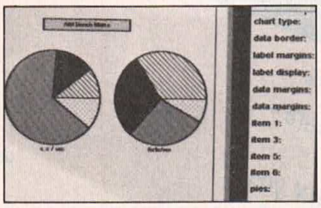
*UNIX is a trademark of AT&T Bell Labs
™PowerSeries and UTX are trademarks of Gould Inc.
™Ethernet is a trademark of Xerox Corp.
™VAX is a trademark of Digital Equipment Corporation

GOULD
Electronics

Please circle Ad No. 10 on inquiry card.

C O N T E N T S

CONTINUED



REVIEWS — 94

THE INTERLEAF PUBLISHING SYSTEM — *by Bruce Mackinlay*

Where giants Xerox and Apple have faltered, startup Interleaf looks to succeed with in-house publishing software for the Fortune 500.

JOURNAL — 98

DISK I/O OPTIMIZATION ON UNIX MICROS — *by David Robboy*

One constant complaint about the Unix system is the disk I/O bottleneck. Our author gives detailed kernel modifications for Xenix 286 and Berkeley VAXs.

104

C TUTORIAL: SOME LOGICAL OPERATORS AND MORE SOFTWARE TOOLS — *by Dr. Rebecca Thomas*

Dr. Thomas' C saga continues this month with a discussion of three commonly used logical operators and some software tools for multiple function calls.

112

ADVANCED SCCS: HOW TO MAKE SCCS AND MAKE WORK TOGETHER — *by Dr. Rebecca Thomas*

In the last article of a three-part series, Dr. Thomas gives some tips on how SCCS and the `make` program work together to manage large programming and text projects.

TRENDS

FROM THE PUBLISHER'S DESK 7
EDITOR'S CONSOLE 9
ON-LINE 13
mail 19
INSIDE EDGE 22
NEW PRODUCTS 121
NEWS FROM AT&T 130
NEWS FROM BERKELEY 132
sync 143

TRAINING

USER SPOTLIGHT 118
WIZARD'S GRABBAG 134
CALENDAR 140

/etc

MARKETPLACE & CAREER OPPORTUNITIES 120
ADVERTISER'S INDEX 136

UNIX/WORLD (ISSN 0739-5922) is published monthly by Tech Valley Publishing, with offices at 444 Castro Street, 12th Floor, Mountain View, CA 94041, telephone (415) 964-0900. Third class permit #3481 Dallas, TX 75260. John M. Knapp, Publisher. RETURN POSTAGE GUARANTEED. Postmaster, send Form 3579 to UNIX/WORLD, 444 Castro Street, 12th Floor, Mountain View, CA 94041.

Subscriptions are $18 for one year within the USA and its possessions and Canada. Foreign subscriptions add $20 for surface mail. Remit in US funds drawn on a US bank. *Subscription questions and problems:* Contact Cheryl Hogan, Circulation Director. Printed in United States of America. Requests for reprints or bulk orders should be addressed to the Circulation Director.

Address all editorial correspondence to the editor at UNIX/WORLD, 444 Castro Street, 12th Floor, Mountain View, CA 94041. Unsolicited manuscripts can only be returned if accompanied by a self-addressed, stamped envelope. UNIX/WORLD is not responsible for lost or damaged manuscripts, photos, or artwork.

FROM THE PUBLISHER'S DESK

Please take a moment to look out the nearest window. Do you see any chunks of falling sky?

Recently several market analysts have raised the time-honored cry, "The sky is falling! The bottom is falling out of the Unix system market!"

The truth is that Unix systems are selling quite well indeed. The problem is analysts' preconceptions. Most doomsayers recently hail from the PC market and expect shipped units to match those of the retail-commodity world. (Not to mention that these analysts have reason to insure their position in an arcane science against being negated by an emerging standard.)

Unix system users are joining the ranks at respectable five-figure numbers every month. For comparison, the best-selling business minicomputer of the 1970s, the HP 3000, took *12 years* to sell 12,000 systems.

It's time analysts realized that professional computing equipment sells differently than amateur. Unix system purchases do not fit comfortably into that neat budget line marked "Other." Rather than a PC system price of $5,000, a prospective user is looking at least at $30,000—but at a cost of only $2,000 or $3,000 per user. (And when your DP management gets wind that you have 10 unconnected PCs sitting around, they're likely to point out heatedly that $50,000 is serious money, and you can get one hell of a twenty-user Unix system supermicro for that price.) The point is that the purchase of a five- or six-figure capital acquisition takes some time—perhaps more than a year. But once made, it will stick—if for no other reason than old-fashioned depreciation and amortization schedules. By definition, supermicros and minicomputers *can't* be a fad. On the other hand, the purchase of multiple PCs is often made in haste and repented at leisure.

And IBM is about to prove my point. With their new, single-user PC AT, they have once again with great fanfare announced a product without a market. How many users out there are grumbling that they're running out of room on their XT's 10 Mbyte disk? How many are complaining that processing on their PC is so slow that they're ready to give up on spreadsheets? No one is about to rush out and replace their perfectly fine PC system for a more expensive system that does exactly the same things somewhat faster. When it's time to replace a damaged PC perhaps, but replacement is not about to engender million-unit sales.

On the other hand, how many users are seriously investigating multi-user, multitasking systems to multiply their production while reducing their budgets? No matter how it's buttered, "Popcorn" will remain a bust until IBM releases Xenix 286 in January.

Single-user PCs are great little machines with great big problems: incompatibility, data security, limited memory, single-tasking for a single user at a single time. Their only advantage is that they are cheap—if you only need one.

Forget AT&T and NCR ads on TV. The Unix system has the ideal marketing position: an honest need that no one else can fill.

John M. Knapp
Publisher

UNIX™/WORLD

STAFF

Publisher
John M. Knapp
Editor Emeritus
Dr. Rebecca Thomas
Executive Editor
Philip J. Gill
Art Director
Dale Bates
Editor at Large
Lauren Weinstein
Copy Editor
Andrew Ould
Editorial Production Coordinator
Monica Berg
Production Manager
Gerald A. McEwan
Production Assistants
James McEwan, Lue McEwan
Staff Photographer
Ralph Cooksey-Talbott
Circulation Director
Cheryl Hogan
Circulation Consultant
John Klingel
Circulation Assistant
Beth Doolin
Assistant to the Publisher
Nancy Thompson
Cover Art
Ralph Cooksey-Talbott

CONTRIBUTING EDITORS

Dr. Brian Boyle, Gnostics Concepts: Market Research
Steve Bourne, Silicon Graphics: Shells
Gene Dronek, Aim Technology: System Optimization/Benchmarking
Dr. Bill King, UC Davis: Writer's Workbench
Bob Marsh, Plexus Computers: Supermicros
P.J. Plauger, Whitesmiths: Work-alikes
Omri Serlin, ITOM International: Market Analysis
Deborah K. Scherer, Mt. Xinu: Software Tools
Lauren Weinstein, Vortex Technology: Networking/Communications

ADVERTISING SALES

New England, New York, New Jersey, Pennsylvania, and Eastern Canada: Charles E. Lynch, Nancy Wood, Merrie Lynch, 61 Adams Street, Braintree, MA 02184, (617) 848-9306.
Midwest: John J. Geier, 444 Castro Street, 12th Floor, Mountain View, CA 94041, (415) 964-0900.
Denver: R.H. "Bud" Foster, 14 Inverness Drive East, Bldg. A, # 232, Englewood, CO 80112, (303) 790-8181.
Dallas: John White, Donald Moeller, 5801 Marvin D. Love Freeway, Suite 303, Dallas, TX 75237, (214) 941-4461.
Los Angeles: Gordon T. Sutton, Carole Ann Philips, 524 W. Commonwealth, Suite G, Fullerton, CA 92632, (714) 525-6686.
Miami: Ray Rickles, Ray Rickles and Company, 3783 Pinetree Drive, Miami Beach, FL 33140, (305) 532-7301.
Atlanta: Scott Rickles, Ray Rickles and Company, 699 Carriage Drive, N.E., Atlanta, GA 30328, (404) 252-4567.
San Francisco: Morton McDonald, Ginger McDonald, Ian McDonald, 265 Bayridge Office Plaza, 5801 Christie Avenue, Emeryville, CA 94608, (415) 653-2122/967-6400.
Oregon: Frank M. Eaton, 510 South First, P.O. Box 696, Hillsboro, OR 97123, (503) 640-2011.

TECH VALLEY PUBLISHING

President	Office Manager	Receivables
John M. Knapp	Verlene J. Perry	Connie Bolak

EDITOR'S CONSOLE

I have a confession to make. I used to think that computer graphics was a boring subject. But I don't really think I was alone in that feeling, either. With the exception of magazines devoted entirely to computer graphics, the computer trade press has generally relegated computer graphics coverage to the background. And who can really blame them? With such excitement and turmoil in the personal computer and software markets, to name a few, graphics certainly doesn't seem to have garnered its fair share of the computer media's attention.

However, I don't really think that we in the press were very much different from our readers and the industry.

The mention of computer graphics brings two immediate images to mind. The first is of some lab where engineers dream up the latest in technology by using enormous and expensive CAD/CAM systems. The second image, and the more sketchy of the two, is that of some lowly office worker sitting at a terminal conjuring up some bar and pie charts for the boss' presentation tomorrow.

The point I'm making here is that, until very recently, graphics, whether for business or engineering applications, has been off on the sidelines, largely separated from the two computing mainstreams, data processing and office systems.

That's changed dramatically in the last year or two, as graphics has become a standard item in most office systems and personal computer applications. Most importantly, however, has been the work done by firms such as Xerox and Apple in pioneering and promoting the graphical or icon interface in the Star, Lisa, and Macintosh. The bottom line is that, from now on, we computer users will be interfacing more and more often with our machines through graphics.

It is in this regard, the graphic or icon interface, that much work needs to be done for the Unix system. Many people assure me that it is only a matter of time until a Star- or Lisa-like interface becomes readily available for Unix system-based machines, but I must implore AT&T and other Unix system vendors to hurry.

Prominent graphics purveyors Jim Warner and Geoff Scott, both of Precision Visuals, Inc., a graphics software house, lead off this issue with the article "Mainframe Graphics Heads This Way," a glowing report on the future of graphics for Unix system users.

Two issues back, we devoted this magazine to Unix system standards. In my *Editor's Console* that month, I briefly mentioned that one particular area of the Unix system standards debate crying out for immediate attention was that of graphics.

In this issue, we delve a little further into the particulars of that debate, with an impassioned (and biased) discussion of one of the emerging graphic standards, GKS, from Bill Elmore, president of Visual Engineering.

We've also brought you three exciting reviews this issue. The first explores the Sun, a leading workstation for lab environments, while the second examines the Intereaf in-house graphics publishing software. These two products will show that bit-mapped workstations may well be worth the money.

The third is an exclusive review of Digital Equipment Corp.'s Ultrix 32 Unix-based operating system. It's the first officially blessed and supported Unix system for the mighty VAX from the mavens of Merrimac themselves. ■

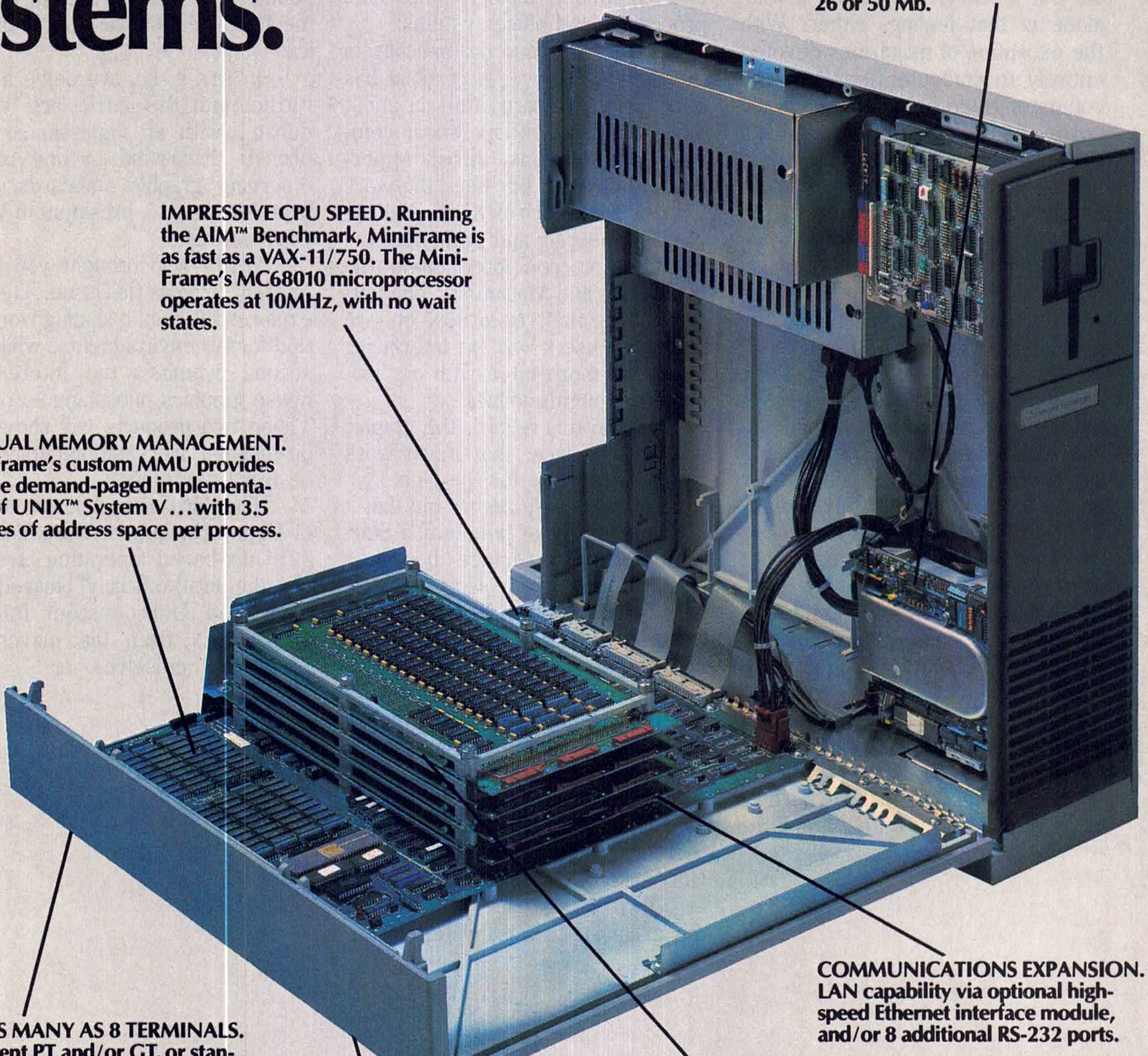
Philip Gill
Executive Editor

An open-and-shut case for making MiniFrame™ your first choice in low-cost OEM systems.

ONE TO EIGHT USERS. MiniFrame can be configured for up to 50 Megabytes of integral mass storage: 5¼" Winchester fixed disks of 13, 26 or 50 Mb.

IMPRESSIVE CPU SPEED. Running the AIM™ Benchmark, MiniFrame is as fast as a VAX-11/750. The MiniFrame's MC68010 microprocessor operates at 10MHz, with no wait states.

VIRTUAL MEMORY MANAGEMENT. MiniFrame's custom MMU provides unique demand-paged implementation of UNIX™ System V... with 3.5 Mbytes of address space per process.



COMMUNICATIONS EXPANSION. LAN capability via optional high-speed Ethernet interface module, and/or 8 additional RS-232 ports.

RUNS AS MANY AS 8 TERMINALS. Convergent PT and/or GT, or standard ASCII. Multidrop RS-422 communications line operates at 307 Kbit/sec.

ECONOMICAL MEMORY EXPANSION. MiniFrame provides ½ Megabyte of RAM standard; up to three boards can be added for system expansion to 2 Mb.

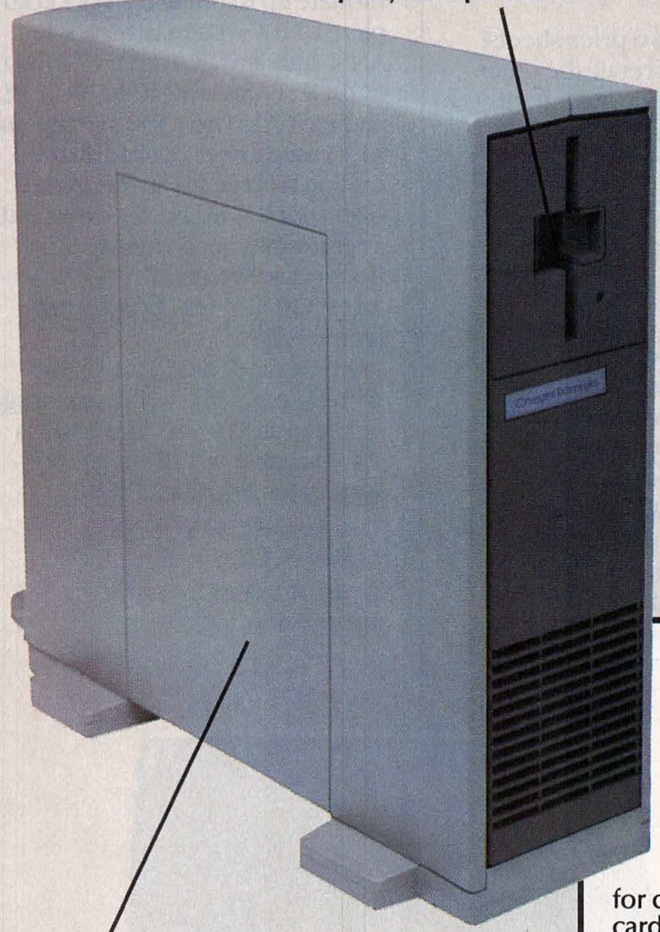
FOR HIGH-SPEED PRINTERS, a Centronics-compatible parallel printer port is standard in the MiniFrame processor.

MiniFrame: another first from Convergent Technologies. Super-minicomputer power at PC prices.

OEMs can now meet the needs of small to medium-sized organizations for low-cost, high-performance systems capable of handling large UNIX-based applications.

The MiniFrame system—which effectively opens a whole new market for you—is unique not just for its price/performance advantages. It also provides complete flexibility in meeting requirements ranging from single-user "personal UNIX systems" to eight-terminal data-processing installations.

BACKUP STORAGE with integral 5¼", 640 Kbyte floppy disk. Optional high-capacity backup available.



WE'VE CLOSED OUR CASE... now send for a complete information package. Write to Convergent Technologies, Data Systems Division, 3055 Patrick Henry Drive, Santa Clara, CA 95050. Phone: 408/980-0850. Telex: 176-825.

Convergent Technologies achieved this breakthrough by adapting the innovative hardware and software developed for its MegaFrame multiprocessor.

The result is a powerful, compact and expandable unit priced to give you the maximum opportunity of capitalizing on a rapidly growing UNIX market.

There are, of course, many other unique benefits of the MiniFrame system.

Convergent provides foundation software for office applications—including a powerful, Wang keystroke-compatible word processor, an advanced financial spreadsheet and complete electronic mail facility.

The Window Manager permits viewing and manipulating of up to four applications running simultaneously on Convergent's PT or GT terminal screens.

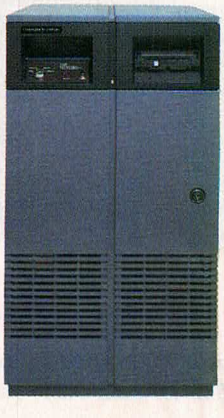
Major performance increases result from utilizing these terminals with the MiniFrame. At only slightly higher cost than standard "dumb tubes", our PT or GT terminals offer high-speed communications plus built-in processor with enough memory to execute key portions of the system code.

Programming languages include industry-standard High Level COBOL and BASIC, full FORTRAN-77, Pascal and C.

OEM prices for the MiniFrame start at less than $5,000; an eight-user MiniFrame can be configured for under $10,000. Prices like these make it an open-and-shut case for choosing the MiniFrame.

Graceful upgrade path to the MegaFrame.™

The MegaFrame, Convergent Technologies' revolutionary super-minicomputer system, utilizes multiple processors; has expansion potential to 128 users, 8 MIPS and gigabytes of disk storage. It enables OEMs to handle today's growing demand for computing services without discarding a single piece of hardware ... or being forced into expensive CPU upgrades.



Applications software can be transported—unchanged—from the MiniFrame to the MegaFrame whenever the workload requires it. The two systems are object-code compatible, allowing OEMs to provide a complete family of systems.

MegaFrame: proof that if anyone can build a UNIX system the way it should be built—it's Convergent Technologies.

Convergent Technologies

Where great ideas come together

Please circle Ad No. 6 on inquiry card.

THE CODATA DIFFERENCE

A Major Improvement in OEM Microcomputers.

With so many microcomputers on the market, and with each manufacturer claiming to have the solutions to everyone's problems, why are more and more OEM users choosing CODATA?

Very simply because the CODATA 3300 works!

That's an important part of what we call THE CODATA DIFFERENCE - a refreshing, new concept in OEM marketing.

More performance...Lower price...Extended warranty.

We've increased the performance of the CODATA 3300 and, at the same time, we've lowered the price. That means our CODATA multi-user, multi-tasking 16-bit computer systems have price/performance ratios nobody else can match. We're so confident in our system's integrity that we've extended the warranty on every CODATA 3300 to 180 days. That's the longest warranty in the business and, let's face it, you've got to be pretty sure of your product to guarantee it for six full months. We're sure.

These are more elements of THE CODATA DIFFERENCE.

An uncomplicated pricing policy.

We have only two price sheets: one for suggested retail, the other for OEM prices. Simple, straightforward, easy-to-understand. Beyond that, there are no quantity commitments, no bill-backs, no hassles.

Full support and technical assistance if needed.

Our customers select the CODATA 3300 because it fulfills their needs, because it does what we say it will do, and because we have taken great pains to assure that it is functionally simple and capable of performing its assigned tasks with no problems. But, if technical support is required it's only as far away as your phone and available at a moments notice via our TOLL FREE number during normal business hours in every U.S. time zone.

A multi-user, 33-megabyte system for only $9,600.

The CODATA 3300 is a powerful 16-bit, 68000-based, MULTIBUS system that can effectively accommodate up to ten users. It's a complete UNIX system that runs full ANSI standard FORTRAN-77, RM/COBOL, BASIC+, SMC BASIC, APL, and PASCAL. The 3300 provides up to 33 megabytes of unformatted on-line storage via an integrated, high speed Winchester drive, and a removable, quad-density 5-1/4″ floppy disc system. The 3300 features 320 K bytes of parity protected RAM.

An expanded CODATA 3300, with 84 megabytes of disc storage is priced at $13,500. The 3300 is also available as a 12-megabyte system for as low as $7,800.

LET US SHOW YOU THE CODATA DIFFERENCE.

Write or call us for more information.

CONTEL CODATA

285 North Wolfe Road, Sunnyvale, CA 94086
1-800-521-6543. In California: 1-800-221-2265

Please circle Ad No. 4 on inquiry card.

KEYBOARD AND MONITOR OPTIONAL

ON NCC, POPCORN, AND CONFUSION

BY PHILIP J. GILL

Despite a general consensus that this year's National Computer Conference (NCC) in Las Vegas was a relatively boring performance, three rather significant trends were obvious from booth traffic and product displays. First, although increasingly vociferous attacks on the Unix system reached a near frenzied pitch just prior to NCC's opening, it was quite clear to all but the most biased that the show was in many ways the Unix system's official coming out party.

The two most exciting new product introductions at the show were in fact Unix system-based software: the Applix, Inc., office automation suite and NCR Corp.'s NCR Officeware. Add to this AT&T's lavish booth display that prominently featured the Unix system and IBM's introduction just days before of VM/IX for the 370-type processors, and you've got the sure makings of an industry standard.

Secondly and tangentially, NCC marked the first appearance of system-level, office automation software for the Unix system environment—again, Applix and NCR Officeware being prime examples.

The show's third major theme was the attention being paid by vendors and users to the man/machine interface. Apple Computer's booth, which featured the Macintosh and which noticeably didn't feature the IIc or IIe, was mobbed with attendees throughout the run of the show. Other booths where the products paid attention to the man/machine design were also similarly busy.

IBM'S POPCORN DUE

As we were closing this issue, numerous reports from our friends in the industry were quoting an August 14th debut for IBM's long-awaited and much-rumored "Popcorn" multi-user supermicro system. This means that if all goes according to plan, you'll have the official word from IBM before this issue reaches your desk.

For the most part, our sources tell us the machine is exactly what we've all been expecting: a multi-user addition to the ubiquitous IBM PC family based on the Intel Corp. 80286 microprocessor, supporting a 20- Mbyte hard disk and running PC-DOS and Xenix.

Now, here's the rub. Before we all cheer this event as a "legitimizing" of the Unix system for commercial environments, our sources tell us that the Xenix operating system won't be available until several months after the machine starts shipping. In its stead, the Popcorn wil run under a new version of PC-DOS—the long-awaited PC-DOS 3.0.

Why, you might ask, is IBM holding back on supporting a Unix system-based operating system for its supermicro-class multi-user system when Unix systems are the de facto standard in this niche?

One observer astute in the ways of IBM harkened back to the days when the original IBM PC was introduced. At that time, the PC was introduced with PC/DOS as its flagship operating system, with both CP/M-86 and the UC San Diego p-System available at some later time as options.

This deliberate holding back on IBM's part of the 16-bit version of the 8-bit de facto standard CP/M-80 created the window necessary for IBM and Microsoft to elevate PC-DOS—MS-DOS to industry standard status. The same game, some say, is being played out with Xenix (read: the Unix system) and PC-DOS 3.0 and its descendants. IBM is afraid of the Unix system bandwagon, especially now that AT&T is leading the horses to water.

This move to circumvent the drive toward standardization on the Unix system and its derivatives could spell trouble for many multi-user system vendors.

CONFUSION IN DAYTON

We're hearing troubling reports of rivalry between two separate divisions at NCR following the introduction by the firm's OEM Marketing Group of the Unix system-based NCR Officeware office automation suite for the Tower 3216 at NCC. It seems that one of NCR's other direct sales arms, the Office Systems Division, is out on the streets selling Convergent Technologies' Unix system-based MiniFrame and MegaFrame computers for office automation applications. Officially, no conflict exists, but most analysts, as well as myself, will bet that some users don't see it quite the same way.

SEQUENT NS32000 SYSTEM DUE

Amid much publicity, a group of Intel managers in Portland, Ore., split from the semiconductor

HANDLE™

Office Automation Software. . .

Selected for Distribution by AT&T.

HANDLE is the number one office automation or "integrated" software system available for the UNIX™ operating system. Selected by **AT&T** Information Services for distribution on the 3B2/300™ and 3B5™ UNIX systems, **HANDLE** is the most powerful full-featured office automation system in the market today.

Some of **HANDLE**'s unique features include:

- Dynamically integrated word processing, graphics, list management, virtual spreadsheet, and spelling correction with a powerful "document metaphor" that provides a common method for creating, printing, merging, archiving off-line, filing under multiple topics and retrieving documents.
- Full multi-user capabilities with extensive security — documents, or portions of docuents, can be marked "author only", "read only" or "general access". **HANDLE** provides its own record locking.
- User preferences — **HANDLE** conforms to each individual's preferences for such things as preferred printers and individual document catalogs. User restrictions can control such things as the ability to delete documents, access archive catalogs, etc.

The AT&T 3B2/300 computer with HANDLE on screen and integrated graphics printouts.

HANDLE™

The first Office Automation System for UNIX.

HANDLE CORP, 140 MACKINAW ROAD, PO BOX 7018, TAHOE CITY, CA 95730 • 916-583-7283 **Please circle Ad No. 12 on inquiry card.**

giant a year or two back to found a system house. That firm, now known as Sequent Computer Systems, is readying an NS32032-based multiprocessor computer running an enhanced version Unix 4.2BSD.

The firm claims that its new system, called the Sequent Balance computer, permits OEMs and end-users to expand from 2 to 12 National Semiconductor Corp. 32032 microprocessors without the necessity of rewriting applications software. Sequent is aiming its Balance computer at the OEM market for CAD/CAE, ATE, and image processing applications.

UNIX SYSTEM PORTABILITY

Officials at Otrona Advanced Systems, a Boulder, Colo., portable personal computer maker, say the firm will support the Unix system on their recently introduced 2001 portable IBM clone during the third quarter of this year. However, Otrona's vice-president of marketing, Joe Calabria, has expressed some concern over which version and what add-on microprocessor board Otrona should implement for the Unix system option.

He said the firm is currently leaning toward AT&T's System V running on the M68000 chip, but he added that Otrona would alter that plan and follow IBM's lead if the computer giant comes out, as all indications are that it will, with an Intel 286-based system running Microsoft's Xenix.

HEARD WANDERING ABOUT THE HALLS

All's not as well as Apple projects, with its much-vaunted Macintosh personal computer. Sales have reportedly flattened out due to lack of software, and some members of the Macintosh team are said to be ready to walk.... Convergent's recent spate of trouble with the WorkSlate portable and N'GEN desktop computers have apparently extended to the MegaFrame as well. Some unspecified hardware bug has already been remedied, however. Meanwhile, WorkSlate inventory is at least 3,000 to 5,000, with some unconfirmed estimates much higher.

A DIGEST OF NOTEWORTHY EVENTS

Systech Corp. (San Diego, Calif.) signed a marketing agreement with Quest International Computer Equipment, Ltd., (England) to distribute Systech's multibus products in the United Kingdom...**Fortune Systems Corp.** (Redwood City, Calif.) announced that LeRoy Cochran has been appointed senior vice-president and CFO...**Callan Data Systems** (Westlake Village, Calif.) agreed with Telesoft to market Telesoft's Ada compiler with Callan's Unistar supermicrocomputer....

Officesmiths, Inc., (Ottawa, Canada) signed an agreement with JTS Computer Systems, Ltd., to put Officesmiths' document management system on Gould's PS 3000 and on Integrated Solutions, Inc.,'s Optimum series of computer systems...**Durango Systems, Inc.,** (San Jose, Calif.) announced a 40-Mbyte hard disk and a 20-Mbyte cassette for the firm's Poppy and Poppy II Unix system-based business systems...**Cado Computer Systems** (Torrance, Calif.) appointed Eystein Thordarson as vice-president of operations. He will have responsibility for the Codata subsidiary....

Softport Systems, Inc., (New York, N.Y.) announced a price reduction in its Muse scientific/technical word-processing package on the Masscomp computers... **Human Computing Resources Corp.** (Ontario, Calif.) and NCR Canada, Ltd., announced an agreement under which HCR's Chronicle business software will be distributed on NCR's Tower computers...**Software Manufacturers, Inc.,** (Carson, Calif.) appointed Exxnet Distributing Co., Creative Business Solutions, Inc., and Oasys, Inc., as domestic distributors of its S-Tran BASIC-to-C translator....

Applix, Inc., (Southboro, Mass.) recently received $4.5 million in its second round of venture financing...**Oracle Corp.** (Menlo Park, Calif.) announced an agreement with Hewlett-Packard to have its DBMS supported on the HP9000 series 500 computer. Oracle's DBMS will be marketed by Auragen Systems Corp. under the name Aurelate and will run on the Auragen System 4000...**Relational Database Systems, Inc.,** (Palo Alto, Calif.) signed a co-labeling and distribution agreement with AT&TInformation Systems to supply File-it, Informix, and C-ISAM on all of AT&T's computers.... ■

CHOOSE A COMPUTER FOR ITS MIND NOT ITS BODY.

Big. Lots of muscle. Sexy. All ways we describe the computer of our dreams. By those standards, the humble Reduced Instruction Set Computer (RISC) seems like nothing to write home about.

Unless someone in your family designs computers.

Because in high-level language benchmark tests* on the Berkeley campus, RISC outperformed heavyweight superminis and micros by as much as four to one. (Incidentally, most benchmarks were written in C, the same language used to implement UNIX.)

Don't be fooled. What RISC lacks in sex appeal it makes up for in shrewdness.

Instead of using raw muscle to speed up memory references, RISC uses logic to eliminate them. Two hundred fifty-six registers, not memory, hold data for processing.

Parameter passing schemes worthy of a doctoral thesis are replaced by simple hardware. A sliding "register window" easily redefines one procedure's temporary data as another's parameters.

And while hundreds of complex instructions on most computers are never used by high-level languages, every super-optimized RISC instruction is. All 31.

But how would RISC perform when it left college and tackled its first real job?

Just fine, we thought. So when we considered architectures for our high-performance UNIX supermini, the Pyramid 90x, we felt quite comfortable taking a RISC.

We combined its large register stack (528 in our case), register window, and reduced instruction set (about half the number in a typical supermini) with some common computer techniques. Like virtual memory, cacheing and floating point.

The results were uncommon.

Two to four times the performance of popular UNIX hosts on standard benchmarks like Ackermann's Function. In less than half the floor space and at about half the price.

The first commercial implementation of RISC worked as well in the computer room as it did in the classroom.

And proved that you don't have to have brawn to make a good UNIX computer. Just brains.

We'll be glad to send you more information on how RISC works, including a reading list of recent research, if you send your name and address to RISC, Pyramid Technology, 1295 Charleston Road, Mountain View, CA 94043.

Your comments and questions are always welcome. Just call (415) 965-7200.

PYRAMID TECHNOLOGY

*Write us for test details. UNIX is a trademark of AT&T Bell Laboratories.

mail

This space is yours. Tell us what we're doing right—or wrong. Raise issues and ideas that other readers may respond to. Propose projects. Reject market trends. Take a stand.

SOFTWARE PIRACY?

Gentlemen:

The feature article "Software Security and the Pin-Striped Pirate," in the March/April 1984 issue of UNIX/WORLD was well thought out and useful. Unfortunately, when Mr. Auditore came to economic analysis, he merely parroted vendors' claims, which have all the sophistication and economic insight of a sales representative calculating the commission. A more thorough analysis is needed.

One may easily observe: (a) Unsupported (pilfered) copies of commercial software represent a competitive force (arguably an unfair one) to keep prices down. If Context, Lotus, Microsoft, and Sorcim all announced the withdrawal of their spreadsheet products from the market and a recall of all copies previously sold, would VisiCorp respond by selling VisiCalc at $50? (b) The widespread use of unsupported (including pilfered) software is indicative of how highly the marketplace values the support currently offered by vendors. One can compare this to the proliferation of "no-frills" stock brokers, "you-bag-it" grocery stores, and "self-serve" gas stations. (This should not be taken as equating the legal status of any of these to pilfered software.) Only as it becomes uneconomic to pilfer software will it cease. Do moral preachments and draconian threats have a good track record?

From a sociological perspective, most people whom I have observed as having any appreciable quantity of unsupported (including pilfered) software generally have vast heaps of it and never use more than even a small portion. Others use pilfered software as a result (they say) of bad experiences with the deplorable quality of software which they legitimately purchased. Paying several hundred dollars for a highly touted product to discover that a far less expensive competing one has far greater functionality can reduce one's good will toward all manufacturers. One could also easily claim that no company with more than 10 employees should even pretend that it supports the microcomputer user (MicroAm and GerundiveStar).

I should also note a most penetrating and provocative analysis of copyright at the end of "Sony Versus Universal Studies: So What?" in *PC Tech Journal,* 1:8 (May 1984), pp. 197–201.

Thank you,

Ross Parlette, Bldg. 0010
computer systems specialist
United Technologies Chemical Systems
P.O. Box 50015
San Jose, CA 95150-0015
(408) 779-9121

Author's response: The perception that software theft is a positive competitive market force is similar to thinking that shoplifting from a department store is helping to keep prices down. Major retail outlets allocate 2-3 percent of their sales total to "lost" merchandise. Clearly they don't accept 2-3 percent less profit. To insure that this percentage does not hit their bottom line, it is added into the price of all products being sold. The software business is much the same. Instead of Mr. Parlette's utopia, where software pirates are keeping software prices down, what we have is the rest of us supporting these Brooks Brothers buccaneers by having the losses attributed to piracy added to the price of our legitimate purchases.

Regarding the value of vendor-offered support, it is important to remember that the software vendor decides what the product is and how much he thinks it's worth. If you don't agree with the vendor's valuation, you don't have to buy it. But one of the options upon disagreement of valuation is not the right to steal it. Certainly a case can be made for "no-frills" software, similar to a self-serve gas station, but remember, you still pay for the gas.

Software piracy is just as much a crime as is shoplifting a pair of Calvin Klein jeans from Macy's. However, because of our inability to fully comprehend the concept of intellectual property, there is no social stigma attached to the rape and pillage of a software program. Instead, the hackers that rip us all off are condescended to: "My, how bright! Isn't that clever?" Indeed. Just as clever as

the people who engineered the Brinks robbery or the thugs who robbed the local 7-11 store.—Steve Auditore

A C− AND A C+

Gentlepeople:

I thrill to see and read the slick QUALITY of the first two editions of your wonderful magazine. (But please include your city name in copyright notice area of the journal, fifth paragraph from bottom of page 5, in the March/April issue. You also omitted it in the first issue.)

The trigger causing this letter was the coincidence of Lauren Weinstein's explanation of HACKER-distortions given to the "copycat" nature of many news stories and your response to Peter Olsen's letter. Peter's concern for speed differences between Unix and VMS can be traced to the inefficiencies of C language versus Assembler and rolling wagons.

In my 8-bit programming, I've concluded that the speed ratio between the two languages is three to one in favor of Assembler. This same ratio applies to program object code size. So I've given up using C in 8-bit micro work. But if you carry this same performance into the 16/32-bit world, the ratios stay the same. My conclusion is that the price paid to use C is just too much compared to using a structured macro assembly library. Programming efficiency can be just about the same using either approach, but the resultant code load and run-time efficiency is greatly in favor of Assembler.

What this all boils down to, it seems to me, is a "copycat" bandwagon. IBM creates a PC and the wagon rolls; AT&T creates Unix and C and the same thing happens. Few question the wisdom of what is happening. Nonetheless, commerce is served since we all see ways to make money out of wagons rolling. But some of us sorrow at the "monkey-see, monkey-do" nature of our human race.

As a fellow human and neighbor, I wish you every success in your publishing venture.

Cordially,

Frank Gaude, president
Echelon, Inc.
10925 Stonebrook Drive
Los Altos Hills, CA 94022
(415) 941-2219

Dear Dr. Thomas:

I've just finished reading my first issue (vol. 1, no. 2) and want to add my compliments to those you've already received. I think you've put together a magazine which is ideally directed to the largest new readership—persons quite knowledgeable in other systems and languages, but not yet specifically familiar with Unix and C. There is an excellent variety, from beginner's material to the somewhat advanced, which will make re-reading back issues very rewarding as we newcomers gain experience with the system's nuances.

I specifically disagree with the reader who alleged that "nobody learns C from magazines." One certainly needs Kernighan & Ritchie, and the Unix Programmer's Manuals, but they contain so much cryptic, obscure, and/or internally inconsistent material that a "plain English" tutorial is very useful. Please keep it up.

There appears to be a small but significantly misleading error in the box "Representing ASCII Characters by Escape Sequences," on page 51. There, you refer to ASCII control-J as the ASCII "newline" character. Strictly speaking, this is correct only in Unix. Under "real ASCII," a newline is denoted by a combination of two characters, linefeed + carriage return (control −J, 0Ahex + control −M, 0Dhex).

Finally, a suggestion. I note that your editorial convention is to denote ASCII characters in their Octal notation; e.g., in the article cited, you show control −I = ASCII TAB (strictly speaking, *horizontal* TAB) = Octal 009. This is fine for most of your current readership, which comes from a minicomputer background where Octal is still in use. But the vast majority of your potential *future* circulation is in the world of supermicros, where Octal is more often than not regarded as a medieval vestige and most of the knowledgeable speak Hexadecimal. I urge you to consider modifying your policy to use both notations in parallel, which will significantly broaden your base of comprehensibility.

Again, from this reader's viewpoint, you're doing a great job. Keep it up.

Very truly yours,

Del Palmieri
3 Maple Ridge
Ballston Lake, NY 12019

Thanks for the suggestion, Del. ■

A CLOSER LOOK AT PLEXUS AND COMPUTER CONSOLES

BY OMRI SERLIN

My lukewarm attitude toward the Unix system is not new and should come as no surprise to my longtime readers. What may be startling is that, despite my doubts about the value and long-term viability of the Unix system, I do not at all dispute its current significance. This issue's column clearly demonstrates the point. Both major stories this month deal with companies whose commitment to the Unix system is a key element of their strategies. Plexus has been an early all-Unix shop and is beginning to reap the benefits of the current Unix system tide. And Computer Consoles has long based its vertical market offerings on a Unix-derived operating system; last month it unveiled an especially powerful Unix system engine.

Without much fanfare, Plexus Computers, of Santa Clara, Calif., has conveyed its early lead in the multi-user Unix system marketplace to a profitable operation running at a $24-million annual rate. While the planned departure of CEO Bill Jobe is a cause for concern, Plexus believes that, given its significant venture investment, a product line that boasts some of the best performing 68000-based Unix engines, and a networking version of the Unix system, it is well positioned to capitalize on the growing interest in the Unix system.

Privately held Plexus has had considerable success in raising venture capital. Total investment is now over $18.5 million, obtained in four rounds. Among the 17 well-known investors are Kleiner, Perkins; Hambrecht & Quist; Alex. Brown; and L.F. Rothschild, to name a few.

Since shipments began in 1981, the company's revenues have grown impressively—to $3.2 million in 1982, $12.5 million in 1983, and to a projected $24 million in 1984. [This rate has already been reached in the second quarter of 1984.] The company has been profitable since the third quarter of 1983 and expected to close its fourth consecutive profitable quarter in June. Plexus' eventual goal is 20 percent pretax, but at this stage, growth takes precedence on the company's agenda.

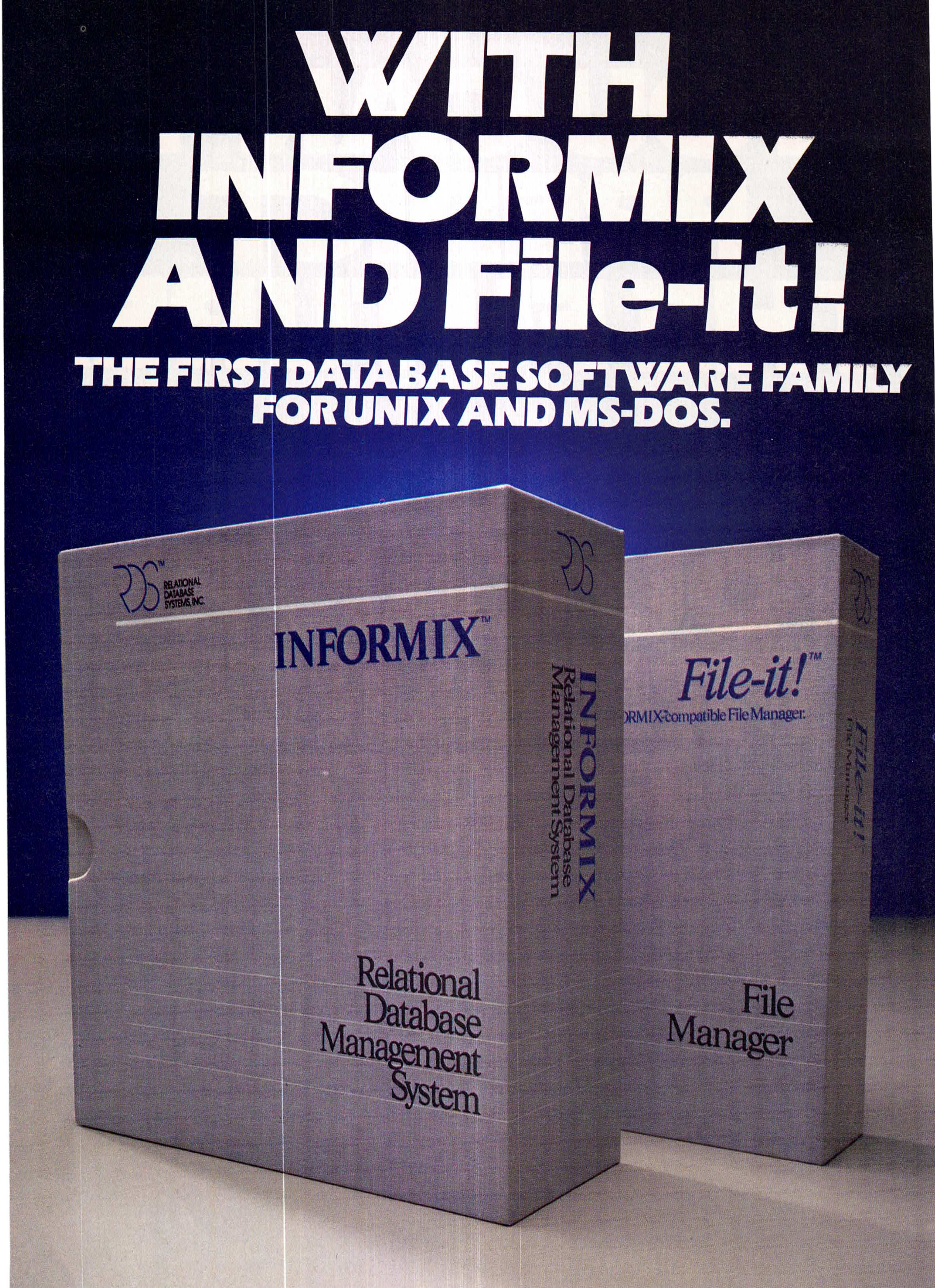
THE ORIGINAL AIM

Plexus set out to be a "minicomputer company using supermicro technology," says chairman Bob Marsh. This meant designing a product that would compete with minis in performance and building a company that would compete for the original equipment manufacturers (OEMs) and value-added resellers (VARs) that previously had based their products on minicomputers.

While OEMs and VARs are also Convergent Technologies' targets, Plexus differs from CT in that it has no low-margin, "blockbuster" deals; instead, it contents itself with smaller accounts that take 100 to 200 units a year. Plexus did sign (in 1983) a $40-million agreement with Philips/Micom, which could amount to several thousand systems, and the company is now negotiating a large, unidentified government procurement.

The product strategy has been basically consistent, even though Plexus has all but abandoned its initial Z8000-based designs (P/40 and P/25) for the current 68000-powered models (P/35 and P/60). The key elements of the product philosophy: First, use reliable, off-the-shelf components, rather than trying to get performance from unproven devices. Second, gear the design to OEMs by providing a multibus backbone to allow configuration flexibility and access to off-the-shelf multibus controllers. Third, provide multi-user performance to match the superminis by clever CPU design and especially by off-loading operating systems functions to a set of intelligent controllers (mainly Z8000-based). Finally, use the Unix operating system. This was initially V.7 (also used by Onyx), but Plexus adopted System III as soon as it was introduced and has stuck with it ever since.

The key architectural concept in the Plexus system is the off-loading of terminal I/O to a separate, intelligent communications processor (ICP). The Onyx experience convinced Marsh that single-board machines, no matter how powerful the MPU, would bog down when serving two or three users. This is due to the way the Unix system handles terminal I/O: Essentially, the kernel is forced to deal with each character separately. To break this bottleneck, the Plexus system


WITH INFORMIX AND File-it!
THE FIRST DATABASE SOFTWARE FAMILY FOR UNIX AND MS-DOS.
RDS™ RELATIONAL DATABASE SYSTEMS, INC.
INFORMIX™
INFORMIX Relational Database Management System
Relational Database Management System
File-it!™
RMIX-compatible File Manager.
File-it!
File Manager

removes the character handling from the kernel and places it in the ICP. The ICP (as with all other Plexus I/O controllers) has a direct memory access (DMA) and communicates with the kernel efficiently, using large blocks.

