the Yates Perspective

JANUARY/FEBRUARY 1984 VOLUME 2 NUMBER 1

UNIX System V.2 Announced

The UNIX system has come of age. No longer an unsupported, low priority product, **AT&T** Technology's announcements, participation, and concern for customer satisfaction at the recent Uniforum show make the future of UNIX look brighter than ever before. The entrance of **Digital Research** into the UNIX market adds another powerful software company to the UNIX market. **Motorola** appears to have placed UNIX on a high priority. They delivered their System V microport to AT&T in November.

At Uniforum, AT&T Technologies announced System V.2. Organizational changes to AT&T, specifically the creation of AT&T Technologies were described, information on System V.2 was given, and some unbundling of System V.2 was announced. Digital Research announced commitment to UNIX products on Motorola as well as **Intel** chips.

Organizational Shift at AT&T

In June 1983, a part of AT&T Technology Licensing was spun off as **Western Electric's Software Sales and Marketing**, headed by Otis Wilson in Greensboro, North Carolina. The group was chartered with responsibility for licensing the UNIX Operating System, related software, training, and support.

The organization is today committed to a commercial approach to software sales. The AT&T Software Sales and Marketing Organization has put away their old overhead used at UNIX meetings which said "Don't call us –we'll call you."

Otis Wilson comments "We're committed to compatibility of UNIX systems and product support." The company has instituted a toll free hot line for ordering and other information (1-800-828-UNIX). They have streamlined the standard agreements and are committed to preparation of standard agreements, in a one week turnaround, very different from the 6-8 week turnaround of the past.

A recent visit to Greensboro confirms the feeling that AT&T Software Sales and Marketing is growing rapidly. The group has gone from four or five people to over twenty, with more expansion to come. Outside sales people assigned to geographic regions would seem the next step. So far, all salespeople are based in Greensboro. To be effective, AT&T needs an outside salesforce and product managers who are frequently in the field and are located in the geographic areas of their clientele. We shall eagerly watch Greensboro's moves.

Table 1
SYSTEM V AND V.2
PRICE AND POLICIES

	System V.1	System V Release 2.0
Basic Price	$43,000 1st CPU $16,000 ea. add'l CPU	$43,000 1st CPU $16,000 ea. add'l CPU
Upgrade Fee		from V.1, $2,500 no upgrade for pre V licensees
Customer Provisions Fee	$25,000	$25,000 for new purchasers free for existing CP licensees *but terminates old CP supplemental agreement*
Binary License Fees		*Users* / *Fee* 1-2 / $60 1-8 / $125 1-16 / $500 1-32 / $1000 1-64 / $3500 >64 / $7000 Discount 2% for each $100,000 paid to ATT in Royalties up to 60%

Because of the shifts in personnel within AT&T and the large number of new people introduced to the software sales and marketing organization, Table 2 gives some of the names and responsibilities of people within AT&T and Western Electric related to UNIX. This is YATES' VENTURES compilation and may not reflect the latest changes in staffing. However, if you are as confused as we are about who to call for what, this table should be of assistance. **continued on page 2**

UNIX System V.2 continued from page 1

Table 2
PEOPLE TO KNOW AT AT&T TECHNOLOGIES

Jack Scanlon
V. Pres., Computers Systems
Lisle, Ill.
(Overall responsibility for AT&T Technologies emergence as a software and hardware company)

Tom Crowley
V. Pres., Software Systems
Summit, N.J.
(Overall responsibility for development and marketing of UNIX System V and related software)

Dick Shahpazian
Director of Software Sales and Marketing
Summit, N.J.
(Responsible for coordination of AT&T Technologies worldwide software interests)

Otis Wilson
Manager, Software Sales and Marketing
Greensboro, N.C.
(Is primary customer interface, responsible for sale and marketing of UNIX software)
(Organization includes the account team, licensing activities, technical sales support, and marketing promotions)

Dan Lankford
Manager, Software Services
Summit, N.J.
(Product management for third party software and training and documentation products)

Mike DeFazio
Manager, Software Systems Marketing
Summit, N.J.
(Product management for internally developed software)

Tom Scanlan
Manager, Engineering, Systems Support and Customer Service
Lisle, Ill.
(Management responsibilities for UNIX Systems V support center)

AT&T announced its latest enhanced version of System V, System V.2. V.2 is described as an upgrade from V, renamed V.1, and not as a new system. From now on, at least in theory, AT&T's enhancements to System V should look more like the kinds of releases from **Microsoft** or **Unisoft,** and like those companies, will attempt to cause a minimum of disruption for existing users.

The biggest news about System V.2 is not related to features, but to changed AT&T policies for distribution.

Simpler Pricing Policies Instituted

As seen in Table 1, pricing for System V.2 has been simplified. The three tier plan is gone, replaced by a simple discount policy. Qualification for discounts is based on a company's previous year sales or on a commitment to AT&T to sell a certain volume. The volume method requires a minimum quarterly payment. An existing licensee's initial discount is based on the total payment they made under provisions for System III and System V. To reach the maximum 60% discount, a company must pay in about $3 million, representing approximately 8,500 1-16 user licenses (Table 3.)

How do the new customer provisions affect Microsoft and Unisoft? They appear to lessen the price advantage of licenses from Microsoft and Unisoft. But did hardware vendors ever really purchase from Microsoft and Unisoft for the price? Undoubtedly, most companies purchased their porting services for the security of knowing they had a source of support. Superior support, a good microprocessor UNIX, and a general perception by the industry that AT&T is committed to supporting them will sway hardware vendors away from Microsoft and Unisoft. Lower and simpler prices are not enough, although the new customer provisions certainly are an improvement over the old ones.

Table 3
System V.2
UNITS SOLD TO REACH MAXIMUM DISCOUNT

Number of users	Undiscounted fee	Maximum discounted fee	Approximate units sold to reach maximum discount
1-2	$ 60	$ 24	73,000
1-8	125	50	35,000
1-16	500	200	8,500
1-32	1,000	400	4,300
1-64	3,500	1,400	1,225
>64	7,000	2,800	615

Source Code Exchange Policies

A major concern to microcomputer vendors is source code exchange across microprocessors. AT&T has stated that source code licensees may exchange source code only between like software products, meaning source code on like CPUs. For example, a company with both *286* and *68000* based computers could not exchange source code between its two CPU implementations.

Note that AT&T has added a new category: the 1-8 user category as differentiated from the 1-16 category. Before, the two were one group. This is a fair change. There is a clear differentiation between systems with a maximum of 8 and 16 users. The Altos 586, Fortune 32:16, and Tandy System 16 should not be included in the same category as Onyx, Plexus, Zilog and other high performance and more expensive systems.

Table 4
RUN-TIME LIBRARIES
THAT DO NOT INCUR PAYMENT OF A ROYALTY FEE

/lib/libc.a
/lib/libm.a
/lib/libld.a
/usr/lib/libF77.a

SPELL FILES:

/usr/src/cmd/spell/extra
/usr/src/cmd/spell/stop
/usr/src/cmd/spell/british
/usr/src/cmd/spell/list
/usr/src/cmd/spell/hashcheck
/usr/src/cmd/spell/hashmake
/usr/src/cmd/spell/local
/usr/src/cmd/spell/htempl

continued on page 3

UNIX System V.2 continued from page 2

Although the idea's purpose, to enforce standardization, is good in theory, reality may be expensive. Do microcomputer vendors have to redo all source code level enhancements from scratch for each microprocessor? We wonder how this could be accomplished, as very few manufacturers can afford to perform the same enhancement twice. If this means source code files from the tape AT&T provides cannot be exchanged, then enhancements performed to a specific program on the 68000 tape would have to be redeveloped to move the enhancements to the 286 version of the program. Sounds time consuming and redundant to us. Let's hope AT&T clarifies this point soon.

Good News for Application Software Developers

AT&T Technologies has announced that certain run-time libraries of UNIX System V may now be included in application software packages without the payment of a licensing fee. As seen in Table 4, some of the /lib/ files can now be included. In addition, spell files are free of any royalty fee.

Table 5
UNIX SOFTWARE
SYSTEM LICENSES

SOFT-WARE	COMM.	EDU-CAT.[1]	GOV-MNT.	TOTAL
M-UNIX	7	131	0	138
UNIX-V6	91	370	64	525
PWB/ UNIX	48	73	87	208
UNIX-V7	145	416	101	662
UNIX-32V	71	233	37	341
UNIX-SIII	238	49	37	324
UNIX-SV	198	98	9	305
UNIX/ TSS	1	0	0	1
UNIX/ 1100	2	8	4	14
TOTALS	801	1378	339	2518

Source: ATT Technologies, Inc.
[1]INCLUDES ADMINISTRATIVE

These run-time libraries have been used by application developers in the past, but the formal announcement of a policy by AT&T will encourage their inclusion. The cloudiness of this issue has prevented some application developers from jumping whole heartedly into the UNIX marketplace. Additional UNIX files may be added to this policy, if AT&T is to encourage application software development.

A Review of AT&T Technologies Source Code Licensee Data

The Greensboro organization has recently released their latest tallies of source code licensee data, shown in Tables 5-9. Noting that this data is for source code licensing only, several interesting trends become apparent.

Consider Table 5. Source code licenses are counted at AT&T in three categories—commercial, educational, and government. These categories define the purchaser of the license. Commercial includes computer companies, commercial corporations, and organizations who not only purchase a source code license but may purchase binary relicensing privileges. Although some large commercial organizations like **Boeing Aircraft** or **General Electric** may purchase source code licenses for use internally, this group is dominated by computer manufacturers.

The educational source code licenses have always been a significant portion of AT&T total UNIX source code users. Educational licensees include universities and research institutions. Typical examples are the University of California at Berkeley, Massachusetts Institute of Technology, and Duke University.

The government category encompasses government agencies and government research institutions. Growth in source code licensing by government agencies is declining as the market shifts to commercial licensing. Many government institutions are now purchasing binary licenses on products from computer manufacturers rather than buying their own source code based system.

Table 6
UNIX SOFTWARE
INSTALLATIONS

SOFT-WARE	COMM.	EDU-CAT.[1]	GOV-MNT.	TOTAL
M-UNIX	9	419	0	428
UNIX-V6	171	1016	181	1368
PWB/ UNIX	133	309	143	585
UNIX-V7	242	1376	176	1794
UNIX-32V	131	618	64	813
UNIX-SIII	448	80	58	586
UNIX-SV	286	100	9	395
UNIX/ TSS	1	0	0	1
UNIX/ 1100	5	16	4	25
TOTALS	1426	3934	635	5995

Source: ATT Technologies, Inc.
[1]INCLUDES ADMINISTRATIVE

AT&T tracks installation of source code licenses as well as license sales. Installations reflect the additional CPUs licensed to run source code at each site. Educational institutions, typically with many CPUs spread out across several departments, are frequent users of the additional CPU privilege. The educational institutions are encouraged to use the system by extremely low licensing fees.