WHAT THE FUTURE MAY HOLD

With AT&T pouring enormous funds into promoting and developing System V, the obvious question is whether System III supporters like Plexus can survive the onslaught. Bob Marsh, however, is not overly concerned. "You can't buy any software now that needs System V facilities," he says, adding that "everybody—and Plexus—is in the process of porting System V, but there are no orders that hang on its availability."

In hardware, Plexus is evidently feeling the pressure from the low end (mainly Altos and Fortune, and possibly the AT&T 3B2). Although the company does not intend to compete at the PC level, it would like to be able to service the Unix system market below its current eight-user low end. Don't be surprised to see a low-end Unix system appearing from Plexus, probably before the year's end. The firm is also thinking about a high-end product as well, although plans in that area are apparently less well defined.

Marsh is unimpressed by the charge that the Unix system lacks "software ergonomics," limiting its appeal. He says, "There are still numerous applications for plain ASCII terminals [where users work from fixed menus and are unlikely to need mice, icons, or multiple windows]." Furthermore, he predicts that Unix system engines could become popular as file servers for multiple desktop PCs, where the latter provide the "soft" interface.

The Unix system's value, Marsh argues, is not so much its actual implementation; rather, it is the idea of a standard "system call" interface that allows independent software vendors (ISVs) and end-users to develop applications that are portable across a wide range of machines so long as all such machines support the Unix standard.

COMPUTER CONSOLES: A PROGRESS REPORT

Buoyed by strong performances in 1983 and in the first quarter of 1984, Computer Consoles, Inc., is aggressively expanding its focus beyond its bread-and-butter telephone systems business and is pursuing new opportunities in office automation and in multi-user Unix applications. The company's FT/OLTP system, the Power 5/55, is nearing beta installations, albeit a year behind schedule. The new 6/32 and 5/30 offerings, unveiled at the annual meeting, were scheduled to begin delivery in June and September, respectively. And the company is very optimistic about its 1984 prospects.

Under the leadership of CEO Vaemond Crane (ex-Intel Systems Group), who joined on in January 1983, CCI is emphasizing its ability to deliver "solutions," meaning systems with vertical market appeal and the ability to accommodate application growth.

Computer Consoles (ASE: CCS), founded in 1968 and headquartered in Rochester, N.Y., is now organized in four groups. The Telephony Systems Group, under Martin J. Cournan, continues to pursue the company's original, and still key, markets in the domestic telephone industry. The company claims a dominant position (70 percent of all installations) in the directory assistance system segment.

The DAS system lets "information" operators retrieve telephone numbers in response to queries based on customer name or other inputs. The market for DAS systems is rather limited, however. Some growth can come from upgrade or replacement opportunities, as demonstrated last April, when a $17 million CCI DAS system displaced an IBM installation at Mountain Bell.

Also, CCI's Audio Response System (ARS—a voice synthesizer that automatically vocalizes the retrieved telephone number to the caller, freeing the operator to handle the next call) is an attractive add-on. However, CCI recognizes that major growth must come primarily from international penetration and from expanding the range of applications sold to the telephone companies.

In the international arena, CCI scored a major success in May 1983, when it concluded a $34-million agreement to supply British Telecom with the world's largest directory assistance system. The company expects $20 million of that to occur in 1984.

The company's leasing subsidiary (CCLC) was initially established to finance the high proportion of systems leased to the telephony industry. The subsidiary, now under CCI vice-president William N. Stirlen, contributed $14.6 million out of a total of $103.5 million revenues in 1983.

OFFICE SYSTEMS

The Office Systems Group, under John E. McNulty (ex-Intel), was officially established in September 1983. It grew out of the acquisition in early 1981 of Reston, Va.,-based RLG Corp. RLG originated a C-coded, Unix system-running office automation package based on YARD (a relational DBMS).

The software was originally developed as a custom project for the U.S. Transportation Department to run primarily on DEC equipment. Later, CCI/RLG used the Z8000-based System 8000 from Zilog, with Zentec terminals emulating DEC VT100, to deliver the software, renamed Officepower.

In September 1982 the Zilog hardware was replaced by the Power 5/20, a 68000-based, CCI--made product that used the same CPU board as the much larger Power 5/55 transaction system. The Personal Terminal (PT) from Convergent Technologies is now used. Over 400 of the 5/20s had been shipped by May 1984, according to CCI.

Significant recent orders for Officepower on the 5/20 system include Merrill Lynch Asset Management, which will also include a Power 5/55 system; a distribution agreement with STC of the U.K., valued at $6 million over 18 months; Mousaw, Vigdor [et al.], a Rochester law firm that ordered six 5/20s and 65 terminals; a $1.5 million order from the Transportation Department; and a $2 million order from Hale & Dorr, Boston's largest law firm.

Last year, the Office Systems Group got a $1.4 million order for Officepower from the U.S. 9th and 10th Circuit Courts of Appeals. Earlier, CCI got a $3.8 million order from the Naval Surface Weapons Center for the Officepower software (to run on VAX systems), along with terminals.

COMMERCIAL PRODUCTS GROUP

This Irvine, Calif.,-based group, now under vice-president and general manager William Deller, was established in 1982 to complete the development of what is now the 6/32, announced in May. CCI originally attempted to develop this 32-bit, C-oriented supermini offshore, but the project came to an ignominious end, amid acrimony that still persists: CCI felt it had been hoodwinked by the developers, who grossly mismanaged the project; on the other hand, several investors who backed the original venture are now suing CCI, alleging that the company transferred the work to Irvine to avoid sharing the profits with them. The suit, filed in October 1983, is pending before the New York State Supreme Court.

Other than the development of the 6/32, the Commercial Products Group's charter isn't all that clear. It will apparently try to sell the 6/32 as a powerful Unix system engine through its own small sales force. In addition, the 6/32 will be used eventually by the Telephony Systems and Office Products groups in their markets as well. Introduced in May, this is an upgrade of the 5/20. However, the main difference is that the 68000-based "CPU 2" board is replaced with the 68012-based "CPU 3" board for added performance and a larger linear address space. Also, a 45-Mbyte streaming tape drive replaces the 20-Mbyte cartridge.

CCI rates the 5/20 at about 1/2 to 1 million instructions per second (MIPS), while the 5/30 is rated at about 1 to 2 MIPS.

Also unveiled in May was the Power 6/32, the first product from the Commercial Products Group. Although the machine boasts impressive price-performance relative to the 11/780, it isn't entirely clear how it stacks up against the 11/785 on the one hand, and such entries as the Ridge and Pyramid machines on the other. CCI plans to market the 6/32 initially as a general-purpose Unix engine; eventually, it may be used as a vehicle for high-end officepower applications and may also power the next version of the 5/55, dubbed the 6/55.

SHORT NOTES

AT&T Information Systems is slowly beginning to respond to queries from prospective resellers of the 3B2 and 3B5 lines. The following OEM schedules are in effect:

QUANTITY	*3B2/300*	*3B5*
1- 4	12%	20%
5- 9	15%	23%
10- 19	15%	27%
20- 49	18%	30%
50- 99	22%	33%
100-199	30%	36%
200-300	33%	38%
301-499	33%	38%
500-800	35%	--

Force Computer, Santa Clara, Calif., the U.S. portion of a Munich, West Germany,-based firm of the same name, offers an assortment of VME bus boards, including two 68000/010-based CPU boards, a host of memory and peripheral interfaces, and software. Plessey Microsystems, Pearl River, N.Y., has recently obtained a manufacturing and marketing license for the Force line. Force was co-founded by Sven A. Behrendt (ex-MMI) and Max E. Loesel (ex-Motorola).

The VME bus is an 8/16/32-bit bus standard that is a subset of Motorola's Versabus, specialized to meet the European DIN connector and Eurocard pcb standards. VME bus was placed in the public domain and is supported by Signetics, Mostek, and Motorola in the U.S., and by Philips in Europe.

Gould/SEL, Fort Lauderdale, Fla., introduced in early May its PowerStation 2000, which is the Convergent Tech MiniFrame. With 512 Kbytes of RAM and a 20-Mbyte Winchester disk, the PS2000 lists for $8,995, including the Unix system. Deliveries were slated to begin during the third quarter of 1984. SEL is already marketing the CT

MegaFrame, under its PS3000 label, and the CT N-GEN as the PS1000.

IBM, which has recently established an IBU called Academic Information Systems (ACIS) to address colleges, universities, and schools, will use the Genix32 operating system from National Semiconductor. Genix32, still under development, is based on 4.2BSD (Berkeley's demand-paged Unix system) and runs on the NS32032 32-bit MPU. A previous Genix version was based on 4.1BSD and ran on the NS16032. IBM is likely to use the system on the IAWS (Interim Advanced Workstation, a PC equipped with an NS32032 board, probably also from Sritek.

National Semiconductor Corp., Santa Clara, Calif., signed up TI as a second source for the NS16000/32000 MPU, which National now calls the 32000 Series. National is far behind leaders Intel and Motorola in the 16/32-bit MPU race; the second-sourcing arrangement with TI is an important plus. For TI, which gave up on its own entry (the 9900/99000) some time ago, cooperation with National offers a way back into the MPU race.

Sequent Computer, Portland, Ore., raised $7.5 million in its second financing round, closed in late April. The company, founded by a group of 16 ex-Intel staffers, is developing a Unix-system running, multiprocessor system based on the NS16032 MPU. ■

Omri Serlin, P.O. Box 1415, Los Altos, CA 94022, 415/948-4516. Mr. Serlin has served on the ANSI X.3 and on the IEEE 802 Local-Area Networks committees and has published numerous reviews in the *ACM Computing Reviews*. He holds B.S. and M.S. degrees in electrical engineering.

Mr. Serlin is the editor/publisher of *Supermicro* and the *FT Systems Newsletter*, a monthly covering developments in fault-tolerant systems. He is quoted frequently in such publications as *Forbes, Business Week, Electronic Business, and Computerworld.*

NORM DE NARDI PRESENTS:

An Industry Salute To High Technology.

The Longest Running Regional Computer Show

- A unique, one day computer showcase highlighting industry trends and product introductions.
- An opportunity to reach a base of qualified customers while simultaneously keeping abreast of the competition.
- An ideal environment for OEMs, dealers and distributors to meet with industry leaders.
- See the latest in computers, CAD workstations, graphic systems, peripherals, communication equipment, UNIX and other software development tools.
- Norm De Nardi's computer industry expertise has made the California Computer Show a most effective marketing tool.
- An extensive promotional effort including trade journal ads, direct mailings and show publicity insures industry awareness and participation.

289 S. San Antonio Rd., Suite 204, Los Altos, CA 94022 (415) 941-8440

The Palo Alto Hyatt Hotel • December 6, 1984 • 12 P.M.–6 P.M.

Please circle Ad No. 54 on inquiry card.

UNIX™ HORSEPOWER!

There are a lot of UNIX based systems on the market today claiming to be "SUPERMICROS". But do they really have what it takes to run multi-user UNIX well? The IBC ENSIGN™ does and here's why:

FAST MEMORY: No computer running at **any** clock speed can run faster than it's overall memory design. The ENSIGN has up to 8MB of 120nsec memory with dual bit error correction. With IBC's proprietary memory management, **all** of this memory runs with no wait states as fast as the 68000 CPU will go. Compare this to other systems running only small cache memories at full speed. Other multiple user systems cannot load all their programs into a small cache memory. Their systems slow down considerably under a heavy multi-user load.

INTELLIGENT SERIAL I/O CONTROLLER: Even the fastest CPU will slow down when it's trying to handle interruptions from multiple on-line users. The ENSIGN provides slave serial I/O CPU's **and** FIFO buffering for both input and output. The result is the ENSIGN's ability to support up to 32 users, with heavy serial I/O demand, while leaving the main 68000 CPU free to run with little serial I/O overhead.

INTELLIGENT DISK CONTROLLER AND HIGH PERFORMANCE DISK DRIVES: The ENSIGN has a slave CPU to handle all disk operations, **plus** 16K of disk buffering. IBC's proprietary disk DMA allows high speed data transfer to main memory without slowing down the main CPU. Further, the ENSIGN supports SMD type 8" hard disks with much faster seek times and transfer rates than 5¼" hard disks usually found in personal desk top computers.

THE RESULTS: The IBC ENSIGN runs multi-user UNIX at performance levels not attainable by other supermicros.

Call IBC and get a copy of IBC's multi-user benchmarks—benchmarks that test 8 users running large CPU programs, with heavy disk I/O **and** heavy serial I/O simultaneously. You'll find that nothing can compare to the ENSIGN.

If you want to run multi-user UNIX on a high performance system with up to 32 users, 8MB memory, and over 1,000MB disk storage, see the IBC ENSIGN.

Outside The USA
IBC/Integrated Business Computers
21621 Nordhoff Street
Chatsworth, CA 91311
(818) 882-9007
Telex No. 215349

Within The USA
IBC/ DISTRIBUTION
1140 36th Street Suite 212
Ogden, Utah 84403
(801) 621-2294

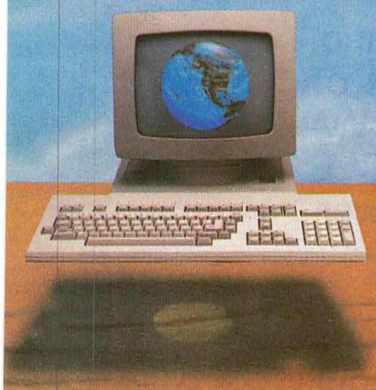
Dealers— Please circle Ad No. 18 on inquiry card.

End Users— Please circle Ad No. 60 on inquiry card.

Unix-based systems, particularly at the microcomputer level, represent a land of opportunity for graphics vendors.

MAINFRAME GRAPHICS
HEADS THIS WAY

BY JAMES WARNER AND GEOFFREY SCOTT



Until recently, Unix system users have been in the uncomfortable position of occupying a kind of graphics no man's land. To the "East," mainframe-caliber graphics systems have been used to produce everything from presentation-quality charts for corporate board meetings to three-dimensional models created interactively by design engineers.

Meanwhile, to the "West," software developed for personal computers, particularly the IBM PC and its compatibles, is producing rudimentary business graphics. The Apple Macintosh and Lisa also fit this category, offering "paint box" and windowing capabilities to the end-user and program developer alike.

For the future, in what direction should the Unix system user look? As spectacular as the advances have been for personal computer graphics, we think the major developments are being played out at the mainframe and "supermini"

CREATIVE CONNECTIONS

Making creative connections for UNIX developers and UNIX users has been the essence of our business since 1977.

We start with the UNIX system itself, adding functionality, performance, robustness and ease of use to produce our UNITY operating system. We then add programming tools and our CHRONICLE business applications family to provide a powerful environment for UNIX-based applications.

To our superior products, we add a superior commitment to customer satisfaction. We treat our customers as our UNIX technology partners. For all our clients - manufacturers, OEMs, VARs and direct accounts - we provide product plus custom development, UNIX components plus enhancements, training plus the industry's finest support. At HCR, our ultimate product is...

PARTNERSHIPS IN UNIXCELLENCE.

Human
Computing
Resources
Corporation

HCR

10 St. Mary St., Toronto, Ontario, Canada, M4Y 1P9 Tel. (416) 922-1937 Telex 06-218072

Please circle Ad No. 16 on inquiry card.

levels. The significant migration is downward.

The problem, at least as far as it affects the Unix system environment, is that personal computer graphics software has been designed with the limitations of the hardware in mind. Hard-copy output is generally draft quality, not presentation quality. The screen display —typically on a low-cost monitor —is marked by low resolution. A limited number of colors are available and displayable at one time.

By contrast, the mainframe/supermini realm has already met the demand for high-quality business graphics, fully interactive design graphics, and, in particular, development tools that can accommodate the idiosyncrasies of a number of advanced display devices, including CRTs, pen-plotters, and laser printers.

Just as in the mainframe arena, graphics development tools will play a major role in the Unix system environment. We will look at these tools, their place in the graphics spectrum, the role and reality of standards, and the advantages of developing graphics applications for Unix system-based workstations.

WHY GRAPHICS?

The demand for graphics is coming directly from the end-user. Consider, for example, the use of computerized graphics in business. A bar chart is not a substitute for a detailed tabulated report—be it sales projections, monthly income statements, or marketing quotas. But it is an excellent tool for showing trends, summarizing results, and illustrating points in a presentation.

The quality and format of such computer-produced charts can be a benefit in itself. A recent study from the Wharton School of Business, for example, demonstrated that presentations utilizing graphics on overhead projectors can markedly affect the course and outcome of both executive and training meetings.

According to the Wharton findings, computer-generated presentation graphics enable participants to arrive at decisions far more quickly than do conventional presentation techniques such as blackboards or flip charts. As a result, the length of meetings can be reduced by 25 percent or more. The study also indicates that presenters employing transparencies are perceived as more professional and effective: These speakers win their point more than two-thirds of the time.

THE SPECTRUM OF COMPUTER GRAPHICS

Graphics software now emerging in the Unix system marketplace falls into three broad categories, each offering the user a particular approach to creating graphics while embodying a particular set of price/performance considerations.

There is a tradeoff at work here. More dedication to a particular operating environment and to a specific application generally provides more ease of use; less dedication provides more flexibility.

Turnkey packages are designed to meet the requirements of a precisely defined problem and are locked into a prescribed hardware environment. Because numerous applications can benefit from graphics output, turnkey packages address a great variety of needs, from artist/design systems to presentation graphics, and from medical diagnostics to electronic component testing.

As the name implies, a turnkey product is analogous to a new car purchased off the dealer's floor—a fully functional system that works from day one while requiring little support from the data processing department. On the other hand, inflexibility can be a problem. A turnkey system lacking certain features also lacks the means to add them. Users must therefore compromise on their "wish list" or add a second package to fill the gap.

Application packages are turnkey systems without the hardware: They provide a specific solution to a specific problem with similar ease of use and limited flexibility. Unlike turnkey packages, however, most application systems are device independent; they are not locked into a particular output device. Moreover, they are generally machine independent as well, able to run on a variety of computers and operating environments.

ONE EXAMPLE

An example is Handle Office Automation Software (Handle Corp., Tahoe City, Calif.), which can run on a variety of Unix-based systems and which provides a range of integrated office tools including a business graphics module. The system enables an executive to access financial information from a database and display it as a pie chart, bar graph, area chart, or line chart. The graphics can be created on a multipen-plotter or on a dot-matrix or letter-quality printer.

Graphics tools packages are subroutine libraries that help programmers build custom graphics applications. Analogous to a carpenter's hammer and saw, these packages require a programmer's skill to use and involve a time commitment for program development. The payoff for this investment is a great deal of flexibility—tools packages can be used to develop virtually any graphics application while supporting most major output devices.

With some packages, such as Precision Visuals' DI-3000 and GK-2000, the set of subroutines is designed independent of any application and can be used by the programmer to display dollars, ohms, meters, or ergs. In other cases, the subroutines are developed with a particular type of application in mind—business graphics, for example, or contour mapping. As a result, these more tailored packages require less programming effort than their more generalized counterparts.

The principal advantage of graphics software tools is that they enable graphics programs to be developed independent of any particular output device. The result is faster programming turnaround time, much less program maintenance, and eliminating the risk of application obsolescence when new hardware technologies are introduced.

DEVICE INDEPENDENCE

One way to view device independence is to think of it as taking the Unix system `termcap` facility several steps further. The `termcap` utility permits the Unix system, and applications running under it, to "know" what kind of alphanumeric terminal is being addressed. As a result, applications requiring full screen addressing —word-processing programs and spreadsheet packages, for example —can rely on the operating system to write to any part of the CRT display. Moreover, this full-screen addressing is accomplished no matter which hardware manufacturer's terminal or terminal model is used —so long as `termcap` recognizes it.

The `termcap` facility is necessary because there is no fully adapted standard for how an alphanumeric terminal is addressed (although the DEC VT-100 has been widely emulated). Therefore, the escape sequence necessary to position the cursor at line 10, column 15, may bear little resemblance from one terminal to another. Likewise, clearing the screen, supporting special function keys, and enabling screen attributes (underline, half-brightness) are all accomplished differently on different devices.

In the graphics arena, the incompatibility problem approaches nightmarish proportions. Graphics programmers can specify line thickness, line style, text style, and text size. They can fill polygon interiors with solid colors or patterns. They can often select from a massive palette of colors (16.7 million is not uncommon), displaying thousands of them at a time. On more advanced displays, a part of the full image can be treated as a separate "segment" that can be individually manipulated on the screen—erased, re-drawn, and re-scaled—apart from the rest of the image.

Moreover, screens are not the only output devices. Graphics programmers must also contend with pen-plotters that have far fewer colors available than a display screen, and with input devices —such as graphics tablets, mice, and joy sticks.

No single graphics standard will dominate the graphics industry —each standard will have its own constituency.

So long as an application is developed with only a single hardware configuration in mind, this bewildering array of variables can be mastered (although not easily). The programmer studies the equipment manuals, learns the necessary escape sequences, and simulates in software those functions not provided in hardware. In other words, the low-level interface that manages every device must be created from scratch.

Graphics tools packages provide an alternative, permitting the programmer to address a single "virtual device" that simulates the combined features found on CRT displays, plotters, film recorders, and other devices. A software module called a "device driver" then translates these generic commands directed at the virtual device into code meaningful to the targeted physical device. An installation acquiring new devices simply adds the appropriate device driver. Except for minimal fine-tuning, applications require no modification.

STANDARDIZING THE PROGRAMMER'S INTERFACE

Over the past several years, efforts have been made to standardize the programmer interface—that is, the programmer's "view" of the virtual device. Two of these standards, Core and GKS, are now vying for prominence. (See the article elsewhere in this issue on the merits of GKS.) The differences between these two standards are hardly major—indeed, there is usually no way to tell which standard was used to produce a given picture. (An exception is a three-dimensional image—GKS is only a two-dimensional standard.)

For example, let's consider how a plot on an X-Y graph might be generated using DI-3000, Precision Visual's Core-based package, and using GK-2000, based on the GKS standard. In both cases, the plot is specified in terms of a world coordinate system—that is, a coordinate system whose units reflect the actual database to be graphed. If an X-Y plot is to show company growth in millions over the past year, the database might, for example, extend from 0 to 100 on the Y axis (for a scale of $0 to $100 million); and from

> **In the graphics field, the incompatibility problem approaches nightmarish proportions.**

1 to 12 on the X axis (for the 12 months of the year). Using DI-3000, a line is specified from a "current position" to a new position on the world coordinate system.

`call jmove (1,5)` would move the "current position" to the 1,5 coordinate point. `call jdraw (X, Y)` would extend a line from the current position to the new position on the world coordinate system. Successive extensions are accomplished by changing the X and Y values and re-calling `jdraw` again.

Under GK-2000, the line is drawn with a single call: `call gpl (2, ptx,pty)`, where `2` is the number of points to be connected, and `ptx` and `pty` are simple arrays containing the coordinate pairs for each point. Successive extensions are accomplished by specifying a greater number of points in the first parameter and adding more coordinate points to the variables making up the other two parameters.

Both tools packages then take the application-specific world coordinates and transform them first into device - independent coordinates, and finally into device-specific coordinates using the appropriate device driver. This mapping from one coordinate system to another can be altered by the programmer who may, for example, want to change the aspect ratio of the final hard-copy image. Essentially, however, the process is automatic and transparent to the programmer.

It is Precision Visuals' view that no single standard will dominate the graphics industry. Instead, like different programming languages, each standard will have its own constituency: GKS for simpler two-dimensional applications and those that must show different views of the same image on multiple devices; Core for higher-performance applications, particularly those using three-dimensional models; and, eventually, PHIGS to support very high-performance displays requiring intensive use of segmentation —highly interactive computer-aided engineering would be a potential user here.

The problem with standards is that they inherently lag several years behind the technology they are trying to address. In an age where new graphics equipment is introduced monthly—the result of heated competition among vendors —the specification of standards remains painstakingly slow—a product of committees, diplomacy, and compromise. Thus, standards must be considered as only a part of the strategy for success in graphics programming.

UNIX SYSTEM-BASED WORKSTATIONS

So where does all this leave Unix system installations and programmers? In excellent shape. The Unix system environment is inheriting a graphics technology that is far more mature than it was only a few years ago. Some vendors have not only refined their products based on suggestions from the marketplace, but they have recognized that many of the programmers using these packages have little graphics experience. As a result, documentation and vendor support have improved significantly; they are no longer geared strictly for the "graphics guru."

But by the same token, Unix-based systems, particularly at the microcomputer level, represent a land of opportunity for graphics vendors. Configured as local-area networks (LANs), these systems can provide faster response time by taking advantage of the dedicated power of a single microprocessor chip.

A graphics metafile system is of particular benfit to users linked in a LAN. The system, which is device- and machine-independent, permits a "snapshot" of the graphic image to be stored on the workstation or on a shared disk. Users of other workstations equipped with a "metafile translator" can readily view the image (although they can't interact with it).

A metafile system is particularly valuable in business and design environments, where graphics are created on one or two workstations to be viewed by others. Without a metafile, the complete application would have to reside at each workstation, consuming disk space and reducing response time.

Unix system-based workstations also represent a way to put graphics in the hands of more people by removing the technology from the confines of the computer room. From their desktops, users will be able to access a variety of devices using a device-independent graphics system to generate everything from high-resolution transparencies on a film recorder to large-format maps on a flatbed plotter to voluminous runs on an electronic laser printer.

The graphics software technology discussed here may have originated on larger machines—but its promise may well be fulfilled at the Unix system level. From all projections, it looks like it's going to be a happy marriage. ■

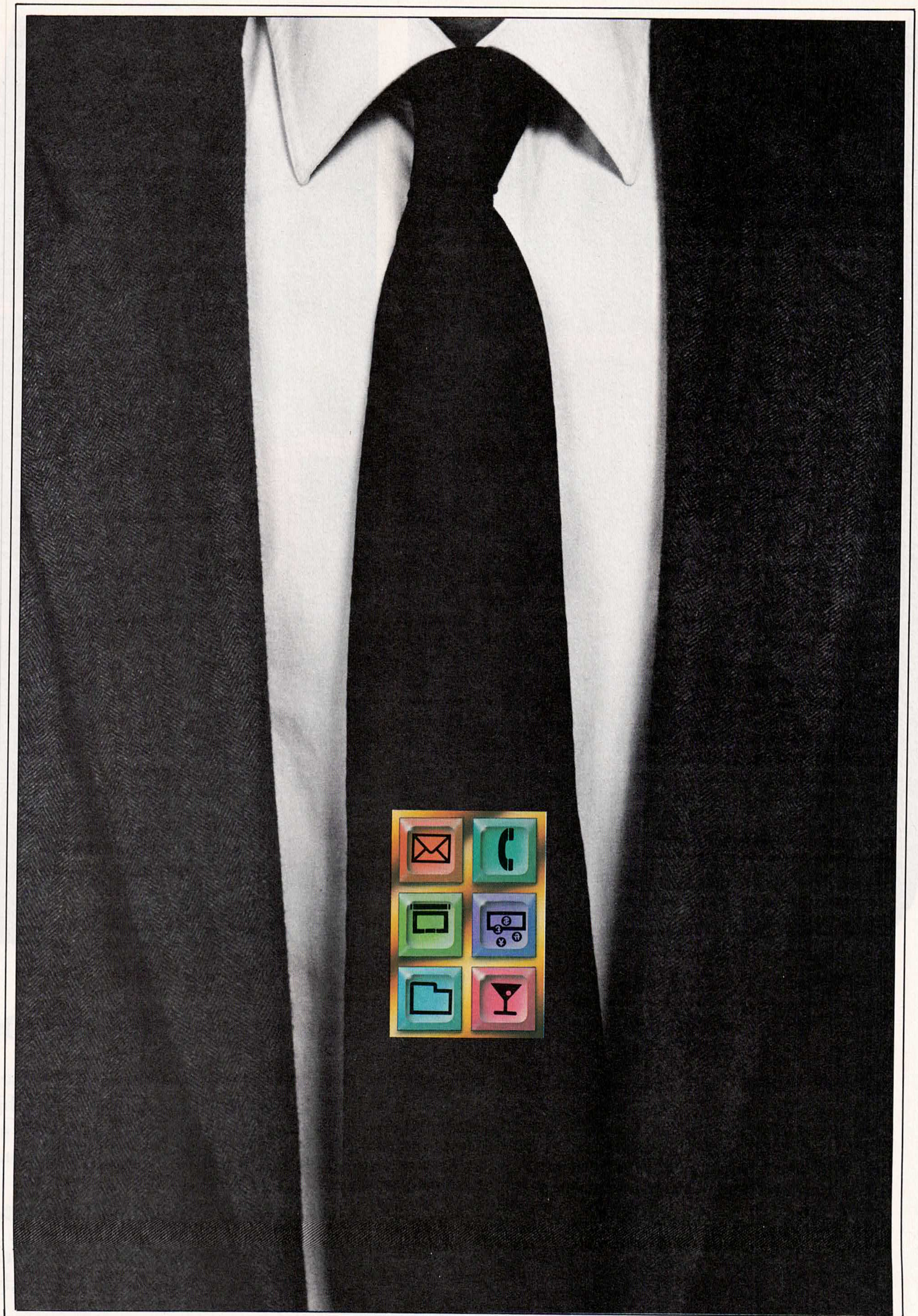
Jim Warner is president and CEO of Precision Visuals (Boulder, Colo.). Precision Visuals' licensees include such Unix system-based hardware manufacturers as Perkin-Elmer, NCR, and Zilog. Geoff Scott is corporate product manager for Precision Visuals and oversees GK-2000, the firm's implementation of GKS. He is a frequent speaker at seminars and trade shows throughout the country and holds a B.A. in mathematics from UC Irvine.

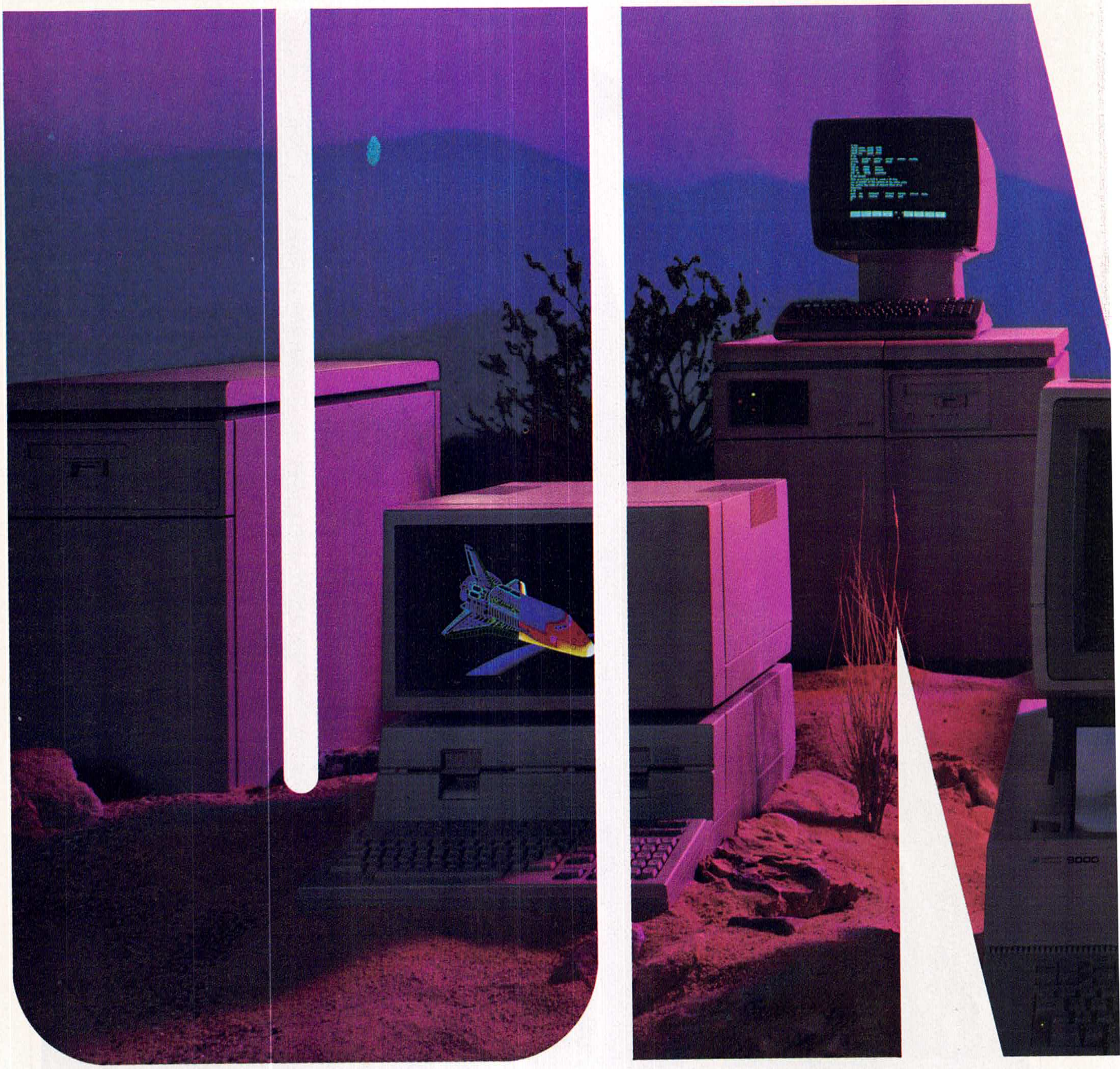
Corporate identity programs are well known to graphics designers, but this approach can also be applied to screen designing.

As computer systems become more sophisticated, a crucial factor in designing them to be friendly, comprehensible, and effective is the quality of communication between user and machine. I propose that this communication can be understood to take place through three faces: *outerfaces* (presentational and analytic displays), *innerfaces* (user-machine command/control and documentation

ICONIC INTERFACES *and* CORPORATE IDENTITY

BY AARON MARCUS





realized.

to answer questions. Working with both end-users and OEMs, we'll find the best solution for any particular application.

Sound interesting? Call your local HP sales office right now about the HP-UX operating system. Or write to Hewlett-Packard, Attn. Pat Welch, Dept.102201, 19447 Pruneridge Ave., Cupertino, CA 95014. In Europe, contact Henk van Lammeren, Hewlett-Packard, Nederland B.V., Dept. 102194, P.O. Box 529, 1180 AM Amstelveen, The Netherlands.

Productivity. Not promises.

HEWLETT PACKARD

BDO2421

dialogue), and *interfaces* (programming and maintenance environments) [Marcus, 1983].

Effective systems imply low cost, sophisticated functionality, friendly interfaces, and good service [Foley and Van Dam, 1982]. Included in all of these features is the quality of communication. This affects the long-term cost of the system by reducing the nonproductive time during training and use of a computer system by providing the means for the user to take advantage of the system's functional power, as well as by enhancing service quality when users easily understand how to achieve their goals. Quality communication generally can be achieved in a task when the following conditions are present [Marcus, 1982]:

> **Human computer screen interfaces are often an unorganized series of frames that are never seen as an entity in themselves.**

Simplicity: Major parts of the system are few in number or are placed in a hierarchy.

Clarity: Parts of the system are evident.

Familiarity: Parts of the system remind the user of things known already.

Integrity: The system is an ordered sum of its parts.

Consistency: What the user knows of one part helps in other parts of the system.

Reliability: The system responds to the user in a trustworthy manner.

Responsiveness: The interactive replies of the system are quick, polite, and helpful.

SEEN AS A WHOLE

Human computer screen interfaces are often an unorganized series of frames that are never seen as an entity in themselves. Once they are viewed as a whole, it becomes apparent that one can speak of the "corporate identity" of an interface; that is, the consistent, articulatable algorithm of typography, symbolism, color, spatial layout, and approach to sequencing/animation that characterize a particular system.

The term *corporate identity program* is well known to the graphic design field, which has applied it to other forms of complex communication. This approach can now be extended to the design of screens, especially for high-resolution, iconic, multi-window interfaces.

The systematic appearance of typography, symbolism, color, layout, and sequencing constitutes a visible language. Corporate identity programs establish the system and record it within graphic design standards manuals, the source code for the graphic design algorithm. Design manuals are well understood in the graphic design field; in fact, a design manual for design manuals has even been written [Chaparos, 1979]. Without reproducing one, the main points of a typical manual's conventions can be delineated. Those that are relevant to high-resolution, iconic, multi-window screen design appear in the subsections below.

TYPOGRAPHY

Typical recommendations limit the variations of typefaces to one or two type families. Many of the leading corporate graphics manuals in the business community have traditionally emphasized a few well-documented type fonts. These have proven their legibility, their flexibility of display in small text sizes and large display sizes, and their availability in many styles.

The following typefaces are among the more popular: Baskerville, Caslon, Garamond, Helvetica, Times Roman, and Univers. Two type families are sometimes combined. A typical combination of two faces might be Helvetica and Times Roman or Univers and Times Roman; that is, a contrast between sans-serif and serif typefaces. Type size is also limited; three (or a maximum of five) suffice for all distinctions such as footnotes, titles, headers, and figure captions. Wherever possible, simple direct changes in size of 2:1 of 3:1 are used to distinguish levels of titling sizes.

Typography appears in columns, usually one to three columns per screen, with 40-60 characters per column. Upper- and lower-case letters should be used wherever possible because they are more legible. Their legibility is due to greater differentiation of letter shapes that contribute to the overall shape of the word.

Reading is accomplished by recognition of word shape as much as the identification of individual letters. All capital letter settings of text may be used for brief titles or for emphasis, but their extensive use can slow reading speed by as much as 13 percent [Chaparos, 1979]. The space between words should be approximately 1/4 the width of the capital "M" for variable-width fonts, while the space between lines may vary according to the design of the type font.

Generally, sans-serif letter forms, like those of Helvetica, require a small amount of extra space between base lines of text. For maximum legibility, the line spacing should produce spaces between lines that are greater than the small amount of space between words. The lines themselves may be non-justified (ragged right); there is no noticeable difference in their legibility [Rehe, 1974].

The differences in line length of nonjustified text can contribute to the visual interest of the screen, but care must be taken to avoid strong, recognizable shapes to the pattern of ends of lines. The columns of text themselves should be separated by a width at least equal to two word spaces for variable-width characters. For many screens, a layout of three columns per screen for text settings or one wide column on the right with a narrow column on the left for marginalia are useful conventions.

SYMBOLISM AND COLOR

According to the language of semiotics [Eco, 1976], signs may be iconic (representational) and symbolic (abstract). The term *symbolism* is used here to refer loosely to all nonverbal signs: illustrations, photographs, diagrams, pictograms, etc. The concept of corporate graphics implies that all images are given attention to their unique communication needs, but at the same time images are adjusted to produce a visual consistency throughout the system.

This can by achieved by the use of constant scale, limited size variations, the orientation of figures with respect to text, the palette of colors, limited variation of line weights, and the treatment of the borders to figures or pictograms. These visual themes help to establish recognizability, clarity, and consistency, just as verbal or linguistic techniques are applied to the text to promote simplicity, clarity, familiarity, integrity, and consistency.

The use of color in computer graphics has often emphasized too many colors, even when only a few are available. The corporate graphics approach to color emphasizes the selection of a limited set of well-chosen colors that meet the criteria of production, the needs of the content, and the preferences or limitations of the viewers. These colors are used repetitively to maintain consistency across content areas and across different display media.

The corporate colors are a primary feature in achieving corporate recognizability. If the set of colors is sufficiently large, the designer can use it in many forms of informational as well as marketing graphics. Some companies choose color schemes that are very simple, while others choose nonstandard, more subtle, muted (low chroma) colors as their unique color identity.

LAYOUT

The approach to spatial organization characterizing corporate graphic design derives from the European constructivist artistic movement of the early twentieth century. As it found its way into the formative years of the international style of corporate graphic design during the 1950s, the approach stressed an articulate, systematic method in assigning areas for text and illustration as well as the background field or format.

Whenever possible, visual references were made to a series of strong, easily recognizable proportions that have been used since classical time, e.g., the square and the golden rectangle.

When multiple columns of text or images are used, the designer can create more interesting and lively compositions of text and illustrations. A typical layout might propose three equal-width columns or one narrow and one wide column. There is typically a large space at the top for important titles or illustrations.

The basis for regulating the varied groupings of text and images is the layout grid. This is a series of horizontal and vertical lines that define certain areas of the screen for the positioning of titles, text, or illustrations. The grid also determines the extent or size of these three elements. In this way it builds in a visual consistency to every possible layout [Mueller-Brockman, 1981].

Traditionally, animation or kinetic movement has not played a major role in corporate design programs because the access to control has been lacking in display media. Where temporal design is possible, the corporate graphics approach again stresses simple, clear, modular temporal constructs. This might apply, for example, to the regular appearance or disappearance of items or the overall dramatic narrative.

Several office automation systems have appeared that display characteristics of the corporate identity approach to the design of the human/machine interface with

Color use in computer graphics has often emphasized too many colors, even when only a few are available.

varying degress of completion. These 16-bit microprocessor systems are supported by high-resolution bit-mapped screens. This article examines three black-and-white systems—the Xerox Star, the Apple Lisa, and the Intran Metaform systems.

THE XEROX STAR

The Xerox Star system appeared in 1981. Based on published documents, a number of the fundamental design principles for the Star interface are known [Smith, 1982]. In creating a system that promoted familiarity and friendliness through the simplicity, coherency, and consistency of its interface, the designers sought to develop a conceptual model of the system in the users' minds that was communicated through the visual features of the interface.

The corporate approach to communication strives for exactly this method—to embody functions and features into an easily grasped and easily learned system. The Xerox team developed the metaphor of the desktop and carried it through in the visualization of all functions and activities. This was accomplished through visual objects called *icons*, which have properties that are summarized for the user in *property sheets*, which may be easily displayed and edited.

By establishing global commands with consistent meanings throughout the interface, the designers were able to develop another systematic aspect of the display. Several other methods are used to establish consistency. Editing is accomplished through a single paradigm of operation, whether one is editing text, graphics, files, the desktop itself, property sheets, or even programs. Retrieving information always takes place through the paradigm of databases, and creating new objects is always achieved through the paradigm of copying.

By reducing the number of parts in the system and minimizing redundancy, the designers were able to achieve large-scale simplicity. One important aspect of this simplicity is mode-less interaction. For example, the keys of the main keyboard are used only for typing, and special keys are used solely for functions. Clearly separating these keys reduces the short-term memory requirements of the user.

The Star organizes the desktop screen into a space for 154 icons centered on fixed locations, each one-inch square on 72 x 72 pixels. The squarish icons tend to fill up their allotted space and use small changes in their edges to communicate the different meanings.

For example, a small corner tipped down on a page represents a file, while a small extension of the top edge of the folder represents a collection of files. Screen buttons differ in the drawing of their corners: Sharp, square corners represent items of data of characteristics on property sheets, while rounded corners represent screen buttons and titling within the top band of windows.

The Apple Lisa followed in many of the directions begun by Xerox's Star.

The Star is a black-and-white system. Color issues concern how white, black, and gray are used. The desktop itself is approximately a 50 percent gray. The dark outlines of the icons and the dark outline of windows show up clearly against it. The area of the icons and windows is white to contrast with the background.

The top border of sub-windows appears as higher-resolution gray to distinguish itself from the two other screen elements—window contents and desktop. Black is used for reversing screen buttons or icons that have been selected; the selected item reverses within the old area, and a thin white rule surrounds it. All of these details represent a particular system-wide approach to the treatment of color.

The Star's icons appear in a fixed grid layout 14 units wide by 11 units high. Another fixed grid aspect of the interface is the top border of a window: It must accommodate two rows of window titles and local screen buttons. The remaining grid features are the right side and button borders of windows—they must accommodate screen buttons for scrolling within the window.

A particularly characteristic feature of the systematic approach in the Star is the standard object-command or noun-verb sequence of selection. Entities appearing on the screen are acted upon or selected. Selecting entities is a primary goal in the user's conceptual model; the user then selects the action or change of state to be effected.

The Xerox Star was a pioneering achievement in the corporate identity of interface design. It represented state-of-the-art, object-oriented screen manipulation in a high-resolution system. Of considerable importance to this discussion is the effort that was undertaken to design not only the algorithms that support the system but also the manner of its representation in a systematic form. The conventions established by the Star have already begun to influence later systems.

THE APPLE LISA

Following upon the Star's approach, the Apple Lisa continued many of the same directions when it appeared in January 1983 [Williams, 1983]. Like the Star, the Lisa offers a selection of typefaces, including serif, sans-serif, fixed-width, and variable-width letters, and a half-dozen variations in type style, including shadowed letters.

To utilize many of these in the interface itself would not represent the corporate identity approach, and the Lisa wisely chooses to use primarily a single size of modern sans-serif letters in upper and lower case, with selected screen buttons appearing in reversed type.

Some of the Xerox Star's development staff came to work on the Lisa. This has influenced the Lisa to adopt the Star's desktop concept as a unifying metaphor. The icons are, in some cases, more highly representational and detailed than in the Star—for instance, in the garbage can that (a little confusingly) represents a wastebasket for unneeded files, indentations appear along the sides of the can as well as a handle on the top to raise the lid.

These details suggest the beginning of more illustrative or anecdotal icons for more personalized workstations. In the Lisa, the icons appear with verbal equivalents below them. The windows themselves and the sub-menus that descend from the menu bar across the top of the screen show slight drop shadows that begin to indicate an implied three-dimensional structure to the flat workspace of the desktop.

A typical difference between the two systems' interface styles can be seen in the Lisa's stronger window scrolling arrows. In the Star they are drawn with three thin lines, while Lisa's are thicker, with an outline and a drop shadow. As mentioned before, the symbolism in the Lisa is richer and more representational from a visual communication point of view. The symbol set also begins to show some weakness as a completely designed system. For example, the reversed titles of windows have unique ornamental additional lines to their sides.

Color considerations in the Lisa are similar to those in the Star. The Lisa also uses several gray value textures to distinguish the primary desktop, the windows, and window borders, but the exact grays are somewhat differently disposed in comparison with the Star. Of special note in the Lisa are the gray right side, gray bottom, and the white top border of windows.

The Apple Lisa permits a relatively unorganized location of icons on the desktop. One strong gridded feature of its screen design, however, is the menu bar that appears on the top of the screen. Submenus "pop down" (rather than up) from it temporarily and may obscure material below. As formulated by both the Star and Lisa, the metaphor of the desktop does not include any corporate standard for desk organization or of windows.

The noun-verb selection paradigm of the Star has been incorporated into the Lisa: The user selects objects, then the transformations intended for those objects. As for the Star, the windows, icons, and sub-menus are intended to appear and disappear instantaneously; if not, this is treated as a deficiency to be hidden with whatever means available.

In the Lisa, the window that appears when an icon is pressed zooms up from the icon position in a noticeable transition. Many computer graphics display systems assume that faster is always better. However, a communication-oriented approach might suggest for novice or occasional users, especially during training periods, that dissolves, fades, wipes, and zooms might be of value in communicating the meaning of change.

The Apple Lisa system represents a substantial refinement of the ideas introduced in the Star system. Earlier this year, Apple introduced the Macintosh. This low-cost version of the Lisa continues many of its graphic features and makes evident the value of corporate graphic standards for interface design.

THE INTRAN METAFORM SYSTEM

The Metaform system from Intran represents a modest development scale in comparison to the 30 work-years of development on the Star's interface and the 200 work-years claimed for the development of the Apple Lisa. Interface design for the Metaform system consumed only a few work-years by the time the system was announced in December 1982.

Metaform provides special-purpose software to accomplish forms design and editing for the Xerox 9700 laser printer. The software resides on a Perq high-resolution display system. Because the author's firm of system-oriented graphic designers was responsible for the graphic design features of the interface, the final product possessed a corporate design approach from the very beginning, when they proposed prototype sketches for typography and screen layout.

The design scheme calls for most screen displays to accommodate Univers typography in a single size for all system messages. Reversed type is used for selected buttons. Multiple lines of type are always stacked flush left, ragged right, making the scanning of lists of buttons and other information easier.

Iconic light button symbolism for Metaform took a special direction: The corporate graphics convention for primary module icons is to use narrative images that tell a brief story about what occurs within a module of the system. These large icons are somewhat like illuminated letters in medieval manuscripts.

They indicate appropriate cursors and explain to the uninitiated what will happen in the modules. Similar to the Star and the Lisa, Metaform uses different cursors to signal processes and states within the system to the user. Pen points, brushes, pencil points, paper clips, and pointing hands all contribute to explaining and differentiating activities.

Color distinctions in Metaform are similar to those of the Star and the Lisa. Gray values are used to distinguish different areas such as light button fields from the work area. Gray allows both white and black to be used for highlighting and lowlighting.

In the original metaform design (actual implementation differs slightly in some features), the screen is a tiled surface of regular areas. The menu appears at the right, while the work area appears at the left. (This could be easily reversed for left-handed users.) By locating screen components in a regular place, users can learn the layout of the system and its features more quickly. The forms designed within the work area can be of any organization depending upon the particular function of the form.

CONCLUSIONS

Although iconic interface design is just beginning to be introduced into the commercial market, the field is rapidly expanding as new systems based on 16-bit processor technology and high-resolution displays are developed. The computer graphics industry already has some strong, clear prototypes upon which to base new designs. In doing so, several important issues arise that will be explored in the next stages of graphically designed interfaces.

What is an appropriate screen format: square, vertical rectangle, or horizontal rectangle? What proportions should a rectangular screen possess?

Where should menus ideally be located: free floating, at the top, right edge, or at the bottom?

Should window organization be free and unorganized, or should some default layouts be imposed to aid comprehension, memory, and user efficiency? What size, shape, and borders should windows have?

Iconic interface design is rapidly expanding, thanks to 16-bit processor technology and high-resolution displays.

Should function buttons, objects, and other illustrative imagery be very representational or abstract?

What is the ideal typeface for an interface: positive, reversed; serif, sans-serif; one size, several sizes?

How should screen elements appear and disappear: quickly or slowly? How can dissolves, wipes, cuts, zooms, and other cinematic techniques be effectively incorporated?

How can color be used effectively to enable users to learn more quickly and to be more efficient in performing their tasks?

The answers to many of these questions will emerge in the iconic interfaces that will be designed in the next few years. Some of these systems will help establish the conventions for the corporate design of systems in areas other than office automation—such as CAD/CAM and computer-aided learning.

Aaron Marcus is the principal of Aaron Marcus and Associates, Berkeley, Calif., an information-oriented graphic design firm specializing in effective computer graphics display. He was recently a staff scientist in the Computer Science and Mathematics Department at the Lawrence Berkeley Laboratory. Mr. Marcus received his B.A. in physics from Princeton and his B.F.A. and M.F.A. in graphic design from the Yale University Art School. He is a member of Siggraph, Ylem Computer Art Society, Sigma Xi, the American Institute of Graphic Arts, and the Semiotic Society of America.

References

Marcus, Aaron: "Graphic Design for Computer Graphics" Proc. Intergraphics-83, session B5-2, Tokyo, Japan, 1983, pp. 1-9; and IEEE *Computer Graphics and Applications,* Vol. 3, No. 4, 1983, pp. 63-68.

Foley, James D., Van Dam, Andres: *Fundamentals of Interactive Computer Graphics,* Addison-Wesley, Reading, MA, 1982.

Eco, Umberto: *A Theory of Semiotics,* Indiana University Press, Bloomington, IN, 1976; and Marcus, Aaron: "An Introduction to the Visual Syntax of Concrete Poetry," *Visible Language,* Vol. 8, No. 4, Autumn 1974, pp. 333-360.

Marcus, Aaron: "Designing the Face of An Interface," IEEE *Computer Graphics and Applications,* Vol. 2, No. 1, January 1982, pp. 23-29.

Chaparos, Ann: *Notes for a Federal Graphic Design Manual,* Chaparos Productions, 1112 6th Street, NW, Washington, DC 20001, 1979.

Rehe, Rolf F.: *Typography: How to make it Most Legible,* Design Research International, PO Box 27, Carmel, IN 46032, 1974.

Mueller-Brockmen, Josef: *Grid Systems in Graphic Design,* Verlag Arthur Niggii, Niederteufen, 1981.

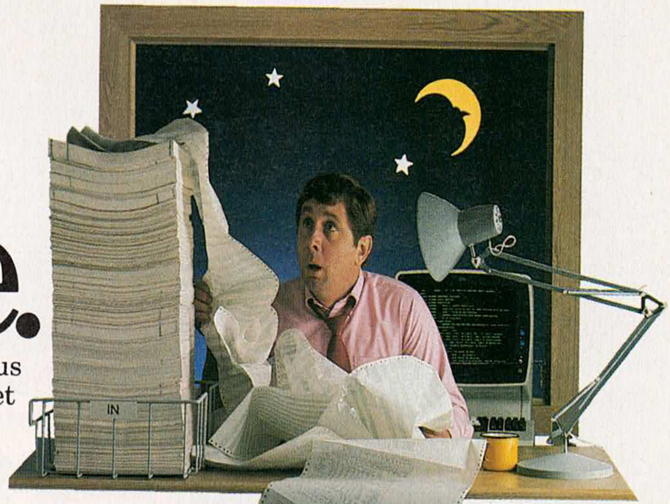
Smith, David Canfield, et al.: "Designing the Star User Interface," *Byte,* Vol.7, No.4, April 1982, pp. 242-282.

Williams, Gregg: "The Lisa Computer System," *Byte,* Vol. 8, No. 2, February 1983, pp. 33-50.

Williams, Gregg: "The Apple MacIntosh Computer," *Byte,* Vol.9, No. 2, February 1984, pp.30-40 ff.

This article is based on a paper presented at the 1984 Conference of the National Computer Graphics Association.

See Software.



Dick is a programmer. Dick gets caught in tedious testing. Drowned in printout. Never to know the sweet smell of success.

Jane has the perfect programming environment. Because it speeds applications to market. It's specially tailored to the IBM® PC.

It's from Micro Focus.

See Software Sell.

Writing programs is fast with integrated PERSONAL COBOL.™ Jane gets powerful editing. Syntax checking. Interactive source code debugging. And forms painting. Plus support for IBM PC keyboard and display functions.

Compiling is also fast. And running too. Thanks to High Performance LEVEL II COBOL.™ With native code generation. And it's certified at the highest federal level.

Maintenance and enhancement is a snap with ANIMATOR.™ So Jane's applications are the latest. The greatest. And Jane's the fastest, most effective programmer around.

This software vendor has just gone public. Because programmers are productive. Customers are contented. And dealers are delirious. All thanks to Micro Focus.

Sell, Software, Sell!

Get on the fast track too. Run to your phone. Or fill out the coupon. Right now.

MICRO FOCUS

2465 East Bayshore Road, Suite 400
Palo Alto, CA 94303, (415) 856-4161

Micro Focus, 2465 East Bayshore Rd., Suite 400,
Palo Alto, CA 94303.

Quick, send me more information.

Name__________ Title__________

Company__________ Phone__________

Address__________

City__________ State______ Zip______

Please circle Ad No. 25 on inquiry card.

Gould... Innovation and Quality in UNIX-based systems

Our Firebreathers are scorching old performance standards.

Gould's PowerNode™ 9000 blasts through UNIX* benchmarks at 4.5 times the speed of the VAX™ 11/780. Sound impossible? Give us your real production code or benchmarks and let us prove it.

Firebreathing Performance.

Now you can run software development and production at the same time, with highly responsive performance. Tightly coupled dual processors nearly double throughput and virtual memory accommodates large programs. Hardware fixed point and floating point accelerators retain high performance in heavy number-crunching situations. The PN9000 handles mainframe jobs in a multi-user UNIX system or serves as a backend processor in a widely distributed network.

Unique UNIX Software.

Gould's own high performance UNIX-based operating system (UTX/32™)—a unique combination of Berkeley 4.2 with special Bell System V features—makes it easy for you to use your VAX-based UNIX software. This allows easy conversion from your system to the increased power of a Firebreather.

Compatible Family.

Gould's Compatibility Suite is a collection of application software packages that are compatible across the entire PowerSeries™ product line. Use C, Cobol, BASIC, or Pascal languages intermixed. This close-knit processor family offers all the advantages of a dedicated system plus the lower-cost-per-user option of sharing resources with Gould's standard networking capabilities including Ethernet™. The Firebreathers are the high end of the widest range of UNIX-based systems in the industry.

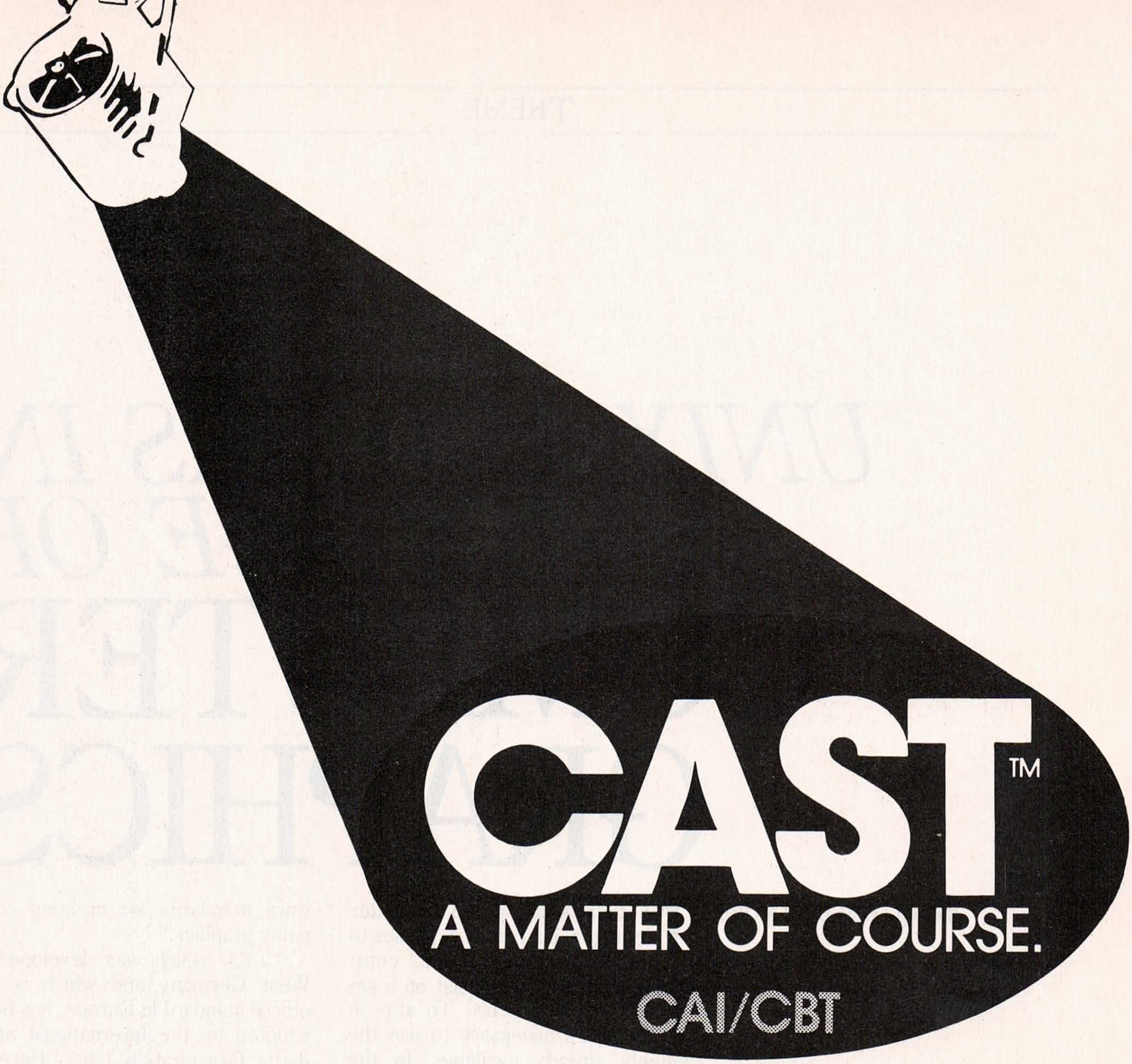
Gould's Firebreathers are scorching the UNIX market.

Gould Inc.,
Computer Systems Division
Distributed Systems Operation
6901 West Sunrise Boulevard
Ft. Lauderdale, Florida 33313
(305) 797-5459

*UNIX is a trademark of AT&T Bell Labs
™PowerNode, PowerSeries and UTX/32 are trademarks of Gould Inc.
™Ethernet is a trademark of Xerox Corporation
™VAX is a trademark of Digital Equipment Corporation

GOULD
Electronics

Please circle Ad No. 56 on inquiry card.


CAST™
A MATTER OF COURSE.
CAI/CBT

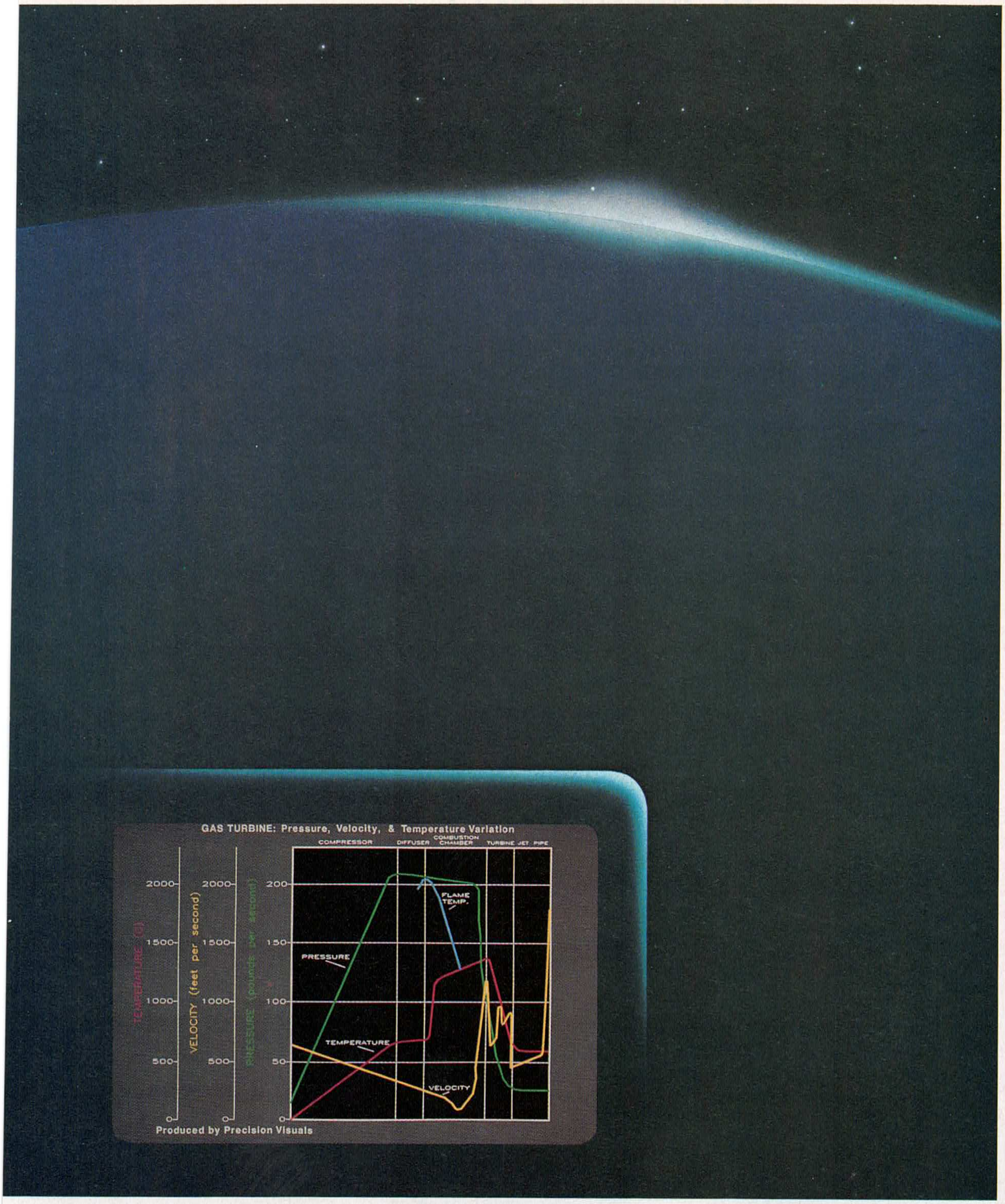
UNIX AND GKS IN A NEW AGE OF COMPUTER GRAPHICS

BY WILLIAM B. ELMORE

In the graphics standards fracas between GKS and the Core System, an advocate for GKS tells why his favorite should be the winner.

In today's fast-paced computer industry, if a company pauses to re-invent the wheel, competitors will whiz past on a set of ready-made wheels. To stay in the lead, it's necessary to use the wheels already available. In the world of computer graphics, those "wheels" consist of solid and substantial standards such as the Unix operating system and the Graphical Kernel System (GKS), a set of software standards for creating computer graphics.

GKS, which was developed in West Germany and which is the official standard in Europe, has been adopted by the International Standards Organization (ISO). Here at home, the American National Standards Institute (ANSI) has endorsed a proposal for adopting GKS as the American standard, making it the principal emerging standard in the U.S.



THIS CHART WAS PRODUCED BY A SYSTEM BASED ON GKS.

TOWER 1632. THE SHAPE'S THE SAME, BUT IT'S GOT AN ENTIRELY NEW DIMENSION.

The most memorable shape in the computer industry just became even more unforgettable.

You might think it would be hard to improve on the computer that took the OEM world by storm only a short year ago, but we've done just that.

First, we added two 46MB, 5¼″ Winchesters to bring total storage capacity to 92MB (unformatted) in a single Tower unit.

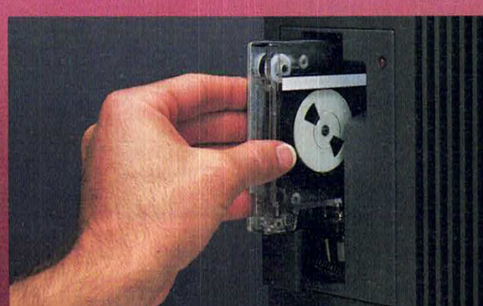
Then we added a ¼″ streaming tape for faster, more convenient back-up. The tape's 45MB capacity allows you to dump the entire contents of a Winchester on one tape cartridge.

Of course, this enhanced Tower still offers the UNIX*-based Tower Operating System for maximum software compatibility and flexibility. And unsurpassed interfacing capacity. And the NCR name. Your guarantee of support. Now and in the future.

And it all still comes in that famous box. You know. The one that towers over the minicomputer.

Call us at 1-800-222-1235 to learn more about the Tower's exciting new dimension.

*UNIX is a trademark of Bell Laboratories.



BUILT FOR SYSTEMS BUILDERS.
TOWER 1632.

NCR
OEM Systems Division

NCR Corporation, World Headquarters, OEM Systems Division, Dayton, Ohio 45479. Nationwide (800) 222-1235. In Ohio (513) 445-2380. In Canada (800) 268-3966.


NCR
Disk
Communications
Fault
Battery
On
Off

The "Personal" Mini

A 32-bit multi-user virtual memory microcomputer by LMC.

The MegaMicro is a "big" computer in a small box. It allows one or up to 32 users to run big applications programs (ones so big they can't even be compiled by smaller 8- or 16-bit machines) simultaneously. Because the MegaMicro is a multi-user system, it allows easy sharing of data bases and peripherals—obstacles that soon haunt business and scientific users of "personals" who find a need to "network" or to add devices such as laser-printers, multi-color plotters and the like.

LMC's MegaMicro is built around the newest state-of-the-art VLSI logic—the 16000 family developed by National Semiconductor. Each MegaMicro is supplied with UNITY—HCR's full Bell-licensed UNIX operating system—as well as FORTRAN AND C. Also standard are hardware virtual memory and hardware floating point, a half Meg. of RAM and a very fast 33 Meg. Winchester hard disk. The result is a computer with the performance of a large mini, at a "micro" price. For example, the MegaMicro does 161,000 double-precision (64-bit) floating point multiplications per second. All this at a realistic price, and even less with government and quantity discounts. The result is a cost per "work-station" far lower than similarly configured (and less powerful) "personals."

The MegaMicro is powerful, inexpensive and designed around the Multibus (IEE 796) which means it has a completely "open" architecture.

LMC MegaMicros *The Logical Alternative*™

LMC™

The Logical MicroComputer Company

A member of the Marmon Group of companies

4200 West Diversey, Chicago, IL 60639, (312) 282.9667, Telex 887787 LMC

Please circle Ad No. 19 on inquiry card.

When linked together, the Unix system and GKS form a powerful combination for graphics. American computer companies now use GKS for two reasons: It is a workable standard, and it provides portability of applications software that use graphics.

GKS provides more attributes than the Core System and offers many ways to manage them.

The chief problem confronting the computer graphics market is the need to deal with devices having dissimilar capabilities. Before people began addressing that difficulty, computer graphics had an esoteric aura. It was a realm open only to companies with mainframes and dedicated graphics devices, and programmers targeted their efforts toward specific systems and output devices. If a firm wanted to expand with different hardware, its software had to be modified or discarded—a luxury smaller companies couldn't afford.

A NECESSARY FEATURE

Today, computer users consider the graphics capability of their hardware a necessary feature, not merely an application. No longer does graphics software play second fiddle to spreadsheet, database, and word-processing packages. Business people, scientists, engineers, analysts—and all computer users —realize that computer graphics is a necessity for decision-making and for presentations.

Thanks to skyrocketing sales, microcomputers occupy desks in homes and offices throughout the world. Lower prices for printers and the affordability of graphics software have triggered the demand for new programs—but programs that run on a single type of computer are no longer satisfactory for most users. Now that we're in an era of networking, users want transportable graphics programs that can produce output on many types of computers, as well as terminals, printers, and plotters.

Mainframes, minis, and microcomputers must be able to communicate and share data with each other. Standards for interactive computer graphics help achieve this goal by facilitating software portability and compatibility, flexibility, program longevity, and ease of application program design. In a company's day-to-day activities, standards enhance productivity by eliminating the need for training newly hired programmers—and retraining them as they move from project to project.

THE GRAPHICAL KERNEL SYSTEM

GKS defines the interface between applications programs and software routines concerned with graphics. It is a "programmer-level" interface consisting of a functional description for a set of sub-routines that programmers can use to perform a wide range of input and output actions for two-dimensional graphics. GKS gives programmers a standard syntax, supports operator input and interaction, and allows storage and dynamic modification of pictures.

The system has been designed to provide graphical management that is (1) complete—it provides all the functions needed for most applications; (2) compact—the set of functions and parameters is small; (3) rich—the range of higher-order facilities is extensive; and (4) portable—graphical application programs can be easily transported between installations.

It also facilitates the portability of graphics applications programs between different computer installations by providing a consistent interface in high-level languages such as C. It also improves a programmer's ability to work on different systems by providing a graphics model and syntax that are common to several systems. This standardization defines the way in which graphics functions are accessed.

GKS makes it easier for users on all levels, from systems programmers to novices, to create graphics. Its portable applications programs provide host and device independence—allowing users greater freedom of choice. As a result, it permits workstation flexibility. Programming on all levels is simplified by the library of graphics primitives, and productivity and creativity are both increased.

The chief problem facing the computer graphics market is the need to deal with devices having dissimilar capabilities.

The system has a straightforward hierarchy of functional levels that makes it possible to develop portable programs with the freedom to configure systems from a wide range of available graphics hardware; it facilitates productive use of a programmer's time.

The standard also supports a full set of drawing primitive commands (with variable attributes) for data input and drawing, support for multiple workstations simultaneously, and device-independent picture segments. While GKS provides device independence for standard functions, it doesn't eliminate nonstandard operations. Thus, programmers are free to access the unique capabilities of a particular device.

FIVE BASIC COMMANDS

GKS uses five basic commands to draw complex pictures, place markers, specify text strings and locations, fill in areas, and transfer pixel arrays. In addition to its drawing commands, GKS can pan, rotate, and zoom images.

When an application program creates a picture with GKS commands, the image is stored as a display list in memory. If this picture, known as the *world view,* is too large to fit on a graphics unit, the user can utilize panning and zooming capabilities which specify that a particular portion of the image be displayed on the CRT screen.

Software based on GKS helps users to work with, or interface to, graphics software utilities and tools. These utilities and tools assist in providing descriptions of the graphics objects that will be displayed and used by the application software. Normally, these objects are defined in terms of two-dimensional coordinates in a coordinate system accessed by graphics primitives (graphics-oriented software commands).

In a CAD application, they might describe the multiple layers of a VLSI chip; in a business application, the coordinates might describe the points on a sales chart. Once the description of an object is incorporated into the application program, the GKS layer provides a standard software interface that converts the descriptions into a "viewing package."

GKS supports a full set of drawing primitives and the setting of primitive attributes. The basic drawing primitives are the polyline, polymarker, and text primitives. Raster devices are supported with fill and cell array primitives. An application program can access capabilities, such as drawing arcs, bars, and circles, through a special escape mechanism known as the Generalized Drawing Primitive.

Graphics objects such as arcs, points, and text are the primitives that can be combined into segments, which become building blocks to create complex pictures. They can be rotated, scaled, re-named, inserted into other segments, made visible or invisible, and manipulated in many other ways. The use of segments allows programmers to access a variety of powerful graphics functions.

The powerful international backing for GKS is complemented by strong support from large U.S. manufacturers.

GKS has several desirable workstation features. It allows multiple-window/viewport transformations (normalization transformations) and allows a special transform (workstation transform) to perform two chores that were previously difficult in device-independent graphics. These chores involve the production of scaled drawings in inches or millimeters, and the utilization of a rectangular screen without restricting graphics to a square area. GKS also supports a full set of input operations, allowing applications programs to receive input from a broad range of interactive input devices.

The device-independent qualities of GKS offer great flexibility. Device drivers are isolated into separate modules at a low level invisible to the user. The portability of applications software between various devices enables users to prepare preliminary work with low-cost devices and to prepare final presentations with high-resolution terminals and high-quality plotters.

OTHER STANDARDS

In addition to standards for writing graphics software, it is necessary to have standards relating to hardware compatibility and graphics information. Two emerging standards are addressing the hardware/driver interface level. The North American Presentation-Level Protocol Syntax (NAPLPS), developed at Bell Laboratories, is now an AT&T standard for transmitting text and graphics over telecommunications lines. NAPLPS provides a compact coding scheme that facilitates the exchange of graphics among people using different computer systems, and its data-compaction abilities enable complex graphics to be transmitted quickly over low-cost telephone lines.

The Virtual Device Interface (VDI) is being developed as a standard interface between device-independent software and graphics devices. VDI is a high-level, bidirectional set of commands used for creating picture elements and for reading from peripherals. By defining a standard input-output protocol, VDI makes all devices appear as identical virtual graphics devices.

Virtual Device Metafile (VDM) is a standard for storing and transmitting pictures, using normalized device coordinates and primitives. VDM, which supplies a means for transferring images from one system to another, is intended to be a standard file format for graphics.

Programmer's Hierarchical Interactive Graphics Standard (PHIGS), now in the early stage of development, will contain the complete set of GKS commands in addition to a set of hierarchical and three-dimensional commands.

GKS VERSUS CORE

The Core System, a graphics standard developed before GKS, has not gained international approval by the ISO. By contrast, the powerful international backing for GKS is complemented by strong support by large U.S. manufacturers such as IBM and Tektronix. In fact, ANSI recently abandoned its work on Core in favor of GKS.

How do GKS and Core compare? Both share a number of concepts, including the following:

(1) World coordinate space—which may be envisioned as a large drawing sheet.

(2) Windowing—which serves to select portions of world coordinate space to display to users.

(3) Normalized Device Coordinates (NDC)—which relegate device dependency to a low level.

(4) Viewports—which determine where the picture will be positioned in NDC space.

(5) Picture segmentation—grouping of output primitives as a single reference. This allows for manipulation and control of an object.

(6) Several levels of implementation ranging from applications with minimum graphic system demands to sophisticated applications.

Despite the similarities between GKS and Core, significant differences in their design, usage, and function have served to shift industry-wide support to GKS.

Core lacks provisions for incorporating new ideas that have evolved during the last 15 years in portability, device independence, performance, and structuring. Core also lacks the language bindings essential for true standardization. At the present time, each Core implementation has its own set of routine names and ordering parameters, so that a program written for one implementation cannot run on another.

AN IMPORTANT CONCEPT

The important concept of a workstation, present in GKS, is missing in Core. GKS has six graphical input classes, which can be defined as logical interfaces through which the application program controls physical devices. Thus, GKS is more in tune with the current popularity of the mouse and touchscreen as a user interface.

In GKS, all output primitives are totally independent of each other. They completely specify the action to be taken—eliminating a programmer's concern about the effect of external procedures.

While the Core System defines only a single window/viewport pair, GKS provides two viewing operations. After transforming output primitives from world coordinates to an enlarged normalized coordinate space of arbitrary size, a workstation window is applied to the image for further clipping and transformation to device coordinates. The concepts supported by GKS's viewing operation allow for the combination of two-dimensional viewing transformation with two-dimensional model transformations. This reduces calculations and provides more efficient throughput.

The capability of GKS in handling attributes is considered a strong plus. Attributes are mainly used to control the geometric aspects of output primitives such as size and shape of text, and the appearance of output primitives such as color. These attributes may be bundled (grouped under a single identifier) or individually specified. GKS provides more attributes than Core and offers many ways to manage them. Since the portability of an application program is affected by attribute management, the Core System is consequently less device independent than GKS.

GKS-defined language bindings provide a consistent interface between GKS and the application. Thus, a GKS-based program from one manufacturer can be recompiled and linked with software from a variety of companies—offering another advantage over Core.

Future demand for graphics software based on the Unix system and GKS will be strong in several areas: Scientific research and development laboratories associated with large companies and government agencies need to create graphical output from data; universities that use computer graphics for research and teaching; businesses where graphics software is becoming the preferred medium for information exchange; and Unix system-based computer manufacturers and vendors.

Powerful graphics sub-routines that adhere to established standards and that are both computer independent and hardware independent will be required to satisfy these needs. Certain applications programs may be extremely graphics intensive. These include statistical plotting, finite element analysis, and contour mapping. Several GKS standards committees are working to add three-dimensional sub-routines to GKS. ■

William B. Elmore is the president of Visual Engineering, a San Jose, Calif.,-based firm that supplies presentation and engineering graphics software for Unix systems based on GKS. He has an M.B.A. from Stanford as well as a B.S. and an M.S. in electrical engineering from Purdue.

Graphics system designers had neglected the Unix system—until they discovered the system's portability and ease of programming.

THE UNIX SYSTEM AND INTERACTIVE GRAPHICS

BY MARK S. CALLOW

Despite the clear advantages of the Unix system as a programming environment, only recently have graphics system designers begun to take advantage of the system's many features. Now, however, this trend is being quickened because of three independent, but converging, changes in the computing industry.

First is the Unix system's low cost—a result of a conscious effort by AT&T to lower the cost of the software to end-users. Second is the availability of low-cost 16- and 32-bit processors capable of taking the fullest advantage of the Unix system, which was developed originally for use on minicomputers. Third is the availability of programmers experienced with the Unix system—an effect of the software's extensive proliferation in universities and colleges.

As a result, graphics designers are taking advantage of the Unix system's portability and the ease of programming in that environment to develop a variety of graphics packages. One of the areas just now getting attention is interactive graphics—an area in which the Unix system has not traditionally been strong, but one where rapid advances are being made.

The advantages of the Unix system to the graphics system designer do not lie in any particular feature; rather, they lie in the system's extreme portability—the ease with which it can be adapted to virtually any hardware or software environment. Anywhere from 70 to 75 percent of the operating system remains identical from system to system. And adapting the Unix system to a particular hardware or software environment means altering at most 25 to 30 percent of the code related to machine-specific hardware and device drivers (written in assembly language and C, respectively).

The hardware interface routines provide such services as enabling, disabling, and vectoring of interrupts, changing the memory map to switch execution from one process to another, and transmitting information between a user's address space and that of the system. The device drivers are programs that provide interrupt handling, I/O command processing, and error recovery for the various peripheral devices connected to the machine.

APPLICATION PROGRAM INTERFACES

As advanced design, low-cost graphics peripherals have been developed for those systems to which the Unix system has been ported, a wide variety of application program interfaces have become available. Several databases of routines have been developed, including `plot`, `graph`, `spline`, `ged`, `stat`, `termcap`, `curses`, and `graphcap`.

`plot`: One of the first sets of graphics routines developed for the Unix system, this family of programs generates, in a relatively device-independent way, graphic output that is then interpreted by a set of filters for specific terminal types, including the Tektronix 4014; DASI Hyterm 450, 300, or 300S terminals; and the Versatec D1200A printer/plotter.

`graph`: This set of routines takes pairs of numbers from standard input as the coordinates of a graph that is then encoded on the standard output for display by the `plot` filters.

`spline`: This routine takes pairs of numbers from the standard input as coordinates for a smooth curve, which is then also encoded for display by the `plot` filters.

`ged`: This interactive graphics editor is used to display, edit, and construct drawings on the Tektronix 4010 series display terminals. Drawings are represented as a sequence of objects in the GPS language.

`stat`: This collection of routines can be interconnected using the Unix shell to manipulate and plot numerical data.

`termcap`: Short for *term*inal *cap*abilities, `termcap` is a database of information and some simple programs to access it that were developed at UC Berkeley. At UC San Diego, this idea was expanded into `graphcap`, which provides additional information for using the graphics capabilities of intelligent terminals.

In the `termcap` database, each terminal is described in terms of three basic types of features or capabilities: (1) Boolean capabilities, which indicate whether or not a terminal has a particular feature (such as direct cursor motion, clear screen, clear to end line, and so on); (2) numeric capabilities, which indicate the display dimensions or the length of a particular delay; and (3) string capabilities, which define a character sequence to be used in performing particular terminal operations. About a hundred capabilities are necessary to describe a terminal, although any one terminal type seldom requires all of them. In addition to the database, the package also incorporates routines to access the database and to handle simple operations such as generation of delays and cursor motion.

NEW DESCRIPTIONS AND PROGRAMS

To date, about 180 terminals have been described in the database, which is constantly being enhanced with new descriptions and programs. With this database, a programmer need only specify the name of the terminal. If it is in the database, a program will find it automatically. If the terminal is new, the programmer need only enter a description in terms of the above standard capabilities with any text editor.

`graphcap` is a quick and easy way of displaying data when a high-resolution device is not available. Enhanced graphics capabilities have made their way into low-priced terminals, allowing programmers to take advantage of the line drawing and special character features for low-resolution plots on terminals. A device-independent `plot` filter has been written that produces graphs on any terminal described in `graphcap`. Unfortunately, `graphcap` has not been widely distributed.

As useful as these databases are, producing screen-oriented programs is a tedious chore. `curses`, a library package from UC Berkeley that was originally developed for the infamous game *Rogue*, allows programmers to do many of the most common type of terminal-dependent functions, such as those involving movement optimization and optimal screen updating, with a minimum of effort. This library would be used, for example, in generating a menu on the screen without knowing the specific type of terminal to be used. It allows development of the particular specifications in a terminal-independent manner, using the data in `termcap` to customize it to a particular terminal at run-time.

In addition to these databases, routines, and filters, new system calls proposed for more recent versions of the Unix system are proving useful in developing graphics interfaces, especially those that employ bit-mapped graphics hardware.

One such system call is `mmap` (originally proposed for Unix 4.2BSD but delayed until 4.3BSD). Partially implemented on some Unix 4.2 systems, `mmap` allows mapping of any piece of virtual memory into a real piece of memory.

In most Unix systems employing bit-mapped displays, the bit map must be inserted at specific known addresses for every process. The limitation of this approach is that even processes that don't want or need it have some of their own address space used up by the video board, and this is especially critical in those systems with limited address space. `mmap`, on the other hand, allows the designer to map the video board into any address in the program. Other related system calls include `getpagesize`, `mremap`, and `munmap`.

INTERACTIVE GRAPHICS AND MULTIPLE WINDOWS

As the graphics capabilities of hardware employing the Unix operating system have become more sophisticated, much effort has been directed to interactive graphics management. Most of these packages employ multiple windows making use of the Unix system's pipe-based interprocess communications capabilities.

Multiple windows allow output from many different tasks to be displayed and processed at the same time. Users are able to view the output of one program while working with a different program. User interface packages generally allow users to perform such graphics management functions as selecting a window for interaction; scrolling up, down, left, and right within windows; moving a window; changing a window size; pulling down a menu associated with a selected window; burying a window behind other windows; and expanding a window to fill the screen.

Each task in this approach has a visual representation on the screen. To switch from one job to another, simply point to the new job with some sort of pointing device. In the meantime, the other tasks are preserved, each in its own window environment.

While interacting with the user on the second task in the second window, the system can be processing the first task in the background. For all practical purposes, this allows the user to perform simultaneously such diverse functions as defining and testing programs, generating and editing graphics, writing and editing, accessing the operating system, and sending and receiving electronic mail.

THE MULTIPLE-VIEW APPROACH

This approach to interactive graphics management builds on the multiple-window approach and allows the user to not only divide the screen into numerous sub-displays, but to move interactively among various views of an object "seen" in each window. In other words, each window is merely a flexible container for the view within it. This is the first step to building a tool that uses multiple windows for a single application.

Each view actually interprets its underlying content (the object) and controls the process used to manipulate it; thus, each view provides its own techniques to see and work on those objects. In this approach, the user can look and operate on the same object from a number of different "points of view," through a different set of filters. Each view will present the user with a different "flavor," aspect, or set of attributes of the object in a way that supports a specific understanding of its underlying content.

Thus, views, rather than windows, provide the central syntactic features common across all elements in the user's screen environment. Windows are just the medium for arbitrating screen space among multiple tasks, allowing the user to move quickly and easily among them. A window's physical representation, beyond delimiting the screen space, identifies its contained object and view, correlates interdependent views as appropriate, and provides access to common window operations.

SIMPLIFYING THE USER INTERFACE

As interactive graphics packages with multiple-window/multiple-view capabilities have become more sophisticated, their developers have found it necessary to simplify the user interface in order to remain commercially viable. Ideally, what one would want in such an approach to interactive graphics management is a command language that is consistent, mode-less, and as context-free as possible. In other words, the command structure that allows the user to manipulate various windows and images within windows should be as close as possible to the way human beings work with and think about everyday objects.

Complete acceptance of graphical interfaces will likely be a lengthy process.

Many of the window-oriented interactive packages developed for use under the Unix system employ an object-oriented approach similar in concept to that used on many popular personal computers, such as Xerox's Star, Apple's Lisa and Macintosh, and others. In this approach, contextual information is kept along with data, making the major command language component (an object) have not just a value, but also a range of inherited characteristics.

Through such a command language, the contents of a window are stored independently of the program providing the output. This permits functions such as scrolling back and forth through previous output into the window. Information on display in a window can be extracted and re-entered, or extracted and moved to a different window.

In this approach, windows are managed by objects that maintain the window data. These objects contain information and descriptions of the valid operations that can be performed on or by each. Messages define the specifications of one of an object's operations, which is then performed on or by that object.

This approach to data manipulation is radically different from the traditional method of separating operations from the data they operate on. In other words, operations are not done to objects under supervision of an intelligent programmer; rather, messages are passed to the object, which performs the operations on itself because it "knows" what must be done.

Traditional implementations employ a message-object structure. But the problem with this grammatical structure, especially concerning manipulating within and moving between multiple windows or views, is that it predetermines the mode within which the user is operating. By selecting a particular function (a message) to be performed before the object, the operator of the graphics interface is directing the system to operate in a particular way, in a particular mode.

In the newer systems, modeled after Smalltalk, the user interface is manipulated with a command language in which this traditional message-object grammatical structure is reversed. By first selecting the object and then the action to be performed, many problems are eliminated or at least minimized. All editing functions, for example, are activated by the user in the same manner, regardless of the environment—first selecting the object (a word, sentence, illustration, entire document, or page layout) and then the action to be performed.

The actions to be performed on any particular view—the messages—are selected by choosing from a set of functions that is essentially identical from view to view and window to window. No matter what operation is to be performed, the same basic set of commands is used because of the object/verb structure of the command language. While the method by which the computer performs the various functions may vary internally, the way the user initiates the action is consistent throughout. Only in specialized cases are additional commands added to the basic repertoire, these being used to perform functions unique to a particular environment.

Portability is the Unix system's major advantage to graphics systems designers.

A TYPICAL MULTIPLE-VIEW SYSTEM

We show in Figure 1 the typical display of a system employing a multiple-view approach to interactive graphics management. In this case it is an integrated text/graphics workstation designed for technical document production, where 90 percent of the work involves graphics manipulation and about 10 percent is text manipulation. As shown, each window has the same general structure:

The *window border* surrounds the window's extent, with the wider tab across its top containing such information as document name, figure name, and so on.

The *command pane* is a vertical region to the right of the window containing the various menu selections relating to the multiple views.

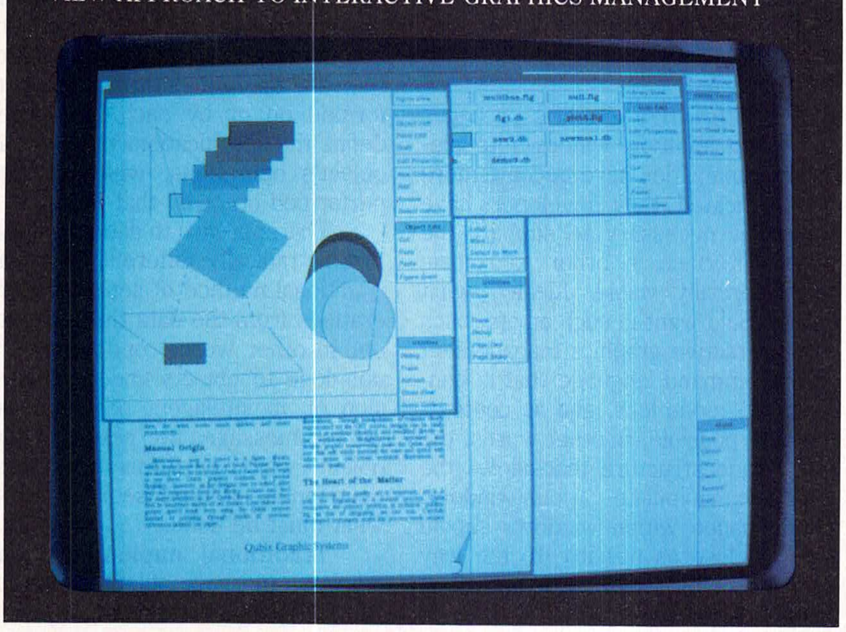
The *presentation pane* is the central area within the window, in which the view's representation of the underlying objects is displayed. Within the presentation pane, one of the objects is always identified as the "current selection," the object that will be acted upon next by user-specified commands.

Depending on the operation to be performed, any one of a number of different views can be displayed, either sequentially in the same window, or simultaneously, with different views in each of several windows. Because all views are connected to the same object (in this case, a document), alterations made to a document in one view are reflected in all other views as appropriate.

From the *outline view,* the writer/editor can revise the order of selected chapters and selections within the document being written by manipulating the appropriate heading entries. After this has occurred, the pages directly accessed in page-level views reflect the reordering.

The *manuscript view* is a display of the text plus the generic mockup commands split into pages.

FIGURE 1:A TYPICAL DISPLAY OF A SYSTEM USING A MULTIPLE-VIEW APPROACH TO INTERACTIVE GRAPHICS MANAGEMENT



This is similar to files in a word-processing system. The *composed view* is the laid-out text with the appropriate artwork and is the view that provides access to all the other ways of viewing and working on a document. In both the manuscript and composed views, the user can directly access and edit the text, all of which feeds back to the finished page's composed appearance.

Within the views, writers use the same basic set of commands or verbs: `delete`, `cut`, `copy`, `paste`, `insert`, `find/replace`, `label`, `mark`, and `select to mark`. While you are in the manuscript view, these commands refer only to text editing; in the composed view, you can use the same basic commands to perform icon and figure editing as well.

For the majority of graphics operations, the operator works mainly with the *figure view*, a display of only the figure(s) on a specific page. It is in this view that the artist or illustrator creates new figures either by drawing directly on the screen with the pen-like sonic stylus or by selecting a previously stored symbol or part from a parts library.

CONNECTED VIEWS AND UNIX 4.2BSD

Despite the sophistication of the multiple-window and multiple-view approaches to interactive graphics, both are limited in their intercommunications capabilities. In the multiple-window approach, it is a case of one window, one function. While it is possible to be doing many functions at the same time, each occupies its own window with its own underlying "object," be it an illustration, text document, or database. To modify the content of a particular window and its underlying object based on new information available in another window, the operator must execute a series of commands that move the appropriate information from one location to another, from one object/window to another.

In the multiple-view approach, the relationship between what is being performed in a particular view and what is going on in another view is much closer, even if they are in different windows. Certain views are related to one another in that they are associated with the same underlying object. Updating one view with information from another is simpler than in the multiple-window approach. This is usually just a matter of the operator sending a message, via some sort of overall screen manager function, to the underlying object with which both views are associated.

In most interactive graphics applications, it is useful, even essential, that the relationship between various windows and views be even closer and that they communicate automatically without the intermediation of a screen manager; in other words, they should be connected. In such a "connected view" interface, a different representation of a single object can be displayed within each view. Changes to the object would be reflected automatically in all views.

Rapid advances are being made in interactive graphics —not a Unix systems stronghold heretofore.

However, this is difficult under most Unix systems because pipes are the only standard mechanism that allow two processes to communicate. Pipes are restrictive in that (1) two communicating processes must be related through a common ancestor, and (2) the pipelines are unidirectional; that is, communication is in one direction—from the output of one program directly to the input of another program.

Some Unix versions provide additional interprocess communication mechanisms that are useful in programming connected views. For example, Unix 4.2BSD provides facilities enabling processors to communicate through a Unix file system-like name space (where all the names are path names) and through a network name space. Once communication is established, the processes may communicate through datagrams in addition to the simple byte stream provided by the traditional pipe.