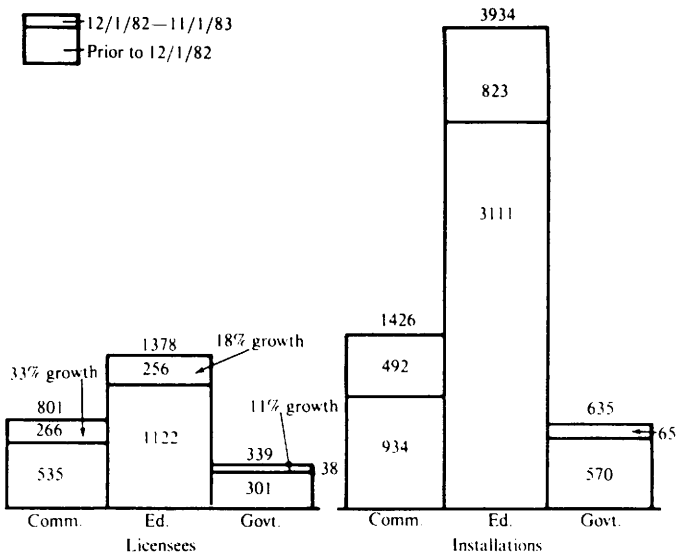
continued on page 4

UNIX System V.2 continued from page 3

Source Code License Growth

Note in Figure 1 the YATES VENTURES analysis of the AT&T data. It is clear that commercial source licensee growth, at 33 percent in the last 12 months, exceeds that of education and government, growing at 18 percent and 11 percent respectively. Installation data is somewhat different, but still reflects a basic trend toward commercial source code licenses as the dominant growth factor in the source code market. We expect to see these licenses continue to rise as more and more manufacturers enter the UNIX market, and educational and government institutions purchase computers with supported binary UNIX rather than purchasing source code UNIX on an unsupported system.

Figure 1
UNIX SOFTWARE
SOURCE CODE LICENSING GROWTH



Source: AT & T Technologies, Inc.

Conclusion

System V.2, with its commercial policies, is an encouraging sign in the UNIX market. AT&T Technologies appears to be addressing many customer needs, and we wait eagerly for additional announcements. AT&T's own source code licensing data indicates an increasing interest by commercial vendors, mostly computer manufacturers and source code licenses. YATES VENTURES' latest database clearly shows that the UNIX enduser has shifted from the government/military or university community running UNIX on a VAX to the software developer, hobbyist enduser, and entrepreneurial high tech user on a microcomputer. This entrepreneurial/leading edge type of user will decline in total percent of the market as the business user begins to purchase $12,000-$25,000 computers running UNIX in 1984. The UNIX market has arrived, and 1984 looks to be a banner year for companies at all price points and market segments.

Table 7
UNIX SOFTWARE SYSTEM LICENSEES
SOURCE CODE INTERNATIONAL

SOFTWARE	COMM.	EDUCAT.[1]	GOVMNT.	TOTAL
M-UNIX	1	47		75
UNIX-V6	23	162	2	187
PWB/UNIX	16	21		37
UNIX-V7	70	204	1	275
UNIX-32V	44	86		130
UNIX-SIII	71	36		107
UNIX-SV	53	43		96
UNIX/TSS				
UNIX/1100		1		1
TOTALS	287	600	3	890

Source: ATT Technologies, Inc.
[1]INCLUDES ADMINISTRATIVE.

International Source Code License Numbers

The international source code license purchasing behavior is indicative of the growing UNIX interest in Japan and Western Europe. Table 7 describes international source code licenses. Of the 53 System V licenses, YATES VENTURES estimates approximately 18 System V licenses in Japan, 10 in the United Kingdom, 8 in Germany, 5 in Scandinavia, 3 in the Netherlands, 2 in Italy, and the remaining 7 are unidentified by YATES VENTURES. Educational international source code licenses tend to center around the major Japanese universities and universities in the United Kingdom and Germany, although this data is only approximate until YATES VENTURES can conduct interviews in Europe, scheduled for the first quarter of 1984.

Table 8
UNIX SOFTWARE SYSTEM INSTALLATIONS
SOURCE CODE INTERNATIONAL

SOFTWARE	COMM.	EDUCAT.[1]	GOVMNT.	TOTAL
M-UNIX	4	72		76
UNIX-V6	38	361	8	407
PWB/UNIX	25	72		97
UNIX-V7	73	436	2	511
UNIX-32V	43	105		148
UNIX-SIII	104	83		187
UNIX-SV	95	158		253
UNIX/TSS				
UNIX/1100		4		4
TOTALS	382	1291	10	1683

Source: ATT Technologies, Inc.
[1]INCLUDES ADMINISTRATIVE.

Sneak Preview

System V Release 2 Overview

by Jean Yates

System V.2 is an upgrade, not a new system. Enduser features and speed have been improved. The new release also incorporates some of the new philosophies of AT&T about its product.

System V.2 reflects the desire on AT&T's part to provide upward compatibility between different processor versions of UNIX. Performance has been improved and performance of AT&T's microprocessor versions is expected to equal anything available from porting houses.

Standards

AT&T has described a set of standards to the /usr/ standards committee, and these standards (Table 1) will allow the industry to develop applications and computers that are compatible. AT&T has also started work on proposed standards for command syntax and error messages. They plan to introduce device naming and other features to promote standardization in the near future.

Table 1
PROPOSED SYNTAX STANDARD
FOR UNIX SYSTEM COMMANDS

RULE 1:	Command names must be between two and nine characters.
RULE 2:	Command names must include lower case letters and digits only.
RULE 3:	Option names must be a single character in length.
RULE 4:	All options must be delimited by "—".
RULE 5:	Options with no arguments may be grouped behind one delimiter.
RULE 6:	The first option argument following an option must be preceded by white space.
RULE 7:	Option arguments cannot be optional.
RULE 8:	Groups of option-arguments following an option must be separated by commas or separated by white space and quoted.
RULE 9:	All options precede operands on the command line.
RULE 10:	"— —"may be used to delimit the end of the options.
RULE 11:	The order of options relative to one another should not matter.
RULE 12:	The order of operands may matter and position-related interpretations should be determined on a command-specific basis.
RULE 13:	"–" preceded and followed by white space should be used only to mean standard input.

Installation for Electronic Address
Add the following entries to the file "/usr/lib/uucp/L.sys"

nwuxd Any ACU 1200 13122601844 in-BREAK-in-BREAK-in unixml word bellmail
NWuxd Any ACU 300 13122601844 in-BREAK-in-BREAK-in unixml word bellmail

Other areas that the System V.2 release starts to address are:

- Standards for object and a.out files,
- Device driver interfaces,
- User interfaces,
- I/O control,
- Networking.

AT&T will utilize BSD features such as vi, large block size, job control, and paging, and in some cases will improve them

Timing and Priorities for Future Releases

AT&T has announced a general policy of a 12-18 month interval between operating system releases. More important than timing is the quality of the offering and the upgradeability. The impact of new releases every 12-18 months in the past was an incredible cost to hardware vendors to keep up with newest releases. Upgrading to a new release must be easier and cannot significantly exceed the cost of upgrading an MS-DOS, concurrent CP/M-86, or other equivalently priced microcomputer operating system.

Unbundling System V.2

At Uniforum, AT&T announced some unbundling of UNIX System V.2. AT&T announced the concept of tool kits: packages of related utilities, unbundled from the system as required and sold separately. AT&T also introduced two new Workbenches. The "Workbench" connotation is applied by AT&T to sets of application programs, versus utilities or programming tools, that may be unbundled as toolkits.

The unbundling concepts announced so far indicate a trend to unbundling UNIX for different markets and different size CPUs. So far, unbundling has centered around related programs and products. The user profile does not appear to have been considered. As seen in Table 2, the UNIX user in the future requires programs that may not appear related to the technical programmer. AT&T must address the needs of specific types of users and the size and types of their computers when determining an unbundling scheme.

Language and Library Changes in System V.2

Enhancements to System V languages and libraries include:

- Arbitrary length variable names in C,
- Changes in archive format for long variable names,
- Addition of debuggers and static analyzers,
- Enhancements to Fortran 77 (F77).

System V.2 allows variable length names to be used in the C language. The changes are compatible with System V.1, although combining V.1 and V.2 files may cause problems as truncated names cannot be recreated. The

continued on page 6

System V Release continued from page 5

format of the V.2 object file has been changed to allow long variable names, and the new format is shown in Figure 2.

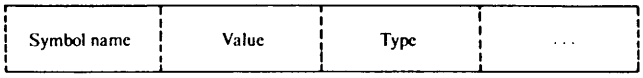
Table 2
UNIX USERS & WHAT THEY DO

Type of use	Major Tasks	UNIX tool	Packaged Application
Word Processing/Text Production	letters, memos, short documents	vi & nroff	sometimes
	reports, papers, books	vi & nroff	word processing packages
	mail merge, forms, typesetting	troff	yes
	system administration	tar, dump, misc. commands	
Bookkeeping and Accounting	accounts payable & receivable, general ledger payroll inventory	They take advantage of UNIX multi-user feature but use few UNIX tools	Always
Administrative and clerical	memos	vi & electronic mail	
	appointment scheduling.	calendar	
	word processing calculations	vi & nroff, file system. dc (desk calculator)	wp
Decision Making	projections and planning		spreadsheet programs
	record keeping	vi & file system	specialized programs
	memos	vi & electronic mail	
Programming	writing and editing programs; debugging, fixing programs, running programs.	ed, ex, sed, Adb, as, awk, ld, many other tools. C, Fortran	cross assemblers
			Pascal, Basic, Cobol

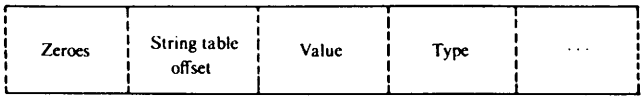
Figure 1 compares the symbol table of V.1 to V.2. The symbol table for V.2 is basically the same as in 1.0 if the name is eight characters or less. Otherwise, the first 8 bytes are different. System V.2 has also added a string table to the object file format. A revised archive file format for V.2 includes provisions for longer file names. The new object code file format for System V.2 is shown in Figure 2.

Figure 1
SYMBOL TABLE ENTRY

System V Release 1.0:



System V Release 2.0:



System V.2 includes an enhanced memory allocation package. It is faster but less space efficient. According to AT&T, it adheres to the proposed/usr/group standard and allows users to access either of the memory allocation packages, old or new.

System V.2 includes new debuggers and static analyzers, including a new tool, *Ctrace*. Ctrace is a C program debugging tool. It follows execution of a C program, statement by statement, printing the text of each statement executed and the values of all variables referenced to modify. Ctrace is practical only for debugging and for relatively small programs at one time since the addition of these statements makes the program much larger. Ctrace is a useful addition to the powerful UNIX program development library.

Enhancements to lint have been added to produce .n files analogous to the .o files produced by cc. Enhancements to prof have also been added to System V.2.

Fortran 77 enhancements include an update of documentation, improvements to the random number generator, and performance improvements at compile and execution time. According to AT&T, F77 in System V.2 is approximately 22 percent improved in sequential formatted read/write.

Improvements in the Shell

The improvements to System V.2 shell include:

- a 25 percent performance improvement;
- command hashing;
- expansion of functionality;
- improvement in quality and maintainability of code.

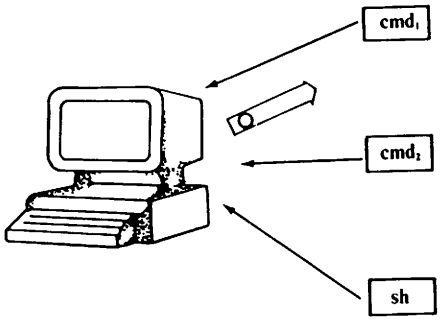
To increase performance, System V.2 has echo and pwd built into it. Directory reads are buffered during file name expansion. New mail is checked for only at user specified intervals, not constantly. Command hashing has been added. With command hashing, when a command is first invoked, the shell locates it by using the normal search mechanism and stores the location as an index item. If the command is typed again, no search is required.