THE BASIC BUILDING BLOCK

In this approach, the basic communication building block is the "socket" rather than the pipe. A socket is an endpoint of communication to which a name may be bound. Each socket has a type and one or more associated protocols. Sockets are typed according to the kind of communications service they allow, and processes are assumed to communicate only between sockets of the same type. However, there is nothing preventing communication between sockets of different types as long as the underlying communication protocol allows it.

Under Unix 4.2BSD, the user has access to three types of sockets: *stream, datagram,* and *raw.* A stream socket provides for bidirectional, sequenced, and unduplicated flow of data without record boundaries. Aside from bidirectionality, communications between a pair of connected stream sockets is nearly identical to that of pipes. A datagram socket supports bidirectional flow of data in which the records are bounded, but they may or may not be sequenced or unduplicated. In other words, a process that receives messages on a datagram socket may find messages duplicated, possibly in a different order from that in which they were sent. A raw socket provides developers with access to the underlying communications protocols that support socket abstractions.

With such a rich interprocess communications structure, it is now possible to split a particular "job" among multiple unrelated processes. With this feature it should be possible for system designers to develop interactive graphics interfaces based on multiple and connected representations, or views, of an object.

Implemented in the example discussed earlier, this connected view structure would allow these "views" and windows to be grouped so that updates in the figure view, for example, are reflected automatically in the composed view. Similarly, changes in the manuscript and layout views would be reflected automatically in the composed view because all are connected to the same underlying object.

In terms of the underlying inter-process communications (IPC) facilities, the underlying object—the document—uses sockets rather than pipelines to tell the various views to examine the database again and to re-create themselves based on the new information supplied by one particular view.

We are now starting to see the development of graphical interfaces for many computer-related tasks, interfaces that could greatly raise the level of abstraction at which users work. But complete acceptance of such interfaces is likely to be a lengthy process. As video display units have become commonly available over the last 10 years, developers have struggled to come to terms with screen-oriented interfaces. Witness how many systems, including Unix, are still essentially line-at-a-time systems. Indeed, much can be gained from graphical interfaces, but much work also remains to be done. ■

Mark Callow, who hails from England, has worked for such firms as Northern Telecom System Corp., Logica Ltd., and Qume Corp., where he worked on the specification of an advanced word-processing and typesetting system.

THE LATEST FROM DEC

ULTRIX 32

BY GENE DRONEK

DEC has done a 180-degree turn and is now the first major player to fully support Berkeley 4.2BSD.

UNIX/WORLD recently visited Digital Equipment Corporation's headquarters in Merrimack, N.H., to follow up on DEC's latest system software announcement, Ultrix 32. Ultrix is DEC's name for Berkeley 4.2 Unix, released only months ago from the UC Berkeley Computer Research Group. UNIX/WORLD wanted to find out how much of Berkeley 4.2 would be in Ultrix and if DEC would really support it. If Ultrix really is ready, then DEC would become the first major player to back 4.2 Unix.

We talked to Bernie Toth, Unix product manager, and Armando Stettner, senior member of the Unix engineering group, to preview DEC's technical plans for the Ultrix release. Bernie told us, "Ultrix is real. It is a complete Berkeley 4.2 distribution, and it will run on any of the VAX series 780, 750, 730, as well as the [announced, but yet unreleased] Micro-VAX." DEC kept as much as was practical from the original Berkeley release, including a plethora of device drivers.

"Nothing useful has been left out," explained Armando. "We have checked that all drivers work on all supported CPUs. Some 50 sites have been testing prerelease Ultrix on an amazing variety of configurations. When you get the distribution tape, you can rest assured that it will BOOT." You will even find some non-DEC drivers in Ultrix, but, of course, DEC will be supporting only DEC equipment drivers. Ultrix, technically speaking, will become the most stable, yet flexible, Unix system running on VAX hardware.

There are additional debugging facilities from 4.2 and an interesting tool, `gprof`, handy for fine-tuning complex C programs. `gprof` automatically produces sophisticated execution-profile reports to help you with program bottlenecks. You will find a network-wise Mail program, Sendmail, and a Network statistics package among the 4.2 goodies supplied in Ultrix.

When UNIX/WORLD arrived at

the DEC development lab, it looked as though a major production effort had just been finished. Furthermore, we noticed that part of the technical staff was about to leave on a well-deserved vacation. Proof copies of the new Ultrix manuals had just come back and were being passed around. The entire Unix manual had been re-typeset into a series of sharp-looking 7-by-9-inch Ultrix binders. Ultrix is indeed real, and so is DEC's support.

Although Ultrix is now based on Berkeley 4.2, it will not remain pure forever. DEC admitted that Ultrix will pick up some System V functionality over time, but DEC was not going to tell us exactly how they were approaching the problem. Future releases and bug fixes from Berkeley will be folded into Ultrix. The specter of "System V compatibility" is not as significant for Ultrix users as are higher performance and using the leading edge in Unix software development.

VAX HARDWARE

Ultrix 32 will run on any of the VAX series processors except the Micro-VAX, which will run Ultrix 32m, a smaller, leaner version, trimmed for DEC's smaller-capacity disk drives. VAX processors include the 730, 750, 780, and the recently announced, faster 785 processor. Although UNIX/WORLD saw several Micro-VAXs in the DEC laboratory (and on their development Ethernet, no less!), we were not allowed to benchmark them because they weren't ready for public announcement yet. UNIX/WORLD will feature the Micro-VAX separately in a future issue.

What software is actually in Ultrix, and what was left out? We used an Aim Technology benchmark, and it reported a total of 280 user commands in the various directories. This is the largest command set we have seen to date. The Aim test shows that Ultrix dropped a few Version 7 commands, skipped a lot of System III commands, but included a whopping 129 additional commands (see Figure 1). Generally speaking, DEC will support the standard commands in `/bin`, `/usr/bin`, and `/usr/ucb`, but it will not support any commands located in `/usr/new` and `/usr/local`.

Naturally we asked what "supported" and "unsupported" meant. DEC intends to maintain supported software from release to release and to fix bugs therein. For unsupported products, bug lists will be kept, and a best effort made to fix them. As Armando Stettner explained, the 4.2 community tends to share code and bug-fixes for orphan software when the problem is important enough. DEC intends eventually to support all the products as its support capacity evolves.

Most of Berkeley 4.2 made it

At $795, our DEC™-comp
a sophisticated image

atible terminal projects even when it's off.

The WY-75 is the one VT-100 software-compatible terminal that looks intelligent even when it's not on.

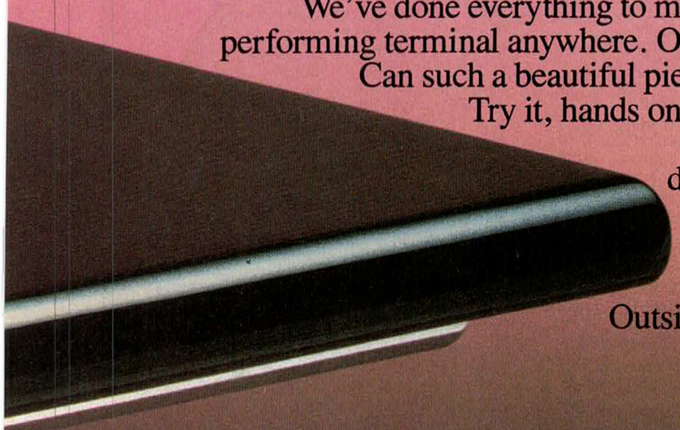
And at $795, it looks as smart to the people who pay for it as it does to the people who look at it all day.

Like all our terminals, the WY-75 combines an unusually small footprint with a very generous 14″ diagonal display. The non-glare screen tilts, swivels, and displays a full 132-column format. The sculpted, low-profile keyboard adjusts, too.

We've done everything to make the WY-75 the best looking, best feeling, best performing terminal anywhere. On or off.

Can such a beautiful piece of equipment be the workhorse we say it is?

Try it, hands on, and see your application software on the screen.



To find out where you can see the WY-75 demonstrated, call the regional office nearest you.

For more information about our complete line of products, write or call Wyse Technology, 3040 N. First Street, San Jose, CA 95134, (408) 946-3075. Outside of California 800-421-1058.

WYSE

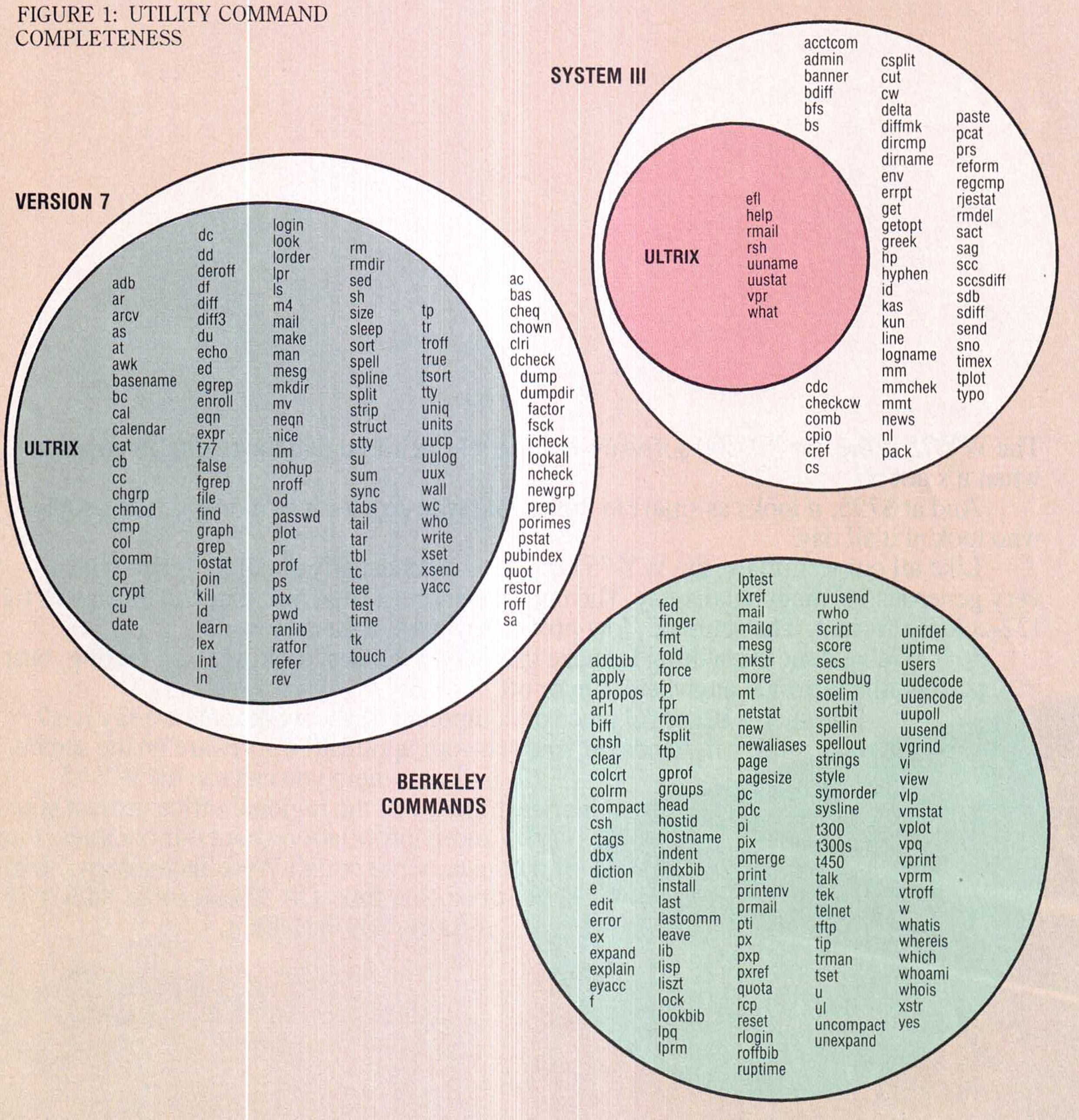
Regional Offices: Northeast (201) 725-5054; Southeast (305) 862-2221; Midwest (313) 471-1565; Southwest (213) 340-2013; Northwest (408) 559-5911.

into Ultrix, but some of the *contributed* category software fell off. This was due mostly to licensing snarls among the many parties involved, not because of technical or support-related issues. Regrettably, Ingres did not make it into Ultrix.

FUNCTIONALITY

What functionality does 4.2-based Ultrix offer above Systems III and V? That depends on when you ask. Both AT&T and DEC claim they already have each other's shortcomings under study and will provide identical functionality in the "next release." For now, local-area networking, the McKusick "fast file system," and paged virtual memory are presently available only in 4.2 systems. However, System V cur-

FIGURE 1: UTILITY COMMAND COMPLETENESS



AIM BENCHMARK MEASUREMENTS

VAX 730 - erehwon, RA81 disk

Arithmetic Instruction Times (microseconds per op)

	short	*long*	*float*	*double*
+ add	10	5	20	15
* multiply	22	15	35	30
/ divide	73	19	35	29

Memory Loop Access Times (microseconds per byte)

	read	*write*	*copy*
CHAR type	10	8	8
SHORT type	5	4	4
LONG type	2	2	2

Input/Output Rates (bytes/sec)

	read	*write*	*copy*
DISK	217k	162k	76k
PIPE			59k
TTY 1		0	
TTY 1+2		0	
RAM 1-byte			124k
RAM 4-byte			476k

Array Subscript References (microseconds)

short[]	long[]
9	7

Function References (microseconds/ref)

0-parameters	1-parameter	2-parameters
funct()	funct(i)	funct(i,i)
51	93	99

Process Forks
9 per second

System Kernel Calls (calls-per-second and microseconds per call)

getpid() calls:	1 kcalls/sec or	709 microseconds/call
sbrk(0) calls:	483 calls/sec or	2070 microseconds/call
create/close calls:	19 pairs/sec or	52632 microseconds/pair
umask(0) calls:	1 kcalls/sec or	889 microseconds/call

VAX 750—Limbo, 6 Mbyte RAM, RM05 disk, Ultrix

Arithmetic Instruction Times (microseconds per op)

	short	*long*	*float*	*double*
+ add	4	1	14	11
* multiply	13	10	45	47
/ divide	37	10	78	81

Memory Loop Access Times (microseconds per byte)

	read	*write*	*copy*
CHAR type	3	3	3
SHORT type	2	1	1
LONG type	489ns	656ns	793ns

Input/Output Rates (bytes/sec)

	read	*write*	*copy*
DISK	408k	321k	139k
PIPE			155k
TTY 1		0	
TTY 1+2		0	
RAM 1-byte			346k
RAM 4-byte			1,261k

Array Subscript References (microseconds)

short[]	long[]
3	3

Function References (microseconds/ref)

0-parameters	1-parameter	2-parameters
funct()	funct(i)	funct(i,i)
28	39	50

Process Forks
11 per second

System Kernel Calls (calls-per-second and microseconds per call)

getpid() calls:	3 kcalls/sec or	310 microseconds/call
sbrk(0) calls:	1 kcalls/sec or	887 microseconds/call
create/close calls:	28 pairs/sec or	35714 microseconds/pair
umask(0) calls:	3 kcalls/sec or	364 microseconds/call

VAX 780--Vacuum, 8 Mbyte RAM, RA81 disk, Ultrix

Arithmetic Instruction Times (microseconds per op)

	short	*long*	*float*	*double*
+ add	2	601ns	5	3
* multiply	3	2	7	5
/ divide	21	10	11	10

Memory Loop Access Times (nanoseconds per byte)

	read	*write*	*copy*
CHAR type	2	2	2
SHORT type	786ns	793ns	783ns
LONG type	297ns	402ns	402ns

Input/Output Rates (bytes/sec)

	read	*write*	*copy*
DISK	202k	142k	74k
PIPE			243k
TTY 1		0	
TTY 1+2		0	
RAM 1-byte			647k
RAM 4-byte			2,486k

Array Subscript References (microseconds)

short[]	long[]
2	2

Function References (microseconds/ref)

0-parameters	1-parameter	2-parameters
funct()	funct(i)	funct(i,i)
17	22	26

Process Forks
28 per second

System Kernel Calls (calls-per-second and microseconds per call)

getpid() calls:	5 kcalls/sec or	183 microseconds/call
sbrk(0) calls:	2 kcalls/sec or	505 microseconds/call
create/close calls:	30 pairs/sec or	33,333 microseconds/pair
umask(0) calls:	5 kcalls/sec or	204 microseconds/call

rently offers shared memory, which 4.2 does not.

Not surprisingly, all three of these additions serve to boost 4.2 performance over standard Unix. Higher performance was the main motivation for the Berkeley modifications, although there are rumors that 4.2 is fickle to tune for heavy multi-user loads.

System V and 4.2-based Ultrix implement a few similar functions differently. System V offers semaphores, messages, and named pipes, while 4.2 offers file locking and interprocess communications sockets. You can achieve the same effect on both systems, but not in a transportable fashion.

NETWORKING CAPABILITIES

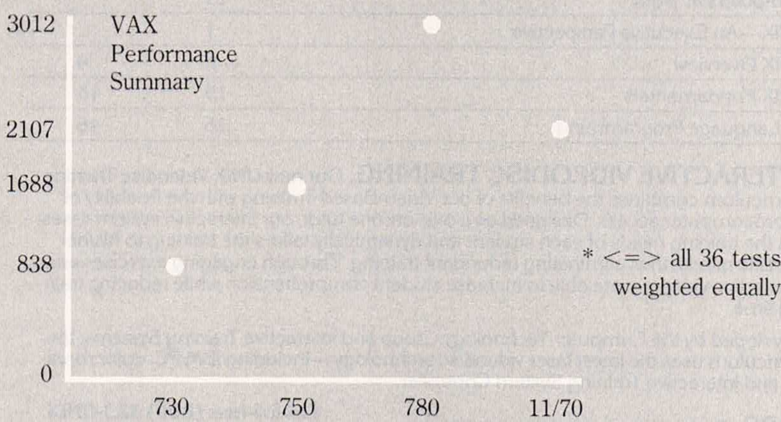
A bonanza of commands is associated with the local-area networking capability. The basic net services arise from implementing process-to-process communication across the network, but there is *no real sharing* of file systems per se. You can use `rcp` to copy remote files, `rlogin` for remote log-in, along with `rsh` for remote shell command execution. Additionally, `rwho` and `ruptime` report who is logged on and system times netwide. All of these programs pass messages over the net to accomplish "virtually" their apparent tasks. Networking undoubtedly is the most exciting area of software research and development in 1984.

In the future, we can hope to see transparent file system sharing. Some non-Unix systems offer it presently, and many companies are at work on their own implementations. Transparent sharing in the fullest sense means that all devices, not just disk files, can be shared across the network without special treatment. There is still plenty of opportunity for network-based application programmers.

DEC's Ultrix stands a reasonable chance of becoming the de facto "standard" supported Unix for VAX systems. We think this is because much effort on DEC's part went into making sure all peripheral drivers work uniformly with all CPUs. All the unusual hardware configurations have already been explored. Secondly, source code will be available to customers with System V and 4.2 source licenses. Thus, as Unix evolves, Ultrix will probably be the purest commercial source of stable, maintained 4.2 code. ■

Gene Dronek, Aim Technology, 3333 Bowers Avenue, Santa Clara, CA 95051. Gene is author of the Aim Benchmark Series; currently, he is the Director of Software for Aim Technology. Mr. Dronek graduated with a Computer Science BS from UC Berkeley. At Berkeley Computer Center, he began as a systems programmer, developed a popular user-friendly debugging compiler, and later crossed over to user services, where he ultimately headed staff and academic consulting services.

FIGURE 2: THREE-WAY PERFORMANCE SUMMARY



This chart shows the overall performance of the three VAXs we benchmarked, along with a reference PDP 11/70. Overall performance points for each machine were computed by averaging the 36 Aim benchmark test measurements.

UNIX™ & 'C'

HANDS-ON TRAINING

SEMINARS • VIDEO-BASED TRAINING • AND NOW INTERACTIVE VIDEO

Select your training medium according to the type of training you prefer and the number of people to be trained.

VIDEO-BASED TRAINING. The Computer Technology Group's Video-Based Training integrates professionally developed and produced video and text material, as well as hands-on exercises, into complete training programs.

Our courses are produced with the highest standards of video quality, applying the latest techniques of instructional design including the use of computer graphics and animation techniques to compress learning time. The students' time is not wasted with the "camera at the back of the classroom" or "chalk talk" approach which is so inefficient, and often ineffective, in transferring skills.

Our Video-Based Training courses are completely self-contained, including the hardware-independent hands-on exercises. All you need is a video cassette player.

COURSE	Number of Modules Video-Based	Interactive
Computers at Work	15	
UNIX—An Executive Perspective	1	
UNIX Overview	6	6
UNIX Fundamentals	15	15
'C' Language Programming	16	16

INTERACTIVE VIDEODISC TRAINING. Our new UNIX Videodisc Training Curriculum combines the benefits of our Video-Based Training with the flexibility of microcomputer access. Designed as a one-on-one tutor, our interactive system assesses the training needs of each student and dynamically tailors the training to his/her specific needs, thus eliminating redundant training. Through engaging exercises and interactive video, we are able to increase student comprehension while reducing training time.

Developed by the Computer Technology Group and Interactive Training Systems, the curriculum uses the latest laser videodisc technology—including IBM PC, color monitor and Interactive Training System Controller.

PUBLIC AND IN-HOUSE SEMINARS. Both public and in-house seminars are offered on a wide variety of UNIX and 'C' Language subjects, including:

UNIX Overview • UNIX Fundamentals for Non-Programmers • UNIX Fundamentals for Programmers • Shell as a Command Language • 'C' Language Programming • Shell Programming • Using Advanced UNIX Commands • UNIX Internals • UNIX Administration • Advanced 'C' Programming Workshop • Advanced 'C' Programming Under UNIX • Berkeley UNIX Fundamentals and "csh" Shell.

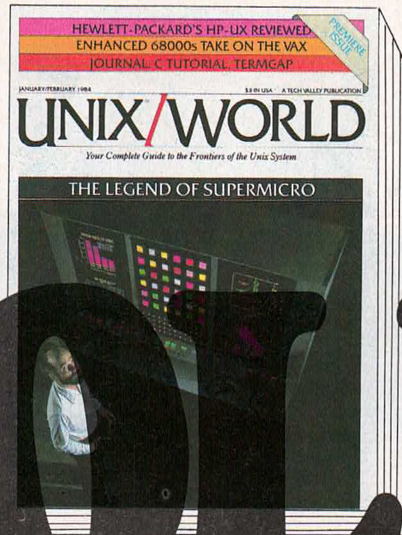
Call toll-free: **(800) 323-UNIX**
or in IL (312) 987-4082
310 S. Michigan Ave., Chicago, IL 60604

COMPUTER TECHNOLOGY GROUP
Telemedia, Inc.

TM UNIX is a Trademark of Bell Laboratories

Please circle Ad No. 5 on inquiry card.

UNIX™/WORLD's Premiere Issue



SOLD OUT!

LIMITED REPRINTS NOW AVAILABLE

All 44,000 copies of our Premiere Issue have been completely sold out! Because so many requests have come in from all of you that did not order the Premiere Issue in time, we have decided to reprint a limited number (only 2,000) of the first issue. If you are interested in Vol. 1, No. 1, please complete the coupon below and mail it to us today.

Only the first 2,000 requests can be honored and must be accompanied by a check or money order for $6.00 per copy. All copies will be shipped third class. Add $1.50 for first class shipment. *No credit orders will be accepted.*

Yes! I must have Vol. 1, No. 1, to complete my collection of UNIX/WORLD Magazine.

NAME: ____________________ TITLE: ____________________

COMPANY: ____

ADDRESS: ____

CITY: ______________ STATE: __________ ZIP: ________

☐ I am a subscriber.
☐ I am not a subscriber.

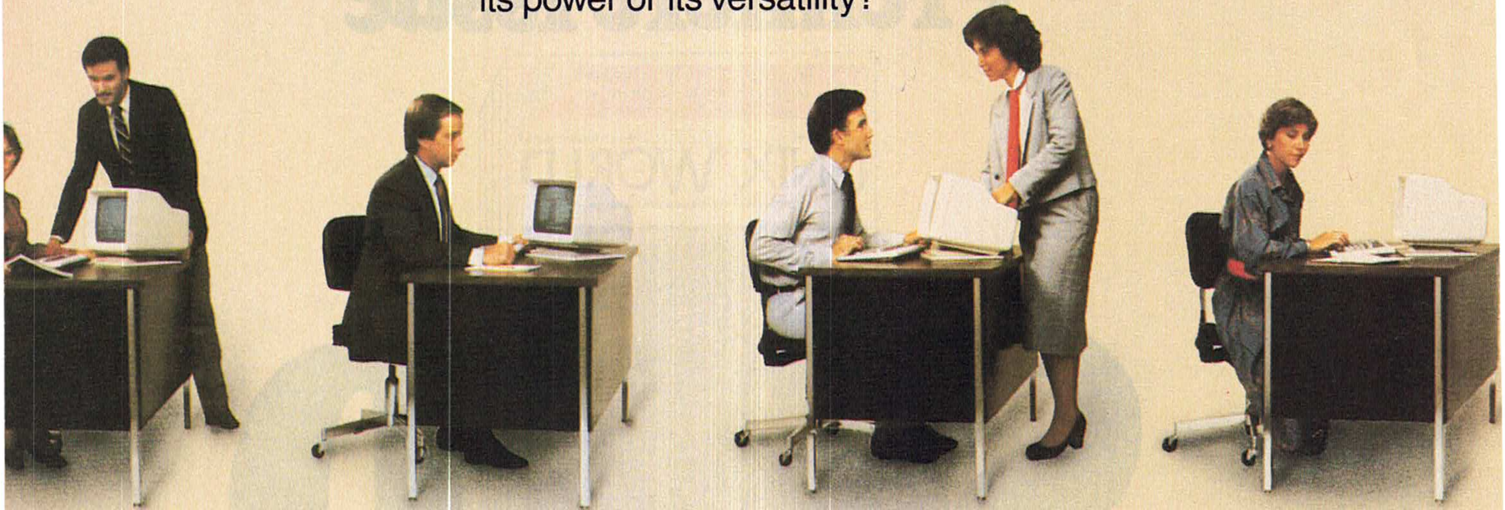
Send coupon to: Cheryl L. Hogan
Circulation Director, UNIX/WORLD
444 Castro St., Suite 1220, Mountain View, CA 94041

(Please allow 4–6 weeks for delivery.) RPRQ11

Please circle Ad No. 55 on inquiry card.

What do you like
better about the VISUAL 2000 . . .
its power or its versatility?

Personally,
I like its price!



Never has a UNIX-based multi-user system given so much to so many for so little.

Introducing the VISUAL 2000

The VISUAL 2000 is the full-featured system with the power and flexibility to support multiple users in real business applications at a surprisingly low cost per station. It can be used with inexpensive video terminals. Or as a database manager or file server for a cluster of intelligent workstations or PCs, including both the IBM® PC and VISUAL's own lightweight, portable, totally IBM PC compatible COMMUTER. In all applications it offers greater performance, more flexibility, and lower cost than any other system in its class.

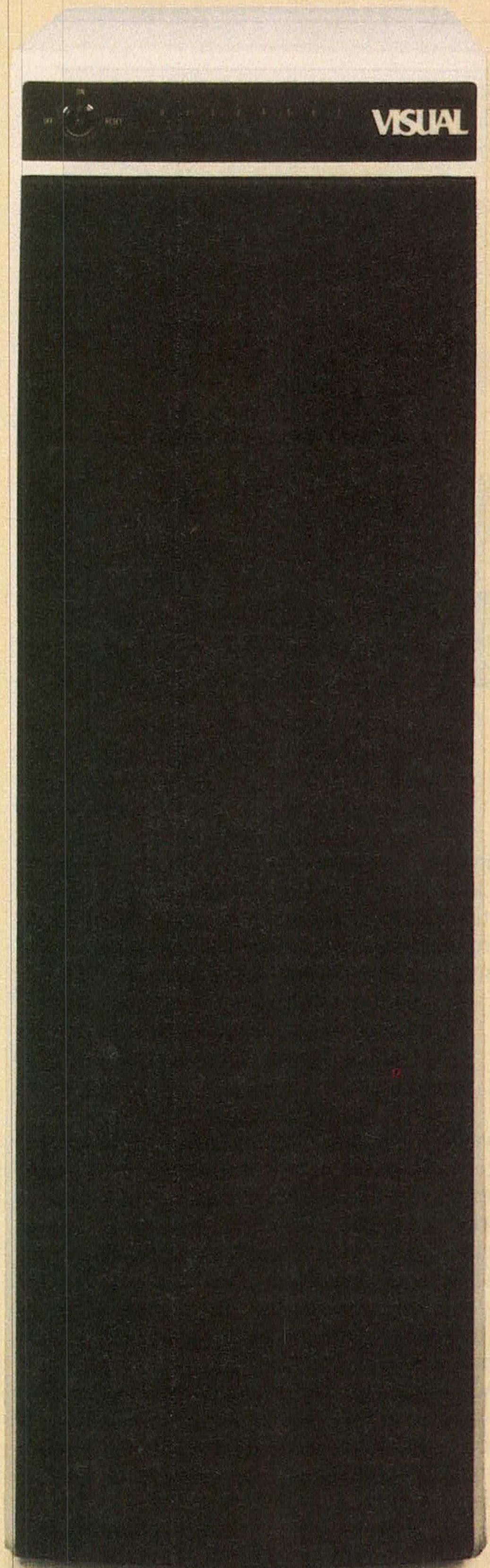
Powerful Intel 286 processor

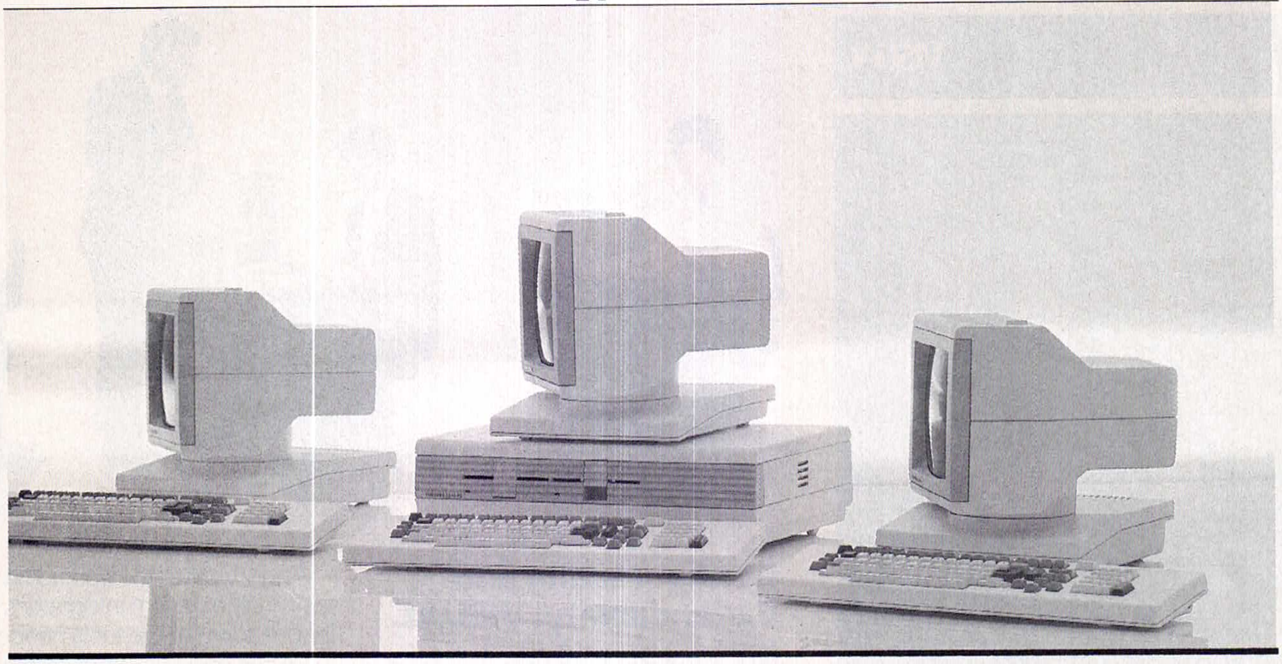
The Intel 286 is today's chip of choice for UNIX™-based systems. Only the Intel 286 has on-chip memory management, an instruction set optimized for multi-tasking, and the optional 287 numeric co-processor to speed up floating point by a factor of 10.

What do these features mean to the VISUAL 2000 end user? Faster response time, more users supported, and lower system cost!

Cost-effective one-board design

A basic advantage of the VISUAL 2000 is its one-board base-level design. A single high-density board includes the 286 CPU, 512KB-2MB of RAM, controllers for Winchester, floppy, and streaming tape, an intelligent communications processor, six RS-232 ports, and a parallel printer port. Even a real-time clock with battery backup. One-board design means higher performance, lower cost, and greater reliability than comparable multi-board implementations.


VISUAL



A REVIEW OF THE REALWORLD ACCOUNTING SYSTEM

BY NICHOLAS L. FEAKINS

The RealWorld Accounting System (Release 3.0) is a menu-accessed integrated accounting system consisting of seven packages: General Ledger (G/L), Order Entry (O/E), Sales Analysis (S/A), Accounts Receivable (A/R), Inventory Control (I/C), Accounts Payable (A/P), and Payroll (P/R). Each is sold separately, and each (with the exception of Order Entry and Sales Analysis) can function independently. Sales Analysis and Order Entry must be run with Inventory Control *and* Accounts Receivable. Accounts Receivable, Accounts Payable, Payroll, and Inventory Control can all be run in conjunction with the General Ledger package.

The system currently lacks Purchase Order and Fixed Assets packages. Chip Payson, RealWorld's development director, told me that the company is currently working on a Job Costing package (which it expects to make available before the end of this year), after which it plans to develop a Purchase Order package. The firm is also currently working on a point-of-sale system, for which there is currently no estimated availability date.

DOCUMENTATION AND INSTALLATION

The documentation for the system is relatively well written and should prove readily understandable to most users. RealWorld takes a rather cavalier approach, however, toward the ease of installing the system:

"You . . . DO NOT have to be an accountant to use RealWorld, but you DO have to have a basic knowledge of bookkeeping. Make liberal use of the RealWorld Glossary and a good dictionary. It may also pay to have at hand a simple bookkeeping workbook which defines terms as it goes along and has plenty of illustrations and examples. If you truly can't tell the difference between a debit and a credit, then you should consider taking a basic bookkeeping course before you start working with the RealWorld software."

To me the foregoing reads like an excerpt from *Do It Yourself Brain Surgery*. Any company seriously considering the purchase of this or any other integrated accounting software would be well advised to have it both evaluated and installed by an accountant with a degree.

Noting that the best way to learn the system is to actually use it, RealWorld provides sample data in the documentation for each of the packages. Users can then enter this data into the system and receive instant feedback as to whether they are handling the procedures properly. The one exception to this approach is the General Ledger package, which comes with sample data files containing "a volume of data that would be the result of two months' worth of General Ledger activity in a fully operational General Ledger system." The data entered in doing the practice exercises for the other packages is a subset of the data contained in the sample data files.

This approach is satisfactory where the system is being installed in a new operation (that is, where there is no pre-existing accounting activity). Usually, however, the user is making a transition from a hand-kept or semi-automated system to a computerized one and needs some guidance on how to make the conversion with maximum efficiency (for example, in which order should the packages be installed) and a minimum amount of downtime.

OPERATION

Central to the system's operation is the Company File, which includes basic company information (name, address, and phone number), system parameters (profit centers, passwords, protected changes, etc.), and print specifications (compressed print option and type of printer). Printers specifically supported by the system include Datasouth, Man-Tally 1805 and 180L/160, General Electric, Apple Dot Matrix, Anadex 9501A and 9620A, Epson, Okidata, IDS, NEC, Digital, and the IBM PC.

The General Ledger, Order Entry, Accounts Receivable, Inventory Control, Accounts Payable, and Payroll packages contain programs for interactive maintenance of the account, customer, item, vendor, and employee files, respectively, including add, change, inquire, and delete functions. Access to records within the file can be alphabetical (by customer name) or numeric (by customer number), at the user's option.

One nice feature of the system is that within all the packages (except Sales Analysis) a data integrity check program allows the user (by running it at the beginning and end of each day) to detect several possible types of data file corruption (sometimes due to hardware or operating system errors). A data integrity report is automatically printed when each of these programs is run, printing warning messages about errors it detects.

Another useful feature is that all of the packages (again, except Sales Analysis) are capable of producing edit lists of various journal inputs, a great help in editing and reconciling entries when a large amount of data has been entered into the system. Although the ultimate goal of any computerized system is to reduce hand calculations and entries to a minimum, they can never be totally eliminated. This feature allows them to be made more efficiently.

All of the packages allow the user to specify that passwords be required to use the system, and each major function on the menus may be assigned its own password.

GENERAL LEDGER

The General Ledger, which is capable of handling up to 13 accounting periods, may be used independently, or it can be tied in with the Accounts Receivable, Accounts Payable, Payroll, or Inventory Control packages.

The package allows only one level of consolidation. This may be satisfactory for smaller retail and wholesale distributors with multiple sales departments or branches, but it is likely to prove inadequate for manufacturers (especially job shops) and others requiring multiple levels of cost breakdown. (The Job Costing package currently under development should satisfy most of these requirements once it is available).

For users with profit centers, the format for the G/L account numbers is a 4-digit main account number followed by a hyphen and then the 3-digit profit center. Users who do not use profit centers only use the 4-digit main account number. Personally, I like to use at least a 6-digit account number, of which two digits constitute subaccount numbers, and I have found a 4-digit number to be unduly restrictive.

In addition to direct posting from the A/R, A/P, P/R, and I/C packages, the system allows direct interactive entry, editing, and posting of general journal transactions. After the general journal transactions are posted to the year-to-date file, a general journal transaction register is automatically printed as a part of the audit trail.

A standard journal transaction file may be maintained for recurring journal entries, each transaction being set up with either a fixed amount that is automatically posted each period (that is, auto-recurring entries like rent and depreciation) or with a variable amount that may be entered each period and then posted. This feature makes it easier to adjust entries at the end of each period. Similarly, general journal transactions may be entered in such a way that they will be automatically reversed the following period. While use of these so-called "reversing entries" is heavily frowned on by accounting purists, they are extensively used by many accounting managers and controllers.

Once all the initial journal entries have been made, the system can produce the following: a general ledger working trial balance (to serve as a worksheet for making adjustments to the regular trial balance) for any specified range of dates and for all or selected accounts; a source cross reference report; and the regular general ledger trial balance, which provides up to nine levels of subtotals and which may also be printed for any specified range of dates and for all or selected accounts.

ORDER ENTRY/BILLING

Order Entry/Billing uses the customer file from Accounts Receivable and the item file from Inventory Control, thus requiring users to purchase those two packages as well. In this regard, it is similar to other integrated accounting packages.

Allowing entry of orders for either immediate billing or for partial or whole billing later, the package offers the user the capability of defining extensive defaults to speed accurate data input during order entry. Its inquiry capability, critical in order processing applications, is well designed, allowing the user to retrieve customer information from the A/R customer file by either customer number or name.

Through the interface to the Inventory Control package, on-line inventory control is a system option whereby inventory is automatically committed when orders are entered and relieved when invoices are printed. Available inventory is verified for each line item as entered, allowing any back order or out-of-stock condition to be handled interactively.

The attractiveness of an order processing package to an individual user often depends on whether or not it offers a number of (seemingly unrelated) features. This package is exceptionally strong in terms of both the number and variety of features offered, including the following:

(1) Line items may be entered by item number or description;

(2) It provides for drop shipments;

(3) It provides on-line credit checking;

(4) Cash received with one-step orders (for example, in mail order companies) is entered directly into the system without having to go through the separate cash receipts program in the A/R package;

(5) Although item pricing is controlled primarily through the item file in the Inventory Control package, a user-maintainable "special sales price" file allows such sales prices to override normal pricing methods for specified periods. And manual price overrides may always be specified;

SALES ANALYSIS

In terms of the bottom line, this package is likely to mean the most to the user because it is a great help in sales and marketing decision-making. Although Sales Analysis is usually included in the Order Entry module in other systems, RealWorld has broken it out as a separate package.

The package offers sales analysis reports by customer, customer type, customer sales volume, responsible salesman, state, item, item category, and item sales volume. The reports are well thought out, flexible, and should prove very useful to virtually all users of the system.

ACCOUNTS RECEIVABLE

The customer file lies at the core of the A/R package. In addition to the usual name and address, this file contains last payment and account balance information, sales volume, and gross profit history, as well as several codes that allow the user to tailor the handling of each customer individually. The package provides for interactive maintenance of the customer file (including add, change/inquire, and delete functions), and either a numeric or alphabetic customer list may be printed upon demand. Miscellaneous customers may be set up in the customer file so that sales or cash receipts for one-time customers can be entered without first having to put the full name and address of such customers on file.

Sales transactions entered through the O/E package are automatically transferred to the A/R package. For users who do not purchase the O/E package and for other transactions, the system allows interactive entry, editing, and posting of sales and credit/debit memos. As sales are entered, the sale amounts, freight, and miscellaneous charges may optionally be distributed to one or more G/L accounts.

INVENTORY CONTROL

The primary purpose of an inventory package is to help a business control one of its greatest costs—that of keeping stock on hand for processing or sale to customers. Ideally, using information generated by the system, the user will arrange for the delivery of inventory items just before they are actually needed (sometimes referred to as "just-in-time" inventories). This package, which relies on a manually input reorder point, is relatively unsophisticated in this regard and is likely to prove less than satisfactory for companies with seasonal or rapidly shifting sales. This defect is mitigated somewhat by the ready availability of printed inventory reports, especially an ABC Analysis.

As in other inventory packages, the central file for this package, which may be used by itself or in conjunction with the O/E package, is the item file. In addition to the item number and description, each item record within the file contains three prices, discount and commission calculation codes, two costs, and various stock control and sales history figures.

Each inventory item may be priced according to any one of several flexible methods that are contained in user-maintainable tables. Item prices may be changed automatically by either a percentage or a straight dollar amount increment of the cost or an old price, or by applying a new price. A price list (including item number, description, category, and any fixed prices) may be printed upon request.

ACCOUNTS PAYABLE

The vendor file lies at the core of the A/P package. In addition to the usual name and address, this file contains information about the normal terms offered by the vendor, together with year-to-date and last-year purchases and discounts. The package provides for interactive maintenance of the vendor file (including add, change/inquire, and delete functions), and either a numeric or alphabetic vendor list may be printed upon demand.

Miscellaneous vendors may be set up in the vendor file so that transactions for one-time vendors can be entered without first having to put the full name and address of such vendors on file.

The system allows interactive entry, editing, and posting of accounts payable, transactions (including invoices, credit vouchers, adjustments, and prepaids). As each new transaction is entered, the system automatically assigns it a voucher number, calculates the due date, discount amount, and discount date (based upon the terms stored in the vendor record), and presents them as default values during data entry.

Each transaction may be distributed to a virtually unlimited number of G/L accounts. On request, new payables are posted to the A/P open item file, and the accounts payable transaction register is printed.

An accounts payable open item report may be printed on request for all or selected vendors. Four user-defined aging periods are provided, and aging may be done by analysis report—showing the total purchases and discounts (year-to-date and last-year) with percent of grand total figures for each vendor—may be printed.

In the generalized payment preparation selection mode, all current or past due items may be selected for payment. Additionally, any items for which the valid discount would be lost may be selected. Individual invoices or all invoices for a specified vendor may be selected for or deferred from payment, and partial payments may be made on any open invoices. After these decisions have been made, a pre-check-writing report showing all selected vouchers is printed, so that any needed adjustments can be made prior to actually printing checks.

As I have said before in this magazine, payroll preparation is a highly complex, ever-changing activity in which economies of scale generally pay off. A lot of software companies prepare payroll packages to keep up with their competitors, a lot of small companies buy those packages because they are included in the overall system, and a lot of those same companies end up having unhappy experiences because preparing payroll in-house is a difficult and time-consuming task.

Additionally, this is one area where the IRS and state payroll tax authorities tend not to be terribly tolerant of errors. Before you buy this or any other payroll package, seriously consider whether the size of your operation really justifies the resulting hassle and whether the whole thing couldn't be better handled by a bank or service bureau. With that in mind, let us proceed!

Central to any payroll system is the employee file, which in this case includes name, address, social security number, hire/review information, pay frequency and rate(s), taxable status and control information, vacation and sick pay control values, and the appropriate monthly, quarterly, and year-to-date figures.

Each employee may be assigned any of three fixed voluntary deductions (union dues, loan repayment, and wage garnishment) plus up to six user-defined deductions or earnings. A frequency code is assigned to each deduction or earning in an employee record. The system computes federal, state, and local withholding, payroll taxes, and workers' compensation premiums on the basis of tax tables, which are maintained by the user and may be printed on request.

CONCLUSION

On balance, I like this system a lot. The major drawbacks are (1) its inability to consolidate at more than one level; (2) the lack of a Purchase Order package; and (3) the lack of a Fixed Assets package. The first two objections should be met by the issuance of the Job Costing package at the end of the year and the subsequent issuance of a Purchase Order package.

The third objection will probably be sufficiently serious to turn off potential users whose requirements include extensive tracking of fixed assets, simply because the only alternatives are a lot of hand bookkeeping (which negates the purpose behind an investment in a computerized system) or the purchase of someone else's Fixed Assets package (in which case you might as well buy that entire system).

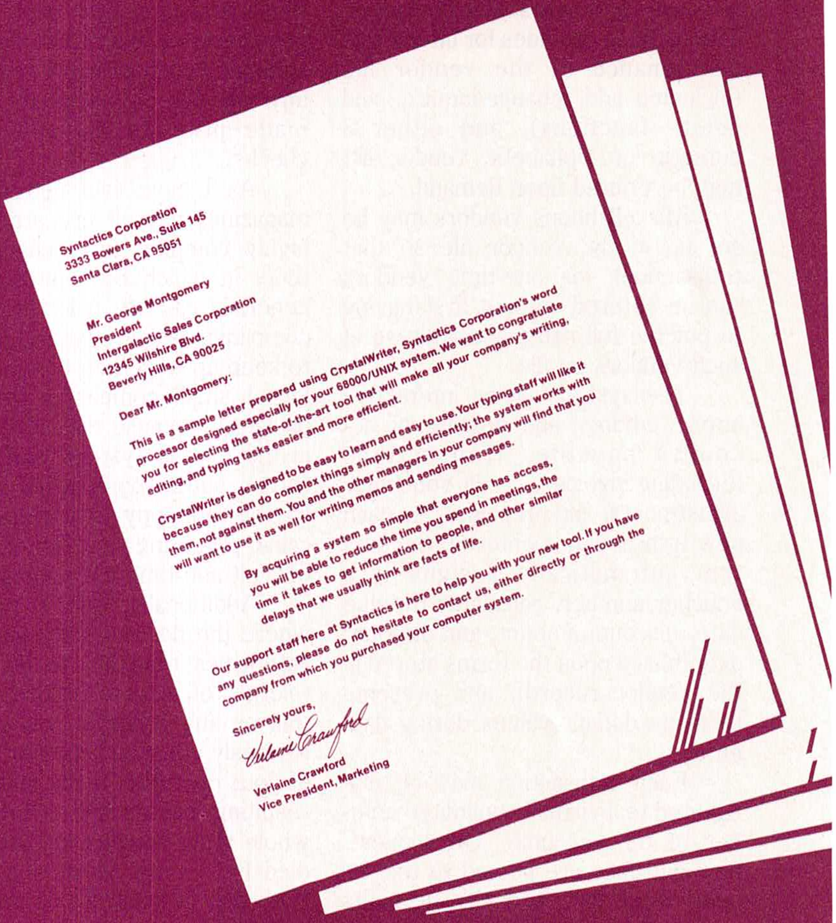
On the positive side, the packages in the sales cycle (Order Entry, Sales Analysis, and Accounts Receivable) are well designed and include numerous features that should result in greater sales efficiency and overall profitability for the system's users.

Similarly, the worksheets, edit lists, registers, and reports should all make the accounting function run more smoothly. The audit trail is clear and difficult to bypass. Finally, the data integrity check programs are a nice feature that should save a lot of controllers and accounting managers a lot of grief.

This system should prove attractive to service businesses as well as wholesale and retail distributors. And it should be able to serve their needs through several stages of growth, probably until they

CrystalWriter™

Complete Word Processing for the UNIX™ System



"...one of the best kept secrets in the world of UNIX System application software..."

"...a truly superior package..."

David D. Coleman
UNIX/WORLD Vol. I No. 3

AVAILABLE NOW, CrystalWriter from SYNTACTICS is the complete word processing system designed to support the first time user–without ignoring the experienced word processor.

For more information on CrystalWriter, call today

800-626-6400
(In California (408) 727 6400)

SYNTACTICS
3333 Bowers Avenue, Suite 145
Santa Clara, CA 95054

SYNTACTICS™

Please circle Ad No. 41 on inquiry card.

have an in-house DP manager. Manufacturing companies, on the other hand, are likely to be turned off by the single level of consolidation, the lack of a Fixed Assets package, and the weaknesses in the Inventory Control package.

If I were buying the system and wanted to be in the business of payroll preparation, I would buy all of the packages. Otherwise, I would skip the Payroll package and buy the rest.

Nicholas L. Feakins is a partner in the Sunnyvale, Calif., certified public accounting firm of Feakins & Feakins. He holds an A.B. from Dartmouth College, a J.D. from Stanford University Law School, and an M.B.A. in accounting from UC Berkeley. Mr. Feakins has served as chief financial officer for several companies in "Silicon Valley." He has provided litigation support and has served as an expert witness in a number of lawsuits involving computer-related issues.

RealWorld Responds:

RealWorld has written and delivered to one OEM a revised installation program that permits a user to install our software in 15 minutes.

COMPANY INFORMATION

The RealWorld Accounting System was developed by the RealWorld Corp. Further information about the system may be obtained by contacting:

RealWorld Corp.
Dover Road
Chichester, NH 03263

Tel: 603/798-5700
800/255-1115

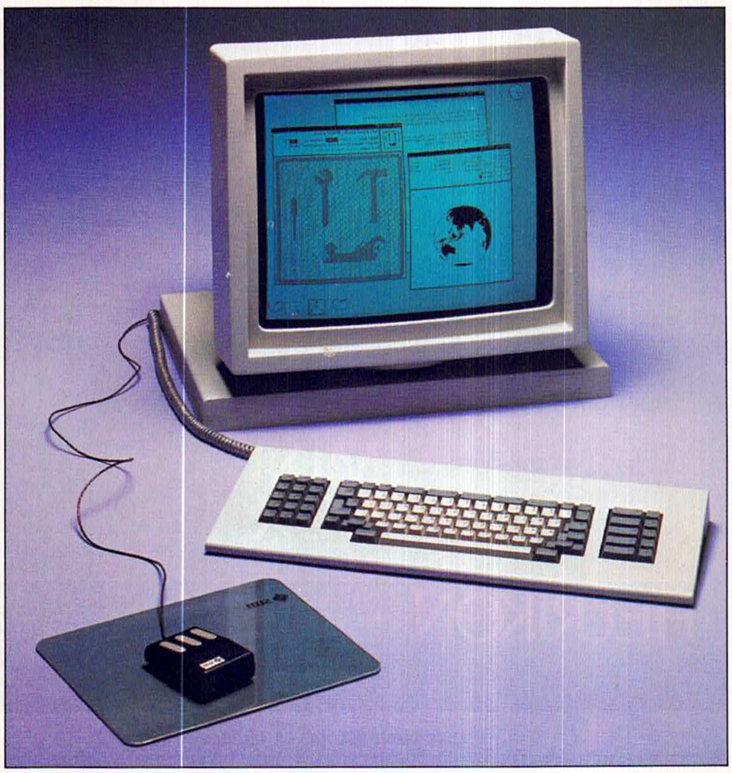
The system is priced at $695 per package, except for the Sales Analysis package, which is priced at $350. Buyers may be able to negotiate price breaks from dealers, especially if they purchase the entire system.

Has the future of intelligent workstations arrived? UNIX/WORLD reviews this bright spot from Sun.

The SUN-2

BY BRUCE MACKINLAY

When Bill Joy, one of the founders of Sun Microsystems, recently spoke on the future of the Unix system, he talked of bit-mapped graphics, networking, advanced word processing, and, most important, computer systems designed for people. What he said went over the heads of half the audience, and the other half shook their heads and said, "What a starry-eyed idealist!" or "Right, but what does that have to do with today?"



Interleaf asked UNIX/WORLD to review its software on the Sun Workstation, and I jumped at the chance to see how the Sun Workstation lives up to Bill Joy's dreams. After working with Interleaf, I chose to do separate reviews of the Sun hardware and Interleaf software.

The Sun-2 is primarily a single-user workstation, providing a sophisticated user with a powerful general-purpose environment. Sun is marketing this workstation to what it likes to call the *knowledge worker*. As I understand it, a knowledge worker is anyone who works with difficult intellectual tasks, including engineering, science, technical publishing, artificial intelligence, and all forms of research.

In the past, workers in these areas have either purchased minicomputers or used mainframes, spending $50,000 or more per researcher. Mainframes provided the researcher with tools and number-crunching, but the price of restrictive centralized control and batch processing was too much to pay. In desperation, many researchers turned to minicomputers, which were not much more than bare computers and which provided researchers with little more than a Teletype and an assembler.

It was from these early minicomputers that the Unix system arose. The high price and the "unfriendliness" of these older sys-

Workstation

tems severely limited the number of researchers able to use them. With the new 16- and 32-bit microcomputers, a number of companies, including Sun Microsystems, Apollo, and Valid Systems, have entered this market with a vengeance, producing "low-cost" research tools.

MULTIPLE WINDOWS AND MICE

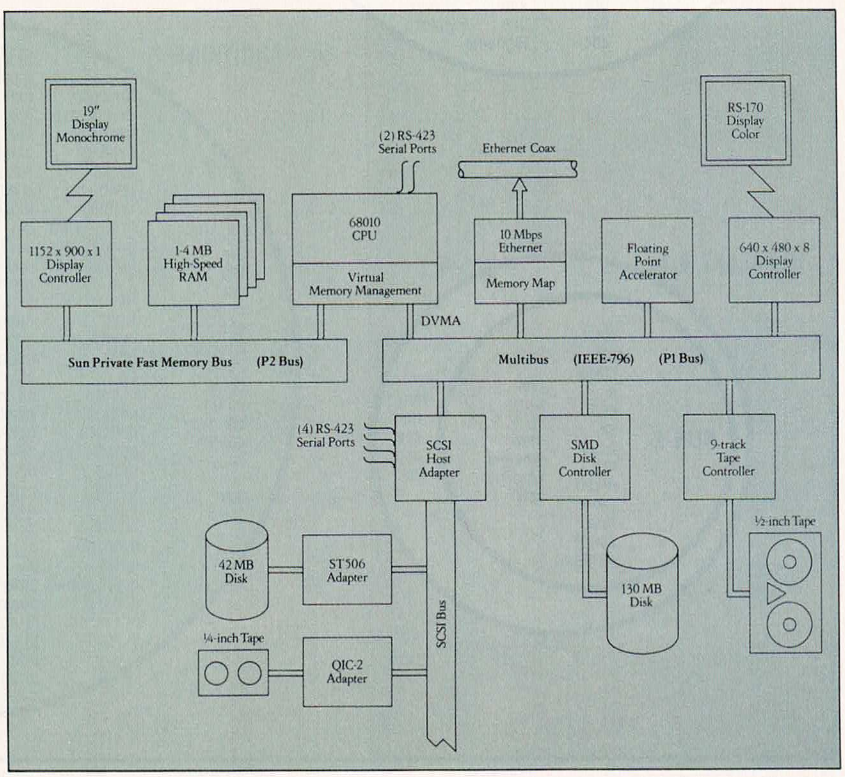
The first thing you notice when you see a Sun Workstation is the 19-inch monochrome display, containing multiple windows and a mouse. Underneath the windows is Berkeley 4.2BSD Unix, extending the machine into a local-area network of cooperating Sun Workstations, super-minis, and mainframes. Within the windows runs a wide selection of software, including some of the latest examples of leading-edge research and development.

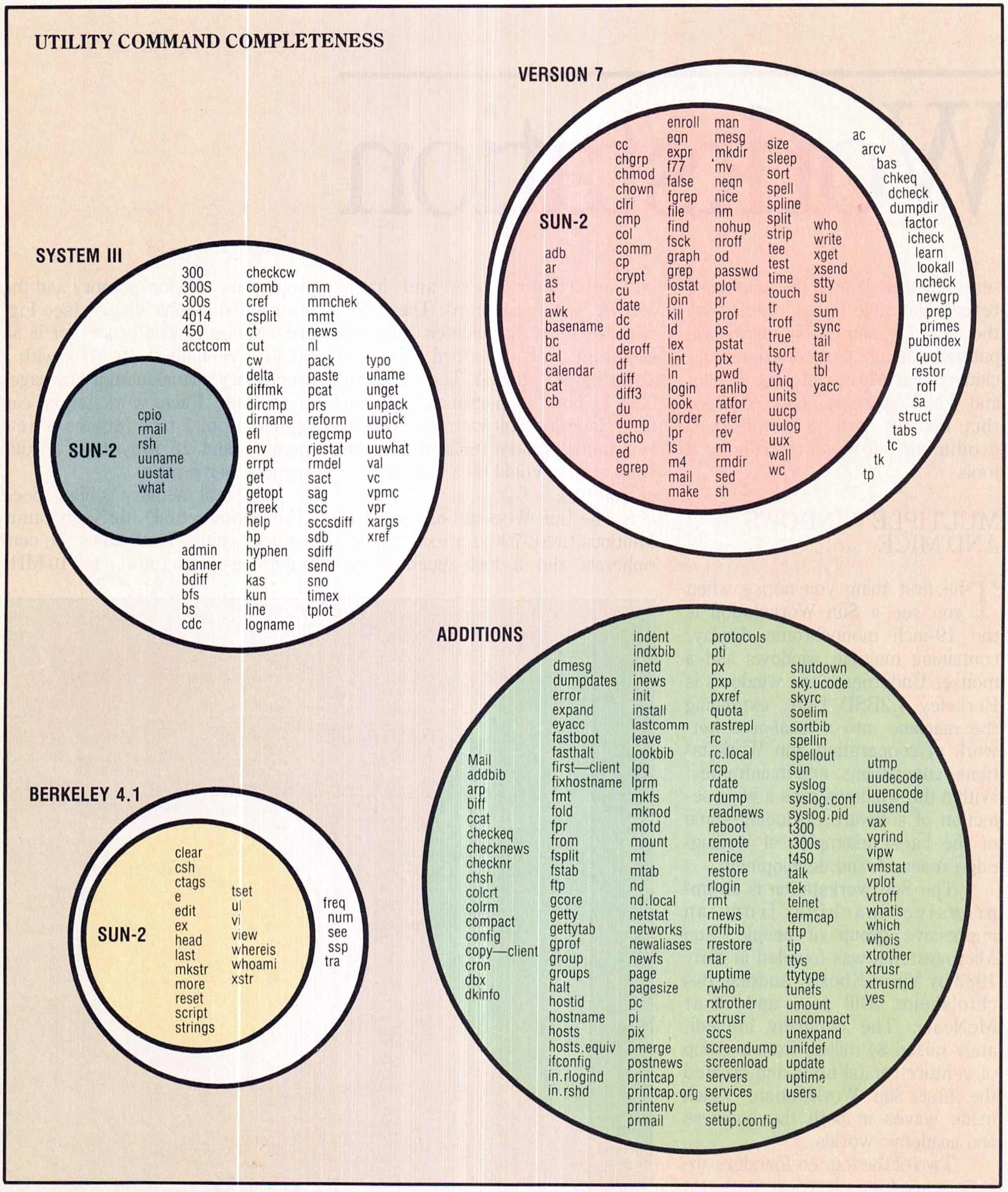
The Sun workstation is an impressive machine from an impressive group of people. Sun Microsystems was founded in early 1982 by Vinod Khosla, Andreas Bechtolsheim, Bill Joy, and Scott McNealy. The company immediately raised $4 million from a group of venture-capital firms and created the first Sun Workstation, which made waves in both the business and academic worlds.

Two of the four co-founders are experienced in electrical CAD/CAM and microcomputers and have M.B.A.'s from Stanford. The other two are Ph.D. candidates from UC Berkeley and Stanford. Stories about Bill Joy abound. The one I like best is how he personally did the first Berkeley software distribution long before DARPA realized that the Unix system could be a valuable research tool.

The Sun Workstation uses the Multibus (IEEE-796) for external peripherals and a high-speed 32-bit proprietary bus for memory and the bit-mapped graphic display (see Figure 1). The central processor is an M68010 running at 10 MHz with a proprietary virtual memory management unit. Each workstation can support from 1 to 4 Mbytes of real memory and 16 Mbytes of virtual memory per process.

The real memory is high-speed (150-nanosecond) 64K dynamic memory chips. This allows the central processor to run at a full 10 MHz




UTILITY COMMAND COMPLETENESS
VERSION 7
SUN-2
enroll eqn expr f77 false fgrep file find fsck graph grep iostat join kill ld lex lint ln login look lorder lpr ls m4 mail make
man mesg mkdir mv neqn nice nm nohup nroff od passwd plot pr prof ps pstat ptx pwd ranlib ratfor refer rev rm rmdir sed sh
cc chgrp chmod chown clri cmp col comm cp crypt cu date dc dd deroff df diff diff3 du dump echo ed egrep
adb ar as at awk basename bc cal calendar cat cb
size sleep sort spell spline split strip tee test time touch tr troff true tsort tty uniq units uucp uulog uux wall wc
who write xget xsend stty su sum sync tail tar tbl yacc
ac arcv bas chkeq dcheck dumpdir factor icheck learn lookall ncheck newgrp prep primes pubindex quot restor roff sa struct tabs tc tk tp
SYSTEM III
300 300S 300s 4014 450 acctcom
checkcw comb cref csplit ct cut cw delta diffmk dircmp dirname efl env errpt get getopt greek help hp hyphen id kas kun line logname
mm mmchek mmt news nl pack paste pcat prs reform regcmp rjestat rmdel sact sag scc sccsdiff sdb sdiff send sno timex tplot
typo uname unget unpack uupick uuto uuwhat val vc vpmc vpr xargs xref
SUN-2
cpio rmail rsh uuname uustat what
admin banner bdiff bfs bs cdc
BERKELEY 4.1
SUN-2
clear csh ctags e edit ex head last mkstr more reset script strings
tset ul vi view whereis whoami xstr
freq num see ssp tra
ADDITIONS
Mail addbib arp biff ccat checkeq checknews checknr chsh colcrt colrm compact config copy—client cron dbx dkinfo
dmesg dumpdates error expand eyacc fastboot fasthalt first—client fixhostname fmt fold fpr from fsplit fstab ftp gcore getty gettytab gprof group groups halt hostid hostname hosts hosts.equiv ifconfig in.rlogind in.rshd
indent indxbib inetd inews init install lastcomm leave lookbib lpq lprm mkfs mknod motd mount mt mtab nd nd.local netstat networks newaliases newfs page pagesize pc pi pix pmerge postnews printcap printcap.org printenv prmail
protocols pti px pxp pxref quota rastrepl rc rc.local rcp rdate rdump readnews reboot remote renice restore rlogin rmt rnews roffbib rrestore rtar ruptime rwho rxtrother rxtrusr sccs screendump screenload servers services setup setup.config
shutdown sky.ucode skyrc soelim sortbib spellin spellout sun syslog syslog.conf syslog.pid t300 t300s t450 talk tek telnet termcap tftp tip ttys ttytype tunefs umount uncompact unexpand unifdef update uptime users
utmp uudecode uuencode uusend vax vgrind vipw vmstat vplot vtroff whatis which whois xtrother xtrusr xtrusrnd yes

with no wait states and without the complexity or expense of memory caching. The HP 200 has a 16-Kbyte cache to allow the processor to achieve the same performance (see UNIX/WORLD Vol. 1, No. 1).

The processor also supports *direct virtual memory access* so that a peripheral can access a byte anywhere in virtual memory, using the same mapping and protection as any process in the system. The virtual memory management unit uses both segment and page translation.

The Sun is faster than the Apollo in most ways, but the Apollo eats floating-point operations for breakfast.

Segment translation allows the operating system and compilers to organize memory into useful structures and shapes. Page translation improves hardware efficiency, allowing the operating system to easily page in and out virtual memory.

Berkeley 4.2BSD Unix supports these features, allowing users to write and run a program that could use a full 16-Mbyte memory on the M68010.

PERIPHERALS

The Multibus handles all input/output. Among the peripherals supported are Ethernet, an SMD hard disk, an SCSI bus adapter handling both a hard disk and 1/4-inch cartridge tape, a 9-track 1/2-inch tape drive, and a floating-point accelerator. The central processor supports two RS-423 serial lines, and the SCSI adapter supports four RS-423 serial lines. RS-423 is a new standard serial interface that is compatible with the older, slower RS-232, but it will allow the upward movement to RS-449 (which will allow serial communications at speeds

BENCHMARK MEASUREMENTS

Arithmetic Instruction Times Without Floating Point Accelerator (microseconds per op)

	Short	*Long*	*Float*	*Double*
+Add	2	957ns	228	127
*Multiply	23	28	294	182
/ Divide	14	32	417	311

Arithmetic Instruction Times With Floating Point Accelerator (microseconds per op)

	Short	*Long*	*Float*	*Double*
*Add	2	960ns	66	41
*Multiply	23	28	72	50
/ Divide	14	32	103	79

Memory Loop Access Times (nanoseconds per byte)

	Read	*Write*	*Copy*
Char type	1	2	2
Short type	575ns	924ns	815ns
Long type	414ns	671ns	645ns

Input/Output Rates (bytes/sec)

	Read	*Write*	*Copy*
Disk	233K	188K	69K
Disk-less	117K	109K	40K
Pipe			171K
TTY 1		702	
TTY 1+2		725	
RAM 1-byte			621K
RAM 4-byte			1552K

Array Subscript References (microseconds)

short[]	long[]
4	4

Function References (microseconds/ref)

0-parameters	1-parameter	2-parameters
funct()	funct(i)	funct(i,i)
13	19	24

C Compiler Performance

1301 symbols, maximum

CPU estim: 8.5 sec + 25 lines/sec

Real estim: 1 X CPU

up to 1 million bits per second).

The RS-449 standard maximum rate is 50 times faster than that of the RS-232. There are a lot of advantages to the RS-449 interface, but the major disadvantage is the large pool of existing RS-232 terminals and computers. These serial lines can be used as terminals, with the workstation acting as a traditional supermicro. The benchmarks show that a Sun could support terminals on all six of these lines without producing unacceptable slowness; none of the benchmarks showed any evidence of thrashing.

The Sun workstation is more than just a pretty toy (like the Macintosh).

The SMD hard-disk controller can support disks in three sizes: 65 Mbyte, 130 Mbyte, and 380 Mbyte. The SCSI adapter supports only a 42-Mbyte disk. The SCSI is slow, with an average access time of 41.3 milliseconds (ms) and a maximum data transfer rate of 5 Mbit/sec. The SMD disk is much faster, having an access time of 28.3ms and a maximum data transfer rate of 15 Mbit/sec. SMD disk controllers support very efficient direct memory access, further reducing the demands on the processor and operating system.

The Sun Workstation also supports a "disk-less mode" in which all disk I/O goes through Ethernet. We were able to benchmark a disk-less workstation, and the disk-less system showed only 10 to 20 percent reduction of performance. In fact, a disk-less system going to an SMD file server is faster than a direct SCSI hard disk.

Comparing the Sun Workstation to the Apollo, we see that the Sun is faster in most respects, but the Apollo eats floating-point operations for breakfast (see UNIX/WORLD Vol. 1, No. 4). Even when the Sun is augmented with a floating-point accelerator, it does not touch the Apollo's raw floating-point speed. Sun Microsystems used the SKY floating-point accelerator. I benchmarked the Sun Workstation with and without the floating-point accelerator.

The floating-point accelerator improved floating-point operations on the Sun by as much as 400 percent, but even with these improvements, the floating-point operations on the Apollo are from 3 to 11 times faster than on the Sun. Even the short and long operations are faster on the Apollo. Sun is faster in both memory I/O (about 2 times faster) and disk I/O (about 2 to 3 times faster), and in most other aspects the Sun outperformed the Apollo.

COMPANY OVERVIEW

Company name:	Sun Microsystems	
Public/Private:	Privately held	
In Business for:	2 years	
Headquarters:	2475 Garcia Avenue Mountain View, Calif. 94043 415/906-1300	
CEO:	Scott McNealy	
VP Marketing:	Carol Bartz	
General Sales Contact:	Joe Roebuck, vice-president sales 415/906-1300	
	This Year	Last Year
Gross Revenue:	$39 Million	$9 Million
Net Income:	$2.5 Million	$700,000
Employees:	400 plus	100
% of total expense spent on R&D:	12%	12%
Units Shipped:	1500	400

Major support centers:
- Palo Alto, Calif.
- Lexington, Mass.
- Frankfurt, West Germany
- Plus 14 other field sales/support offices

Major funding:
- 1st round — $4.5 million
  - Kleiner Perkins, TDI, U.S. Venture Partners, West Coast Venture Capital.
- 2nd round — $11 million
  - Numerous institutional investors

NETWORKING

The Sun Workstation supports TC/ICP networking over Ethernet at 10 million bits per second because it supports Unix 4.2. Unix 4.2 provides the best networking environment in the Unix world, with remote log-in, remote shells, and a whole set of remote utilities. Sun is working on the next step, the remote file system. This will allow a person to `rmount` a remote file system onto the local file system.

This capability was planned for the Berkeley 4.2 release, but Bill Joy did not have the time to write the code. I was told that Sun has finished the code and that all that needs to be done is to integrate the code into the kernel. A number of other leading-edge computer manufacturers are rushing to produce this remote file system. AT&T plans this capability for a future Unix VI or VII. The only question now is who will produce a good, deeply integrated network file system.

I was very pleased to see the depth of integration of the network on the Sun Workstation. The laser printer I was using was on its own workstation shared by over 100 different Sun Workstations. The list of machines on the different networks at Sun world headquarters contained over 300 machines.

The other advantage of 4.2 networking is compatibility with other 4.2 machines including VAXs, the interesting new Pyramid RISC machine, and soon the IBM 3081 mainframe. When Sun finishes the remote file system, they plan to give the software to these other noncompetitive machines. Soon we will see the realization of the dream of a network of workstations and background number-crunchers.

HARDWARE OVERVIEW

Model:	Sun-2/120 Workstation
Price:	$16,900 + $3,300 + $3,300 + $13,900 + $4,000 + $3,500
Configuration:	3 Mbyte of RAM, 130 Mbyte hard disk, black and white 900 x 1152 high-resolution, KBD, mouse, Ethernet LAN, floating point
Related Models:	Sun-2/170 rack mountable
First Delivered:	1982
Processor	
CPU:	M68010
Cycle Time:	12.5 MHz
Bus:	Nine-slot IEEE-796 (Multibus)
Min. Memory:	1MB DRAM
Max. Memory:	4MB DRAM
Floating Point:	IEEE co-processor.
STORAGE MEMORY	
Floppy:	none
Winchester:	42-380MB (several can be placed on one machine)
Backup:	1/4-inch cartridge tape 1/2-inch nine-track reel tape
NETWORK HARDWARE	
LAN	Ethernet
Software:	Unix 4.2 (BSD) Suntools (windows) Remote file system
Shell:	Bourne, `csh`, suntools
Libraries:	Complete
Utilities:	Complete
Languages:	Pascal, C, FORTRAN-77

WINDOWS AND GRAPHICS

There is more to good graphics than just a large bit-mapped display and laser printers; these are just the raw materials that good graphics are built upon. To have good graphics requires software, lots of software, much more than a low-level subroutine library. The Sun Workstation provides a lot of software, and the independent software provides tools. The Interleaf Office Publishing System software provides very pretty word-processing and very advanced graphic presentation tools. Q-CALC is a spreadsheet with graphics, similar to Lotus 1-2-3.

Donald Knuth's TeX and MetaFont typesetting allows very advanced typesetting of technical and foreign language composition. There are also a number of different CAD/CAM/CAE software systems with extensive graphics and windows. Sun provides these third-party software developers extensive tools to develop graphic and window software. The content of these tools raises bloody debates between enthusiasts of different approaches. I have always believed that graphics tools should insulate the programmer from the messy hardware details.

The tools are fairly high level, with very few hardware details showing through. What most impressed me was the ease with which graphics designed for the large resolution display could be displayed on the medium-resolution color display. The window tools are also fairly high level. The drive-less workstation I reviewed had both monochrome and color displays. John Gage of Sun Microsystems demonstrated the ability to move from one display to the other as if they were one screen. He could open a window on the left screen and then move the mouse to the right and open a window in the right screen.

> **Sun is marketing this workstation to what it likes to call the *knowledge worker.***

UTILITIES AND OTHER SOFTWARE

As you can see from the list of third-party software, the Sun workstation is more than just a pretty toy (like the Macintosh). There is a full set of 4.2 Unix utilities, including `rcp`, `rdate`, `rdump`, `rlogin`, `rsh`, `rnews`, `rrestore`, `rtar`, `ruptime`, and `rwho`. `rtar` was used to download the Aim benchmarks from a tape drive on a remote machine, creating files and directories on the local machine. Most of the System III commands were missing.

The only useful System III command included is `cpio`. Missing

SOFTWARE OVERVIEW

Sun Microsystems has a very aggressive third-party software program, and the following software packages are currently available from Sun Microsystems:

OFFICE AUTOMATIONS	
C-CALC	Electronic spreadsheet
CCA EMACS	Full-screen editor
EMACS	Full-screen editor
Horizon	Office automation software
Interleaf	Text and graphics document
LEX	Menu-driven word processing
MISTRESS	Relational DBMS
Q-CALC	Spreadsheet with graphics
RAPID/USE	Prototype constructions
SCRIBE	Document formatter
MicroIngres	Relational DBMS and application tools
TeX82 & TeX	Technical composition and typesetting
Troll/USE	Relational DBMS
WordMARC	Word processing
XED	Document processing
XDE	Data entry and data edit facility
GENERAL ENGINEERING	
MACSYMA	Computer algebra
ELECTRICAL ENGINEERING	
CADroid	Logic design system
HSPICE	Circuit simulator
REDUCE	Logic reduction
SILOS	Logic simulator
LANGUAGES AND SOFTWARE TOOLS	
DYALOG APL	APL interpreter
FRANZ LISP	LISP interpreter and compiler
Lattice C	Lattice C cross-compiler for Apx86
MAINSAIL	MAINSAIL compiler
Pascal-2	Optimizing Pascal compiler
Q'Nail	Interactive programming environment
RM/COBOL	GSA-certified ANSI standard COBOL compiler
Telesoft-Ada	Partial implementation of Ada
MECHANICAL DESIGN	
Amethyst	Interactive solid modeling
CatSolid	Computer-aided engineering
GENERAL GRAPHICS	
Dataviews	Interactive Color Graphics Package

Version 7 commands are unimportant, except for `dumpdir` (which has been replaced by the `-t` option of `restor`) and `learn`. I was surprised that some Berkeley 4.1 commands were missing, but the missing commands were of very little importance. I like the `see` and `num` commands, but these are very easy to duplicate.

The list of additions includes a large number of useful commands. The Sun includes C, FORTRAN (`f77`), and Pascal. Third-party languages include APL, LISP (FRANZ LISP), MAINSAIL, Q'Nial, RM/COBOL,

There is more to good graphics than just a large bit-mapped display and laser printers.

and Telesoft-Ada. Sun also has an aggressive third-party software program and will extend its software base even further.

In the Sun Workstation I can see the future Bill Joy described, but much work will have to be done before we can reach this future. The system is much more integrated than its predecessors, but it is still very fragmented. Major work needs to be done connecting the Sun to other systems. The cost is still very high—I won't have a Sun Workstation on my desk for a number of years. If the price of a drive-less workstation dropped to $5,000 or less, then we would see a real beginning of the intelligent workstation.

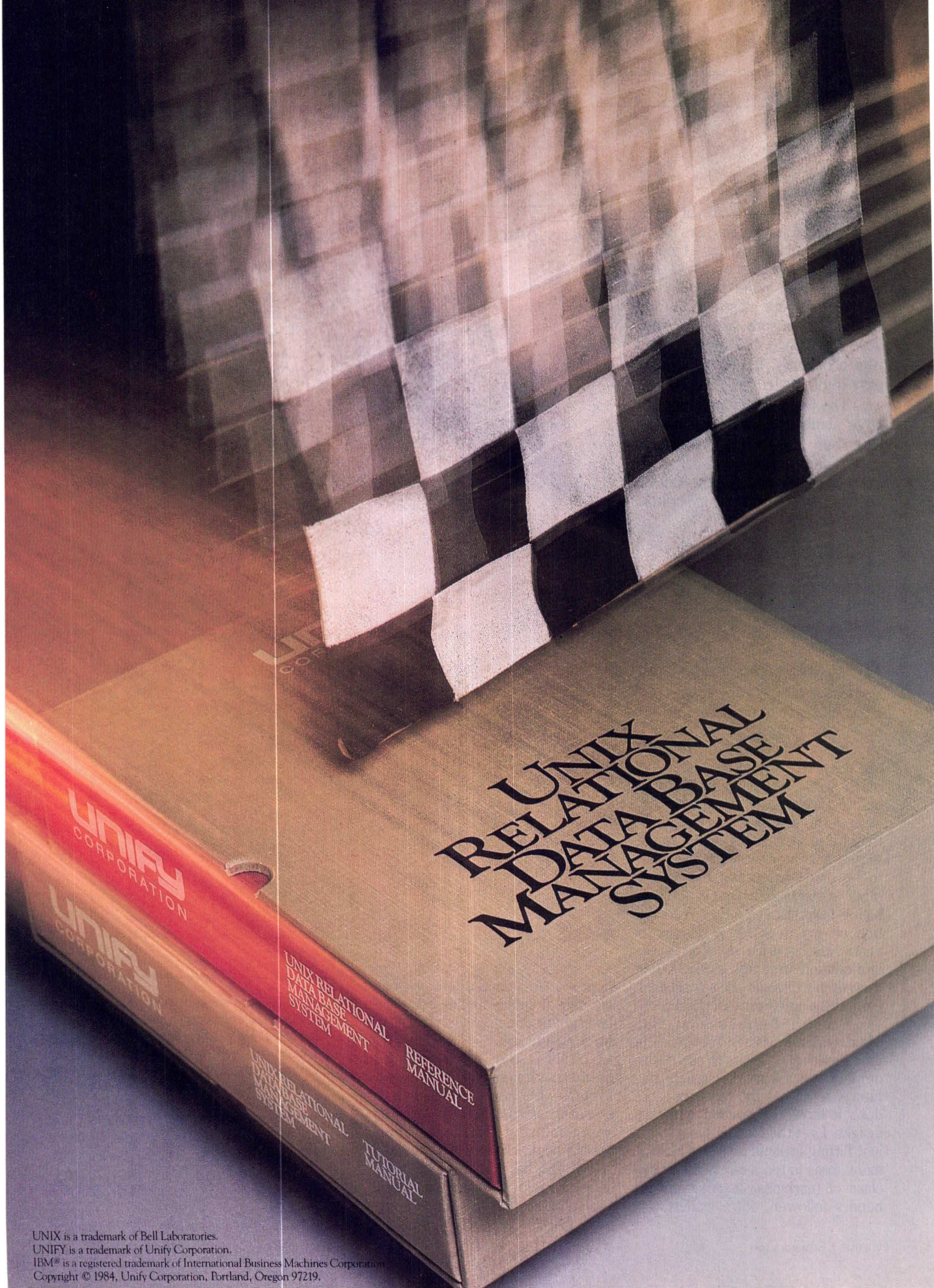
I don't believe that Sun Microsystems has the resources to fund the level of research and development necessary to reach this future. AT&T has the resources, but does it have the needed guts and insight. I get the feeling that AT&T is just flitting around, and IBM has too many irons in too many fires. In this class of machines, IBM has always been a follower, not a leader. Sun hopes that cooperation among a number of noncompeting companies and government researchers will forge this future.

This is why Sun is willing to license out its jewel—its remote file system—because in exchange the company would receive important research that it cannot afford to do. DARPA could change this, however, by getting back into the funding of Unix system research on a large scale. ■

ERGONOMIC EVALUATIONS

Packaging:	Console	Good, but large for desk
	System Box	Good, desk side
	Printer	Separate (Imagen laser printer)
	Fan noise	Ok, noticeable
Monochrome:	Graphics	Excellent
	Characters	Excellent
	Brightness	Excellent
	Grays	Excellent
Color:	Graphics	Good, medium resolution
	Characters	Good
	Brightness	Good
	Colors	Excellent
Keyboard:	Action	Good
	Special Keys	Excellent
	Weight	Good
	Mouse	Excellent, optical
Mass Storage:	Hard Disk Noise	Ok, noticeable
	Backup Tape Access	Excellent
Software:	Languages	Excellent
	Kernel	Good, 4.2BSD
	Version 7 Commands	Ok, some missing
	System III	Poor, many missing
	Berkeley Commands	Excellent
	Additions	Many for network and windows
	Applications	Excellent, aggresive third party
	Games	none
	Debuggers	adb

Bruce Mackinlay studied computer science, electrical engineering, and math at UC Berkeley. He is now a senior partner in NOVATECH Systems, Inc., a software development and consulting firm. He is currently working with UNIX/WORLD Magazine in developing its computer facilities. His last article for the magazine, a review of Honeywell's Scorpion, appeared in Vol. 1, No. 3.


UNIX RELATIONAL DATA BASE MANAGEMENT SYSTEM
unify CORPORATION
UNIX RELATIONAL DATA BASE MANAGEMENT SYSTEM
REFERENCE MANUAL
unify CORPORATION
UNIX RELATIONAL DATA BASE MANAGEMENT SYSTEM
TUTORIAL MANUAL

The INTERLEAF Publishing System

BY BRUCE MACKINLAY

Our fearless reviewer examines the Interleaf Office Publishing System. He finds it a winner —with a few loose ends.

The Interleaf Office Publishing System is a document manipulation package that is much more than a word processor. You can divide the Interleaf software into three integrated parts: (1) text manipulation with automatic proportional formatting; (2) a diagramming and drawing tool that allows users to create diagrams within the document; and (3) a charting tool that lets users create and manipulate business graphics. But it's the level of integration of these different parts that helps make Interleaf unique.

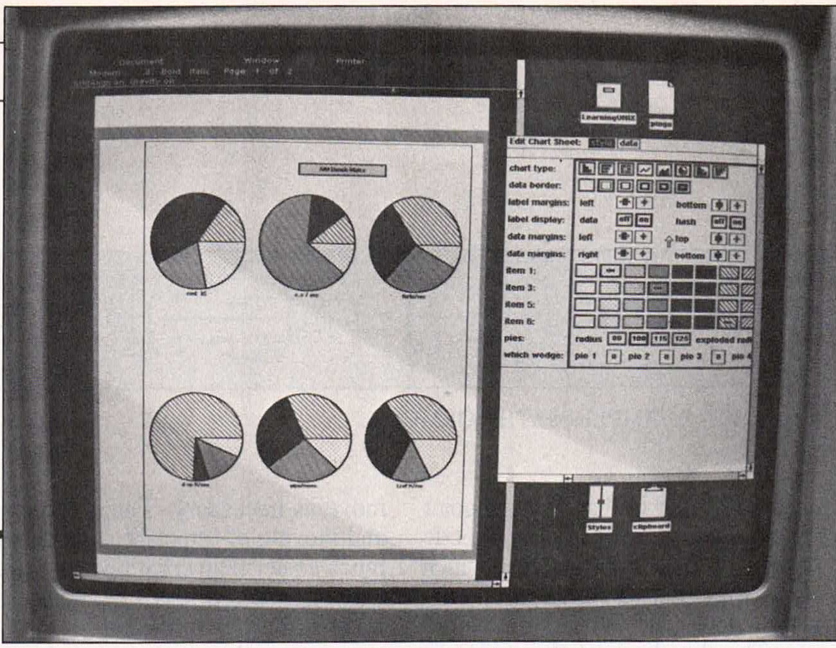
With older tools like Lotus 1-2-3, you can create the graphics, but you must resort to scissors and paste to include them in a document. The Macintosh has MacWrite and MacPaint, but again you have to resort to scissors and paste. Neither Lotus 1-2-3 (and its ilk) nor the Macintosh provides the high-quality visual resolution that Interleaf does, and Interleaf combines these tools into one coherent package. The package's other special trait is its speed. While other "see-is-what-you-get" systems slow to the intolerable level with even small documents, Interleaf takes only a second to drastically re-format most documents.

Historically, Interleaf is derived from the Xerox Star and the Apple Lisa. But sales of both the Star and the Lisa have failed to reach even the most pessimistic manufacturer projections. Xerox removed the Star from the market after dismal sales, while the sales failure of the Lisa has been supplanted by the Macintosh's phenomenal success. The Star was not a complete failure, however, as Xerox did manage to sell some units to Fortune 500 firms, specifically to departments needing high-quality mixed text and graphics.

Currently, these departments use outside graphics and typesetting or a combination of word processing, graphics software (like Lotus 1-2-3), and a lot of paste. Any software replacing these time-honored methods must produce the same results without resorting to paste or going out of house. The people who run these departments are cynical and will ask hard questions, and successful marketing of the Interleaf Office Publishing System will require that it be complete, easy to use, and reasonably priced. And, most important, Interleaf must convince the same people who have seen and rejected the Xerox Star and Apple Lisa.

INTERFACE AND DOCUMENTATION

The user interface is very reminiscent of the Xerox Star, having icons for documents, folders, drawers, and cabinets. Even though



> **If anything will open the doors to the Fortune 500, it will be Interleaf's charting tools.**

I was distressed to find that Interleaf runs independently of the Sun windows, I can understand why Interleaf did this. Sun windows are unique to the Sun Workstation, and, in order to remain independent, Interleaf had to provide its own windows. Also, Sun windows in their raw form do not provide the different levels with folders, drawers, and cabinets that Interleaf needed to provide a good user interface.

I reviewed a draft of the new documentation, and it is a good example of what you can produce using the Interleaf package. The mixture of text and graphics is impressive in this 153-page document, with no need to resort to the familiar cut-and-paste method. While playing with the program, I had to resort to the manual only a few times. But when I did, it was hard to find what I was looking for. That I seldom needed to use the manual is a big plus for the software. However, that I had to search around in the manual to find what I wanted to know is a strike against the documentation.

The training portion of the manual looked good, but one thing I have learned is that users do not read manuals, even training manuals. Good software goes a long way to mitigate poor documentation. I suppose I'm very hard on documentation because I have written so much of it. For example, I expect that documents both explain and give step-by-step examples of how to do things. The reference section must be complete and the index robust—a training manual does not replace a reference manual.

While the Interleaf document was strong on examples, it was short on explanations. Another problem was the index, which was too small and which tended to use Interleaf's own terminology instead of terms found in common usage. In sum, the manual does not match the high quality of the software and needs work.

TEXT EDITING AND FORMATTING

At first I thought that comparing the Interleaf product to a standard word processor would be a little like comparing apples and oranges. After a little thought, I realized that to reach its goal, the Interleaf software must supply *at least* the same tools and capabilities as a standard word processor. I used the Arthur Naiman 100-point checklist described in his *Word Processor's Buyer's Guide* (New York: McGraw-Hill, 1983) to compare the product to other systems on the market.

The product received a very high score of 86.25 points, placing well above most dedicated word-processing systems, second only to

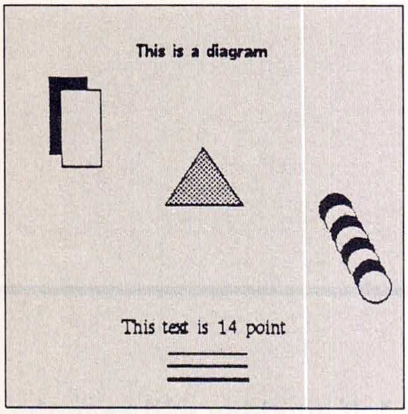


FIGURE 1: A PINGO DIAGRAM

$$f(x) = \sum_{i=-\infty}^{\infty} \left[C_i^k \left(\frac{x - t_i}{t_{i+k} - t_i} \right) + C_{i-1} \left(\frac{t_{i+k} - x}{t_{i+k} - t} \right) \right] B_i^{k-1}(x)$$

FIGURE 2: A SAMPLE MATH FORMULA

CPT. To surpass CPT and become acceptable to the Fortune 500, Interleaf must remove some of the more glaring failures of its product, including no underlining, no automatic hyphenation, no double or multicolumn, no automatic table of contents, and no automatic appendix or index.

The table handling is also very weak. I suggest that Interleaf consider integrating table handling and charting (see below). A table really is a special form of text/graphics and should be handled differently. Of all my complaints about this software, I think the lack of common word-processing features will be the greatest stumbling block to widespread sales to the Fortune 500 companies.

Also, a problem with the combination of graphics and text will cause serious difficulties. A rectangular graphic area can't have text (in the normal sense) running next to it on the page. To be fair, I must point out that Interleaf does true and continous on-screen formatting of the same quality produced by some of the best batch-mode formatters using laser printers.

It's hard to describe the advantages of using a mouse as a pointing device—you really have to use one to understand. Using a mouse is much more akin to using a pencil on paper in that you can point at the text you want and directly modify it. You don't have to inch along with the arrow keys or constantly scroll the text up and down.

The drawback of the mouse is that you must move your hand from the mouse to the keyboard, a movement requiring both time and energy. Good software requires this only when you are actually entering new text. Using this criterion, Interleaf is definitely good software. I have been told that a tracker ball is even better than a mouse, but I have yet to use one and am skeptical. (Of course, I also used to be skeptical about the mouse.) Arguments over pointing devices will rage until terminals are banned as tools of torture, along with the rack and thumbscrews.

DIAGRAMMING

With the Interleaf product, you can open a rectangular area that can contain either a diagram or a chart. A diagram can be anything from the company's logo (for the letterhead) to math formulas (see Figures 1 and 2). With the diagramming tools, you can create arbitrary boxes, lines, or ovals (including circles). I found it easiest to open a very large diagram area and create a collection of template figures (box, line, and oval) off to the side of the area. Then, when I needed a particular figure, I could duplicate the template, move it to the desired location, and adjust its size—a technique I used to set math formulas.

From four different fonts I collected a set of template math symbols that I could duplicate and move as necessary. This picking up and moving of template objects was much easier than creating new items when I wanted them. (Thanks to John Gage of Sun Microsystems for suggesting this technique.) I suggest that Interleaf adopt a palette of objects from which to select in order to replace the current method.

The big problem with most computer drawing is placing objects precisely, and Interleaf uses two methods to improve placement. The first, *grid alignment,* is useful when placing letters or math symbols into a diagram. Without grid alignment, it takes a delicate touch to place items on a line. Grid alignment "draws" a symbol onto the line. The second method is *gravity,* which is used when two objects need to touch. As in grid alignment, objects are "drawn" together so they simply touch. Again, it does not require a fine hand to make two objects touch. What I missed most was the ability to paint arbitrary shapes with a brush and a palette and to zoom in to manipulate a drawing at the bit level.

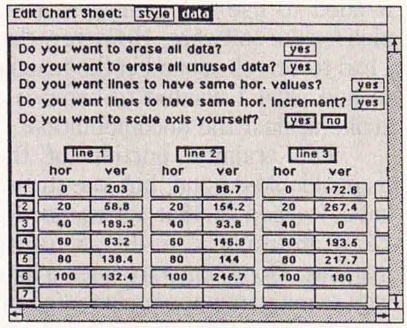


FIGURE 3: EDIT CHART DATA SHEET FOR A LINE CHART

CHARTS

I had the most fun with the charting tools. Figures 3, 4, and 5 are examples of the Data Sheet, the Style Sheet, and a Line Chart. Enter the data you want to chart into the Data Sheet, choose the chart type and style with the Style Sheet and, voilà, you get a chart like Figure 5. Even more impressive is the large selection of predefined charts in the Chart Wishbook. It contains over 40 different sample charts, including horizontal and vertical bar charts, line charts, pie charts, scatter diagrams, and more.

Also, you can add to the Wishbook any time you want. With other systems, taking the same data and plotting it 40 different ways would require weeks or months of effort. With Interleaf, you can play with a chart until it is perfect, and you can do this in an afternoon. If anything will open the doors to the Fortune 500, it will be Interleaf's charting tools.

I have very few complaints, but, since I want this product to succeed, I must complain. First, there is no easy way to bring an external table of numbers into a chart. This was a real problem when I went to chart the Aim benchmarks (see Figure 6). The benchmarks created a nice table of numbers, but I had to type them into the Data Sheet to chart them. Another gripe has to do with placing random labels in and around a chart. I would like to use the diagramming tools on charts so I could place comments.

The next suggestion follows from comments made by John Gage of Sun Microsystems. It would be nice to place equations into the data chart so I could do summations or linear regression analysis. It would also be nice to be able to place the numbers from the Data Sheet into the text, making a table. Even with these complaints, I have only the highest praise for the charting tools.

If Interleaf can meet two conditions, then they will succeed where Xerox and Apple have failed. First, the firm must fix the technical problems with its software and, second, Interleaf must create a sophisticated national sales and support organization that can talk to Fortune 500 companies in their own language. The current package is very, very good. Yes, there are loose ends, but Interleaf will clean them up. The only remaining questions are whether Interleaf can make the changes fast enough and whether the firm can set up the sales and support organization. Time will tell, but I wish them the best of luck!