New functionality has been added to the shell. Shell scripts are now stored in a parsed form in the shell database. They execute more quickly, and state and variable changes are visible after execution completes. Several new built in commands have been added to the shell including unset, to remove variables and functions typed to tell how a command will be interpreted when executed, and hash, giving information about the command hashing which has been performed. continued on page 7

System V Release continued from page 6

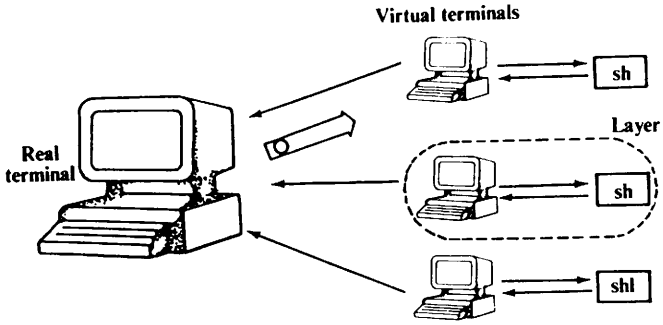
Figure 2

BSD Model



- Only "foreground" job receives input and can modify the terminal state
- stty option control "background" job output
- sh arbitrates access to the terminal
- jobs are signaled when moving between foregound and background

Release 2 Model



- stty option on virtual terminal controls whether its output goes to the real terminal
- programs manipulate virtual terminals which can then be mapped to the real terminal
- only current layer receives input
- utility shl provides control

The shell and its new enhancements are written in pure C, increasing maintainability and ease of customization.

Job Control

Figure 2 shows the BSD 4.1 and System V.2 job control models. With BSD, only the "foreground" job can receive input and modify the terminal state. The stty option controls background job output. Jobs are signaled when moving between foreground and background. With the System V.2 model, the stty option on a virtual terminal controls whether its output goes to the real terminal. Programs manipulate virtual terminals which can then be mapped to the real terminal. Only the current layer receives input, and the utility shl provides control.

Other Enhancements in System V.2

Other enhancements to System V.2 include a library of screen handling and optimization routines which support development of terminal independent programs. This library is an enhancement to the original BSD curses, including support of several new commands, more than one kind of highlight, and multiple terminal handling capabilities.

Termcap has been expanded to terminfo. Support for a larger class of terminals and cleaner syntax has been incorporated into this enhancement.

The cron facility has been improved in both structure and administration and is better set up for commercial use. Mail has been enhanced with a new mailx alternative message process interface. Mail still exists as a command, but mailx provides commands to facilitate saving, deleting, and responding to messages and includes folders and message specifications. It supports editing and formatting of outgoing mail and is a derivative of the BSD mail system.

New or improved commands include PG, a CRT file perusal filter (derivative of the BSD more program). Password has been enhanced to be more secure.

Several enhancements have been made to the V.2 accounting package, making it up to 25 times faster. A file's disk blocks are now charged to the file's owner, not the user whose log in the directory hierarchy that contains the file. Allowance for prime time and nonprime time accounting is included in System V.2.

Good Dirt

The Seiko/SMC Connection . . Commodore Selects COHERENT . . Pyramid Spans the Gap

Another company jumped on the UNIX bandwagon. **Plexus** signed a three-year, $40 million contract with **Philips Information Systems**. Under the agreement, Philips will purchase the *Plexus P/35* and *P/60* 32 bit computers. The machines will be used as shared resource and network processors in future office automation products. The computers may also be marketed as part of turnkey vertical market packages. The machines run UNIX version III, support one to 40 users, and range in price from $15,000-$80,000.

Onyx lowered the price of its Professional Service Program (PSP) by 25 percent and increased the margin on PSP contracts it pays to dealers. The service is provided by **RCA Data Services**. Price varies according to system purchased; it usually starts at around $800.

Speaking of servicing: **Honeywell** will be providing service for **Coleco's** *Adam* home computer system. Unfortunately, Coleco wasn't able to get enough units on retailers' shelves for the crucial Christmas season.

Xerox gets help: **The Genra Group** will take control of Xerox's 54 U.S. retail outlets. Genra is a Dallas-based company with a *strong* management team. For example, Genra Chairman Joseph Verdesca was the founder of Computer Roomers, a leading manufacturer of retail computer store furniture. Xerox isn't really giving up on direct retailing, as it will quietly take an equity position in Genra. In addition, Xerox will supply the company with "management talent," and will continue to finance customer purchases through Xerox Credit Corporation.

Look out for Seiko: **SMC** and **Intec** have joined forces to distribute Seiko computer products in the United States. **Seiko** appears committed to a UNIX-type operating system, as they ran XENIX on their *8610* box at COMDEX. Rumor has it, however, that the American side of the company is not convinced of the value of UNIX in raising the marketability of their products.

In any case Seiko's 8610 box will compete against the **Altos** *586*. The 8610 may capture a significant portion of this market segment due to its low price (under $8000) and its excellent construction and quality design. The Seiko/SMC connection is a potentially strong one because of SMC's library of application packages. These applications are written in *SMC Basic*, and could quickly be converted to run on XENIX.

Microsoft decided to set up a training facility: it chose **KNOW HOW, Inc.** to do the job. The San Francisco based company is backed by the Pacific Technology Venture Fund and Prentice-Hall. KNOW HOW Vice President Lawrence Magid has a Ph.D. in Education and writes a syndicated computer column for the *Los Angeles Times*. The company will train endusers on *MS-DOS*, *Multiplan*, Microsoft *Word*, and the *Microsoft Mouse*. XENIX isn't included.

Sun moves into vertical markets: **Compugraphic** signed up to buy Sun workstations. Compugraphic is the world's largest supplier of typesetting and graphics communications systems. The *SunStations* integrate UNIX with networking, graphics, and an advanced user interface. Sun is selling its workstations in many markets. Last summer (1983) it signed **Computervision** to supply the CAD industry. This vertical market is one of the places for UNIX suppliers to be.

Software wanted: **Callan Data Systems** will now give software developers a 35 percent discount off the retail price of its *Unistar* microcomputers. In return, developers must grant marketing rights to Callan, who will offer the software to its customer base. The company hopes to get vertical market applications which will run under UNIX System V on its multiuser Unistar systems. Sounds like a good deal for all parties.

For those who value such information: Callan Unistar workstations were sold to Ford Aerospace, for use at the Johnson Space Center in Houston. The computers will monitor and control mission objectives of the space shuttle. We see a trend emerging—UNIX in space.

Commodore chose *COHERENT* as the operating system for its next generation of computers. **Mark Williams Co.** agreed to port COHERENT to a *Z-8000* based computer. The new Commodore system is scheduled for shipment by the third quarter of 1984.

Commodore signed agreements with **Zilog** which permit Commodore to manufacture its own Z-8000 chips. The company also manufactures its own disc drives through a joint venture with **Mitsumi** of Japan. Similar production arrangements for other systems allowed Commodore to post recent profits of 12 to 15 percent, while competitors posted (e.g., Mattel and Texas Instruments) substantial losses.

We anticipate a *significant* impact on the multiuser market due to Commodore's mass production strategy. The best-selling *Commodore 64* reportedly ships at a rate of more than 100,000 units per month, with gusts of up to 150,000 during the Christmas season. While Commodore surely does not expect as high a volume for its multiuser models, its vertical integration will allow Commodore products to be manufactured and sold at prices which are considerably lower than competing products.

Addendum: A number of factors influenced Commodore's decision to choose COHERENT. COHERENT was written specifically to run on small systems. According to Mark Williams President Bob Schwartz, "COHERENT has better code, more secure files, and is more user friendly than XENIX. And COHERENT comes with all of the UNIX utilities." He forgot to mention the price. Since COHERENT is not a licensed version of UNIX, royalties need not be paid to Western Electric. Lower costs mean lower retail prices. **continued on page 9**

Good Dirt continued from page 8

November was a big month for **Relational Technology** (RTI) of Berkeley, California. The company signed OEM pacts with **Callan Data Systems, Computer Consoles** (CCI), **Codata, Dual Systems** (Berkeley), **InDaSys** (San Rafael, California) and **Sequoia Systems**. These companies agreed to distribute RTI's relational database systems: *INGRES* (the original minicomputer version) and/or *MicroINGRES*.

We got a look at a new word processing system developed by **Leading Edge Products**. The program we saw at COMDEX ran on the Leading Edge 8088 based **IBM** *PC* look-alike machine with a color screen. It includes several modes which allow for recall of deleted letters or blocks, and several document reformatting functions that can be used while inside the document. The software was written in part by J.B. Royal, one of the key authors of **Wang's** excellent *Wangwriter* word processing program. Conclusion: word processing on micros is rapidly reaching a higher level of sophistication.

Also from COMDEX: **Pyramid Technology** announced that its *Pyramid 90x* microcomputer is now set up to run *both* UNIX 4.2 BSD and System V UNIX *at the same time*. According to Pyramid Marketing Director Barbara Kline, "The user can dynamically select which configuration to use at any given point in time." If this system works as advertised, it might begin to span the gap between business and scientific markets for UNIX.

Eagle goes abroad: **Thomson-CSF 'Communications'** will market Eagle's 16 bit IBM compatible microcomputers in France. This thrust into the PABX/telecommunications vertical market can only be seen as beneficial. In addition, Eagle recently added networking capability to its product line. **Nestar** of Palo Alto, California, will provide its *PLAN Series* hardware and software as the basis for Eagle's first networking product. *EagleNet I* can connect the *Eagle PC*, the *Spirit*, and the *600*.

Angry **Apple Computer, Inc.** shareholders barred from attending the annual meeting due to lack of meeting space, circulated a petition outside in hopes of invalidating the meeting with the SEC. Press people from ABC to The Wall Street Journal were also barred from attending the meeting, with all the seats filled on a first-come, first-served basis. Apple had announced it would introduce the new Macintosh at this meeting. Approximately 500 people were observed futilely trying to get into the shareholders meeting to vote. Even Bill Gates, Chairman of the Board for Microsoft, Inc., a major contributor to the development of the newly released Macintosh was seen having difficulty entering the Apple meeting. Those shareholders left outside were forced to leave their proxy cards with Apple representatives, or wait to view video tapes of the meeting at an as yet undisclosed place and time. Introducing the Macintosh on the same day as the annual shareholders meeting caused more excitement than Apple anticipated. This excitement has now backfired, with angry stockholders and others feeling slighted at not being able to attend.

People In The News

Ray Moffa, formerly a *VAX* production manager at DEC, recently took over the presidency of **Pixel Corp**. He won't have to worry about cash flow problems, due to a recent successful public stock offering. Moffa is planning a 3-5 year program to position Pixel in the 8-10 multiuser market. Pixel will de-emphasize technological improvements in favor of conforming to the emerging industry standards. The company claims that new Pixel boxes will be "perfect environments for application software developers."

Kathryn Gould was named vice president of marketing for **Oracle Corp**. Gould was formerly a consultant with the technology management practice staff of Arthur D. Little, Inc., and has held several marketing positions with Data Systems Design, Gould, and Bell & Howell.

Al Sisto transfers from Intel Corp. to **Relational Technology, Inc.** (RTI). Sisto will leave his job as Intel's marketing manager of database operations to become RTI's new vice president of sales. RTI also snagged **Peter Tierney**, a 16-year IBM veteran. Tierney joins RTI as vice president of marketing.

Altos appointed three regional sales managers to oversee the company's OEMs and Fortune 500 enduser sales effort. **Gary Gentges** leaves Interel Data Communications to oversee Altos' midwestern region from its Chicago area office. **Arthur Thibodeau** heads up the new Boston area office in Altos' New England territory. He previously worked for Beehive International. **Walt Vanator** moves into the new metropolitan New York regional office. He was formerly a vice president with MTI Systems.

Steven Shedivy and **Gary Hinrichs** were hired at **Open Systems** in an effort to expand its Marketing Service Group. Shedivy will become marketing services manager, and Hinrichs will serve as marketing communications representative.

Also expanding marketing: **Lantech Systems**, the maker of uNETix, creates two new positions. **John Saba** leaves his regional sales manager at the consulting firm of Tres Computers Systems to become Lantech's national sales manager. **Martin Schelling** hires on as Lantech's new regional sales manager. He leaves Fortune Systems.

From the Lab

A UNIX Benchmark Blitzkrieg

By Peter Marvit

Once Upon A Time . . .

Benchmarking computer systems seems to be a popular pastime of many vendors and users. UNIX based systems, especially, tend to invite performance comparisons. Unfortunately, results published by computer manufacturers highlight their own machines (often using programs designed to cast favorable spotlights) and results obtained by users are rarely published. This article attempts to provide seller and purchaser alike a starting point for comparing performance of a broad range of computers running UNIX.

You will find a number of different benchmarks (written in C or using UNIX utilities) which try to measure different aspects of a computer system's performance. Tables show the results of actual runs and compare both computer (CPU) time and execution (real) time. You will also see a complete listing of machines used for this blitzkrieg so you can compare the actual configurations.

What is the purpose of this madness? The benchmarks, results and format are offered as a beginning approach to measuring different UNIX based computers. Since we have no affiliation with any particular manufacturer, you might presume the results are unbiased. However, we present a number of issues which you should keep in mind while looking at the tables. These benchmarks are but a part of our ongoing effort in this area. We hope to stimulate general interest in benchmarking and develop a set of useful benchmarks for various classes of machines.

But First . . .

Before delving into the meat of the subject, you should be aware of several caveats. You should first shed your rose colored glasses and don your critical spectacles by reading "Benchmark Confessions" (see page 15) or other philosophies of benchmarking. No benchmark report, including this one, ever tells the entire story of either computer performance or total capabilities.

The computers included in the tables range from **IBM** *PC* with UNIX systems shoehorned into it to the mighty **Hewlett-Packard** *9000 Model 500.* Each computer involved in this benchmark effort has its own market niche, strengths, weaknesses, and (especially) price point. Readers should note relative price as well as relative performance.

The benchmarks themselves constitute an initial attempt at using a set of programs which would explore many aspects of a system's performance, yet could run on a wide variety of configurations. *The benchmarks therefore tend to favor the measurement of the smaller machines.* Most of the machines tested top off at 8 to 16 users, for example. A much different, or at least more comprehensive, set of benchmarks should be used to exercise the computers which can support 30 or more users. The subject of measuring multiuser computers under typical loads is one we hope to develop in the near future. At the moment, however, we present the results at face value.

Likewise, the intended uses of each computer are not really taken into account in this study. For example, an engineering workstation which is part of a network requires a much different performance profile than a multiuser system used for word processing, data entry and other business applications. For this series, due to time and other limitations, we decided to use programs which were part of the public domain. A short history accompanies the description of each benchmark below. We felt that we could not re-invent the wheel and still produce a good set of valid measurements without much more investigation and data. In fact, we will use the results from this battery as a basis for refining our own benchmarking standards.

The benchmarks were performed during November and December of 1983. By the time you read this article, many (if not all) vendors will have introduced new hardware and/or software revisions—including **Pixel, Sritek,** and **VenturCom,** amongst others.

How We Did It . . .

All benchmarks were run four times using the UNIX "time" utility. "Time" reputes to be accurate within one tenth of a second for elapsed computer time and one second for clock time. Please note the error factor in the results. Some people have questioned even *that* accuracy, and for good reason. However, since it was infinitely more reliable than the bell and stopwatch method, "time" served its purpose.

In UNIX, "time" has several components: user, system, real. User time is the duration the CPU actually processes information using non-privileged instructions. Sorts, mathematical manipulation and other CPU-intensive activites consume a high amount of user time. System time includes all operating system calls. Disk access, starting and stopping jobs and writing to a terminal all take a large amount of system time. Real time can be measured by a wall clock. It is the actual amount of time a program takes to run while you sit at your terminal twiddling your thumbs. Real time includes all the moments when you are "swapped out" and the computer is working on someone else's task.

In the tables, we report computer time (user + system) and real time. We chose this method to reduce the size of the matrices and to make the results more meaningful to a general audience. Most managers and users primarily care about the real time component anyway.

continued on page 11

Blitzkrieg continued from page 10

Why These Benchmarks?

We wanted a set of benchmarks which were short, concise and yet thorough. We opted for benchmarks which could encompass a broad range of machines, ignoring the special considerations of particular machine sizes.

The benchmarks selected for this battery excercise three separate areas: Compiler, multiuser capabilities, and basic system speed. As an aggregate, they represent many of the possible uses of a particular computer system. Some benchmarks are comprehensible only to people with a heavy technical background. We have attempted to produce performance profiles which can be understood by both managers and hackers.

The benchmark programs are written in C since all UNIX systems support that language. In addition, you should be able to run the programs in different environments (providing you have a C compiler) to produce new results. The multiuser benchmarks use utilities very specific to UNIX. In some of the UNIX-like operating systems (e.g. QNX on the IBM *PC*) we had to slightly alter the syntax of the commands. We would love to see a generalized C program which tests a multitasking enviroment. continued on page 12

Table 1
SYSTEM CONFIGURATIONS

Manufacturer	Model	Price as Configured	Processor & Speed	Operating System	Disk	Memory	I/O Ports	C Compiler
Altos	586	$14,985	8086 @ 4.7 MHz	Xenix 2.3 (UNIX V7)	40MB	512K	6	Microsoft
Apple (Uniplus+)	Lisa	$ 8,685	68000 @ 5 MHz*	Uniplus+ (System III)	20MB (Corvus)	1MB	2	Unisoft
Apple (XENIX)	LISA	$ 7,385	68000 @ 5 MHz	Xenix 3.0	5MB	1MB	2	Microsoft
Callan	Unistar 200	$15,150	68000 @ 8 MHz	Uniplus (UNIX V7)	21MB	1MB	4	Unisoft
Codata	3300/84	$19,415	+ floating print board 68000 @ 8 MHz	Unisis (UNIX V7)	84MB	1.25MB	10	Unisoft
Corvus	Uniplex	$ 9,000	68000 @ 8 MHz	Uniplus (UNIX Sys. III)	20MB (external)	512K	8	Unisoft
DEC Professional (VENIX)	350	$ 9,020	fll + floating chip	VENIX	10MB	512K	1	VenturCom
Dual Systems	83/80	$21,000	68000 @ 10MHz	Uniplus+ (UNIX V7)	84 MS (SMD)	1 MB	4	Unisoft
Fortune	XP30	$14,900	68000 @ 6 MHz	Fortune Op. System FOS (UNIX V7) 1.7	30MB	1MB	5	PCC
Hewlett-Packard	9000/220	$32,595	68000 @ 12.5 MHz	HP-UX UNIX System III	65MB	1MB	4	Hewlett-Packard
HP	9000/500	$59,650	Proprietary 32-bit @ 18 MHz	HP-UX	65MB (external)	1.5MB	8	Hewlett-Packard
IBM (QNX)	XT	$7,575	8088 + 8087 @ 4.7 MHz	QNX	10MB	512K	8	Quantum
IBM (VENIX)	XT	$7,390	8088 + 8087 @ 4.7 MHz	VENIX	10MB	256K	2	Johnson
IBM (Sritek/Xenix)	XT	$10,610	8088 @ 4.7 MHz + (Stritek) 68000 @ 10 MHz	XENIX 2.3	10MB	640K	2	PCC
IBM (XENIX)	XT	$7,605	8088 @ 4 MHz	Xenix 3.0	10MB	640K	8	Microsoft
Intel	286/380	$31,860	286 @ 5 MHz	XENIX 2.3	35MB	512K	13	Microsoft
Onyx	C8002M		Z8000 @ 4MHz	Onix 3.0.1 (UNIX Sys. III)	20MB	512K	8	PCC
NCR	Tower 632	$13,295	68000 @ 12.5 MHz	Enhanced Uniplus+ (V7)	32MB	1MB	8	Unisoft
Plexus	P35	$19,950	68000 @ 12.5 MHz	System 3	22MB	512K	8	MIT
Pixel	80	$17,444	68000 @ 10 MHz	Enhanced Uniplus+ (V7)	40MB	1MB	4	Unisoft
WICAT	155	$23,000	68000 @ 8 MHz	Uniplus (V7)	30MB	1MB	6	WICAT
Zilog	8000/31	$36,900	Z8000 @ 5.5 MHz	Zeus 3.2 (UNIX V7)	84MB	1MB	8	Zilog (Enhanced PCC)

Blitzkrieg continued from page 11

The Results

The accompanying table provides that which everyone wants — numbers. Although this article is already sprinkled with caveats, you should still remember that the table represents a snapshot in time on particular boxes. There is no magic single number upon which to hang your hat and cry, "This is the best."

Compiles

In a development environment, compile speed becomes a critical bottleneck. Although you choose a compiler on factors other than speed (documentation, completeness, error messeges, et al.), the time taken during the necessary and often tedious task of compiling, fixing bugs, and then recompiling should be as short as possible. Compiler efficiency is measured by running an optimizer (if available) on the final code. You should note the comparative sizes (measured in bytes) of the programs as well as execution speed. In most cases, we took advantage of only the first level of optimization. Some vendors, such as **Zilog,** supply multiple levels for different applications which can produce dramatic differences in execution speed.

The two programs used are the old and venerable hello.c and equally history-laden whetstone.c. The First does nothing more than type "hello" when run. However, the C statement "printf" tends to drag a large number of systems calls with it and so this seemingly simple program can emerge quite ungainly from a compiler.

The whetstone benchmark is an example of a "synthetic benchmark." As you can see from the listing, various routines in the program are performed a prespecified number of times. This program theoretically simulated an actual computational mix of scientific processing tasks. The program is interesting from historical perspective (published in 1976 but written many years earlier) and is a good example of an attempted synthetic job. The actual run times are greatly improved if the computer system has a separate numerical processor for floating point operations, so be aware of system configurations when comparing the results of this highly CPU intensive program.

Multiuser

The multiuser benchmarks consist of various combinations of different modules. The fundamental portions are the files "disk.proc", "cpu.proc" and "term.proc". Respectively, they are disk intensive, CPU intensive and terminal I/O intensive - the three possible areas of system use. Not all machines will perform equally well in the three sets of tests; a compute-bound environment (i.e. scientific processing) will not need great terminal I/O performance while a disk-bound environment (i.e database maniuplation) doesn't require CPU performance (at least not as much).

This set of benchnmarks is not straightforward to run. To simplify the explanation, the entire text of the procedure and appropriate files is reproduced. The various combinations simulate how a system performs running a single task, under a load and under a larger load. The results of these benchmarks are good indications of how the system will respond in a real-life environment with several people pounding away on data entry while others are trying to recalculate a spreadsheet.

General System

The next set of programs are variations on a theme of CPU efficiency. As you can see, they are exactly the same addition loops with different data types used as variables. These programs have circulated on the UNIX networks for a long time and were first performed in 1982. Note that differences between hardware architectures show up most prominently in these tests. For example, a computer which actually uses memory to implement register longs will perform slower than another which uses the hardware registers, even though it might really be a much faster machine.