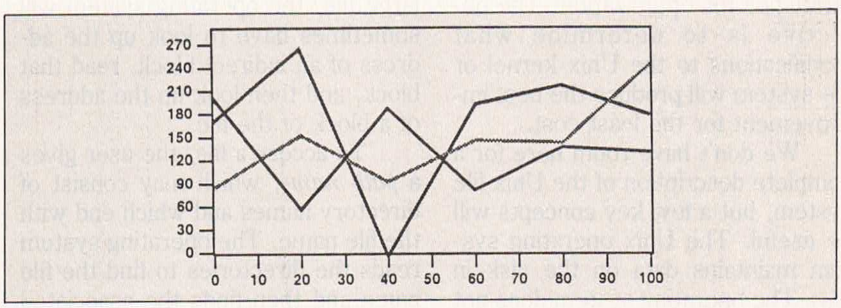


FIGURE 5: LINE CHART FROM THE WISHBOOK

FIGURE 6: AN EXAMPLE OF INTERLEAF'S CHARTING CAPABILITIES

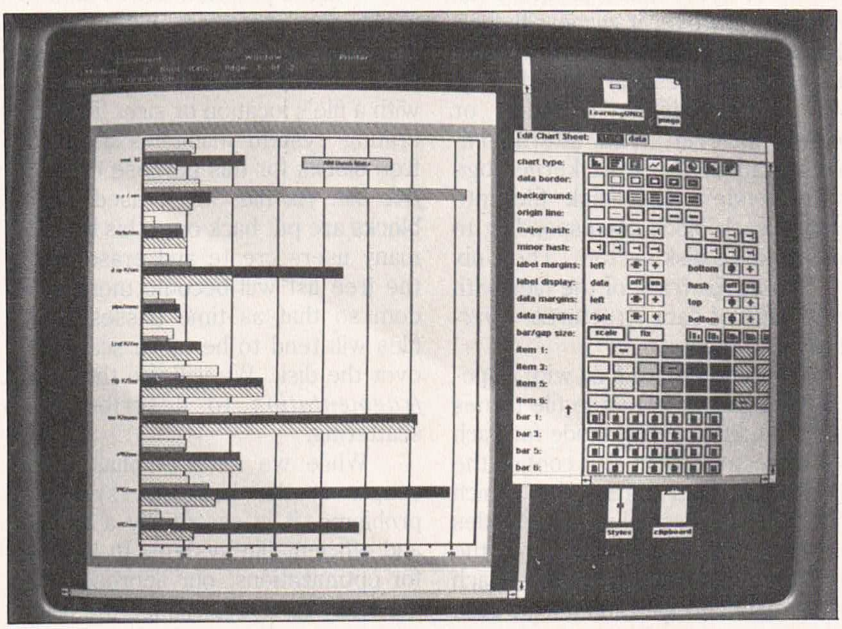
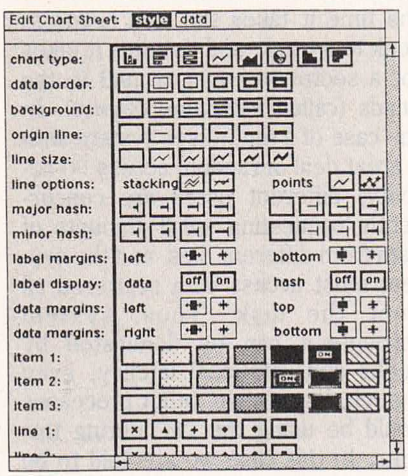


FIGURE 4: EDIT CHART STYLE SHEET FOR A LINE CHART



UNIX BLOCK I/O OPTIMIZATION ON MICROCOMPUTERS

BY DAVID ROBBOY

As microprocessors become more powerful, file systems become more of a bottleneck, particularly in microcomputer systems with low-cost hard disks. This article analyzes the sources of file system bottlenecks that arise in one Unix environment and presents some strategies for optimization. The objective is to determine what modifications to the Unix kernel or file system will produce the best improvement for the least cost.

We don't have room here for a complete description of the Unix file system, but a few key concepts will be useful. The Unix operating system maintains data on the disk in *files*. The operating system does not enforce any format or structure on the contents of a file; it is just a stream of bytes that a program can interpret however it wishes. In fact, the user does not have to distinguish between disk files and other devices such as terminals, printers, or pipes. However, in its internal implementation, the Unix kernel logically divides each disk file into *blocks*, each block corresponding to one or more disk sectors. The Unix system keeps track of the files with two kinds of data structures: *directories* and *inodes*.

Directories are files with a special type of content—the file names and the address of an inode for each file. Directories can also contain the names of other directories which are, after all, just files. The inodes are structures maintained on the disk by the operating system. Each inode maintains data about a file such as its size and the addresses of the blocks containing the data. When a file is large, there is not enough room in the inode for all the block addresses. In these cases, the inode contains the addresses of *indirect blocks*, which contain more addresses of blocks allocated to the particular file. Thus, to find data in a large file, the operating system will sometimes have to look up the address of an indirect block, read that block, and then look up the address of a block of the file.

To access a file, the user gives a *path name*, which may consist of directory names and which end with the file name. The operating system reads the directories to find the file name and then finds the associated inode, giving the system the information it needs to access the file.

When a program writes data to a file, the operating system automatically allocates more blocks so that the user need not be concerned with a file's location or size. The operating system maintains a list of free blocks for this purpose called a *free list*. As files are erased, their blocks are put back onto this list. As many users create and erase files, the free list will become more random so that as time passes, new files will tend to be more scattered over the disk. We will use the term *fragmentation* to describe this scattering.

While we have emphasized a couple of the Unix file system's problems, it is essentially a simple and efficient file system. In looking for optimizations, our approach will generally be not to change the basic file system design, but to look for implementations that avoid some inefficiencies.

TYPES OF BOTTLENECKS

The performance bottlenecks associated with disks fall into three general categories. If performance is limited by speed of data transfer, we are said to be "I/O bound." For example, when backing up a disk to a tape or another disk, the speed of the devices, controllers, and memory interfaces usually determines how fast the system will perform.

If system performance is limited by the CPU, we are said to be "CPU bound." This would arise in the case of a slow CPU combined with fast devices and complex software, so that system performance is limited by the processing of operating system requests and the dispatching of different processes. A program doing intensive calculations is, of course, CPU bound, but this article is more concerned with a situation where the operating system is CPU bound while doing I/O.

The third bottleneck is subtler. The overhead of each I/O access is a combination of both of the above, the time it takes the disk heads to seek to a track, and the time it takes for a sector to come around to the heads (called *rotational latency*). In the case of a multi-user system with a great deal of random access I/O activity, different users are concurrently requesting small amounts of data from different files, and the system must access data scattered all over the disk. Thus, system throughput can be dominated by seeks and rotational latency, even though, in principle, other processes could be using the CPU during this time. In this case we are said to be *access bound*. In such a system, data is not being transferred at the maximum rate of the I/O devices; yet, the CPU is often idle, waiting for I/O.

STRATEGIES

Several optimization strategies are described here, and each might be appropriate for a particular kind of I/O bottleneck. Faster disks or controllers are the obvious brute-force solution if the system is I/O bound, but we will also look at software strategies for the other types of bottlenecks.

"Read-ahead" means reading more data than a program actually requests on the assumption that it may be needed later. In case the system is access bound, it is better to do fewer, but larger, accesses to the disk. Read-ahead is useful in case of sequential file I/O, but the worst case involves random access of small records by many users, in which case reading ahead will be useless. In practice, however, some accesses are likely to be sequential, such as loading programs and printing spooled files.

One way of reading ahead is to increase the logical block size in the file system. A larger logical block size means that more data is read or written on each access, thus potentially reducing the number of accesses. However, this solution also increases the wasted space on the disk. On the average, at least half of a block will be wasted in a file because the end of each file will occupy only part of a block. This problem is particularly acute in inexpensive systems with limited disk space.

Another way to accomplish read-ahead is to cache entire tracks. Reading a track at a time can save many accesses if more blocks on the same track will be needed, or it can lead to thrashing if many users demand data scattered all over the disk. This strategy is likely to be of little use in the current Unix file system, where files are fragmented, but it may help in conjunction with a strategy that reduces the fragmentation.

BLOCK CACHING

Block caching, buffering blocks that have already been read, assumes that the same data or other data in the block will be needed again. It is both a form of read-ahead and a method of keeping frequently needed data available. The Unix system already caches blocks, but there is much room for more sophisticated caching algorithms to increase the probability that a needed block will be present in memory. There is also the brute-force method of simply storing more blocks.

Organizing the disk to reduce the number of seeks is a very important strategy. Currently, Unix system files tend to be fragmented all over the disk, with the result that we are access bound because of seeking. The fragmentation also frustrates attempts at read-ahead because, for example, each block of a file may be on a different track. Thus, it would seem worthwhile to look at strategies that keep files clumped together in contiguous blocks.

Even if files were not fragmented, concurrent random access by several users would still cause the system to make I/O requests all over the disk. Thus, reducing the fragmentation might not, by itself, significantly improve performance. However, if we know that the files are not fragmented all over the disk, then we can take advantage of other read-ahead strategies, as we shall see.

Organizing the file system can include other strategies than just reducing the fragmentation. For example, increasing the block size is a form of read-ahead, but it also means that a file will contain fewer blocks. Therefore, the file blocks can be found from the inode with less indirection, resulting in fewer accesses to the device.

Seeks may also be reduced by sorting the list of free blocks when files are deleted, so that newly created files will tend to occupy consecutive blocks. Another strategy is to rearrange the order of the partitions on the hard disk and to rearrange the locations of the inodes within the partitions to minimize the average distance of seeks between files and inodes.

Large sequential reads, such as loading of programs, will be faster if they go directly to the target area of memory, bypassing the buffer system altogether. This is known as *direct memory access*, or *DMA*; it has several benefits: (1) it loads a program by reading it all at once instead of a block at a time; (2) it saves the time it would take to move the data to a buffer and then to memory; and (3) it avoids filling up all the system buffers with a program, thereby saving data in the buffers that may be needed again.

CURRENT SYSTEM CHARACTERISTICS

Wayne Smith, of Intel Corp., has studied the behavior of Xenix-286 using several software tools he developed. One of these was a CPU histogram tool that interrupted the CPU at regular intervals and stored the location of the instruction pointer. Another program analyzed this data and printed a list of the most heavily used kernel functions and how much time was spent in them. Smith also added code in key places in the kernel to record how many I/O requests resulted in buffer hits, how many actual disk requests required seeks to new tracks, and similar data. To simulate the system's behavior in a more or less typical commercial environment, the tests were done with four to eight processes doing I/O-

-intensive, random-access disk processing.

In the case of large raw data transfers by a single process (such as a disk backup), almost any system will be I/O bound. On the other hand, we observed that for small records (32 bytes), both the system throughput and the amount of physical I/O drop off greatly, and the system is CPU bound by the sheer overhead of system calls and buffering. However, for the multitasking, random-access situation described above, the system was shown to be access bound. We will briefly and qualitatively summarize the test results that led to this conclusion.

Using a faster disk controller increased the raw disk throughput greatly, but Xenix system throughput for a disk-intensive, multi-user application increased much less. This shows that we are not predominantly I/O bound; that is, we are not moving data as fast as the hardware can move it. Using a faster CPU (the 286 versus the 8086) did not increase system throughput in proportion to the speed of the CPU, showing that we are not predominantly CPU bound. This leads to a conclusion that we are access bound.

This belief is supported by two other facts. One is that the kernel histogram shows that the operating system is idle much of the time; that is, all processes are waiting for something, and the CPU has nothing to do. The other is that the typical disk activity is to read one block and then seek to another track for the next block. Frequently these seeks cover a great distance and jump over many tracks.

We can make a couple of observations here. One is that the Unix operating system is designed for multi-users, and it attempts to smooth out the random seeking all over the disk by sorting the I/O requests by cylinder. In this way, it attempts to cause the disk heads to move more or less smoothly up and down the disk, and to do as many reads or writes as possible on one cylinder before moving on to the next cylinder. On a small system with just a few users, this may not work as well as it would on a larger system because there are not enough I/O requests to be statistically smoothable.

For example, if there is an average of only two requests in the I/O queue at any given time, then the disk is working all the time and processes are waiting for I/O. But the disk still is likely to have to seek to a new track for each request. Another observation is that if there were a process doing CPU-intensive activity such as number crunching, then these I/O bottlenecks would not matter very much; the system as a whole would be CPU bound. We have intentionally avoided this situation to discover what the I/O bottlenecks are, and it seems likely that in many commercial environments the I/O bottlenecks will be significant to system throughput.

BERKELEY FILE SYSTEM RESULTS

Our other empirical results come from the Berkeley Unix "Fast File System." Berkeley claims to have gotten about a tenfold throughput improvement by using essentially two types of strategies. One was a larger block size, and the other was a set of algorithms designed to keep directories and all their associated files and inodes clumped together in local areas of the disk called *cylinder groups*, which are discussed below. Berkeley did not fundamentally change the Unix file system; their changes can be thought of as optimizations of the implementation, and the performance improvements (if true) are impressive.

The larger block size is useful because, as we have mentioned, it is both a form of read-ahead and a way of reducing the amount of indirection needed to access a file. Also, it seems intuitively that it may help to reduce file fragmentation by breaking each file into fewer, larger pieces. Its disadvantage is that it wastes disk space, which is a worse problem for microcomputer systems than it was for a VAX system at Berkeley. This problem was dealt with by allowing a block to contain addressable fragments so that several small files can occupy parts of a block. This in turn necessitates some dynamic reorganization in case a file outgrows its block fragment.

Keeping the files clumped together in cylinder groups was a way of dealing with file fragmentation. Cylinder groups can be thought of as partitioning the file system into several smaller quasi-file systems, making the files in each one closer together and more compact. But the operating system controls the allocation of files in the various cylinder groups so the user doesn't have to actually mount many little file systems or refer to them by separate path names. Because blocks are allocated according to their location, this method necessitates maintaining a bit map of available blocks and some more complexity in maintaining the free list itself. Another complexity is that large files would fill up a cylinder group and leave no room for other files, which would defeat the attempt to clump them together. Therefore, as files grow large, additional blocks must be allocated for them outside of their cylinder group.

OPTIMIZATION STRATEGIES CONSIDERED

Since we are not CPU bound, there is limited value in improving the CPU performance. However, there may be cases where this is justified because it is easy, especially for such things as using machine DMA instructions to move data around in memory or clear buffers.

The brute-force solution to block caching is to have more buffers. As processors and memory become faster and cheaper, the block I/O becomes relatively more of a bottleneck on microprocessor systems. At the same time, however, brute-force solutions, such as larger buffer caches, become relatively more feasible.

REDUCING ACCESS LATENCY

The latency of each access depends on the time needed to seek to the track and for the sector to rotate around to the heads. An important method of reducing the number of seeks is to reduce the fragmentation of files, discussed below. A method of reducing the average seek distance is to rearrange the order of partitions (that is, file systems) on the hard disk and the locations of inodes within the partitions, thus reducing the average distance of seeks. This is possible independently of the actual implementation of the Unix system. Finally, the rotational latency can be reduced by having the appropriate sector interleave for the particular combination of disk, CPU, and operating system. Most disk controllers automatically interleave the sectors when the disk is formatted so the operating system software does not have to be concerned with this.

A way to reduce file fragmentation, and thus minimize seeking, is to sort the list of free blocks when deleting files so that new files will tend to be allocated in contiguous blocks. While sorting the free list does not guarantee that files will be contiguous (because different tasks may be requesting blocks concurrently), it should help reduce file fragmentation.

CYLINDER GROUPS

By "cylinder groups" we mean the method of organization used by the Berkeley fast file system. As mentioned above, this system is rather complex to implement, and we face the question of whether this is as useful for a microcomputer in a commercial environment as it is for a VAX at a large university. We hope to answer this question by experimentation.

One possible argument against cylinder groups is that, by increasing the complexity on the disk, they may make the file system less robust in case of system crashes or power failures. Another weakness is that although they keep the blocks of files close together, they do not keep them consecutive. Thus, a DMA program loader would still have to load programs block by block instead of all at once. On the plus side, this method would keep inodes and directories relatively close to the corresponding files, reducing the distance of many seeks.

PERIODIC REORGANIZATION OF THE FILE SYSTEM

Another way to reduce or eliminate fragmentation is to have a utility that periodically consolidates all files into contiguous blocks (that is, logically contiguous, with an interleave built into the disk format). This solution is not aesthetically appealing, but it may have some advantages for small systems. It requires no changes to the kernel, being a utility, and introduces no incompatibilities in the file system. It guarantees that the blocks of files are contiguous rather than just close together. Thus, a DMA program loader could load programs very rapidly, and various read-ahead methods would be more likely to actually read ahead.

Reorganizing the files would mesh very well with sorting the above-mentioned free list since it would be easier to find sequences of consecutive blocks if the free list was already sorted. A single utility could do both.

On the minus side, this utility would have to be run periodically by a system administrator in single-user mode. This chore might be neglected unless it were run automatically at bootstrap time, which would make booting the system time-consuming. Also, if the disk was very full, it might be impossible for the utility to do any reorganizing for lack of space. The first objection might be overcome by making the utility a perpetual background task, but this would be tricky with multi-users running. It would require mutual exclusion while the file system was being manipulated, and it would tend to violate our bias toward keeping things simple. The second objection might be overcome by using the swap space as a work area.

The worst case for this solution is probably a situation where a few large, permanent files grow slowly. In this case, reorganizing would be both frequently necessary and slow —it might even be impossible on a nearly full disk. A way around this might be to reorganize very large files into smaller clumps of contiguous blocks. Another undesirable situation is one where many files are often created and erased, such as a software development environment. In this case, the file system would fragment itself rapidly, and reorganizing would be frequently necessary.

A variation of this strategy is to reorganize each new file when it is closed, thereby maintaining files in consecutive sequences of blocks. The "close" operating system call could invoke a background process that checks to see if the file is contiguous. If not, it then checks to see if enough contiguous blocks are available. If so, the background process could allocate the free blocks, copy the file over, free its old blocks, and re-sort the free list. A possible disadvantage of this strategy would be a perceived degradation in performance by other users because mutual exclusion would be required while manipulating the list of free blocks. However, this would probably be more than offset by the resulting overall improvement in performance.

LARGER BLOCK SIZE

Increasing the block size is an important optimization in the Berkeley file system, and it deserves attention. As mentioned above, it offers an inherent reduction in fragmentation, is a form of read-ahead, and saves on indirection. However, it wastes space. Since increasing the block size is only practical if it does not waste space, some space-saving strategies are considered here.

The Berkeley system of allowing files to occupy fragments of a block is complex, as befits an academic institution. However, this fact may not be very important since they have tested the design and it can be imitated. It apparently solves the problem of increasing the block size without wasting space.

Another way of achieving a large effective block size is to allocate several blocks at a time and have the kernel do all of its I/O in chunks of several blocks. So far, we have not saved any space, but it should be relatively simple to devise a way to free the unused odd blocks and use them for small files or the ends of large files. Some dynamic reorganization would be necessary to copy small files into odd blocks and to copy files out of odd blocks if they outgrow them.

On the face of it, this method seems simpler than the Berkeley fragmenting strategy. The next step would be to do a more detailed design and see if it is still simpler.

MORE INTELLIGENT BUFFERING

The objective of the buffer system is to save data that might be needed again, and it may be worthwhile to try to predict which data is most likely to be needed. The current kernel uses a "least recently used" (LRU) algorithm that is implemented very simply by maintaining a free list of buffers in that order. We could consider more sophisticated general algorithms, such as "least frequently used," but this is probably the road toward greatest possible complexity.

Aside from general algorithmic approaches, we can try to distinguish between types of data that may be needed more frequently. We can distinguish between the contents of files and the file system structures such as inodes, directories, and indirect blocks. Directories are needed only for opening, creating, or deleting files, which is to say not very often. On the other hand, traversing path names might involve searching the same few directories over and over. This suggests that it might be useful to cache the parent directories of the current directory.

Inodes are stored in memory independently of the buffer system, but they must be updated if a file is being written to. They must also be time stamped, and thus buffering them might be worthwhile.

Indirect blocks might also be buffered. For a file large enough to need indirect blocks, the kernel must refer to them to find the data blocks of the file. When writing to a large file, the kernel must update the indirect blocks. The rationale for indirect blocks was that the user must pay a price for large files.

RELATED STRATEGIES

All of the strategies are closely interrelated. For example, if we do program loading by DMA to the user memory, we avoid the overhead of buffering these requests block by block. We also save our buffer cache from getting flushed out by programs. However, there are also some other implications. First of all, this would reduce the amount of sequential I/O (as opposed to random I/O) passing through the buffer system, reducing the need for read-ahead algorithms.

It would also put a premium on maintaining files in contiguous blocks on the disk, and large I/O requests could proceed at the data transfer rate of the I/O devices. If files are maintained contiguously, then it might make sense to use a track-caching disk controller, where otherwise it would not. Lastly, by reducing the amount of sequential I/O passing through the buffers, it might make buffer caching more effective by preserving more frequently used blocks in the buffers. Thus, certain optimization strategies go together and compound each other's effects, reducing or eliminating the need for other strategies. These strategies in turn influence the most desirable hardware architectures.

As we mentioned at the beginning, these optimizations are necessary because advances in silicon technology are outstripping the performance of disks and file systems. Conversely, some of the strategies we are considering are made feasible only because of hardware advances such as cheaper memories and fast, track-caching disk controllers at competitive prices.

Our plan for future investigation at Intel is to try as many of the optimizations mentioned in this article as possible to find out which ones and which combinations are most effective for various types of applications and system loads. ■

References

McKusick, M., Joy, W., Leffler, S., and Fabry, R., "A Fast File System for Unix," Computer Systems Research Group, Dept. of Electrical Engineering and Computer Science, UC Berkeley.

David Robboy, an avid white-water canoeist, has been a farmer, bass player, carpenter, and software consultant. He is currently working at Intel Corp. in Hillsboro, Ore., as a software engineer.

THE C TUTORIAL: SOME LOGICAL OPERATORS AND MORE SOFTWARE TOOLS

BY DR. REBECCA THOMAS

As you learned in the first installment of this tutorial series, the C language has an extensive list of operators. This issue we'll discuss the three most commonly used logical operators: && (logical AND, or intersection), || (logical OR, or union), and ! (logical NOT, or negation). The first two are binary operators because they operate on two operands. The negation operator is a unary operator since it takes only one operand.

These operators are usually employed to join C language subexpressions to yield combined *test expressions,* which are used mainly in flow control constructions. The operands associated with each operator in each subexpression are first evaluated to give a logical value, TRUE or FALSE. Then the logical operators combine these operand values according to the rules shown in Figure 1. This figure depicts the combining rules for the AND, OR, and NOT operators with *truth tables,* which conveniently show the logical (or Boolean) value that results when different Boolean value operands are joined by the logical operators.

PRECEDENCE OF C LANGUAGE OPERATORS

The C language operators have a property known as *precedence* that may be used to determine the order in which expressions with more than one operator are evaluated. The expressions using operators with a higher precedence are evaluated first. If an expression contains several operators with the same precedence, the expression is evaluated from left to right. A table of the operators introduced so far, as well as their precedence (or priority), is shown in Figure 2.

Examples of how precedence influences the order of expression evaluation are given in Figure 3. Figure 3A illustrates precedence of arithmetic operators and how it may be changed by use of parentheses. Since the assignment operator (=) is evaluated right to left, the expression on the right side of the equal sign is evaluated first, with the resulting valued stored in x on the left-hand side of the "equation." On the left-hand side of Figure 3A the arithmetic expression evaluates to give "2" since the higher-order operation, division, is performed before the addition. However, in the right-hand side of the figure, the ex-

FIGURE 1: LOGICAL OPERATIONS ILLUSTRATED BY TRUTH TABLES

```
Given: The logical operands A and B that take on
logical values TRUE (denoted by T) or FALSE (denoted by F).

         A   B   A && B   A || B   !A   !B
        --- --- -------- -------- ---  ---
         T   T     T         T     F    F
         T   F     F         T     F    T
         F   T     F         T     T    F
         F   F     F         F     T    T
```

FIGURE 2: THE PRECEDENCE AND EVALUATION ORDER FOR SOME OPERATORS

```
The operators with the highest precedence are listed first.
Operators with the same precedence are listed together on a
given row. The order in which the expressions containing
these operators are evaluated is listed in the right-hand
column.

        Operator     Order of Evaluation
        --------     -------------------
           ()           left to right
        ! ++ --         right to left
         * / %          left to right
          + -           left to right
       < <= > >=        left to right
         == !=          left to right
           &&           left to right
           ||           left to right
           =            right to left
```

FIGURE 3A: EXAMPLES OF CHANGING ARITHMETIC OPERATOR PRECEDENCE

```
x = 2 + 3/5                             x = (2 + 3)/5
```

pression evaluates to yield "1" because the highest-order parentheses operators force the addition to be performed first, and then the sum is divided.

In Figure 3B we show an example of using the AND logical operator and the equality operator (--). The test expression for the `if` statement on the left-hand side says "if x AND y is zero (or FALSE)." Without the inner set of parentheses, an entirely different interpretation is obtained—namely "if x AND if y is zero (FALSE)" because the equality operator (==) binds its operands more tightly than does the AND operator.

As a final example, consider the expressions shown in Figure 3C. The expression labeled WRONG doesn't correctly collect the character and detect the end-of-file (EOF) condition. The inequality operator has a higher precedence than the assignment operator (=), so the value returned by `getchar()` will be compared directly to EOF instead of first being stored in `c`. Thus, `c` would only take on two different values: one (for TRUE), or zero (for FALSE). If `getchar()` doesn't return EOF, then `getchar() ! =` EOF is true and c becomes 1; otherwise it becomes zero. Simply add an extra pair of parentheses as shown in RIGHT to force the storage of the value returned by `getchar()` into c followed by the comparison of that value to EOF.

The C language allows a convenient shorthand for testing the equality or inequality against zero (FALSE). For instance, in Figures 4A and 4B we compare two common approaches. The expression on the left side of Figure 4A could be read "if on screen is zero (or FALSE)" and the right-hand side expression "if not on screen." The left side of Figure 4B reads "if on screen is not zero (or TRUE)" and the right side "if on screen." Thus, in these cases at least, the shorter form reads quite naturally. However, you should exercise caution when using more complicated combined expressions because an abbreviated form can be difficult to understand.

SOME GUIDELINES FOR WRITING READABLE C EXPRESSIONS

A good rule of thumb for readability is the "telephone test." Your code should make sense when read aloud as if over the telephone. If it fails that simple test, then you should rewrite your code. Remember that even though *you* may understand your clever, esoteric algorithm, you or another programmer may later find it difficult to comprehend. The ability to understand code quickly is essential for maintaining it or improving it economically at a later date.

Some other guidelines to improve the readability of expressions: (1) High-precedence operators, such as ( ), !, ++, and --, should not have white space (space characters or tabs) around them. (2) Low-precedence operators, such as = and the comma (used in function argument lists), should always have white space associated with them. Generally you use white space on both sides of the equal sign and one space after a comma. (3) Medium-precedence operators, such as the arithmetic, relational, and logical operators, generally are surrounded by white space.

EVALUATION OF COMPOSITE LOGICAL EXPRESSIONS

Expressions connected by the && and || operators are evaluated from left to right. As soon as the program can determine if the overall expression would be TRUE or FALSE, the evaluation stops. When subexpressions are joined with the OR operator, evaluation terminates as soon as any subexpression is TRUE since then the entire expression will evaluate TRUE. If subexpressions are joined with AND, evaluation stops as soon as any subexpression is FALSE since the entire expression will evaluate as FALSE.

Frequently you can change the placement of the subexpressions within the larger composite expression to give more efficient execution. Place the subexpressions most likely to end evaluation of the combined expression near its beginning. Then the evaluation should end sooner (on the average) since not all

FIGURE 3B: EXAMPLES OF CHANGING LOGICAL OPERATOR PRECEDENCE

```
if ((x && y) == 0)          if (x && y == 0)
```

FIGURE 3C: A COMMON EXAMPLE OF A PRECEDENCE ERROR

```
WRONG: while (c = getchar() != EOF)
RIGHT: while ((c = getchar()) != EOF)
```

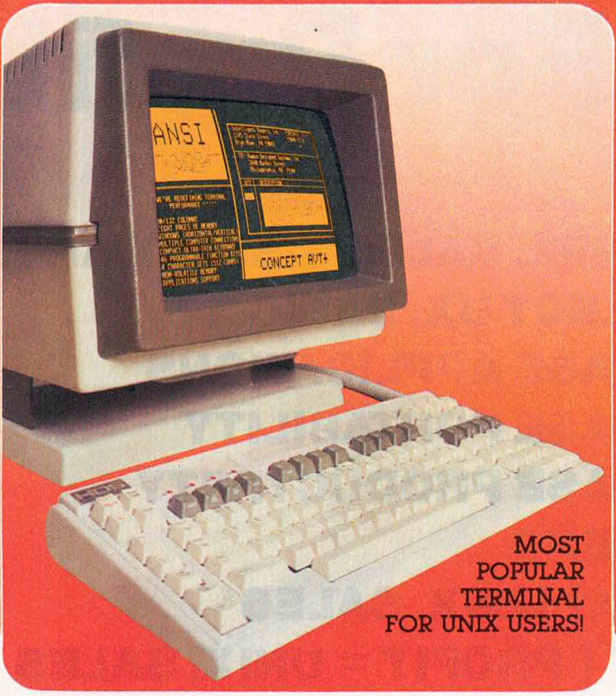
FIGURE 4A: EQUIVALENT TESTS FOR FALSEHOOD

```
if (onscreen == 0)          if (!onscreen)
```

FIGURE 4B: EQUIVALENT TESTS FOR TRUTHFULNESS

```
if (onscreen != 0)          if (onscreen)
```

The new CONCEPT AVT+
In a word: capability.



MOST POPULAR TERMINAL FOR UNIX USERS!

THE CONCEPT AVT+ DISPLAY TERMINAL

ANSI standard conformance, DEC software compatibility, eight pages of memory, 80/132 columns, windowing, multiple computer connections...and a full range of practical ergonomic features. Capabilities which enable all terminal users to maximize their productivity and explore their creativity. Only from Human Designed Systems.

Introducing the new *concept* AVT+ display terminal from Human Designed Systems. Simply the smartest interactive display terminal available today. And at a very competitive price.

It starts with ANSI standard (X3.64-1979) conformance and DEC software compatibility, and incorporates superior human design features, advanced functionality and highest quality construction. A commitment to quality that is reflected in each of the individual components.

The *concept* AVT+ keyboard is ergonomically designed for optimum operator comfort, incorporating a VT100-style numeric pad for DEC software compatibility. And it builds on those human design features with a range of capabilities that make the *concept* AVT+ an ideal choice for smart terminal users.

The *concept* AVT+ offers up to eight pages of display memory that eliminates unnecessary hardcopy printouts and provides a powerful tool for applications requiring multiple formats and storage of large volumes of text; non-volatile memory that enables users to permanently configure a terminal for their needs or applications; windowing that allows users to create individual displays within display memory; programmable function keys which transmit data and/or execute terminal commands; up to three additional communications ports for connection to other peripherals and computers; flexible user networking and functionality for use in a wide range of different applications, including multiple computer connections; and much more.

$1295*

The new *concept* AVT+. More to offer at its price for terminal users than any other terminal available today.

*Quantity one. DEC and VT are trademarks of Digital Equipment Corporation

human designed systems, inc.
3440 Market Street, Philadelphia, PA 19104
215-382-5000

Human Designed Systems. We're redefining terminal performance.

Atlanta — (404) 391-9763; Boston — (617) 449-6446; Chicago — (312) 825-2960; Dallas — (214) 437-1888; Delaware — Infocon: (302) 239-2942; Denver — (303) 469-1953; Detroit — (313) 434-2729; Hawaii — Gray Associates: (808) 261-3751; Houston — (713) 952-1403; Los Angeles — (213) 410-9454; Northern New Jersey — Infocon: (201) 624-1372; New York City Area — Infocon: (212) 689-8833; New York State — Naco Electronics: Rochester: (716) 223-4490; Syracuse: (315) 699-2651; San Francisco — (415) 692-4184; Washington, DC — International Systems Marketing: (301) 279-5775; Argentina — Itron SA: (01) 774-9369; Australia — Computer Clarity Pty. Ltd.: (02) 241 3385; Belgium — BELCOMP: 091-31.52.22; Canada — CAIL Systems: Toronto: (416) 362-1063; Denmark — ADCOM Data Aps: 1-19 44 66; Finland — Valtamatic Oy: 0-742 011; France — Walton: (1) 226.06.90; Japan — Ampere: 03 (365) 0825; Portugal — Soc. Com. Crocker, Dellaforce Co. Ltd.: 1-6801 41; Singapore — DTS Singapore: 33-88-566; South Africa — Psitec (Pty.) Ltd.: (11) 836-9181; Switzerland — Mitek ag: 01/461 22 52; United Kingdom — Shandell Systems Ltd.: 2407-2027; Venezuela — H. Blohm SA: 2 541.21.22; West Germany — COMKO Computersystemges, mbH: 221-48 30 51. INTERNATIONAL DISTRIBUTORSHIP INQUIRIES INVITED.

Please circle Ad No. 17 on inquiry card.

UNIX™ Program For Profit

```
-10  FOR UNIX. SALES = SOFTWARE. SOLUTIONS
-20      SELECT UX. BASIC
-30          CASE APPLICATIONS
-40          CASE PORTABILITY
-50          CASE PRODUCTIVITY
-60          CEND
-70      NEXT UNIX. SALES
-80  PRINT ( PROFIT = UNIX. SALES ) ^UX. BASIC
     RUN  (Other Basics may not give you the same result).
```

– UX-Basic™ Provides Applications

- From OASIS™ Basic, accessing existing applications base
- Through UX-Link™ interfacing to DBMS and graphic libraries
- Via UX-Translate™ tapping applications from other Basics

– UX-Basic Provides Portability

- Language is available across a range of UNIX machines
- Programs will execute on any machine running UX-Basic
- Device independence allows program use on any terminal

– UX-Basic Provides Productivity

- With a full featured editor, development and debugging tools
- With easily readable, highly structured, modular style code
- With business features including: C-ISAM™ and BCD arithmetic

UX-Basic®

UNIX is a Trademark of AT&T Bell Laboritories.
UX-Basic, UX-Link and UX-Translate are Trademarks of UX Software, Inc.
OASIS is a Trademark of Phase One Systems, Inc.
C-ISAM is a Trademark of Relational Database Systems, Inc.

See us at
COMDEX™/Fall '84
Las Vegas Convention Center
Booth 1196

UX-Basic is available directly from equipment manufacturers and selected software distributors.
UX Software, Inc.
10 St. Mary Street, Toronto, Canada M4Y 1P9
(416) 964-6909

Please circle Ad No. 49 on inquiry card.

the subexpressions have to be evaluated.

As an example, compare the code fragments in Figures 5A and 5B. Assume for this example that a SPACE is input more frequently than a NEWLINE, which occurs more often than the end-of-file code (EOF). Test for SPACE, NEWLINE, and finally EOF in that order to end execution of the loop most quickly. The `while` loop continues execution as long as `c` is not one of these characters. As soon as one of these characters in input, the composite `while` loop test expression becomes FALSE, and execution continues with the next statement in the program.

A SOFTWARE TOOL USING MULTIPLE FUNCTION CALLS

Two issues back we introduced the notion of software tool program design. You learned how to design a simple software tool so that it worked most efficiently with the Unix shell and other utilities supplied with the Unix system. In this issue you will learn about designing a tool program using a *top-down* approach. This program example illustrates the use of several modular functions.

The `locate` program is used to find all lines in a text file that contain a given pattern and display them. This example is a simplified version of the Unix `grep` software tool. The first step in designing a program, even a modest one like `locate`, is to outline the general steps. One possible outline:

(1) While there are input lines, read the next one.

(2) See if the line contains the desired pattern.

(3) If so, display the line.

The next step could be to translate the outline into an English-like programming language called *pseudo-code*. This "language" is actually a mixture of C and English. Writing in pseudo-code lets you work out the flow of logic without getting bogged down in syntactic details. Figure 6 shows one of many different possible pseudo-code versions.

Now that the hard part of the design has been done, all that is left is to translate pseudo-code into C. The final result is shown in Figure 7.

This program examines each character position in the input line to see if the substring that starts at that character position and continues toward the end of the line matches the desired pattern. If so, the line will be sent to the standard output; otherwise it will be ignored.

The `match()` function tells the function that called it (`locate()`) if the substring matches the pattern. The `locate()` function scans the input line character by character position and tells its calling function, `main()`, if a match has been found. We could have combined the code in `match()` with that in `locate()` for this simple example. However, we decided to leave them separate because we shall illustrate another useful tool in the next installment by simply changing `locate()` and leaving `match()` as it is. We'll keep both of these functions independent, which helps make the overall program more modular.

The major disadvantage of this particular program is that the search pattern has been "hard-wired" into the program code. In a future article we'll show you how to specify the

FIGURE 5A: A LESS EFFICIENT ORDERING OF SUBEXPRESSIONS

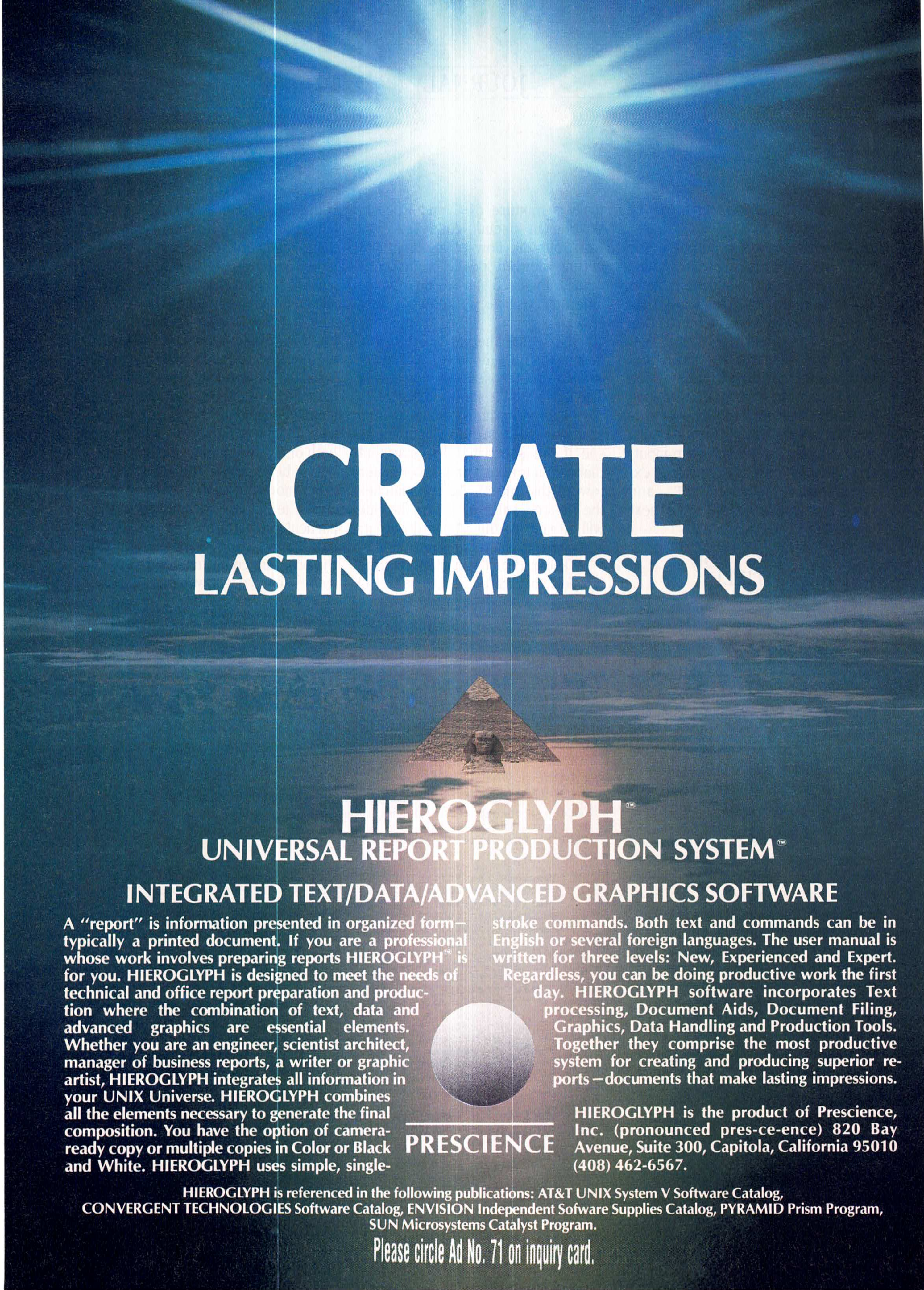
```
...
while ((c = getchar()) != EOF && c != NEWLINE && c != SPACE)
      ;
```

FIGURE 5B: A MORE EFFICIENT ORDERING OF SUBEXPRESSIONS

```
...
while ((c = getchar()) != SPACE && c != NEWLINE && c != EOF)
      ;
```

FIGURE 6: ONE PSEUDO-CODE VERSION FOR THE `locate` PROGRAM

```
Define a constant for the maximum number of characters
in input line.
Declare the pattern string to be found.
main()
{
     Declare the string to hold the input line;
     Declare and initialize any other required variables;
     while(can gets() another line from the standard input)
          if(can locate() the pattern)
               puts() it to the standard output;
}
locate()
{
     Declare and initialize any variables;
     while(there are characters, examine each in turn)
          Find any match();
          If there's a match
               return the position where it was found;
          else if not a match
               examine the next character in the input line;
          return(-1) if no match after examining all characters;
}
match()
{
     Declare and initialize any variables;
     while(characters left to scan in the pattern)
          If a character of the string doesn't match the pattern
               return(-1);
          return(last position in input string);
}
```


CREATE
LASTING IMPRESSIONS
HIEROGLYPH™
UNIVERSAL REPORT PRODUCTION SYSTEM™
INTEGRATED TEXT/DATA/ADVANCED GRAPHICS SOFTWARE
A "report" is information presented in organized form—typically a printed document. If you are a professional whose work involves preparing reports HIEROGLYPH℠ is for you. HIEROGLYPH is designed to meet the needs of technical and office report preparation and production where the combination of text, data and advanced graphics are essential elements. Whether you are an engineer, scientist architect, manager of business reports, a writer or graphic artist, HIEROGLYPH integrates all information in your UNIX Universe. HIEROGLYPH combines all the elements necessary to generate the final composition. You have the option of camera-ready copy or multiple copies in Color or Black and White. HIEROGLYPH uses simple, single-stroke commands. Both text and commands can be in English or several foreign languages. The user manual is written for three levels: New, Experienced and Expert. Regardless, you can be doing productive work the first day. HIEROGLYPH software incorporates Text processing, Document Aids, Document Filing, Graphics, Data Handling and Production Tools. Together they comprise the most productive system for creating and producing superior reports—documents that make lasting impressions.
PRESCIENCE
HIEROGLYPH is the product of Prescience, Inc. (pronounced pres-ce-ence) 820 Bay Avenue, Suite 300, Capitola, California 95010 (408) 462-6567.
HIEROGLYPH is referenced in the following publications: AT&T UNIX System V Software Catalog, CONVERGENT TECHNOLOGIES Software Catalog, ENVISION Independent Sofware Supplies Catalog, PYRAMID Prism Program, SUN Microsystems Catalyst Program.
Please circle Ad No. 71 on inquiry card.

search pattern on the command line used to invoke `locate`. This additional feature would help give this program the flexibility necessary to be an effective general-purpose software tool.

Now let's see how this program operates. Figure 8A shows the command line for compiling the source code. Figure 8B shows a sample input file and Figure 8C the result of running `locate` on the text from that file.

FIGURE 7: THE `locate` SOFTWARE TOOL PROGRAM

```
$ cat -n locate.c
#define MAXLINE 120    /* increase to avoid core dumps */
char pattern[] = "the";     /* pattern to locate */
main()
{
        char line[MAXLINE];      /* storage for input line */
        while (gets(line)) {
                if (locate(line, pattern) >= 0)
                        puts(line);
        }
}
locate(line, pattern) /*if pattern is not in line return -1 */
char line[], pattern[];
{
        int m, i = 0;
        while (line[i]) {          /* not end of line */
                m = match(line, i, pattern);
                if (m >= 0)        /* there's a match */
                        return(m);
                else
                        i++;    /* point to next character */
        }
        return(-1);
}
match(string,i,pattern) /* if pattern is not in string return -1*/
char string[], pattern[];
{
        int j = 0;
        while (pattern[j])        /* more to scan */
                if (string[i++] != pattern[j++]) /* mismatch */
                        return(-1);
        return(i); /* must be a match */
}
$ []
```

FIGURE 8A: A COMMAND LINE FOR COMPILING `locate`

```
$ cc locate.c -o locate
```

FIGURE 8B: FILE CONTAINING SAMPLE INPUT

```
$ cat appeal
Now is the time
for all good citizens
to come to the aid
of their country.
$ []
```

FIGURE 8C: RUNNING `locate` ON CONTENTS OF `appeal`

```
$ locate <appeal
Now is the time
to come to the aid
of their country.
$ []
```

ATTENTION READERS:

We are looking for readers interested in contributing useful software tool programs for inclusion in this series of articles. Authors of published entries will be acknowledged and will receive $50. Please follow these simple guidelines for your contributions:

(1) Submit your C programs to *Dr. Rebecca Thomas*, UNIX/WORLD, 444 Castro St., Suite 1220, Mountain View, CA 94041.

(2) Employ concepts and features that have been discussed already in this tutorial series. Avoid using code that contains, say, pointers and structures, until we've had a chance to introduce them.

(3) Write C programs so they are portable across different versions of the Unix system. If possible, they should run without change on Bell Version 7, System III, V, Berkeley 4.2, and most Unix work-alikes. You should use the standard I/O library for input/output. Also use the `lint` syntax checker to eliminate nonportable constructions, and compile the code with a portable C compiler such as `pcc` to help ensure portability. Hardware dependencies, such as terminal control codes, should be eliminated or at least minimized and isolated to one code region or to a separate module.

(4) Keep your example short —under 100 lines of source code.

Dr. Rebecca Thomas, 1839 Tenth Avenue, San Francisco, Calif. 94122. Dr. Thomas is an author of the best-selling *A User Guide to the Unix System.* This series of articles is excerpted from her forthcoming book on the C programming language. Dr. Thomas also works as a consultant specializing in the Unix operating system and the C language. She lives with three cats; one is retired, but the other two still keep warm on top of her terminals. These very independent "termcats" defy description in anyone's database.

References:

Kernighan, B.W. and Ritchie, D.M.: *The C Programming Language*, Englewood Cliffs, NJ: Prentice-Hall, 1978.

Kernighan, B.W. and Plauger, P.J.: *Software Tools in Pascal,* Reading, MA: Addison-Wesley, 1981.

Plum, T.: *C Programming Guidelines*, Cardiff, N.J.: Plum Hall, 1984.

ADVANCED SCCS: HOW MAKE AND SCCS WORK TOGETHER

BY DR. REBECCA THOMAS

Automatic program administration is made possible by the *Make system.* As you learned in the first article of this series on SCCS, the compiling and linking steps for producing an executable program can be automated by using the `make` program. First the programmer creates a file that describes how one or more *target files* (or simply *targets*) depend on other files known as *dependency files* (or simply *dependents*). If one or more of the files that the target depends on has been modified since the target was last updated, then it must be updated.

The Make system may be employed for maintaining a wide variety of targets, not just programs —including text documents; however, we shall only consider target files that are either object files or executable load modules (programs) in this article. In general, the dependents will be source files, object files, and header (also called *include*) files. Header files are so named because they usually have an "`.h`" suffix added to their file name. The preprocessor `include` directive is generally used to "include" (add in) the header file to the source file.

The `make` program determines which targets are out of date with respect to their dependents by examining the modification times in their disk inode entries. If the target has not been updated and its dependency files have been, `make` will then create another instance of the target.