Table 2
SIEVE RESULTS

	Comp. Time	Real Time
Altos 586	4.4	4.5
Apple Lisa (Uniplus+)	8.2	8.4
Apple Lisa (XENIX)	8.1	8.0
Callan Unistar 200	5.4	6.0
Codata 3300/84	5.4	5.3
Corvus Uniplex	7.2	7.6
DEC Professional 350 (VENIX)	10.5	10.5
Dual 80	7.7	8.0
Fortune XP:30	7.6	7.8
HP 9000/220	4.3	4.4
HP 1000/500	2.5	2.7
IBM PC (QNX)	—	13.5
IBM XT (VENIX)	11.8	12.8
IBM XT	5.2	5.5
IBM XT (XENIX) (Sritek/Xenix)	11.5	11.3
Intel 286/380	4.6	5.0
Onyx C8002M	5.9	6.0
NCR Tower	4.0	4.0
Plexus P35	3.3	3.4
Pixel 80	4.3	4.3
WICAT 155	7.6	8.0
Zilog 8000/31	2.4	2.5

Pipe.c attempts to measure the efficiency of pipes in UNIX. Pipes are used to communicate from one process to another and are a frequent trouble spot. The program spawns a child process (so the two are running concurrently) and then writes a large file through a pipe. The time taken by this program is dependent on disk access time, CUP efficiency and compiler implementation.

Finally, the *Sieve of Erathosthenes* (Table 2) is included for historical reasons. This CPU intensive program has won the hearts of many readers and continues to be one of the most popular "benchmarks." Unfortunately, its results have been overemphasized and, as with most narrow focus benchmarks, tell only a small tale in the larger story of system performance. Yet the "Sieve" has become a standard of sorts and many publications have reported the results for numerous machines.

continued on page 13

Blitzkrieg continued from page 12

And So . . .

We hope to stimulate you into thinking about developing valid objective comparisions of computer system performance. We present a compendium of pre-existing benchmarks which singly are incomplete but taken together form a reasonable measure of a total system. Remember that benchmarks should be only a part of the entire evaluation/comparison process. The ultimate use of any particular machine will dictate its requirements.

For us, this blitzkrieg begins our ongoing program of benchmarking. As mentioned before, we hope to develop a series of tests which are geared to specific classes of machines. In particular, the problems and promise of synthetic benchmarks to simulate different real-world environments tempt us for further investigation.

Although this article includes tests for UNIX systems only other computer applications lend themselves to objective measurement. Word procssors, database management systems, scientific processors and network software need further study. Perhaps you, gentle reader, might take up the task and embark into the perilous world of benchmarking?

References

1. *Hagen, T. and Tenenbau, A. "Two programs, many Unix systems." unpublished.*
2. *Grappel R.D, Hemeway J.E., "A Tale of Four Microprocessors: Benchmarks Quantify Performance", EDN April 1, 1981, pg 179-185*
3. *Gilbreath, J. "A High-Level Language Benchmark," BYTE, September 1981, page 180-198.*
4. *Curnow H.J. and Wichman B.A., "A Synthetic Benchmark", Computer Journal, Vol 19 Number 1, February 1976.*
5. *Benwell N., BENCHMARKING, J. Wiley & Sons, Toronto (1975*
6. *Gilbreath J. and Gilbreath G. "Erathosthenes Revisited", BYTE Vol 8 Number 1, 1983, pg 283-326.*
7. *InfoPro systems (publisher), UNIQUE: Your Independent UNIX and C Advisor (newsletter), East Hanover NJ, Vol 2 #10, June 1983, pg 3-8*

Benchmark Blitzkrieg: Postscript

After completing this set of benchmarks, we have produced a large body of numbers and much source of controversy. When this project started, we didn't realize the full scope of the ardous and time-consuming task we so innocently set out to accomplish; ultimately we made the self-evident discovery that benchmarking is non-trivial. Yet, in spite of all the problems and possible complaints, results were produced.

In retrospect, we might have done things quite differently. Certainly astute readers could criticize many aspects of the benchmarks and methodology. Although our stated goal was to compare complete UNIX systems, it is really unfair to compare 8088-based IBM *PC* to a 32-bit *VAX* costing several magnitudes more. The series itself is biased toward CPU bound jobs. Each computer's target market and thus potential use profile is basically subjugated to a standard profile. The list can go on *ad nauseum.*

Keeping all the caveats and weaknesses in mind, the numbers do validly reveal performance differences. When you look at the tables of benchmark times, you should constantly refer to the system configuration table. Particularly note the price, processor and speed, and disk capacity of each system.

In general, compiler results measure compiler implementation primarily and hardware performance secondarily. The compile results for the hello.c program showed a moderate range of times and wide range of object sizes. Surprisingly, most compilers saved only 10 to 20 bytes while often taking much longer to run! You should note here that QNX on the IBM *PC* has no computer time component; that operating system measured only real (elapsed) time.

The whetstone compiles similarly demonstrated a modest savings in program size using the compiler's optimizer. For both compiles, we probably should have used a program which lends itself to optimization (although we're not sure how to construct one fairly and simply). However, especially in the whetstone compiles, the differences in times (both real and computer) between optimized and unoptimized compiles highlights the compilers, "overhead".

Floating point hardware seemed to give a tremendous increase in the whetstone run times. You can also see that processor cock speed affects this CPU bound program. Optimizing the program had little impact on the run times. Originally developed to model a scientific application's instruction mix, the whetstone benchmark can be used to compare numeric processing capabilities.

The multiuser series is perhaps of most general interest. What you should look for is not so much the absolute times (although that's important, too), but how steep the time slope becomes as loads increase. That is, notice the relative increase of real time as additional tasks run in the background. As you can see, the computer time portion remains fairly constant across any three related tests. Although publication deadlines prevented us from including graphs, ambitious readers could draw line charts which would graphically illustrate where and how the various systems "bend" under multiple loads.

We should note that the performance of machines in this series seems to be most influenced by disk speed and CPU time-slicing (i.e. the amount of time the processor works on any one task). Disk access time is a hardware limitation; for example, Winchester disks are slower than SMD-type drives. The time-sharing environment, on the other hand, can be software tuned by the vendor for maximum efficiency in a particular application. For example, a system used extensively for data entry will need a small time-slice so screen display refreshing is not delayed. A development environment would need larger time-slices so compiles (essentially CPU-bound jobs) occupy the maximum amount of the processor's attention.

continued on page 14

Blitzkrieg continued from page 13

The UNIX operating system in its current iteration is by nature disk bound. Thus the results of the disk intensive tests indicate possible fundamental performance weaknesses in a computer system. However, since the test is strictly a disk read and most applications have a wide mix of disk activities, you should hesitate before generalizing the results. It is interesting to note that the magnitude of variation between systems tested here is generally small compared with the CPU or terminal tests.

The CPU test had a wide spread of numbers. A surprise was the **HP** *9000/220* which ran very fast under a single task but experienced significant degradation under multiple loads. Other systems did well with a single task and single load but "bent" severely under a double load. This benchmarks could be used to model a heavy development environment or one in which major computational jobs need to run concurrently with data entry or other screen oriented software.

The terminal tests seemed to be favorably influenced by intelligent I/O controllers, proprietary workstations, and hardware terminal "handshaking." Perhaps the most frustrating experience with computers in waiting for a display to output on a terminal. Improvements in this area can often ameliorate deficiencies in other parts of a computer system and will enable more users to be added to the system without significant performance degradation.

As you might expect, processor speed affected the CPU loop series the most. When looking at the results, please note the processor's architecture. For example, QNX and the IBM *PC* emulates the 32-bit "long" words in software and so runs extremely slowly. The loops with register variables indicate how system performance can be improved by taking advantage of certain hardware strengths. These benchmarks are favorites as "quick and dirty" tests of processing power, although they are better used as software efficiency design aids.

Piping, a fundamental UNIX concept, allows one process or program to send information to another using transfer mechanisms built into the operating systems. Most vendors have done a good job of implementing pipes, although you will notice some glaring exceptions in the table. The program itself uses both disk and CPU time and so measures several bottleneck areas.

Finally, you will see the numbers produced by the infamous *Sieve of Eratosthenes.* Since a great deal of discussion (and controversy) has preceded this attempt, we won't belabor the intricacies of the program. Worthy of note, however, is the fact that we did not use register variables as some people have suggested and implemented. Because of the historical context of the program, we wanted to keep it as general as possible. Although different approaches and tricks would certainly improve performance, continuity and validity of comparison would be lost.

We hope you will carefully draw your own conclusions from the results published herein. In general, we have avoided taking a stand on which computer is "the best" since none showed a clear performance advantage across the board of tests. In addition, a careful analysis of computer performance for any one application is obviously beyond the scope of this article. Thus, we present the results at face value with some guidelines for interpretation. We welcome suggestions for improvements in the blitzkrieg and hope to spark renewed interest (and skepticism!) in benchmarking.

(The complete set of benchmarks is available to all multiclient subscribers.)

Appendix 1

PROGRAM FOR SIEVE OF ERATOSTHENES

```
#define true 1
#defie false 0
#define size 8190

char flags[size+1];
main() {
       /* ADDED REGISTER DECLARATIONS */
        int i,k,prime,count,iter;
       for (iter = 1;iter <= 10;iter++)
              {
              count = 0;
              for (i=0;i <= size; i++)
                     flags[i] = true;
              for (i=0; i <= size; i++)
                     {
                     if(flags[i])
                            {
                            prime = i + i + 3;
                            for (k = i+prime; k <= size; k+ =prime)
                                   flags[k] = false;
                            count++;
                            }
                     }
       }
```

Benchmark Confessions

by Peter Marvit and Mohandas Nair

Everyone has opinions on three subjects—sex, religion, and benchmarks. While the first two are influenced by cultural and personal tastes, benchmarks often masquerade as objective, cloaked in scientific methodology and absolute numbers. Figures don't lie, but liars figure, as the saying goes, and benchmarks are prime targets for selective interpretation and general confusion.

This article presents our philosophy of benchmarking. Our discussion focuses on complete computer systems, but the principles involved are general and can be extended to other systems. We have not tried to encompass the entire field of benchmark methodology, but rather to provide guidelines and considerations for all who read and perform benchmarks—computer users, salesmen and designers alike. Benchmarking is, at best, problematic, and at worst, a gross distortion of reality, but you can make the best of an impossible situation if you enter the fray armed with specific directions.

We look at the role of benchmarks. What are they? Who uses them? How are they used? Understanding the background and intentions of benchmarks is key to interpreting the results. How those results are reported is crucial. Complete information must be given if those results are to be meaningful. We recommend a point-by-point list of what may be included in a good benchmark write-up. The delicate issue of benchmark design and validity requires a great deal of thought. We skim many parts of the problem and point toward several avenues of investigation.

Why are benchmarks the way they are? What makes a good benchmark report? These questions don't have definitive answers and, as textbooks say, are left as exercises to the reader!

The Role of Benchmarks

A benchmark is an objective, reproducible measure of performance (e.g. execution speed comparisons, object size or device interrupt latency measures). It assists us in placing systems within a continuum, be it a list of computer times measuring I/O performance, CPU performance, etc. Thus, from an individual standpoint benchmarks are a means of comparing one system to other ones. Benchmarks form a strong feedback mechanism to manufacturers and software shops, allowing them to gauge where their creations fall in the marketplace. However, the inherent attractiveness of benchmark reports to the "seller" demands a certain skill, on our part, to differentiate the sales-pitch from the benchmark information. But even if we are skilled at this differentiation, what's so important about benchmarks?

Consider the thoughts we go through in buying a car. Obviously, we differ in our decision-making techniques but it would be one-sided to think about performance and nothing else. Clearly, many other factors may be brought into the decision-making process. Benchmarks but a small (albeit important) part of the selection, decision or evaluation process. Other influencing factors include:

- Price
- Service
- Longevity—will it last, will the company selling it last
- Robustness
- Reputation
- Add-on/Upgrade capability
- Aesthetics e.g., color, design, size

Some people take benchmarks very seriously while others are unaffected but not highly influenced by them. All in all, benchmarks take a place in the comparison/evaluation process.

The Audience

We classify the audiences for benchmarks into engineers (makers), marketeers (sellers) and users (buyers). Each has its own needs and methods of evaluating benchmarks. The engineer wants to optimize her system's design. The benchmark she uses tests pieces of a computer and displays bottleneck areas or components which can be improved. By varying a single element in the system, an engineer can tweak the system, using a benchmark program as her measuring instrument, to compare one configuration against another.

The marketer wants to sell a product. He would love nothing better than a single number which conclusively shows that his computer outperforms the competition. Benchmarks also provide some indication about the strength of competing products, and what market niche the computer should target. More often, they serve as little more than advertisements.

On the other end, the users want objective comparisons between different systems they might purchase. Benchmarks appeal to their need for (theoretically) unbiased reporting of a system's performance.

Naturally, each user's needs vary depending upon the individual's experiences and tastes. The ideal solution to benchmarking is to take the final applications that the users actually perform and run them on the various machines. Unfortunately, this usually proves impossible, especially for complex applications. Needs can rarely be anticipated, which makes much of the testing speculative. This situation leaves us looking again to existing benchmarks, which are armed with the abilities to discern fact from fever. Hence, let's consider how benchmarks are misused to gain momentum in our attempts to isolate or solution.

Misuses of Benchmarks

In benchmark reporting, we have discovered a distinct path between truths and lies. Developing the skill of telling not-really-the-truly and not-downright-lies graduates the benchmark reporter to a new and significant level of "successful benchmarking." Many look upon benchmarks as "marketing-hogwash" which are only coincidental with facts.

continued on page 16

Confessions continued from page 15

In two reports with the same results, we encounter two different messages to the public. Such situations are common but this does not negate their contradicting effect on a fast-reading, often overlooking audience. Possibly, more detailed information could clear up any confusion about how these conclusions were derived.

But the lack of detailed information is not the only stumbling block in benchmark reporting. Consider person M who endeavors to buy a used car. M visits a reputed dealership, isolates a few choices and finally decides on a beautiful car which is kept indoors under beautiful lighting. The car is dry, clean and reasonably priced. M purchases the car, drives it off the lot and discovers, on a rainy day, that the car leaks! Benchmarks can bring colored light and shelter to the leaking, flawed products in a similar manner. By highlighting the good and overshadowing those flaws, a benchmark report could deceive an audience until that terrible rainy day.

For example, incomplete information is as dangerous as lying in the benchmark world. Consider an article published in the April issue of *EDN Magazine* highlighting a UNIX based benchmark with performance numbers for various competitive systems. Under the subheading "Test Results Tell the Story" we read

> *System capabilities are subject to interpretation, but the results of a simple benchmark provided by the manufacturer tend to support claims of fast processing.*

But "results" *don't* tell the story. The story requires much more information such as details of configurations used, the methodology used in tests, etc. A comment on performance serves no purpose to the true benchmarker. It also feeds incomplete information to readers (who admittedly enjoy sweeping, unsupported ideas, however true they may be.)

Ironically, we received a copy of a benchmark report, generated by Teus Hagen and Andrew Tenenbaum from Amsterdam, titled "Two Programs; Many UNIX Systems" that highlighted the same CPU-bound program. They ran this program on a multitude of competitive systems. With amazing coincidence, the results they derived on the systems were identical to those published in the EDN article. However, Hagen and Tenenbaum drew the following conclusion in their article:

> *None, you should take these measurements with a grain of salt, or better yet, an imperial gallon of salt."*

Picking Valid Benchmark Reports

As we discussed before, benchmark reports can deceive more than inform by giving incomplete data. The effects of benchmarks on us can be approached rationally, however; as rationally as one approaches any form of advertising without discounting everything. The following considerations may assist in developing this rational approach towards reading and writing benchmarks by being less caught up in the results and more involved with knowledge of how the information is presented.

Look at any benchmark report and consider the following:

(a) *Who originated/authored the report*

Obviously a report on XYZ done by the company that created the product will show XYZ successful. Unbiased reports are hard to obtain but one technique would be to get benchmarks from other companies that include XYZ as a competitive measure. Thus, if you want to bench XYZ, don't obtain benchmark reports from them. Instead, ask other companies for their reports that involve XYZ. Chances are they would report XYZ objectively.

(b) *Determine the objective of the report*

The need emerges for abstracts, detailed introductions, and summaries. Readers who have no time tend to read the introduciton, the graphs, and then run to the conclusions. Thus, we have the need to identify and establish the message in any report. If you don't get the message clearly, drop the report—it will harm more than help. Examples of clearly directed reports/articles are References 13 and 5.

(c) *Descriptions of Methodology Used*

Without a crisp description of how the benchmarks were performed, there is no use describing the results. The reader wants to reproduce these results to trust them and a comment on the technique used would be sufficient. For example, "CPU, I/O-bound benchmarks were compiled with optimize, run five times each and the average user system time taken" would help. Questions can pop up like. . .

- was it optimized, pessimized on compile?
- is it a single user benchmark are other tasks running?
- any estimations?
- are they run from the same directories?
- were the programs manually timed or was a utility used?

Descriptions of the programs or benchmarks used and further descriptions of the environments where testing took place are crucial to a successful benchmark report.

(d) *Source Listings*

Documented source code or listings of the actual keyboard instructions will give us, the readers, a clear understanding of what was performed. From this, we can reproduce the benchmark and criticize or applaud the technique. There's something about actual "code" that makes it all seem so technical and accurate!

(e) *Details of Systems Used in the Benchmark*

The report should give detailed information systems upon which the benchmarking occurred. This data would guard against readers who assume too much about the results obtained, and can possibly detect systems which were unmarketed souped-up versions used only for the benchmark. Still sticking to our concern for reproduceability, we need details of the system configuration.

continued on page 17

Confessions continued from page 16

Detailed descriptions of system configuration can be found in Reference 2, the **Intel** benchmark report on "SBC 286/10 Single Board Computer as a XENIX Engine." This is an example of how configuration details can be displayed, so that readers can be instigated to further discussions about running, etc. We suggest that the following information be included in appropriate reports:

SYSTEMS USED
MANUFACTURER:
MODEL:
Processor type:
Clock Speed:
Memory:
offboard:
onboard:
Disk Manufacturer/size
Operating System/Compiler
Price
Special features: tape etc.

(f) *Results are never forgotten but. . .*

Ah! Benchmark reports always display results, results and more results. "We beat them by 5X performance difference" is all we hear. The public seems to crave a single, definite "X" value that, to them, described complete system competence in comparison to other competitive systems. We realize that an average number like 1.5x or 2x is merely nothing but an average estimation of comparative system performance, comparative "X" values are useless unless linked with what is being compared e.g. 2x better than system Y in I/O capability. Long-live the single "X" values but let us just move away from this mentality and consider what could make valid benchmark-result reporting.

Benchmark results are usually represented graphically as well as in raw number form. We suggest that results be displayed in absolute raw numbers in combination with either a relative graph or an absolute graph.

The main aim of benchmark reports is to inform, display results and discuss findings. As in any scientific experiment, raw numbers or colored graphs are irrelevant scribblings without careful analysis and interpretation. In other words, next time you hold a benchmark report in your hand, look for a position or conclusions taken by the author. If he/she has not made one, don't make one yourself.

Benchmark Design?

Questions about the benchmark and its design still remain. A look at benchmark programs today is a study in dichotomies: simple single task processes vs. complex multiuser job streams. Current taste favors the former.

First and foremost, small programs are easier to type! People can usually comprehend their purpose and method without difficulty. They seem to test a single element in a computer system (CPU speed, disk access, etc.). Like minimalist art, streamlined programs have an elegant aesthetic. Because they are (presumably) easy to run, reproducing their results presents few problems. Yet, these small programs often suffer from their simplicity. As mentioned earlier, the well-placed spotlight of a single benchmark focuses attention on one aspect of system performance, ignoring the rest. The question of "exactly what the one program actually measures" rears its formidable head again.

The gargantuan global benchmarks provide a stark contrast. Usually transported via many reels of magnetic tape and the child of laborious years of effort, these complex tasks require considerable expertise and time to set up and run them. They produce voluminous statistics on many aspects of a system's performance. Since their design supposedly reflects the requirements of typical computer loads, the results should predict real-life situations. Unfortunately, the design is frequently not verified and the benchmark numbers are often cryptic—hardly the stuff for managerial decisions. Completeness can also be a problem. One missing software utility used at the beginning of the benchmark stymies the rest of the run because subsequent tests rely on previous results. To compound the problem, standardization from system to system is difficult to achieve due to differing language and operating system versions, enhancements, and omissions.

One approach to system benchmarks is to use the following paradigm. To admittedly oversimplify the matter, a system benchmark could consist of an I/O loop and a CPU loop. You introduce parameters which determine how many times the individual loops are iterated. That way, you can have a program which is as I/O bound or CPU bound as desired. To test different application conditions, you merely vary those parameters and the number of programs running concurrently. This conceptual mode offers these advantages:

- Basic modules can be easily coded
- Each module can test a discrete function
- Individual modules are easily run and understood
- Programs can be combined to produce complex tasks which simulate real-world applications

This method has been called a synthetic benchmark. A synthetic job is intuitively appealing and offers great promise in industrial practice. Although synthetic benchmarks have historically received great interest, little work has appeared attempting to formalize the concept. Academics and others should research its theoretical basis so we might be able to get one step closer to a benchmarking standard.

Many problems and considerations are inherent in benchmark implementation. For example, the actual coding is non-trivial. Each language has certain strengths and weaknesses, but a programmer must be careful that the benchmark measures system performance and not her own cleverness. Certain languages (and hardware, for that matter) tempt the programmer to use "tricks" which bias the test. For example, a microprocessor might decrement faster than add and so ap-

continued on page 18

Confessions continued from page 17

pear faster with an addition-only program. Code transportability must be considered. Benchmark programs should be well commented and conform to commonly accepted language standards.

Compiler efficiency also has a tremendous impact on system speed. Assembly code can circumvent this problem, but it also introduces machine dependencies. The method in which a higher-level language is implemented can provide hidden optimizations. Two different compilers operating on the same source code can produce dramatic differences. In this light and considering language structures, benchmarking two *different* languages is ludicrous.

Determining the weight to assign each portion of a synthetic benchmark requires a good understanding of a computer system's actual use. Many operating systems can provide empirical data about the nature, frequency and requirements for performing each task.

To design even an individual module, you must clearly define its purpose. An ill-conceived program measures many aspects of a system—none of them properly. A well constructed and targeted program can effectively focus on a single performance element and form a valid building block for a synthetic benchmark. For example, an I/O loop might exercise moving data within pages on memory, from disk to disk, from memory to disk, etc. You must isolate individual performance factors, although don't get caught up in itsy-bitsies, until you form an adequate model on a computer system. Some possible factors are shown in figure 1. This list is only an example of what might affect system performance; the list will vary with the benchmark requirements.