A MAKE FILE

The programmer must supply a *description file,* more commonly known as a *makefile,* which describes the relationships between the targets and their dependent files. The makefile contains descriptions of the dependency files and commands that generate the updated target files.

If the makefile is named either as `Makefile` or `makefile`, the `make` command will automatically recognize it. If the capitalized name is used, it will be listed before the other lower-case filenames by the Unix directory listing command, `ls`.

A separate directory should be devoted to contain the makefile, the various dependency files, and their associated target files. Figure 1 shows the contents of the makefile that we'll use as our example for this article. The `pack` program is used to compress a text file, while the `unpack` program can expand a file compressed by `pack`.

The makefile contains *dependency lines,* which specify the target on the left-hand side (such as `pack:` on the first line of Figure 1), followed by the dependents to the right (such as `pack.o`, `getargs.o`, and `compress.o`). In our example, the *command lines* specify the `cc` invocation line used to generate the corresponding target (such as the second line in our example makefile, `cc pack.o getargs.o compress.o -o pack`, to generate `pack`).

This particular makefile may be used to generate any of the targets, `pack`, `pack.o`, `compress.o`, `unpack`, `unpack.o`, `uncompress.o`, and `getargs.o` by specifying that particular target file as an argument to the `make` command. The first target in the makefile, `pack` in this example, is "made" by default if no target file is indicated on the `make` command line. The general syntax for invoking `make` to update a target file is

$ **make** [*target file...*]

AVOIDING SYNTAX ERRORS

Caution. The makefile must be constructed carefully to avoid syntax errors. The target name *must* be followed by a colon (:), and its dependents separated from the target and themselves by a variable amount of white space (blanks and tabs). Note that you may continue long dependency lines on the next line by typing a backslash (\) immediately before the NEWLINE. Command lines *must* begin with one or more tab characters, and the tab *must* be present and cannot be replaced by spaces. Omitting this tab character is a common mistake when constructing makefiles.

You may visualize the relationships between the target files and their dependents as a tree. Figure 2 shows a tree for our `pack` example program.

The dependent of each target is a branch of the tree. The dependents `pack.o`, `getargs.o`, and `compress.o` are listed just above the `pack` target. These dependents are targets also, where `pack.o` depends on `pack.c`, `getargs.o` depends on `getargs.c`, and `compress.o` depends on both `compress.c` and `compress.h`.

Note that even though the source file `compress.c` contains the directive to include `compress.h` (**include "compress.h"**), the source file is *not* dependent on the contents of `compress.h` and would *not* need updating if this header file were changed. However, the targets `compress.o` or `pack` would need updating if `compress.h` were changed.

The commands listed above each target (in parentheses) represent the "rule" for moving toward the root of the tree. Thus, the first rule for moving from the upper branch, `pack.c`, toward the root is "cc -c pack.c."

FIGURE 1: OUR SAMPLE DESCRIPTION FILE

```
$ cat Makefile
pack:                pack.o getargs.o compress.o
                     cc pack.o getargs.o compress.o -o pack
pack.o:              pack.c
                     cc -c pack.c
compress.o:          compress.h compress.c
                     cc -c compress.c
unpack:              unpack.o getargs.o uncompress.o
                     cc unpack.o getargs.o uncompress.o -o unpack
unpack.o:            unpack.c
                     cc -c unpack.c
uncompress.o:        compress.h uncompress.c
                     cc -c uncompress.c
getargs.o:           getargs.c
                     cc -c getargs.c
$ []
```

FIGURE 2: THE RELATIONSHIPS BETWEEN TARGETS, DEPENDENTS, AND COMMAND LINES

```
+---------+      +------------+      +-------------+         +-------------+
| pack.c  |      | getargs.c  |      | compress.c  |         | compress.h  |
+---------+      +------------+      +-------------+         +-------------+
     |                 |                    |                       |
(cc -c pack.c)  (cc -c getargs.c)           +-(cc -c compress.c)-+
    \|/               \|/                         \|/
+---------+      +------------+            +-------------+
| pack.o  |      | getargs.o  |            | compress.o  |
+---------+      +------------+            +-------------+
     |                 |                          |
     +-(cc pack.o compress.o getargs.o -o pack)-+
                      \|/
                    +------+
                    | pack |
                    +------+
```

FIGURE 3: AN ABBREVIATED DESCRIPTION FILE

```
$ cat Makefile
pack:                pack.o getargs.o compress.o
                     cc pack.o getargs.o compress.o -o pack
compress.o           compress.h
unpack:              unpack.o getargs.o uncompress.o
                     cc unpack.o getargs.o uncompress.o -o unpack
uncompress.o         compress.h
$ []
```

MODIFICATIONS AND UPDATES

As an example, if the `make` program is invoked to update the target `pack`, it would search such a tree beginning with the uppermost branches, `pack.c`, `getargs.c`, `compress.c`, and `compress.h`. If any of these dependents have been modified more recently than the corresponding target, the rule (command) that connects the dependent to its target is invoked.

This process is continued until each intermediate target (`pack.o`, `getargs.o`, and `compress.o`) is as up to date as its dependents. The final target, `pack`, would be updated according to the rule "cc pack.o getargs.o compress.o -o pack" if any of its dependents, which are the intermediate targets noted above, have been modified more recently than `pack` itself.

The `make` program contains an internal table that describes the relationship between some files. For instance, such a table describes how to transform a C language source file (named with a "`.c`" suffix) into its corresponding object file (with the "`.o`" suffix). Figure 3 shows how we may shorten our description file because the rule for transforming "`.c`" files into "`.o`" files is inherent in the `make` program.

Note that all dependencies such as "`pack.o: pack.c`" are described internal to `make` and do not appear in this makefile. It is easier to visualize the dependencies in this form of the makefile because it is less "busy" than the form shown in Figure 1.

HOW make MAY BE USED TO HANDLE SCCS FILES

By now you may be wondering how the Make system can be used to process SCCS files. Well, early versions of `make` could *not*, in fact, handle the SCCS filename format because they couldn't reference prefixes (such as "`s.`") to filenames. Augmented `make`, introduced in Bell System III, contains a rule to recognize SCCS filenames.

Place the program modules in the custody of the SCCS system file by using the `admin` command as you learned in the last article on SCCS. The general command-line syntax would be:

$ admin -i ***basename*.c**
s.*basename*.c

where *basename* would be replaced by the basename of the program module (such as `pack`). As with the document text example used in the last issue, use the SCCS `admin` command as shown in Figure 4.

The basic usage of augmented `make` is identical to the older version. The SCCS files are named as before, and the makefile construction remains the same. With augmented `make`, when a request is made to "make" a target, whose dependents are in the custody of an SCCS file, the following operations ensue:

(1) A "get -p" operation is done, which writes the source text to the standard output. This output is then redirected by the shell to a disk file. In this way, a writable *g-file* is created, no *p-file* is updated, and all ID keywords are replaced by their textual equivalents.

(2) The target is "made" by compiling and linking the appropriate source modules.

(3) The retrieved source file is erased.

The `make` command may be employed now to create the executable programs for testing. Figure 5 shows a typical "make" session to create the target `pack`.

USING AUGMENTED make AND SCCS

Figure 6 shows the text compression (`pack`) and expansion (`unpack`) utilities being tested by compressing and expanding our text file, `poem`. The Unix system `wc -c` command displays the character count for the specified file argument. The compression may be computed independently as: (62- 45 = 17)/62 * 100% = 27.4%, the same result displayed by the `pack` program.

The `unpack` program issued an error message that says, in effect, that it did not decompress `poem.z` successfully; however, the directory listing and character counting operations show that it did!

To correct the "bug," the next step would be to perform a "get-for-

FIGURE 4: ADDING THE SOURCE CODE MODULES TO THE SCCS SYSTEM

```
$ admin -ipack.c s.pack.c
$ admin -iunpack.c s.unpack.c
$ admin -icompress.c s.compress.c
$ admin -iuncompress.c s.uncompress.c
$ admin -igetargs.c s.getargs.c
$ []
```

FIGURE 5: "MAKING" THE pack PROGRAM

```
$ make pack
        get  -p s.pack.c > pack.c
1.1
78 lines
        cc  -c pack.c
        rm -f pack.c
        get  -p s.getargs.c > getargs.c
1.1
28 lines
        cc  -c getargs.c
        rm -f getargs.c
        get  -p s.compress.c > compress.c
1.1
55 lines
        cc  -c compress.c
        rm -f compress.c
        cc pack.o getargs.o compress.o -o pack
$ []
```

editing" operation to retrieve an instance of unpack.c for updating. Then execute the think-edit-make-test cycle until you are satisfied that the offending module has been debugged. Finally, the changes necessary to debug the source module are incorporated into the SCCS file by the delta command, yielding version SID 1.2. These steps are shown in Figure 7.

In this case make employs the unpack.c file instead of s.unpack.c to construct the target unpack since the dependent unpack.c was changed more recently than the equivalent SCCS-format-dependent s.unpack.c. Also, since a writable *g-file* (unpack.c) existed in the same directory, it *must* be removed before get will create another copy of this module for updating purposes.

In this series of articles you have seen how both the SCCS and the Make system can automate, thus saving time and reducing errors when programs or text documents are being developed in the Unix environment. Actually, we have only scratched the surface to give you an idea of the capabilities of these systems. By using these facilities in your daily work to maintain your own document or source code, you will discover new and powerful uses for these administrative tools.

FIGURE 6: USING THE pack AND unpack PROGRAMS

```
$ wc -c poem
       62 poem
$ pack poem
pack: poem: 27.4% Compression
$ ls poem*
poem.z
$ wc -c poem.z
       45 poem.z
$ unpack poem.z
unpack: poem.z: unpacking error
$ ls poem*
poem     poem.z
$ wc -c poem*
       62 poem
       45 poem.z
      107 total
$ []
```

FIGURE 7: CORRECTING THE unpack PROGRAM

```
$ get -e s.unpack.c
ERROR [s.unpack.c]: writable `unpack.c' exists (ge4)
$ help ge4
ge4:
"writable `...' exists"
For safety's sake, SCCS won't overwrite an existing g-file if it's writable.
If you don't need the g-file, remove it and rerun the get command.
$ rm unpack.c
$ get -e s.unpack.c
1.1
new delta 1.2
92 lines
$ ed unpack.c
[ EDITING SESSION ]
$ make unpack
        cc  -c unpack.c
        cc unpack.o getargs.o uncompress.o -o unpack
$ pack poem
pack: poem: 27.4% Compression
$ unpack poem.z
unpack: poem.z: successful
$ delta s.unpack.c
comments? Erroneous "unpacking error" condition fixed.
1.2
1 inserted
1 deleted
91 unchanged
$ []
```

A GLOSSARY OF TERMS

Augmented make—the version of make that can process SCCS files. First introduced in Unix System III.

Command line—the line in a makefile that specifies the shell command that must be executed to update the target.

Dependency file—a file upon which a target file depends.

Dependency line—the line in a makefile that specifies a target file and the file(s) the target depends on.

Description file—also known as the makefile. This file contains a sequence of entries that specify target files, their dependencies, and the commands to be executed to create an updated target.

Header file—a file containing definitions and declarations that are used by a C program source module. The filename usually has the ".h" suffix.

Include directive—the file inclusion directive for the C language preprocessor. The file specified in this directive replaces that directive.

Include file—see header file.

Linking—the process of combining one or more relocatable object files to create another (larger) object file.

Load module—the program image on disk that is ready for loading into memory for execution.

Makefile—see description file.

Object file—the file produced by compilation of source code.

Preprocessor—the first program invoked in the C language compilation sequence. This program can do file inclusion, token replacement, and control conditional compilation.

Rule—the command for updating a target file.

SCCS file—the file that is the custodian of the source text module in the SCCS system.

Source code—the text file that consists of C programming language statements.

Source text—the file of human-readable text that is incorporated into the SCCS system.

Target file—the file to be "made" (or updated) by the Make system.

Text file—a file of human-readable characters. The Unix system employs the ASCII character set in text files.

Dr. Rebecca Thomas, 1839 Tenth Avenue, San Francisco, CA. 94122. Dr. Thomas is an author of the best-selling *A User Guide to the Unix System.* Dr. Thomas also works as a consultant specializing in the Unix operating system and the C language. She lives with three wonderful cats—Beetle Bailey, who has retired, and Nubbins and Gray Matter, who still keep her company by lying on Dr. Thomas' terminal while she works.

Bibliography

Feldman, S.I., *Make—A Program for Maintaining Computer Programs.* Bell Laboratories, August 1978.

Bradford, E.G., *An Augmented Version of Make.* Bell Laboratories, January 1981.

Bell Laboratories, *The Unix User's Manual.* Release 3.0. June 1980.

Roome, W.D., "Programmer's Workbench: New Tools for Software Development." *Bell Laboratories Record,* vol. 75, no. 1, pp. 19, January 1979.

Flourney, M.L., "Software Workbench Automates Program Modification." *Data Communications,* vol. 11, no. 2, p. 107, February 1982. ■

THE SEATTLE UNIX GROUP

BY IRENE PASTERNACK

The Seattle/Unix Group is a new organization of Unix system users in the Seattle area. The organization was born from the efforts of David Herington and Fred Weir, both of Sun Microsystems, who organized the first meeting in May 1984. For that meeting, they invited Bill Joy to present a talk on "Networking with Unix" and mailed invitations to the 30 or so organizations they knew that were involved with the Unix system.

From there, news of our group spread by word of mouth, and interest in our group grew quickly. The first meeting drew 160 people, while a list of those interested in the group approached nearly 400 people. By the second meeting, which was hosted by AT&T Information Systems, we had over 100 paid members. AT&T Information Systems used the event to present information on the availability of their 3B2 supermicrocomputer, 6300 Personal Computer, printers, and related software.

At that stage, the Seattle/Unix Group was still operating on an ad hoc basis. The Seattle community of Unix system users had been growing steadily, and people needed a way to share information and resources. We were rapidly outgrowing our old informal ways, the phone, and the Usenet network. Of the 100 members, fewer than 20 sites are connected. I had heard many discussions on the need for organizing a user group, but no one wanted to tackle the logistics of starting a new organization until Sun took the initiative.

The initial mailing called for people willing to be on the board of directors of the new organization, and a board was elected from the people who responded. The first board was elected for a one-year term and will serve until May 1985.

The board's first tasks were to survey participants to find out what they wanted from a users' group in order to create a simple organizational structure that would allow the group to focus on sharing Unix system information. We found a regular meeting space that could grow with us and found vendors to continue the tradition of sponsored meetings. Bylaws were written, and the Seattle/Unix Group is in the process of incorporating as a nonprofit corporation.

Monthly meetings are held on the fourth Tuesday of the month at Seattle University. The quarterly meetings are a bigger occasion, with a vendor-sponsored dinner and speaker. We plan to publish a monthly newsletter and already have organized a local news network on Usenet. Our goal is to stay local, serving the needs of the Seattle-area community rather than competing with Usenix or /usr/group.

Special-interest groups (SIGS) have formed for people interested in graphics, real-time, Xenix, foreign languages, networking, databases, and business applications software. Plans for additional member services include a public domain software library and a job resource file.

It is still too early to know where the Seattle/Unix Group is going. The survey results show that most of us are relatively new to the Unix system, with an average of one to two years experience. Surprisingly, the two most popular types of machine are the TRS-80 Model 16 and VAXs. This illustrates the diversity of the group: cottage industry software developers, researchers in artificial intelligence, sales and technical staff from major vendors, programmers at all levels, business end-users, hackers.... We're using a wide variety of versions, including the first Berkeley releases, the newest System V, and several look-alikes (so much for standards).

The momentum of the Seattle/Unix Group belies the friendly acronym (slug) the group goes by. If you're in the area and interested in the Unix system, please join us.

For membership and services information, please write the Seattle/Unix Group, P.O. Box 58852, Seattle, WA 98188; call (206) FOR-UNIX or send mail to ...uw-beaver\!teltone\!ssc\!slug. ■

Irene Pasternack, who heads the board of directors of the Seattle/Unix Group, is a relative newcomer to the Unix system. Her association with the system began two years ago as a trainer and technical writer. She is the director of Specialized Systems Consultants and has authored several of the firm's pocket references for the Unix system and C. Irene holds a B.A. from Princeton University and an M.A. from Whitworth College. She lives in Seattle with her bicycle, a kayak, and a computer.

MARKETPLACE
CAREER OPPORTUNITIES

Readers who have computer merchandise or services to sell, buy, or trade and who want to be included as a classified advertiser in UNIX/WORLD Magazine should write to: Tech Valley Publishing, 289 S. San Antonio Rd., Los Altos, CA 94022, or call (415) 949-3737. The rate for the classified section is $12 per line. The deadline for ads is the first of the month prior to publication. Please include complete name and address in every ad.

Cucumber Books—UNIX and C Language books. Whether you need one of fifty, we stock all titles. Cucumber Bookshop is located at 5611 Kraft Drive, Rockville, MD 20852, (301) 881-2722. Mail and phone orders.

Dyalog APL—Today's most productive application development tool for commercial and educational users. Implemented in C, it gives you the power of UNIX directly from within APL itself. It is a new generation APL plus UNIX, NC. Available for VAX, HP-9000, Gould, Zilog, Fortune, Xenix, and Unisoft. Call or write to: Notation, 10 Jackson St., Los Gatos, CA 95030. (408) 354-3274.

WMZ Consulting Group—System Consultants for total implementation of information systems. WMZ provides design, implementation, training, and ongoing support. We have experts in UNIX on staff and we are not committed to any software or hardware. We sell solutions not iron. Call (415) 945-2070, or write WMZ Consulting Group, 1483 Ashwood Drive, Martinez, CA 94553.

A User Guide to the UNIX System—by Rebecca Thomas, PhD, and Jean Yates. This easy to learn, self-teaching UNIX book is an essential part of the UNIX users library. Was $17.95, now $14.95. Write or call Tech Valley Publishing at 289 S. San Antonio Rd., Ste. 205, Los Altos, CA 94022, (415) 949-3737.

ELXSI VIRTUAL SYSTEM V

ELXSI has announced its version of the Unix System V operating system to run on the System 6400, a 64-bit computer that can be modularly expanded from supermini to supermainframe size. ELXSI's Enix implementation is available separately or running in parallel with the ELXSI message-based operating system EMBOS. EMBOS provides enhancements, including demand-paged virtual memory.

ELXSI's Enix operating system provides a fast file system, virtual process forking for running parallel jobs, and generalized interprocess communication for increased speed. Fourteen separate CPU or I/O processors can be attached to the System 6400 running Enix. The Enix system also offers the TCP/IP software protocol.

Enix runs in parallel with EMBOS on the 6400 and provides users with ELXSI's data management system, derived from UC Berkeley's Ingres, and keyed file access method KAM for fast record access using relational data. It also allows files to be mapped directly into memory for speed, performs batch processing, and provides mirrored (carbon) files. The EMBOS operating system supports many programming languages, including Pascal, COBOL, FORTRAN, C, and Mainsail.

The ENIX operating system was to become available in July and is priced at $24,000 for one to 32 users, $30,000 for 33 to 64 users, and $36,000 for more than 64 users.

For more information, contact ELXSI, 2334 Lundy Place, San Jose, CA 95131, (408) 942-1111.

Please circle Reader Service Number 100.

UNIQ RESOURCE PLANNING SYSTEM

Uniq Digital Technologies has introduced the Spider II Manufacturing Resource Planning System, a modular software system composed of integrated manufacturing and accounting packages developed by MCBA, Inc. Spider II was developed specifically for use with AT&T's Unix System 5.2 operating system and the DEC VAX series of superminicomputers.

Spider II capabilities include tracking the basic elements of planing control—inventory, material requirements planning, custom order processing, standard product costing, job costing, and shop floor control—as well as providing integrated accounting modules. The package also offers word processing, financial modeling, with Unicalq electronics spreadsheet, computer-to-computer communications, and electronic mail capabilities.

Unix System 5.2 is the latest update from AT&T and is designed to prevent "lock-in" to the products of any single computer manufacturer. This means that some existing hardware can be integrated into the Spider II System.

For more information, contact Grant F. Thomas, Uniq Digital Technologies, 28 South Water Street, Batavia, IL, (312) 879-1008.

Please circle Reader Service Number 101.

PHILON AND PIXEL COMPILER MARKETING AGREEMENT

Philon, Inc., and Pixel Computer have announced an agreement through which Pixel Computer will market Philon's series of fastexecuting, portable language compilers for the MC68000/Unix operating environment.

Although details of the terms and conditions were not disclosed, Pixel retains the rights to market the four currently announced Philon Fast/Compilers—Philon Fast/BASIC-C, Philon Fast/BASIC-M, Philon Fast/COBOL, and Philon Fast/C—in addition to all future Philon Fast/Compilers developed for the MC68000 systems.

For more information, contact Philon, Inc., 50 Cooper Square, N.Y., NY 10003, (212) 420-0317.

Please circle Reader Service Number 102.

INTERNATIONAL MEDICAL CORPORATION'S MULTI-MED

International Medical Corp.'s new entry into the medical computer field, Multi-Med, manipulates patient medical and administrative data in one central office computer system.

Multi-Med can generate comprehensive reports for 12-lead resting ECGs, Holter ECG tests, or exercise ECG procedures. Once Multi-Med has acquired a patient's physiological data, a comprehensive risk factor profile can be automatically generated using a patient database for comparison. The risk factor analysis program provides valuable information to the physician about the likelihood of latent cardiovascular disease for each patient.

Because Multi-Med has a menu interface, it is easy to learn and operate. This system can accommodate up to 20 terminals simultaneously. with simultaneous access to all parts of the program. Hard-disk storage capacity ranges from 10 to 336 Mb.

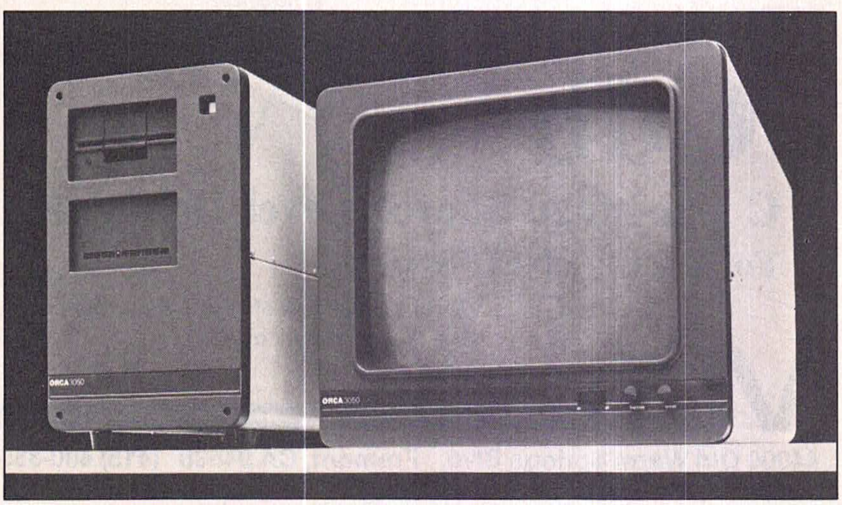
Multi-Med has data security with automatic backup to protect patient records, direct communication with other physicians or hospitals, and hook-up to various medical databases and research centers.

Standard Multi-Med sells for $17,000, including practice management software, terminal, printer, and 10-Mb mass storage. The standard system will accommodate patient files, accounting information, and the Unix operating system.

Free four-day training workshops for physicians and office staff are offered in Denver. Continuous Medical Education accredited Multi-Med seminars will be presented at several locations nationwide through International Medical Education Corp., a nonprofit educational organization accredited by the American Medical Association.

For more information, contact International Medical Corp., a division of Medtronic, Inc., 3055 Old Highway Eight, P.O. Box 1453, Minneapolis, MN 55418, (612) 574-4000.

Please circle Reader Service Number 103.



ORCATECH WORKSTATION

Orcatech, Inc., a Canadian supplier of graphic computer CAD/CAG systems, has introduced its new workstation, the Orca3050. Orcatech's workstation is portable, runs the Unix operating system, and is priced at $20,000-$50,000. This system includes a refresh rate of 30Hz interlaced or 60Hz non-interlaced, 5 1/4 Winchester disk, multiprocessor architecture, and Ethernet networking capability.

The Orca3050 has a wide range of applications, including CAD/CAE, presentation graphics, command and control simulation graphics, and mapping.

For more information, contact Malcolm Cocks, national sales director, Orcatech, Inc., 1000 Morrison Drive, Ottawa, Ontario K2h 8K7.

Please circle Reader Service Number 104.

WICAT'S NEW BASIC PROGRAMMING LANGUAGE

Wicat Systems, Inc., has released a new enhanced BASIC programming language known as W-BASIC, designed primarily for technical and scientific applications. W-BASIC is fully Microsoft-BASIC compatible and compiles with the ANSI standard ANSI X3.60-1978 for BASIC. Any program written in Microsoft BASIC should run unmodified under W-BASIC (except for operating-system-dependent programs or those that use special graphics or sound functions).

Some W-BASIC features: programs are stored in their original text format; user-defined functions can be created with any name; any BASIC command can be executed from a program including EDIT, AUTO, LOAD, DELETE, RENUM, KILL and DIR; program continuation (CONT) can be performed even after the text of the program has been changed; and error messages specify the line number and statement causing the error.

W-BASIC is supported on Wicat systems 140, 150, 160, 200, and 220, and is available under Wicat's proprietary MCS (multi-user) control system.

For more information, contact Wicat Systems, Inc., P.O. Box 530, Orem, UT 84057, (801) 224-6400.

Please circle Reader Service Number 105.

FORTRAN 77 FOR UNIX ENVIRONMENT

Absoft Corp. has introduced several new versions of its FORTRAN 77 compiler for the Unix environment. These compilers are designed for the different Unix-based operating systems running on any computer using the MC68000 family of processors as a CPU.

The Absoft FORTRAN 77 compiler is written in 68000 assembler and is a superset of the ANSI/IEEE Standard X3.9-1978. Major extensions of the compiler include a full-screen source-level symbolic debugger, virtual array management, profiler, and Sky FFP hardware support.

This series of compilers is currently available from several OEMs whose Unix implementations include Unix System III, Unix Version 7, Berkeley 4.1 BSD, Unix System V, Unix V/68, Idris, Xenix, Regulus, and Unos.

The compiler comes with a 300-page reference manual, symbolic debugger, profiler, and linker. Compile speeds are in excess of 4,000 lines per minute. Execution speed is in excess of 160K Whetstones.

For more information, contact Absoft Corp., 4268 No. Woodward, Royal Oak, MI 48072, (313) 549-7111.

Please circle Reader Service Number 106.

TANDY/RADIO SHACK SOFTWARE RELEASES

The Precedent Legal Accounting System, just introduced by Tandy/Radio Shack, is based on American Bar Association accounting standards. This system integrates time and billing, general ledger, and cash disbursement capabilities, thereby speeding accounting procedures. Transaction information is entered only once, and the system automatically updates all appropriate accounts and client files. For example, if a check is written on behalf of a client in cash disbursements, the time and billing (unbilled inventory) for that client/matter and the general ledger modules are automatically updated for that client.

The Precedent, offered at $795, operates on TRS-80 Model II and Model 12 microcomputers, with a hard disk recommended. Meant for use by law firms or the sole practitioner, the system can track the billing hours of up to 50 people; the number of transactions is limited only by available disk space. Numerous management reports can be prepared, including an unbilled inventory report, trust deposit client advance report, and client/matter report. A standard chart of accounts, included in the program,

A tutorial for first-time users is included in the program manual, along with sample data on the program disk. The Precedent Legal Accounting System (26-4620) is available at Radio Shack Computer Centers.

The second Tandy/Radio Shack software release is a RM/FORTRAN 77 program for the TRS-80 Model 168 microcomputer. With RM/FORTRAN 77, software developers, scientists, engineers, or other programmers now can use the TRS-80 Model 16B microcomputer to write and execute mainframe-level FORTRAN applications. Used with the Model 16B's TRS-XENIX operating system, the program is a complete implementation of the most recent ANSI FORTRAN standard, known as FORTRAN 77.

RM/FORTRAN 77 (26-6451) for the Model 16B is available for $699 at Radio Shack Computer Centers.

The third Tandy/Radio Shack software release is dBASE II, which runs on the Tandy TRS-80 Model 2000 personal microcomputer. This program from Ashton-Tate is used to create, store, edit, retrieve, and manipulate information. With dBASE II, large volumes of data can be organized and structured for rapid access to specific information.

dBASE II contains an application development language that allows users to develop specialized applications. It offers many of the programming capabilities of high-level languages such as BASIC, Pascal, and C and also enables the user to write programs with fewer lines of code and in less time than other high-level languages.

dBASE II may be used on Tandy 2000 computers with two 720K floppy disks or with a 10-megabyte hard disk and one floppy disk. A 256K memory is required, and an optional printer is recommended.

dBASE II (26-5352) is available for $595 at Radio Shack Computer Centers.

For more information, contact Tandy Corp., 1800 One Tandy Center, Fort Worth, TX 76102, (817) 390-3300.

Please circle Reader Service Number 107.

PRODUCTIVITY TOOLS FOR UNIX SYSTEMS

SouthWind Software, Inc., has introduced its new Integrated Productivity Tools (IPT) software packages for Unix-based computer systems. IPT offers work environment tools for the small business and corporate markets. The products feature a spreadsheet, analytical and presentation-level graphics, and a direct interface to popular existing DBMS under a common user interface. Data can be moved directly from the spreadsheet to the DBMS and back, and graphics can be composed from either spreadsheet files or DBMS files.

Tactician, the spreadsheet section of the IPT package, has on-line help and command prompts and an electronic worksheet. Tactician includes an interface to Unify, Informix, or Micro Ingres DBMS. Formatting and output options include floating $, imbedded commas, and parentheses or CR for negative numbers.

The graphics application of IPT —Grafsman—can supply both analytical and presentation-quality business graphics from data stored in the spreadsheet, DBMS, or standard files. Grafsman is built on top of the DI-3000 tool set by Precision Visuals. It is an independent device, allowing for a variety of output devices. Many graphics formats are available: horizontal, vertical, side-by-side, stacked and floating bar graphs, absolute and additive line graphs, pie charts with or without exploded segments, scattergrams, and text-only graphs. Grafsman features multiple charts per picture, multiple text fonts, color graphs, and basic templates for graph design or individualized templates created by the user.

OEMs and VARs will find IPT easy to integrate into their own systems, and IPT can be added to many machine configurable on the NCR Tower 1632 and the Plexus P/35. Future ports include Altos, Sun, Apollo, Wicat, Zilog 8000, and Pyramid Systems. IPT applications may be purchased individually or as a set.

The Integrated Productivity Tools set is the first commercially packaged software system from SouthWind Software. Demo packages became available July 1.

For more information, contact Bill Podlena, SouthWind Software, 4520 E. 47th Street South, Wichita, KS 67210, (800) 346-3025.

Please circle Reader Service Number 108.

UNIX-BASED CROSS-DEVELOPMENT TOOLS

Unix system-based cross-development tools from Uniware are hardware independent, fully portable, and run on almost any 16- and 32-bit Unix-based system. The modular design uses a base module consisting of a macro preprocessor, link editor, and utilities, plus a target microprocessor.

This new release (3.0) features a link editor and object file format, allowing an unlimited size symbol table that is saved for symbolic debugging. The link editor can resolve an unlimited number of object modules and library references into one object module. It supports multiple overlays and uses a C-like specification language that allows users to define load maps, define and use symbols in arithmetic expressions, and create linkages between overlays.

Host processors supported include: DEC PDP-11, VAX; Plexus Z8000, 6800; Onyx; Zilog; Altos 586; and IBM/Amdahl under UTS (by special arrangement). Target supported microprocessors include: Intel 8041/8042, 8035/8048/8049, 8051, 8080/8085, 8086/8088; Zilog Z8, Z80, Z800; Motorola 6800, 6801/6802/6803, 6805, 6809, 68000; Hitachi 6301, 6305; TI TMS 7000 series; General Instruments 1650, 1670; National NS16032, NSC800; Rockwell 6502, 65C02; Mostek 3870/F8; and RCA 1802.

Utilities provided with the base module include link map, size, symbol "name" list, and CRC generator with user-supplied algorithm. The broad selection of object reformatters also offered support Intel Hex, Motorola S-Record, MOS Technology Hex, Tektronix Hex, and Human Readable. Downloaders include: Data I/O PROM programmer; EH4 PROM programmer; Tektronix 8002, 85xx development stations; and user-configurable "generic" downloader.

Uniware cross-development tools Release 3.0 comes with a six-month warranty that includes any program updates occurring within that period. List prices for the base module range from $1,200 to $2,000, depending on the number of users for the host processor. Target module cross-assemblers list at $800 for 8-bit, and $1,200 for 16-bit target microprocessors. Two levels of support are available: full service with telephone consultation, and update-only support.

For more information, contact David Weaver, general manager, Uniware, division of Nuvatec/Inc., 261 Eisenhower Lane South, Lombard, IL 60148, (312) 620-4830.

Please circle Reader Service Number 110.

VAX UNIX PROJECT MANAGEMENT SYSTEM

National Information Systems announced that VUE, a menu-driven computerized project management system, will now run under VAX Unix.

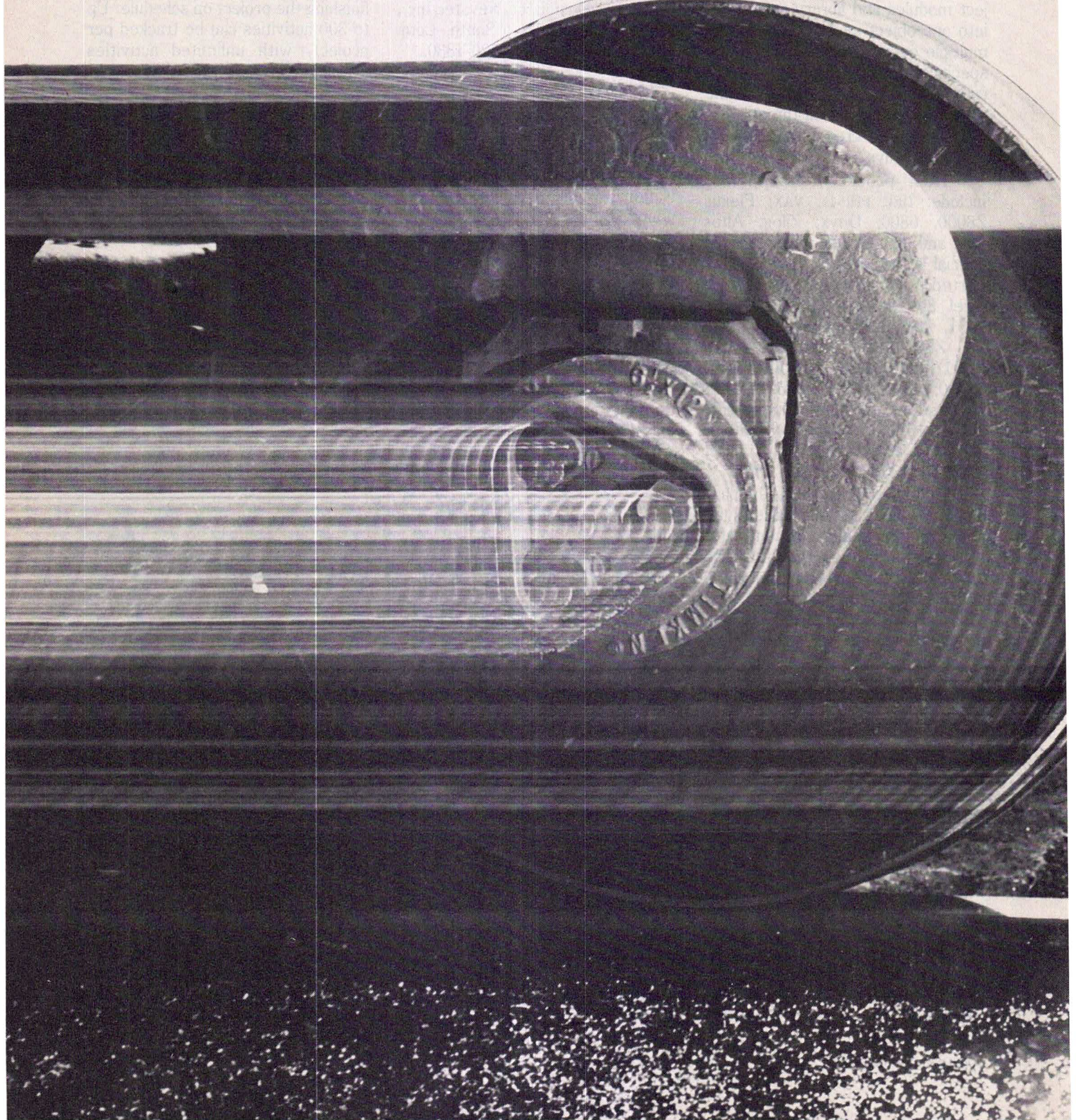
VUE is an interactive project management system designed to help managers establish and monitor project management plans. VUE uses the critical path method, spotlighting those activities whose timely completion is critical to finishing the project on schedule. Up to 300 activities can be tracked per project, with unlimited activities available through the multiproject option.

VUE runs on DEC 10/20, VAX, VMS, PDP-11, HP 300, Honeywell, and Perkin Elmer computers.

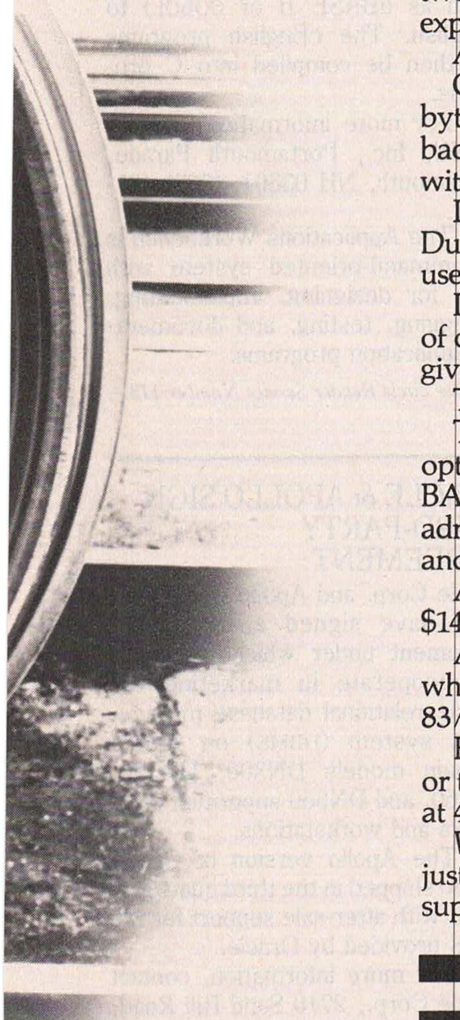
For more information, contact National Information Systems, Inc., 20370 Town Center Lane, Suite 130, Cupertino, CA 95014, (408) 257-7700.

Please circle Reader Service Number 109.

The Association of
didn't spin its wheels with



American Railroads the wrong microsystem.



We're Dual Systems.

And when the railroad people needed a super microsystem to run real-time testing of train wheel axle assembly units with measurements taken at the rate of 500 times per second, they turned to us.

After all, our super microsystems have enjoyed a rather enviable track record with Boeing and Sony and Ford and United Press International, just to name a few.

And for three years we've been delivering 68000-based systems, full UNIX* Version 7 with Berkeley enhancements, and IEEE-696/S-100 bus conformance. That's longer than anyone else.

Which brings us to our newest high-performance micro, the 83/80. Featuring the full UNIX capabilities of Version 7 or System III, the 83/80 can handle up to twelve users and provides a standard 512 KB of dynamic RAM with parity that's expandable to 3.25 MB.

And we've backed it with a full one-year warranty.

Our 83/80 incorporates a high-throughput SMD controller and an 80 megabyte Winchester disk drive with 20-25 milliseconds average seek time. And our backup memory is well worth remembering — it consists of an 8" floppy disk with 1 MB of storage.

In addition, you'll find our 83/80 delivers increased performance through its Dual ported full-track disk buffer and proprietary controller circuiting. More users can access with better response time.

It's also very well-educated. Our 83/80 can read or write up to an entire track of data in a single disk rotation, regardless of where the disk-head settles on a given track.

That's smart.

The C programming language comes standard with UNIX, of course. Other optional languages include FORTRAN-77, PASCAL, RM/COBOL®, LISP and BASIC. And that's just for starters. Optional software includes data base and administrative packages like INGRES and UNIFY.

And the sticker price? Quantity 10 at $14,693.

As you can see, we had our own wheels turning when we designed the 83/80.

For further information, please write or telephone our Marketing Department at 415/549-3854.

We'll be glad to give you more than just a whistle-stop tour of one powerful super microsystem.

DUAL

Dual Systems Corporation
2530 San Pablo Avenue
Berkeley, CA 94702

*UNIX is a trademark of Bell Laboratories.
® RM/COBOL is a registered trademark of the Ryan McFarland Corporation.

Please circle Ad No. 7 on inquiry card.

CSSI FINANCIAL ACCOUNTING SYSTEM

CSSI Corp. has introduced Bizpak, a financial accounting system available on Pixel, Altos, and Convergent Technologies computer systems. Bizpak runs on the Unix operating system and is implemented in RM/COBOL. The system is fully integrated, menu-driven, and supports multiple companies, multiple warehouses, and multiple users. Bizpak consists of accounts payable, accounts receivable, general ledger, inventory management, sales order processing, sales analysis, and payroll modules.

Bizpak source code licenses are $500 per module, and a complete system license is $3,000, a savings of $500.

For more information, contact CSSI Corp., 10700 Lyndale Ave. So., Bloomington, MN 55420, (612) 881-4501.

Please circle Reader Service Number 111.

AUTOMATED FORMS MANAGEMENT SYSTEM

This new computer-based system is designed to automate the manual systems and procedures that Forms Management Systems (FMS) has developed. The AFMS will automatically provide middle- and upper-level management with information needed to reduce costs on printing, forms, and supplies. The system also promotes efficient use of existing paperwork, supplies, time, and space. The system will enable users to obtain and control data pertaining to the use and flow of all forms, paperwork, and supply items used by their companies.

Five integrated modules comprise the system. The base module incorporates a DBMS, report generator, and various utility programs. Once the base module is installed, other modules may later be added to the system at the user's option.

The forms management module is the key component in the AFMS, serving as the basic information resource for each installation. This module is a master relational database for all forms and supply item specifications. It provides the basis for the system's ability to track and monitor forms activity. The system contains and organizes all forms and supplies specifications, characteristics, and functions for sorting on any desired attribute. It provides functional codes for all forms and supplies, letting the user identify materials on the basis of their use, thereby preventing duplication and promoting consolidation.

For more information, contact Forms Management Systems, 350 Half Day Road, P.O. Box 212, Prairie View, IL 60069, (312) 634-1120.

Please circle Reader Service Number 112.

NEW C DEVELOPMENT TOOLS

cLINE, Inc., has introduced a family of integrated software development tools for the business applications programmer. The product line consists of cEnglish, cEnglish compiler, Applications Workbench, and Language translators.

cEnglish, an application development language that uses high-level commands, combines the best features of two languages: the power of C and the ease of English. Intended for use by programmers of all levels, the language is primarily procedural in nature and supports a complete set of standard control flow and data manipulation facilities. It provides RDBMS and screen management capabilities, as well as a unified interface to a variety of database management systems; yet the language retains its independence from any specific DBMS.

The cEnglish compiler is a stand-alone program that provides for file and macro expansion, semantic and syntactic error checking, and strong data typing with implicit conversion. It translates cEnglish programs into functionally equivalent C programs, and most cEnglish constructs map directly into equivalent object level constructs to yield object code.

The cLine includes language translators that convert source programs written in other languages (such as dBASE II or COBOL) to cEnglish. The cEnglish programs can then be compiled into C programs.

For more information, contact cLINE, Inc., Portsmouth Parade, Portsmouth, NH 03801, (603) 431-2111.

The Applications Workbench is a command-oriented system with tools for designing, implementing, debugging, testing, and documenting application programs.

Please circle Reader Service Number 113.

ORACLE & APOLLO SIGN THIRD-PARTY AGREEMENT

Oracle Corp. and Apollo Computer, Inc., have signed a third-party agreement under which the firms will cooperate in marketing the Oracle relational database management system (DBMS) on Apollo Domain models DN300, DN320, DN460, and DN660 supermini computers and workstations.

The Apollo version of Oracle will be shipped in the third quarter of 1984, with after-sale support for the DBMS provided by Oracle.

For more information, contact Oracle Corp., 2710 Sand Hill Road, Menlo Park, CA 94025, or Apollo Computer, 15 Elizabeth Drive, Chelmsford, MA 01824. ■

Please circle Reader Service Number 114.

NEWS FROM AT&T

AT&T OPENS UNIX SYSTEMS OFFICE IN JAPAN

AT&T has opened a Unix systems office in Tokyo that will be responsible for the marketing, licensing, and support of software products in the Far East, including Japan, Australia, Hong Kong, Korea, Malaysia, New Zealand, Singapore, and Taiwan. AT&T also said it will produce a standard Japanese version of the Unix system.

Both announcements were made in Tokyo by John A. Hinds, vice-president of market development for AT&T International, the overseas marketing unit of AT&T, and Thomas H. Crowley, software systems vice-president in AT&T's Computer Systems Division.

Larry L. Crume, head of Unix Systems Engineering at AT&T Bell Laboratories, was named director of the Unix systems office, with an initial team of five people. The operation will be located at the headquarters of AT&T International (Japan) Ltd. in Tokyo.

As a first step, Crowley said the office's beginning operations will include the creation of a Unix systems standards committee, comprised of representatives from leading Japanese companies. The committee will be charged with investigating many areas of the Unix system.

PRIMARY AIM

Its principal objective, however, will be to advise AT&T on the key elements of a standard Japanese version of the Unix system that will be compatible with the U.S. standard System V. The Unix systems office will handle the administration of current Unix systems licenses as well as all future licensing activities. It will also be responsible for the distribution of software and documentation, service, and maintenance.