Benchmark designers and discerning readers will remember that different types of users need different benchmarks. Special consideration is commanded to fulfill the needs of office automation, word processing, scientific, multiuser, or database applications. Indeed, system performance in each area depends on the proper match of hardware and software as well as basic machine speed. For example, the public is acutely interested in benchmarking multiuser environments, although they are especially difficult to measure accurately. The entire topic of benchmarking requirements covers a wide and barely touched area. Given the confusing variety of factors involved in benchmarking, it's a wonder anyone even attempts such an onerous task.

Conclusion

In this article, we have attempted to explain a formalism that creates a framework for credibility in the development and reporting of benchmarks.

There is still much more to investigate and discuss. We hope that areas such as detailed benchmark design for specific applications, actual mechanics of benchmark execution, benchmark evaluation and the possibility of standardized benchmarks will be attacked and courageously covered in the future. Unfortunately in the benchmark world, it's usually not "who knows" but "who shows" that counts. We have no lemon laws for benchmarking that guard against misrepresentation. But we do have methodologies and general techniques that assist us in our understanding or development of benchmarks.

Figure 1

Hardware factors

CPU speed in instructions per second
Logic speed
Memory access speed
Slave store hit rates
Store access path loading
Interrupt handling facilities
Store organization
Drum address organization
Drum channel loading
Disc rotational latency
Disc seek times
Controller loadings
Magnetic tape channel loadings
Card reader speeds
Line printer speeds
Communications processor loading
Multi-processors

Software factors

Slave store algorithms
Virtual store management algorithms
Backing store management algorithms
Scheduling algorithms
Compiling algorithms
Editing procedures
Interrupt handling
Physical file handling
Logical record management
Multiple buffering
Indexing methods

The role of reader is seldom confronted but demands mention. The audience, if polite, will permit misrepresentation in benchmarking and if aggressively critical, will nurture clear and honest benchmark reporting.

In short, this article is not a thorough treatment of the deep subject of benchmarking. Rather, consider it to be a starting point for discussion and ammunition against credible reports. You are encouraged to pursue the references for more insights. We are also anxious for your involvement and opinions in this controversial subject.

(Peter Marvit is a senior analyst at Yates Ventures. Mohandas Nair is a technical marketing engineer at Intel Corporation. *The authors would like to thank Gene Dronak of AIM Technology, David Billstrom of Oregon Software, the Berkeley marching band, and Madeline for suggestions and inspirations. This article will appear in the February issue of Byte magazine.)*

REFERENCES

1. *Legg, G. "Virtual memory workstation achieves high graphics resolution" EDN, April 14, 1983, pg 67-78.*

2. *Benchmark report "The Intel SBC 286/10 Single Board Computer as a Xenix Engine" Number 8 May 1983 (order #230676-002)* continued on page 19

onfessions continued from page 18

3. *Benchmark report "Erathosthenes Sieve Prime Number Benchmark on the SBC 26/10 Board" Number 7 March 1983 (Order #21094-002)*

4. *Hagen, T. and Tenenbaum, A. "Two programs, many Unix sytems." ?? unpublished*

5. *Grappel R.d., Hemeway J.E., "A tale of Four Microprocessors: Benchmarks Quantify Performance," EDN April 1, 1981, pg 179-185*

6. *Gilbreath, J. "A High-Level Language Benchmark," BYTE, September 1981, page 180-198*

7. *Curnow, H.J. and Wichman B.A., "A. Synthetic Benchmark", Computer Journal, Vol 19 Number 1, February 1976.*

8. *Benwell, N., BENCHMARKING. J. Wiley & Sons, Toronto (1975)*

9. *Spooner, C.R., "Benchmarking interactive systems: modeling the application", proceedings of the 15th meeting of the Computer Performance Evaluation Users Group, pg. 53-63.*

10. *Davies D.J.M., "Benchmarking in selection of time-sharing sytems, proceedings of the 14th meeting of the CPEUG, November 1981, pg. 27-36.*

11. *Cabrera, L.F., "Benchmarking UNIX: A Comparative Study, in Experimental Computer Performance Evaluation," (D. Ferrari and M. Spadoni, eds.) North-Holland, Amsterdam.*

12. *Cabrera L.F. and Paris J-F, "Comparing user response times on paged and swapped UNIX by the terminal probe method," Proceedings of the 14th meeting of the CPEUG, November 1981, pg 157-168.*

13. *Gilbreath J. and Gilbreath G. "Erathosthenes Revisited " BYTE Vol 8 Number 1, 1983, pg 283-326.*

The Future of UNIX

by Jean Yates

This article summarizes the key changes that must be made to System V.2 to provide a truly commercial UNIX product across micro, mini, and mainframe computers. It reviews some basic concepts and looks at AT&T proposed methods of dealing with them.

As seen in Table 1, the UNIX System today and the changes that need to be made to it fall into functional groups.

Today the expert user or programmer is typical of the UNIX enduser. In 1983, the dominant type of user shifted from university/government programmers to commercial programmers and members of the small computer manufacturing and distribution industries. This "priming of the pump," a painfully long process for UNIX systems, appears to be about complete, and by the end of 1984 the dominant user will be a novice business user interested in general business or business problem solving applications. The UNIX system must adapt to this new user with a cohesive menu/multiwindow user interface that eases UNIX use.

Table 1

THE UNIX SYSTEM TODAY	AT&T TECHNOLOGIES: WILL THEY MEET THESE CHALLENGES?
Expert user/programmer	novice/business users
Software development system	office uses
Technical documentation only	Good beginner manuals, online help improved
Berkeley Net, other non-ATT nets	A good ATT supported UNIX to PC to 3270 network
Only a few, specialized applications from ATT	Demand for large "IBM/PC/Charlie Chaplin" library of applications
No standard microcomputer UNIX in enduser's hands	Need for 16-bit, slow disk, office-oriented UNIX
$, #, >, etc.	Menus, windows, graphic representation (icons)
300-1200 baud most common terminal speed connection	Bit mapped, very fast, high band width terminal connection.

The UNIX system today was first designed for a software development environment, and was most useful on 32 bit minicomputers. The traditional UNIX user has developed software and performed tasks within the general topic of "program development." By the end of 1984, the dominant business UNIX user will use UNIX as an office tool, enhancing its electronic mail and networking facilities with packaged applications for accounting, spreadsheet, etc. These applications and the smaller systems on which they run require a UNIX operating system unbundled from the software development system.

Technical documentation is available for UNIX today, but dividing the manuals into smaller and smaller subsets is not enough. Good beginner's manuals, online help, and support of training facilities is necessary to make UNIX meet office enduser needs.

A Standard UNIX Network

Today the *de facto* UNIX network is a variation on the BSD software, with many other non AT&T networks under development. As UNIX becomes a dominant vehicle for networking different vendors' equipment together, AT&T must provide a supported UNIX network that connects personal computers and IBM 3270 terminals to UNIX systems. This will probably be *the* issue of 1984,

continued on page 27

Marketing Corner

Selling UNIX to the ISO

This article is intended to be used as a presentation guideline for the professional sales person who is confronted with the task of presenting UNIX to a Dealer or other Independent Selling Organization (ISO).

When UNIX first became available on the new generation of supermicros vendors could count on the UNIX world to consume enough of the machines to get the lines rolling. But, as competition has grown, and as UNIX becomes more of an industry standard, it becomes necessary to reach beyond the confines of those who know and understand UNIX. Vendors must broaden their customer base to include all avenues of distribution. In order to do this, vendors must show the ISO how and why he should sell UNIX based computer systems. No small chore! The ISO must also be able to present UNIX to the end user marketplace in a manner which makes sense to the customer. An even more formidable challenge!

So, the problem becomes one of how to present a UNIX based system to a prospect in a sufficiently attractive package to make that prospect want to buy a UNIX system. The outline presented herein depicts a typical sales scenario and how a typical sales person can present the product to the prospect in a manner designed to demonstrate how powerful UNIX is and how easily it can be used to sell computers and solve application problems.

In order to demonstrate just how easy UNIX is to use and to sell, the vendor must be able to demonstrate the power of UNIX and how it can be adapted to meet a specific enduser's needs.

Since enduser sales are unquestionably 'near and dear' to the heart of every ISO, the vendor needs to show all the benefits of his system and how those benefits are going to help the ISO sell more of his systems than of "Brand X". Remember, the best approach to selling computers is to "sell a solution to a problem". The ISO has two major problems: First, how to sell computers to endusers. Second, once sold, how to successfully install the system at the least cost and greatest profit.

The classic approach in this problem is to sell 'features, functions, and benefits'. In the past vendors have demonstrated the hardware features that are not available on the competition's box. Sometimes the programs included word processing software, a spread sheet, and some nifty accounting programs. Or maybe, if the ISO was a little more sophisticated and had some programming capability, the vendor could show off the operating system features which allow developers to get systems out the door quicker and at a lower cost than a competitor's. But, having been around for a while now, most vendors have noticed how hardware, operating systems, application software (such as word processing and general accounting), and all the other features they've been selling are starting to look pretty much the same from one computer to the next. As the industry matures this similarity is going to become even more visible. Yet vendors know, all else being equal, that features are still the best way to sell a computer.

So, distinguishing the product from all the other 200+ on the market becomes a major concern. After all, if the vendor can show the ISO how to sell a system, he will sell a system. The more the ISO can move, the more the vendor will move. Elementary logic. But how?

The Typical Sales Scenario

First, let's set up a 'typical' ISO sales scenario. You're at a major trade show and you've got a prospect who is looking for a new multiuser system to sell.

Like all professionals, you have qualified your prospect and know that you can close the sale if you can just give him what he wants. Your price is right, your discounts competititive, your product's performance is excellent, you don't have any representation in the prospect's marketplace, the prospect has good credit, and you believe he can sell a reasonable amount of product for you. You want this ISO on your team.

What do you do next? You proceed to demonstrate why he should be selling your product rather than someone else's. You tell the prospect that you can go up to X-MB of high speed RAM, you have the latest processor technology, your disk storage and back-up capability is incredible, you have applications, and you talk about all the other benefits your sales and support program offers. You even explain that you have UNIX. At this point you are ready to measure the prospect's reaction; which will probably go one of two ways. Either he says, "That's nice, but what's different about your product?" Or, there is a spark of interest, (followed immediately by a compelling need on your part to proceed with the rest of your story). In either case you have soon reached the point that you must demonstrate the product.

If you are using a systems engineer to tell your story you now have to announce to the prospect that you will call in someone who can operate the machine. (Always a bad way to start a demo and you hope that you can make this announcement in comforting undertones that are only somewhat less audible than the prospect is likely to grasp.) If you are doing your own demos you proceed boldly and fearlessly to the keyboard with your new-found disciple, the prospect, hovering intensely over your shoulder.

First you go through UNIX, showing how everything works. You show the shell, directories, pathnames, processes, login, passwords, file permissions and options, background processing, redirection of standard input and standard output, environments, devices and files, pipes, switches, and a number of other basics of UNIX. You also mention all the languages that your system supports and that a number of popular accounting packages, data bases, and other applications are available and supported.

continued on page 21

Selling to the I.S.O. continued from page 20

Once you have the preliminaries out of the way, assuming the prospect is still with you, you must now address yourself to the specific needs and problems of the prospect.

Now, in our example, let's suppose that this prospect has special interests in electronic mail, word processing, and communications. The prospect has also explained to you that a certain big company back in his home town has some branch offices and wants to standardize on a method of sending inter-office memos and correspondence between the home office and these branches. Plus, they want to standardize all company policies and keep them updated and on-line for all employees to see at any time. This big company is also very conscious of productivity in the word processing department. They want to use a system that is cost effective, simple to operate, and, at the same time, very powerful.