In explaining the firm's decision to enter the Far East market, AT&T's Crowley noted that the current worldwide Unix system market breaks down as follows: 73 percent in North America, 17 percent in Europe, 9 percent in the Far East, and 1 percent other. However,

FIGURE 1: AT&T 3B SOFTWARE

Product Name	Configuration	Price	Availability
Microsoft Word	3B5 w/1 Mb memory 3B2 w/512 Kb memory	3B5 $ 1,300 3B2 $ 650	10/84
Multiplan	3B5 w/1 Mb memory 3B2 w/512 Kb memory	3B5 $ 1,000 3B2 $ 500	9/84
RM COBOL		3B5 $ 3,000 run-time version $ 600	8/84
dBase II	3B5 3B2	3B5 and 3B2 (both) $ 1,200 $ 600	7/84
AT&T Ingres/C5-Ingres	3B5 2/160 Mb disk, 4 Mb memory 3B2 32 Mb disk, 2 Mb memory	3B5 $17,000 3B2 $ 2,000	8/84 9/84
Informix	3B5 w/1 Mb memory 3B2 w/1/2 Mb memory	3B5 $ 3,000 3B2 $ 1,600	8/84 8/84
File-it!	3B5 w/1 Mb memory 3B2 w/1/2 Mb memory	3B5 $ 815 3B2 $ 495	9/84 9/84
PC-Interface	3B2 w/1/2 Mb memory	3B2 $ 500 Software: $ 1,000	10/84
AT&T Business Auditing System	3B5 (5 modules) 2 Mb memory, 160 Mb disk	All 5 Modules: $10,000	7/84
	3B2 (5 modules), 1 Mb memory 10 Mb disk	All 5 Modules: 3B2 $ 5,000	7/84
Gift Registry	3B5 2 Mb memory & 160 Mb disk 3B2 2 Mb memory & 32 Mb disk	3B5 $ 3,000 3B2 $ 3,000	7/84 7/84
Communications Management Control System	3B5 w/1 Mb memory min. 3B2 10 Mb disk	3B5 (Total Package) $ 4,500 3B2 (Total Package) $ 2,500	10/84
C-ISAM	3B5 w/1 Mb main memory 3B2 w/1/2 Mb main memory	3B5 $ 900 3B2 $ 450	9/84

For more information on these 3B and PC software offerings, contact AT&T software sales and marketing at 800/828-UNIX.

since early 1983, growth rates have been strongest in the Far East and Europe, at 69 percent and 53 percent, respectively. North America, on the other hand, has grown 30 percent.

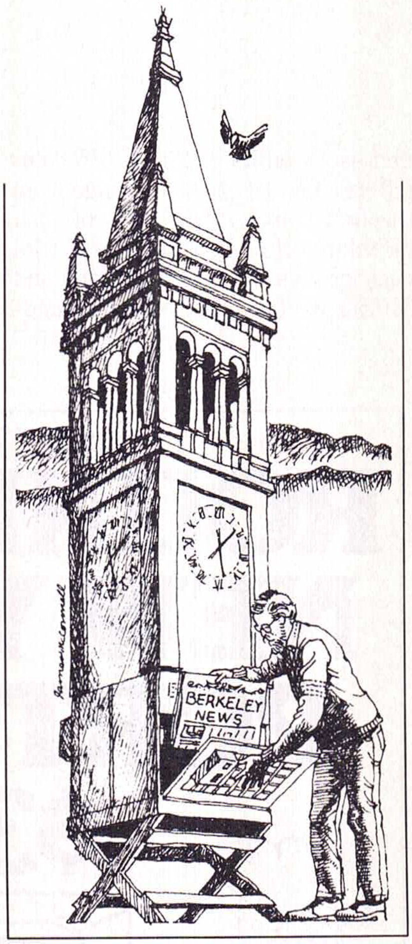
Earlier, AT&T and Ing. C. Olivetti & Co., S.A., in which AT&T holds a 25 percent stake, disclosed the formation of Unix Europe, a joint-venture company to promote Unix System V and to serve the needs of customers in Europe.

SOFTWARE FOR THE 3BS

AT&T has introduced the first software packages for sale to end-users on its 3B2 and 3B5 computers. The 12 packages range from the first Unix system ports of such popular office productivity tool packages as Microsoft's Word and Multiplan (an electronic spreadsheet) and Ashton-Tate's dBase II database manager, to more traditional Unix system favorites such as the Informix and Ingres relational database management systems (see Figure 1). ■

WHO WILL DEVELOP BERKELEY UNIX?

BY ALAN TOBEY



The emergence of the Unix system as a potential "standard" operating system for the 1980s—on a wide range of machines from micros to mainframes—has raised important questions about how its development will be encouraged and controlled in the future. Now that Unix has caught the attention of marketeers as well as technical gurus, the forces pulling the Unix system in different directions will be coming from the marketplace as well as the labs. The resultant tug-of-war between market needs and technical needs will profoundly influence the nature of future Unix systems.

Nowhere is the incipient struggle between market and technology more noticeable than in the split between AT&T's "official" Unix System V and the 4BSD "research" Unix from UC Berkeley. Looking at their current states and predictable development paths means untangling differences that cut to the heart of Unix system philosophy and implementation.

AT&T's System V (now supplied as "System V Release 2") is being widely promoted by AT&T as *the* Unix standard. It is intended to be a stable base for a complete range of application programs that will run with minimal changes over a wide collection of hardware. System V is being promoted as "business Unix," the system software that will be the backbone of multi-user office automation systems, even if the Unix system itself is completely invisible to the end-user.

Its attraction *as Unix* will be primarily to the developer, for whom it will provide a consistent working environment independent of the target hardware, and to the hardware manufacturer, for whom it will provide a nonproprietary industry-standard operating system. Most end-users of System V will never see a naked Unix prompt. Their view will be just the application they're using at the time, or a "user-friendly" menu shell controlling access to a restricted set of Unix commands.

With this business market goal in mind, AT&T is promoting System V as a stable, fully supported applications base that will maximize compatiblilty through succeeding versions. This is appropriate for "business Unix," for which rapid change is not desirable; it's also good business strategy for establishing a new standard quickly. And the fact that the development of System V will now be controlled by a large company with deep resources is reassuring to those segments of the industry uncomfortable with smaller startup companies.

A SIGNIFICANT DEPARTURE

From a historical perspective, though, AT&T's attempt to restrict System V's development is a significant departure from the traditional guiding principles of the Unix system, which was developed as an open-ended experimental system. It was explicitly designed to allow for change and improvement, and it was meant to remain at the cutting edge of what is technically and conceptually possible at a given time. Designed as a tool for sophisticated users, the Unix system was never intended to be static over time, or to hold upward compatibility as the pre-eminent virtue. In that sense, the attempt to make System V into a controlled vendor-proprietary operating system is a radical break with the Unix tradition.

It can be argued that this "formalizing" of the Unix system is an inevitable step in its growing maturity. This argument would be true—if it were not for Berkeley Unix. UC Berkeley's work over the past five years has significantly extended the frontiers of Unix—into virtual memory management, networking and distributed Unix systems, screen editors, improved electronic mail, improved interprocess communication, and a number of other areas. Few of Berkeley's improvements are identifiably in the current System V release. AT&T's market caution —together with a healthy dose of a "not-invented-here" syndrome

—have made System V a very tight filter for new Unix system capabilities.

Its improved capabilities and its open-ended nature have made Berkeley Unix the system of choice for technically demanding users who need to push at the state of the art. Since these users are generally focusing on specific applications, the question of a universal standard is less important. Virtually all Unix engineering applications run on Berkeley-based systems, from CAD/CAM to cross-development microprocessor tools to circuit simulators; very few can run on System V, nor are they likely to in the near future.

The Unix camp, then, is dividing along technical lines. Business-application developers with modest capabilities will do well working under AT&T's System V umbrella. Others with more demanding technical applications will need to pull features from any available Unix source, "standard" or not. For the foreseeable future, "technical" Unix will still be identified with Berkeley's 4BSD.

But for technical Unix to survive and thrive, it will have to overcome its negative image of being a riderless horse. It's apparent that System V will develop in a tightly controlled and conservative way under AT&T's direction. Who will develop Berkeley Unix?

BERKELEY UNIX SYSTEMS

It now appears that Berkeley Unix will have a future more in line with the Unix system's traditions, as development occurs in many places and with different goals in mind. UC Berkeley's Computer Systems Research Group is continuing work on performance enhancements to 4.2BSD and is busily conducting research on distributed Unix and networking issues. Professors Domenico Ferrari and Sue Graham and a technical team that includes Unix system wizards Mike Karels and Kirk McKusick are active and visible figures at Berkeley these days. Other universities, such as Purdue and the University of Utah, have active Unix research groups working on different technical issues. And a number of less formal groups and individuals throughout academia continue to contribute to the ongoing demands of technically advanced Unix.

In addition, Berkeley Unix is being actively developed in the private sector. Many of the companies who have incorporated 4BSD into their hardware products—Sun, Pyramid, and Masscomp are significant examples—are working hard to extend the basic 4BSD system in directions helpful to their own long-range plans. Other companies that have recently adopted 4BSD as an alternative to their proprietary operating systems—Digital Equipment Corp. is the most prominent —are looking at diagnostic and support questions as well as enhancements. And the engineering-oriented user base of 4BSD has also managed to improve the tools 4BSD provides for their work.

The open question is how all these scattered efforts will be coordinated. No one in the 4BSD community wants to see Berkeley Unix diverge into dozens of incompatible semi-proprietary systems, but there is no AT&T equivalent capable of imposing a single direction. What is emerging are coordinating efforts in three different areas: (1) UC Berkeley may continue to act as a clearinghouse for 4BSD, even independently of work toward another formal release; (2) more representation on the /usr/group Unix standards committee by 4BSD advocates will tend to ensure at least base-level compatibility among different 4BSD implementations as well as between 4BSD and System V; (3) a now-embryonic consortium or ongoing working group of independent 4BSD developers, including private companies as well as funded research groups, may produce important agreements on 4BSD directions. Some companies, for example, have expressed a willingness to share the protocols they have developed to support their private implementations. This would be another aid to widespread compatibility and would also lessen the need for duplicated effort.

In any case, the open development path of "technical" Berkeley Unix will prove an interesting contrast to the more monolithic AT&T approach. We can expect continuing technical advances, efforts toward consensus and nonduplication, and an ongoing struggle to keep the 4BSD universe within reasonable common bounds. This may prove frustrating for applications developers until further standards decisions are reached, but open development is much more in line with the traditional Unix system philosophy, which has proven its major strength up to now. ■

Alan Tobey is the marketing director of Mt. Xinu, a Berkeley-based company that specializes in Unix systems software.

A C SHELL SCRIPT

DEAR SYSTEM DOCTOR:

We have a C-shell script, memo, that works great for us. The major function of memo is to send a message or a message file with file name filename.ext to someone's terminal at a specified time. If the person is not logged in, the message will become a mail message automatically.

After we read your introductory issue of UNIX/WORLD, we felt

FIGURE 1: MANUAL PAGE FOR memo SYSTEM

```
NAME
     memo - send memos to oneself or to others
            at a specified time
SYNOPSIS
     memo
     memo LOGIN_NAME
     memo TIME [#]MESSAGE
DESCRIPTION
     MEMO is a csh script used to send messages to individuals. Without arg-
     uments, MEMO asks for the delivery time and message one wishes to send to
     oneself. "Time" in this case is as described in the AT(1) command.  If
     the message is a singlular name preceeded by a "#" sign, then MEMO treats
     the message as a message file containing text to be sent. The message file
     must already exist in the users Message file directory before using MEMO.
     If not preceeded by a "#" sign, the body of the message, up to a line
     containing a single period, ".", is assumed to follow the message prompt.

     With the argument LOGIN_NAME, MEMO sends a message in the same interactive
     format given above, but to the person with the login name, LOGIN_NAME,
     instead of to oneself. LOGIN_NAME may also be in the ARPA address format:
        "name@site" -- where "site" is the VAX/SUN system site (such as
        "orville", "wilbur" or "wiley" at NASA-Ames) and "name" is the users
        login account name on that system.
     Again, a message file may be substituted for a message, as indicated
     above.

     With more than one argument, MEMO silently sends the given MESSAGE (any
     number of lines, up to a line containing a single ".") or message file
     (again indicated by using a "#" before the filename, MESSAGE) to oneself
     at the specified TIME. TIME in this case is as given by AT(1) without the
     optional day, as described there.

     A directory, $HOME/Messages, should be created to hold the routines
     which comprise the MEMO command (memo, memo_act and memo_sub) as well as
     the files "msg*" for storing memos to be sent.  These routines (which are
     all executable, mode 755 files) are used to create and destroy these msg*
     files, and the file "msgcnt" keeps track of the (ordinal) number of them.
     All "#" message files must also be located in the same $HOME/Messages
     directory as for the MEMO files listed above.

     MEMO sends its messages by first attempting to write to the tty at which
     the user is located.  Failing this, the message is sent via MAIL(1) to
     the users mailbox.  If "BIFF(1) y" is set, then the user is informed
     immediately that a message has arrived.

FILES
     memo_act - used in .login files to do housekeeping for MEMO
     memo - csh script file which executes MEMO commands
     memo_sub - csh script file which runs memo inputs in bg
     msgcnt - numeric ASCII file containing current msg count
     previousdate - date last logged-in, used for bookkeeping purposes
     msg* - actual messages files ("*" indicates msg number)

SEE ALSO
     at(1), cron(8)

BUGS
     Because of the granularity of the running of the "atrun" file by cron(8),
     times are accurate to within only 5 minutes (15 minutes on the SUN) of the
     given time.
```

FIGURE 2: THE memo MAIN DRIVER csh SCRIPT LISTING

```
$ cat -n -v memo
     1  # memo.csh
     2  pushd $HOME/Messages > /dev/null
     3  set noglob
     4  set CR = "^M"
     5  set person = `who am i|sed -e "s/.*\\!//"`
                                #get addressee default name
     6  if ($#argv > 1) then    #memo being run non-interactively?
     7      set tmsg = ($1)     #yes : get time msg is to be sent
     8      set apptmt = ($argv[2-$#argv])      #and body of msg
     9  else                    #memo being run interactively:
    10      if ($#argv == 1) set person = ($1) #get name of
        addressee<>default
    11      echo 'Delivery time/date of message?'
    12      set tmsg = ($<)             #and the time/date of the msg
    13      echo 'Enter message now, please.'
    14      set apptmt = ($<)           #get msg body/msg filename
    15      set apptmt = `echo $apptmt:q`
    16  endif
    17  echo '' >! $HOME/tmpmsg
    18  if ($#apptmt != 1 || $apptmt[1]:q !~ "#"*) then
    19     while ($apptmt[1]:q != '.')
    20         echo $apptmt $CR >> $HOME/tmpmsg
    21         set apptmt = ($<)
    22     end
    23  else
    24     echo $apptmt $CR >! $HOME/tmpmsg
    25  endif
    26  (memo_sub $person[1] $tmsg > /dev/null)& #run memo pgm in bg
    27  popd > /dev/null
$ []
```

that *Wizard's Grabbag* was the ideal place for us to share this C-shell script as a response to your solicitation. Enclosed is the source code listing and the "Doc" files.

Sincerely yours,

George N. Pornaras

Yen-whei Chow

George N. Pornaras and Yen-whei Chow work at NASA's Ames

FIGURE 3: THE memo_sub SUBROUTINE csh SCRIPT LISTING

```
$ cat -n -v memo_sub
     1  # memo_sub (subroutine to memo.csh)
     2  set noglob
     3  set person = ($1)                       #get name of addressee and
     4  set tmsg = ($argv[2-$#argv])            #time/date of msg
     5  unset msgname
     6  set aptmtcnt = `cat $HOME/tmpmsg | wc -w` #establish msg/msg-file flag
     7  if ($aptmtcnt == 1) then                #is it a msg-file?
     8      set apptmt = `cat $HOME/tmpmsg`
     9      if ($apptmt[1] =~ "#"*) set msgname=(\\$apptmt[1]) #yes, flag it
    10  endif
    11  unset noglob
    12  if (! $?msgname) then                   #is msg-file flag set?
    13      set msgcnt = `cat msgcnt`           #no, get next available
    14      while (-e msg{$msgcnt})             #msg count number
    15          @ msgcnt++
    16      end
    17      set msgname = (msg{$msgcnt})    #label the msg with a msg number
    18      cat $HOME/tmpmsg >! $msgname    #dump msg into this numbered file
    19      @ msgcnt++
    20      echo $msgcnt >! msgcnt              #set msg no. to next available
    21  endif
    22  echo \# >! at_pad                   #initialize at-file
    23  echo set logged_in = \`w -h $person\` >> at_pad
    24  echo set nomsgs = \`finger \| grep $person \| grep \"\*\"\` >> at_pad
    25  echo 'if ($#logged_in && \! $#nomsgs) then' >> at_pad #logged-in/mesg n
    26  echo '  set ntty=\`who\|grep $person\`' >> at_pad #okay
    27  echo '    echo ^M^G Reminder Message...^G^M > /dev/$ntty[2]' >> at_pad
    28  echo '    'cat $msgname \> '/dev/$ntty[2]' >> at_pad #write on screen
    29  echo '    echo ^M^G > /dev/$ntty[2]' >> at_pad
    30  echo else >> at_pad                     #user not logged-in :
    31  echo "   mail $person < $msgname" >> at_pad #mail msg to user
    32  echo endif >> at_pad
    33  echo /bin/rm $msgname >> at_pad     #discard msg afterwards
    34  chmod 755 at_pad
    35  at $tmsg at_pad                     #send msg at appropriate time/day
    36  /bin/rm at_pad $HOME/tmpmsg         #clean-up files
$ []
```

ADVERTISERS INDEX

If you are interested in products or services advertised in UNIX WORLD, our advertisers are pleased to send you additional information.

Simply fill out the reader service cards and circle the numbers of advertisers you are interested in. Then mail the card and expect prompt, pertinent information.

Reader Service No.		Page No.
1	**Accucom:** /USR/TROFF—Lets your PC set 15½ feet of typesetting per minute.	131
61	**Aim Technology:** Evaluating Unix/Xenix systems? Contact Aim Technology for a free benchmark manual	143
2	**AT&T:** Unix System V training —comprehensive, hands-on courses from the creators of the Unix operating system.	29
3	**CMI Corporation:** SERIX—The high-performance CMI version of AT&T's Unix System V operating system with Berkeley 4.1 enhancements ported to the IBM Series/1 minicomputer.	119
4	**Contel Codata:** The Codata 3300—a powerful 16-bit, 68000-based multibus system that can accommodate up to 10 users.	12
5	**Computer Technology Group/ Telemedia, Inc.:** Unix system and C hands-on training: Seminar video-based training, and interactive videodisc training.	74
6	**Convergent Technologies:** The MiniFrame- Superminicomputer power and Unix versatility at PC prices. Also part of the Convergent family is the MegaFrame, a revolutionary superminicomputer system.	10,11
54	**DeNardi Enterprises:** The California Computer Show—Palo Alto Hyatt Hotel, Palo Alto, Calif., December 6, 1984.	30
7	**Dual Systems Corporation:** The 83/80 —A new high-performance micro with full Unix system capabilities of Version 7 or System III that can handle up to 12 users.	126, 127
8	**Excelan:** Change a 12-month design cycle to 12 days with EXOS, the technology that provides complete LAN capability.	1
9	**Elxsi:** Performance and expansion potential: Inside the 64-bit multiprocessor system 6400.	Back Cover
32	**First Computer Corporation:** Your compatibility connection. Services include: systems integration, distribution of computer products and service contact.	18
10	**Gould, Inc.:** Manufacturers of the Power -Series 20 Computers—Unix-based systems that span the performance range.	4
56	**Gould, Inc.:** The PowerNode 9000, Power-Series and UTX/32 —firebreathers that are scorching old performance standards.	48
11	**Gould Software Division:** New Unix-based software tools that bring projects in on time and on budget.	Outside Back Cover
12	**Handle Corporation:** The first office automation system for the Unix System —selected for distribution by AT&T.	14
13	**Hayden Book Company:** Hayden's videotapes and books make it easy to learn the Unix system and write programs in C.	20
14	**Heurikon Corporation:** OEMs will like the one-size Minibox, a Unix system multi-user, single-board workstation.	83
15	**Hewlett-Packard:** The HP-UX. HP's enhanced version of Unix. Available on the M68000-based machines and 32-bit systems, along with a growing array of applications software.	40,41
16	**Human Computing Resources Corporation:** Making creative connections between Unix developers and Unix users —partners in Unix excellence.	34
17	**Human Designed Systems, Inc.:** The Concept AVT+—a smart interactive display terminal widely used by Unix users.	107
18,60	**Integrated Business Computers:** Ensign—The Unix-based system with up to 32-user capacity, up to 8 Mb memory, and over 1,000 Mb disk storage.	31
59	**ITOM International Co.:** The Supermicro Newsletter—written by the folks who invented the term.	106
19	**Logical MicroComputer Company:** The MegaMicro—a 32 bit multi-user, virtual-memory microcomputer. Ideal for the OEM.	54
20	**Logical Software Inc.:** Softshell—a convenient interface for Unix users at all levels of expertise.	102
21	**Mark Williams Company:** COHERENT, a multi-user, multi-tasking operation system compatible with the Unix system for the IBM PC, the 8086, and the PDP-11 Series.	58
22	**MC2/Measurement Concept Corporation:** CAST (Computer Assisted Self-Training), a new and powerful authoring language and information management system.	49
23	**MCBA, Inc.:** Shrink-to-fit software. Integrated manufacturing, distribution, and accounting packages.	141
24	**Medical Informatics, Inc.:** Affordable power and performance—the M-5 computer system is 68010 CPU, 2¼ Mb memory, 10 user, 80 Mb removable disk drive, 75 ips, 1600 bpi tape drive, and Unix 4.2 BSD.	117
25	**Micro Focus:** Personal COBOL, Level II COBOL and Animator—for the effective programming environment.	47
26	**Mt. Xinu:** The Unix system for the DEC VAX.	117
27	**NCR Corporation:** For OEMs, the Tower 1632. Built for system builders.	52,53
29	**Philon Inc.:** PHILON/FAST compilers—now for M68000/Unix.	116
30	**Productivity Products International:** Objective-C—the language that adds the concepts of messaging, objects, and inheritance to any C compiler.	140
	Plexus Computers, Inc.: Beat our benchmark and receive a Heuer chromograph watch valued at $485.00.	Inside Front Cover
71	**Prescience:** HIEROGLYPH—The universal report production for the Unix System: Integrated text data and graphics software.	110
	Pyramid Technology: Pyramid Technology 90X—the supermini born to run Unix. No reader service number; please contact by mail or phone.	16
33	**Quality Software Products:** Unix spreadsheet Q-CALC—full Lotus 1-2-3 capabilities, including graphics and menu interface and full access to Unix.	140
34	**Relational Database Systems, Inc.:** Informix and File-It!—the database software family for Unix and MS-DOS.	23,24, &25
35	**Scientific Placement:** National registry of candidates and jobs in the Unix system field.	120
36	**The Santa Cruz Operation (SCO):** For the IBM and Lisa 2-Xenix, the Unix-based operating system, along with a comprehensive line of applications software.	6
37	**SHA Computers, Inc.:** Tower Power —give your Tower more power!	73
39	**Southwind:** Tactician and Grafsman —fast, versatile, and powerful tools.	17
40	**STSC:** The APLPLUS/Unix System —APL for the Unix environment.	129
41	**Syntactics:** Crystal Writer—complete word processing for the Unix System.	82
57	**Tech Valley Publishing:** Unix shell and vi posters. Please clip and send the provided coupon.	63,72
43	**Unify Corporation:** UNIFY—The fast, powerful and flexible relational database management system.	92,93
45	**UniSoft Corporation:** UniPlus+—Unix adapted for the M68000-based microcomputer.	27
46	**Unisource Software Corporation:** Venix/86 for the IBM PC/XT—offering a 30-day free trial.	8
47	**UNIXEXPO:** Unix Operating System Exposition and Conference, New York City, NY, October 16-18, 1984.	65
70	**UNIX/WORLD Magazine:** UNIX/WORLD—your complete guide to the Unix frontier. Subscribe now and receive a 25% discount off the newsstand price!	139
55	**UNIX/WORLD Magazine:** Limited reprints of UNIX/WORLD's premiere issue are now available! To receive yours, please clip and send the provided coupon.	75
48	**User Training Corporation:** Great learning exhilarates and saves you money —the performance courseware for Unix and C.	103
49	**UX Software:** UX-Basic—provides applications portability and productivity.	108
50	**Visual Technology:** High performance, flexibility, extensive software—at a low price. The visual 2000.	76,77
67	**VLS, Inc.:** Cdb C language source level debugger gets all the bugs before they eat all your productivity.	121
68	**VLS, Inc.:** VAX-based systems, custom configuration, sales integration, installation, maintenance.	123
69	**VLS, Inc.:** 4.1- and 4.2-based systems —sales enhancements, custom configuration, and support.	125
52	**Wyse Technology:** WY-75—the DEC-compatible terminal that projects a sophisticated image even when it's turned off.	68,69
53	**Yates Ventures:** The Yates Perspective —a monthly newsletter on the software market. Subscribe now and receive a reduced rate.	2

Research Center as engineers in the NAS project office. George is interested in the development of a workstation network on the Network Systems Corp. Hyperchannel data link. He graduated from the EECS Department at UC Berkeley. Yen-whei Chow is interested in the computer automatic database for NAS system workload and computer system simulation modeling. She graduated in 1983 from the EECS Department at the University of Santa Clara.

FIGURE 4: THE `memo_act` ACCOUNTING `csh` SCRIPT LISTING

```
$ cat -n -v memo_act
     1 #memo_act : memo accounting pgm
       (used for any once-daily routines)
     2 set maxmsgcnt = 15                   #user-defineable
                                             msg count limit
     3 if (! -e previousdate) then
     4     echo '' > previousdate           #initialize previousdate
     5 endif
     6 date|colrm 11 23 >! todaysdate       #get todays date (only)
     7 cmp -s todaysdate previousdate       #is it same as before?
     8 if ($status) then                    #if not, then:
     9       if (! -e msgcnt) then
    10           echo 0 > msgcnt            #initialize msgcnt
    11       endif
    12       set msgcnt = `cat msgcnt`             #get msgcnt of
                                                    present msgs
    13       if ($msgcnt >= $maxmsgcnt) then #does it exceed max?
    14           echo 0 >! msgcnt                  #restart the
                                                    present msg count
    15       endif
    16       /bin/cp todaysdate previousdate #indicate new date
    17 endif
    18 /bin/rm todaysdate
$ []
```

FIGURE 5: AN EXAMPLE OF AN `at` FILE READY FOR EXECUTION

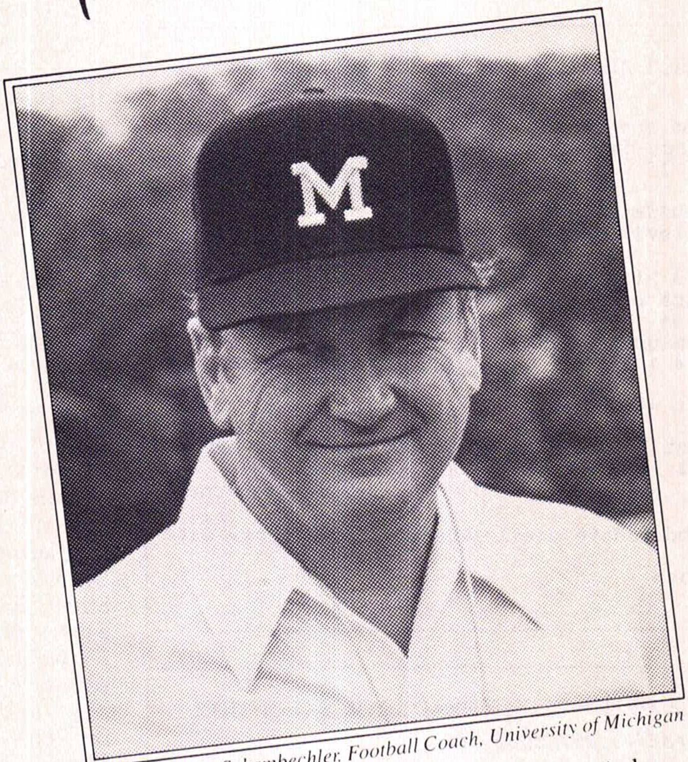
```
$ cat -n -v /usr/spool/at/84.207.0900.25
     1  cd /gc/rathomas/Messages
     2  umask 77
     3  HOME='/gc/rathomas'
     4  export HOME
     5  SHELL='/bin/csh'
     6  export SHELL
     7  PATH='/usr/ucb:/bin:/usr/bin:.:/gc/rathomas/
        Scripts:/gc/rathomas/Bin'
     8  export PATH
     9  TERM='ku'
    10  export TERM
    11  USER='rathomas'
    12  export USER
    13  EXINIT='set wm=10'
    14  export EXINIT
    15  /bin/csh << 'xxFUNNYxx'
    16  #
    17  set logged_in = `w -h rathomas`
    18  set nomsgs = `finger | grep rathomas | grep "*"`
    19  if ($#logged_in && ! $#nomsgs) then
    20     set ntty=`who|grep rathomas`
    21     echo ^M^G Reminder Message...^G^M > /dev/$ntty[2]
    22     cat msg > /dev/$ntty[2]
    23     echo ^M^G > /dev/$ntty[2]
    24  else
    25     mail rathomas < msg
    26  endif
    27  /bin/rm msg
    28  xxFUNNYxx
$ []
```

[*Doctor's note:* The script as written works without change on a Berkeley 4.x BSD system that has the Berkeley-developed utilities `w`, `colrm`, and `finger`. Also, the C shell needs to recognize the `pushd` and `popd` commands and be able to read keyboard input using `set` *variable* `=$(<)`. Figures 1 through 4 were submitted by the authors. Figure 5 shows the contents of my "`at`-file" that was readied for execution by `at`.] ■

Wizard's Grabbag is a regular feature of UNIX/WORLD. Submit your questions, answers, shell scripts, C programs, or tips and techniques that ease the programmer's burden to the *Wizard's Grabbag*, Unix/World, 444 Castro Street, Suite 1220, Mountain View, CA 94041. Authors of published entries receive $25 for questions, $50 for shell scripts, C programs, or tips. By the way, system administrators at UC Berkeley were known as system "doctors" because of their ability to keep the Unix system running smoothly, their cpu-side manner, and those three magic letters that tended to follow their names, Ph.D.

Guidelines for Reader Contributions: (1) Write your shell scripts and C programs so they are portable across different versions of the Unix system. If possible they should run without change on Bell Version 7, System III, V, and Berkeley 4.x. Thus you should employ "universal" Unix utilities when coding shell scripts and the standard I/O library when writing C code. Also use the `lint` syntax checker to locate non-portable constructions and compile with a portable compiler, such as `pcc`, to help ensure portability. (2) Hardware dependencies, such as terminal control codes, should be eliminated or at least minimized and isolated to one code region or to a separate module. (3) Keep your example as short as possible, say under 100 lines of code.

THANKS.
YOU SAVED MY LIFE.



Bo Schembechler, Football Coach, University of Michigan

Not long ago, Coach Schembechler needed 5 units of blood during heart bypass surgery. And if it weren't for someone like you donating blood through the Red Cross, he might not be here today.

The need for blood never ends. The demand is continuous. And there is no substitute. A healthy person can donate blood several times a year.

Make sure enough blood will be there if you, a loved one, or anyone in your community ever needs it.

Call Red Cross for a blood donor appointment. And bring a friend. Thanks.

Donate Blood.

American Red Cross

We'll Help. Will You?

CALENDAR

CALENDAR KEY

(1) Integrated Computer Systems (ICS) Seminars (contact: Ruth Dordick), 6305 Arizona Place, P.O. Box 45405, Los Altos, CA 90045, 800/421-8166, in Calif. 800/352-8251.

(2) Computer Faire, Inc., (CFI), 181 Wells Avenue, Newton, MA 02159, 617/965-8350.

(3) Center for Advanced Professional Education (CAPE), 1820 East Garry Street, Suite 110, Santa Ana, CA 92705, 714/261-0240.

(4) National Expositions Co., Inc., (NEC), 14 West 40th Street, New York, NY 10018, 212/391-9111.

(5) Bunker Ramo, Information Systems Division, Director of Training Services Group, 35 Nutmeg Drive, Trumbull, CT 06609, 203/386-2223.

(6) Structured Methods, Inc., (SMI), 7 West 18th Street, New York, NY 10011, 212/741-7720.

(7) UNIOPS (contact: John Bass or Paul Fronberg), P.O. Box 1456, Cupertino, CA 95015, 408/996-0557 or 408/988-1755.

OCTOBER

October 1-3

"Unix System Internals," New York, N.Y. (See calendar key #6, SMI, for contact details.)

October 1-3

"Unix," Rochester, N.Y. (See calendar key #3, CAPE, for contact details.)

October 1-5

"Advanced Unix," Trumbull, Conn. (See calendar key #5, Bunker Ramo, for contact details.)

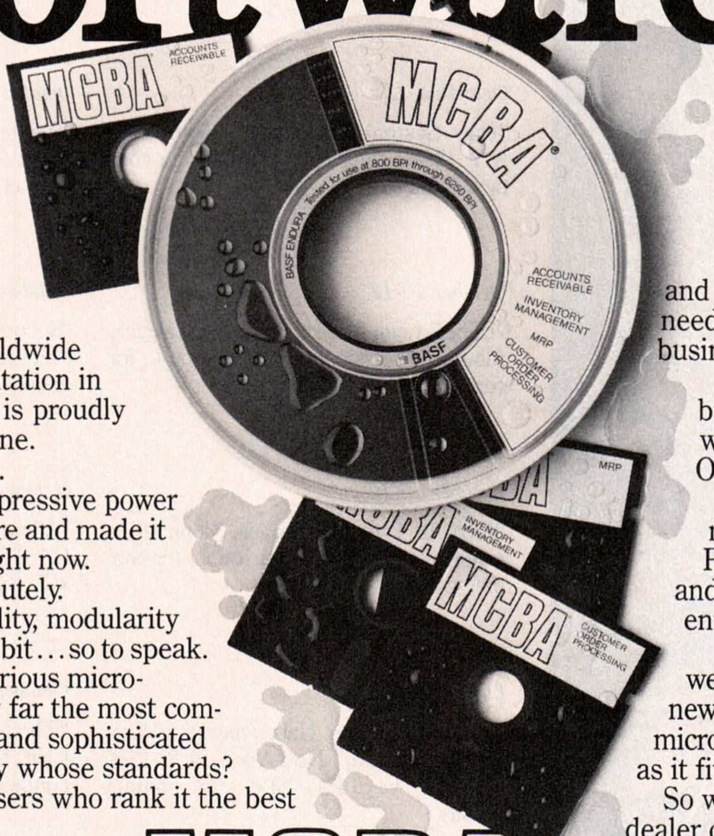
October 2-4

"East Coast Computer Faire," Boston, Mass. (See calendar key #2, CFI, for contact details.)

October 2-5

"Hands-On Unix Workshop," Los Angeles, Calif. (See calendar key #1, ICS, for contact details.)

MCBA introduces shrink-to-fit software.



With nine years in minicomputer software, 15,000 installations worldwide and an established reputation in the mini world, MCBA is proudly shrinking its software line.

Down to micro size.

We've taken the impressive power of minicomputer software and made it available for micros. Right now.

Alter the fit? Absolutely.

Alter the functionality, modularity and capability? Not one bit... so to speak.

This new line of serious microcomputer software is by far the most comprehensive, well-tested and sophisticated in the industry today. By whose standards? Thousands of MCBA users who rank it the best in the business.

MCBA's library of 16 integrated manufacturing, distribution and accounting packages can be installed in whatever combination and sequence a user needs for his or her business.

It grows with businesses. No matter what size they are now. Or want to be later.

And MCBA software now runs in RM/COS,® PC-DOS, UNIX™ and UNIX look-alike environments.

In other words, we've tailor-made our newest software to fit micros—as comfortably as it fits user needs.

So whether you're a dealer or a user, find out about it. Call us now at (818) 957-2900.

Shrink-to-fit software.
For growing businesses.

MCBA®

Minicomputer Software for Micros.

2441 Honolulu Avenue, Montrose, California 91020

Also for DEC, Wang, HP, TI, and Perkin-Elmer minis.

Please circle Ad No. 23 on inquiry card.

October 3-5

"Unix," Calgary, Alberta (See calendar key #3, CAPE, for contact details.)

October 4-5

"Using `lex` and `yacc`," New York, N.Y. (See calendar key #6, SMI, for contact details.)

October 8-12

"Unix System Workshop," New York, N.Y. (See calendar key #6, SMI, for contact details.)

October 10-12

"Unix," Long Island, N.Y. and Calgary, Alberta (See calendar key #3, CAPE, for contact details.)

October 15-19

"Intro to Unix," Trumbull, Conn. (See calendar key #5, Bunker Ramo, for contact details.)

October 16-18

"Unix/Expo: The Unix Operating System Exposition," New York, N.Y. (See calendar key #4, NEC, for contact details.)

October 17-19

"Unix," Raleigh, N.C. (See calendar key #3, CAPE, for contact details.)

October 22-26

"Advanced C," Trumbull, Conn. (See calendar key #5, Bunker Ramo, for contact details.)

October 23-26

"Hands-On Unix Workshop," Boston, Mass. (See calendar key #1, ICS, for contact details.)

October 25-28

"Second PC Faire," San Francisco, Calif. (See calendar key #2, CFI, for contact details.)

October 29-November 2

"Advanced Unix" and "Programming in C," Trumbull, Conn. (See calendar key #5, Bunker Ramo, for contact details.)

October 30

"The Mac as a Workstation," Palo Alto, Calif. (See calendar key #7, UNIOPS for contact details.)

So bye, bye, Ma Bell, why did you die?
We get static from Sprint and echo from MCI,
"Our local calls have us in hock!" we all cry.
Oh Ma Bell why did you have to die?
Ma Bell why did you have to die?

I drove on out to Murray Hill,
To see Bell Labs, some time to kill,
But the sign there said the Labs were gone.
I went back to my old CO,
Where I'd had my phone lines, years ago,
But it was empty, dark, and ever so forlorn...

No relays pulsed,
No data crooned,
No MF tones did play their tunes,
There wasn't a word spoken,
All carrier paths were broken...

And so that's how it all occurred,
Microwave horns just nests for birds,
Everything became so absurd,
The day... Bell System... died.

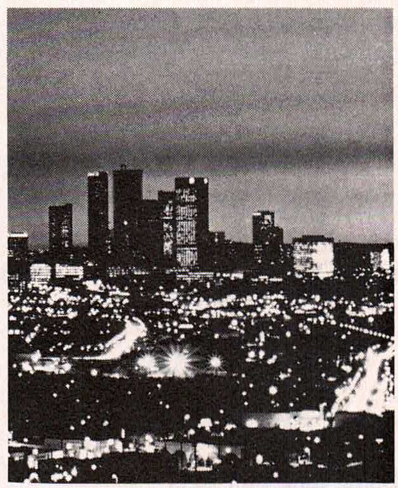
So bye, bye, Ma Bell, why did you die?
We get static from Sprint and echo from MCI,
"Our local calls have us in hock!" we all cry.
Oh Ma Bell why did you have to die?
Ma Bell why did you have to die?

We were singing:

Bye, bye, Ma Bell, why did you die?
We get static from Sprint and echo from MCI,
"Our local calls have us in hock!" we all cry.
Oh Ma Bell why did you have to die?

(The End)

Lauren Weinstein is a computer/telecommunications consultant based in Los Angeles. He has particular expertise in the fields of computer networking (over a decade working with ARPANET and other networks), the Unix system (around ten years), microcomputer technology, and telecommunications systems ranging from dials and ringers to modern satellite systems.



THE DAY BELL SYSTEM DIED

BY LAUREN WEINSTEIN

Greetings. With the massive changes now taking place in the telecommunications industry, we're all being inundated with seemingly endless news items and other information concerning the various short- and long-term effects. The breakup (some prefer to call it *divestiture*) of AT&T and a multitude of related topics (including long-distance service "competition" to name but one) are indeed very serious matters about which I have very strong opinions. I'll be commenting at some length on these subjects (which are beginning to affect us all) in a future column. This month, however, I thought I might take a break from the "conventional" column and try something a bit different.

Instead, I'd like to address a largely neglected aspect of the new "era" of telecommunications—the need for a song! Since the great satiric songwriter Tom Lehrer has retired from the composing world, I have attempted to help fill this void with my own light-hearted, non-serious look at a "possible" future of telecommunications. This work is entirely satirical. None of its lyrics are meant to be interpreted otherwise. A warning: There's a fair amount of telco lingo included, but even if some of the terms might be unfamiliar to some of you, I hope you will find the song amusing.

These lyrics should be sung to the tune of Don McLean's classic *American Pie.* I call my version *The Day Bell System Died. . .*

—Lauren—

UUCP: {decvax, ihnp4, seismo, harpo, allegra}!vortex!lauren

Notice: This is a satirical work

THE DAY BELL SYSTEM DIED

(To the tune of *American Pie*)

(With apologies to Don McLean)

Long, long time ago,
I can still remember,
When the local calls were "free."
And I knew if I paid my bill,
And never wished them any ill,
That the phone company would let me be...

But Uncle Sam said he knew better,
Split 'em up, for all and ever!
We'll foster competition:
It's good capital-ism!

I can't remember if I cried,
When my phone bill first tripled in size.
But something touched me deep inside,
The day... Bell System... died.

And we were singing...

Bye, bye, Ma Bell, why did you die?
We get static from Sprint and echo from MCI,
"Our local calls have us in hock!" we all cry.
Oh Ma Bell why did you have to die?
Ma Bell why did you have to die?

Is your office Step by Step,
Or have you gotten some Crossbar yet?
Everybody used to ask...
Oh, is TSPS coming soon?
IDDD will be a boon!
And, I hope to get a Touch-Tone phone, real soon...

The color phones are really neat,
And direct dialing can't be beat!
My area code is "low":
The prestige way to go!

Oh, they just raised phone booths to a dime!
Well, I suppose it's about time.
I remember how the pay phones chimed,
The day... Bell System... died.

And we were singing...

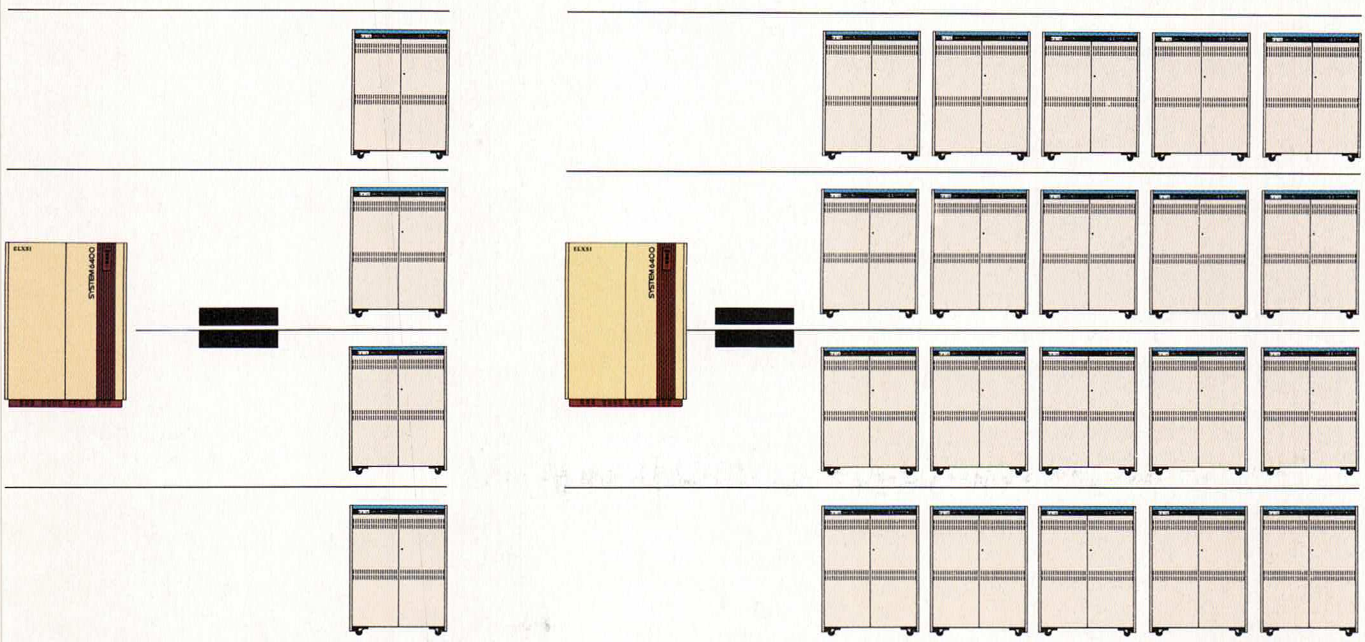
Bye, bye, Ma Bell, why did you die?
We get static from Sprint and echo from MCI,
"Our local calls have us in hock!" we all cry.
Oh Ma Bell why did you have to die?
Ma Bell why did you have to die?

Back then we were all at one rate,
Phone installs didn't cause debate,
About who'd put which wire where...
Installers came right out to you,
No "phone stores" with their ballyhoo,
And 411 was free, seemed very fair!

But FCC wanted it seems,
To let others skim long-distance creams,
No matter 'bout the locals,
They're mostly all just yokels!

And so one day it came to pass,
That the great Bell System did collapse,
In rubble now, we all do mass,
The day... Bell System... died.

ELXSI vs. VAX



NOW OR LATER

Unmatched performance, deliverable now, and potential for expansion. That's ELXSI. The ELXSI System 6400 with one CPU has 4 times the performance of the VAX 11/780. Plus you can add up to 5 CPUs in the initial cabinet, and deliver 20 times VAX performance. 10 CPUs in two cabinets. Now. Or later.

The 64-bit multiprocessor System 6400 offers a choice of either EMBOS, our proprietary message-based operating system, or ENIX, our virtual memory UNIX System V operating system. You can choose either one or both. All configurations feature 32, 64, or 80-bit floating point arithmetic hardware, virtual memory, Ethernet link, the largest physical memory in the business, and more.

The ELXSI System 6400. No one can match our performance or our potential. Not now. Not later. Contact ELXSI today for complete information.

THE ELXSI GUARANTEE:
Find a computer system that can match our performance and our potential for expansion, and we'll raise our price to match theirs.

ELXSI

ELXSI, 2334 Lundy Place, San Jose, California 95131
408/942-1111, Telex 172320.

Please circle Ad No. 9 on inquiry card.

Gould . . . Innovation and Quality in Software

CELEBRATE!

New UNIX™-based software tools bring projects in on time, within budget.

Get your hands on our UNIX tool kit. Take out the Gould Software Division 4.2 UNIX with System V enhancements. You'll know it. It has a fast file system, automated system configuration, interactive debugging, and a Source Code Control System that compresses files for more efficient disk storage.

Then grab The Newcastle Connection, an ingenious tool that lets you create a distributed UNIX file system in a LAN by uniting heterogeneous UNIX versions and hardware environments.

Next, try the Unicellerator™, a high efficiency, user transparent tool that dramatically increases productivity under UNIX by offloading CPU intensive jobs from the host. The Unicellerator is available to *all* users of a system simultaneously.

The kit also contains a Cross Development System for developing and interactively debugging C language programs under UNIX that will run on Motorola® and Intel® microprocessors. The heart of the system is a C language compiler that generates compact, efficient code that reduces code space by 20–40% and increases performance by an equal amount. For programmers who want to develop Macintosh™ applications in C under UNIX, our Macintosh Cross Development System will do it four times faster than Apple®'s Lisa™-based environment.

Our UNIX tools are available individually or as a complete, well-integrated set. The sooner you get your hands on them, the faster you'll finish your development projects. And that's worth a toast any time. Contact Gould Software Division, 1101 East University Avenue, Urbana, Illinois, 61801. (800) 952-8888 or (217) 384-8500.

UNIX is a trademark of Bell Laboratories
Apple is a registered trademark of Apple Computer, Inc.
Motorola is a registered trademark of Motorola, Inc.
Intel is a registered trademark of Intel Corporation
Lisa is a trademark of Apple Computer, Inc.
Macintosh is a trademark licensed to Apple Computer, Inc.

Please circle Ad No. 11 on inquiry card.

GOULD
Electronics