UNIX obviously has all these capabilities. In fact, it sounds like UNIX was made to order. So, first you proceed to demonstrate "vi", or some other word processor. Next, you decide to show "grep" and "fgrep", with the count options set to demonstrate how word processing and other text or lexical statistics can be generated. Maybe your systems engineer has also explained how easy it is to write "shell scripts". Sensing that the prospect is still with you, you move right along to "mail" to show how to handle the electronic mail problems. For communication needs you decide to show "cu" and "uucp". As for the on-line documentation requirement, you demonstrate "nroff" using the "an" macros to format the documentation before you "mv" it to "/usr/man/cat?". As *everyone* knows, this is how documentation is made available to the "man" command. Maybe you even remember to put the "-e" option on "nroff" to get full terminal resolution.

Of course, during the demonstration there have been some questions. Most of these you have been able to field and have only had to allow your systems engineer to speak on a limited basis. You have shown as much of UNIX as the time permits; even if you did have to show it just like it comes out of the box.

What happens next? The prospect, after a brief pause, may ask you how long it takes to learn all this. He may say nothing at all. Or, he may ask if you sell anything with MS-DOS or CP/M. If you have done your job extremely well the response might be, "Well, I guess we're going to have to learn UNIX. But right now I just don't have time. I'll get back to you when we decide to get into UNIX."

Naturally, there are many other possible responses. Some are good, and some are not so good. In any event, you know that there must be a better way to tell your story on UNIX.

There are better options. Instead of going through all of these technical gyrations vendors can show the same things, or even more things, using Office Automation menus. Vendors can also generate special menus just for sales presentations. In either case, they would not have needed system engineers to intervene, and they can show the prospect exactly what he said he wanted to see and can do it in words the prospect understands. The vendor can thus make a better presentation.

Shells give endusers a way to use UNIX without having to know UNIX. These products bring out all the best features in UNIX for the Office Automation Environment. Considering the immense capabilities of UNIX, there is no reason to duplicate the software already contained in the operating system. UNIX is already the most powerful office automation system on the market. Vendors simply need to make UNIX usable by the average office worker.

The main purpose of shell products is to give vendors a method of presenting their solutions to the prospect's problems in an intelligent and understandable manner. Each of these Office Automation menus is designed to show the specific features, functions, and benefits of UNIX. The vendor's own narrative is the final ingredient to a successful presentation of UNIX on his system. In other words, make sense out of UNIX. Make UNIX usable.

(This article was reformatted and reprinted with permission from UniComp Corporation).

Company Profile

Adobe Systems

by Joann Andrushko

Adobe Systems was founded in December 1982. Its staff of 12 includes President John Warnock, who was Principal Scientist at **Xerox's** Palo Alto Research Center, where he was involved in advanced raster display techniques. Vice President Charles Geschke, also from PARC, helped design the **Xerox** *8000* network series product family. Completing the management team is V.P. of Marketing Stephen MacDonald, who was Marketing Manager at **Hewlett-Packard.** The first round of financing was obtained from venture capitalists **Hambrecht and Quist.**

Adobe's product, known as *POSTSCRIPT*, is a software interface between a document composition program and a raster output device. It integrates text and graphics, giving the user wide options to scale, rotate or transform all character shapes. The company believes that until now, the user has not had such a powerful tool to control text/graphics integration.

POSTSCRIPT is written in the C programming language and runs under UNIX. It was developed on a *VAX 11/750* and ported to a 68K **Sun** *Microsystem.* Because POSTSCRIPT has few UNIX dependencies, it will port to other operating systems. Adobe plans to develop a TROFF driver. POSTSCRIPT will hook into a word processing package from **Unilogic** known as *SCRIBE.*

The product, still in R&D, features a full programming language, the POSTSCRIPT language.

Adobe will target the large hardware vendors such as **Hewlett-Packard** and **DEC**, at a price of around $4,000 with quantity discounts. The package will introduced in first quarter of 1984.

Market Research Study

Part II: Vertical Market Opportunities for UNIX Based Systems

by Eileen Skrabutenas

In last month's issue of *The Yates Perspective*, we discussed factors and trends driving the vertical market acceptance of UNIX based systems uncovered by YATES VENTURES exhaustive survey of systems integrators and vertical market software developers. This month's article takes a closer look at trends *within the following individual vertical markets*:

- Communications
- Education
- Financial Services
- Government/Military
- Manufacturing
- Professional/Health Services
- Retail Trade
- Transportation
- Wholesale Trade

Communications—A UNIX Stronghold

Until 1975, vertical market acceptance and use of UNIX was almost exclusively limited to the communications industry. Created by the research arm of **AT&T**, one of the largest telecommunications companies in the world, UNIX has been used internally by the **Bell System** to control everything from huge, electronic switching systems to office automation systems in the front offices of Bell Operating Companies.

The communications market consists of three major areas of opportunities to vendors of UNIX based equipment and software:

- **Bell Operating Companies** (BOCs);
- Value Added Independents—**SP Communications, MCI**, etc.;
- Enduser markets—PBX/Computer controlled communications networks.

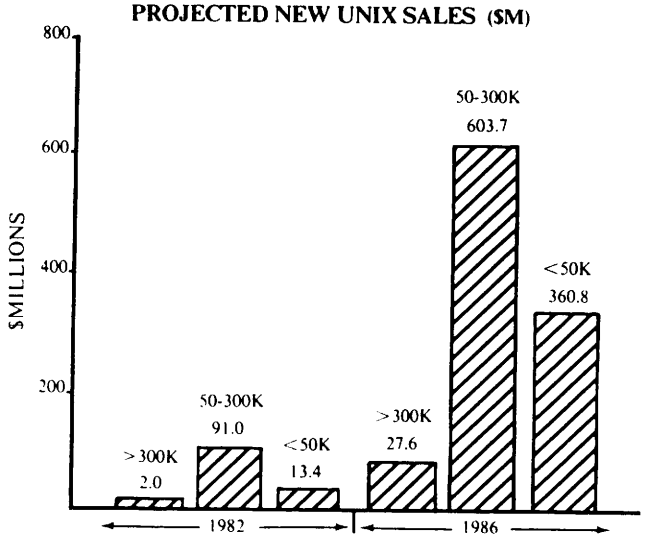
Vendors of UNIX based products have a new opportunity to sell communications software to the divested operating companies. It is unreasonable, however, to expect that vendors will simply be able to replace **Western Electric** (WECO) as the BOC's primary supplier. Loyalty and equipment familiarity will tie the BOCs to Western Electric in the short run. Furthermore, AT&T's use of a central sales organization to market WECO manufactured equipment to the BOCs indicates that the divested parent is not abandoning this market segment.

DEC, IBM and **Computer Consoles** (CCI) are the three main computer system suppliers to the BOC's outside of Western Electric. Their high end systems with UNIX offer a natural replacement for Western Electric *3Bs*. Digital Equipment and CCI publicly announced their commitment to UNIX, and IBM is expected to release the *4300 series* with UNIX during the first quarter of 1984.

Market Projections

The high end ($50-300K) market segment will offer the greatest enduser dollar opportunity for vendors of UNIX based products. Sales are expected to increase from $91 million in 1982 to approximately $604 million in 1986, as indicated in Figure 1. This increase will be the result of rising sales of expensive fault tolerant, distributed network based systems. In comparison, sales in the <$50K category will rise from $13 million to $361 million during the same time frame. However, unit sales, shown in Figure 2, will dominate in the low end (<$50K) category as higher performance capabilities are added to machines at lower price points.

Figure 1
COMMUNICATIONS
PROJECTED NEW UNIX SALES ($M)



Educational Services

The educational services market can be split into two segments:

- universities;
- elementary and secondary schools;

and two applications;

- R&D or training;
- administration.

Universities

Outside of internal use at the Bell System, UNIX is most widely accepted by the university community. AT&T's dissemination of UNIX under inexpensive educational license in 1976 directly contributed to its subsequent commercialization and acceptance in markets other than communications and education. Computer science students were trained in the rich program development environment provided by UNIX, and upon graduating they demanded access to it in industry.

continued on page 23

Vertical Markets continued from page 22

The availability of this low cost, research and programming oriented, portable operating system has created a virtual UNIX monopoly in the computer science departments of most universities. UNIX is also widely used within university and university-to-university networks primarily on DEC supplied minicomputers. Universities are currently making a great push to supply students with their own PCs or provide them with access to less resource intensive multiuser microcomputers running UNIX.

Elementary and Secondary Schools

Opportunities for vendors of UNIX based products in the elementary and secondary school systems are limited. This market is dominated by a rapidly growing installed base of **Apple** *II's* and **IBM** *PCs*. It is characterized by a one student, one computer learning environment. Demand for a sophisticated computing environment (i.e. UNIX) is non-existent at the level of classroom instruction. Furthermore, the financial pressures burdening most public schools limit the funds available for computer purchases. If recent efforts by Apple to donate Apple II's to every school in the country are successful, the elementary and secondary schools will be "owned" by Apple.

In both market segments, systems offering administrative and business application solutions represent viable targets for UNIX vendors. The ideal product consists of inexpensive multiuser solutions offered by UNIX based systems in the <$50K price point, bundled with accounting applications. Such a system would appeal to systems houses trying to sell into an educational services market that is marked by a chronic shortage of funds.

Market Projections

YATES VENTURES projects that the largest dollar and unit volume opportunities for UNIX vendors will exist in the university/R&D related segments of the educational services industry. Vendors offering systems in the under $50,000 price point are expected to have the greatest impact as single user workstations and the "supermicros" attract budget minded administrators. **Sun Microsystems** workstations are an example of one company selling CAE/programmer workstations as alternatives to additional minicomputer purchases. These products are networked into minis running UNIX. Large compiler/ programs are performed on the mini, while smaller tasks run on the workstation.

Financial Services

The two main opportunity areas for UNIX in the financial services market are:

- Insurance;
- Banking and Securities.

Insurance

Within the insurance industry, UNIX has been limited to a few selected and supported corporate sites using DEC *PDP-11s*. Outside of the corporate environment, companies providing insurance related services (independent agents, etc.) have traditionally relied on service bureaus and time sharing services for the bulk of their data processing requirements. **IBM** *370s* and *4300s* dominate the installed base of these services.

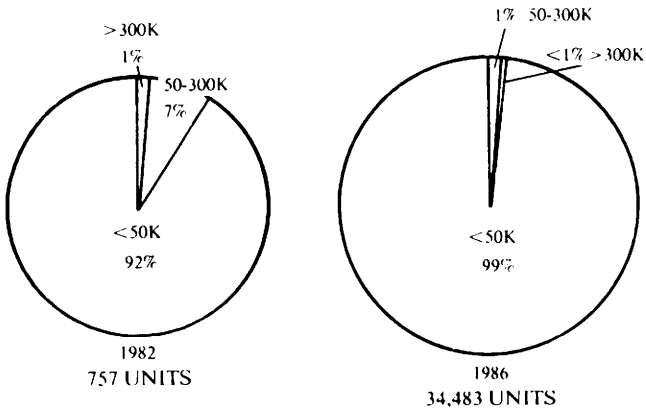
Independent agents and regional claims processing centers represent two of the prime opportunity areas for vendors of UNIX based products. Users in this market are in a transition period, switching from renting time sharing services to individual PC ownership, multiuser and distributed networked systems. The upswing in purchases is due to the demand for access to general purpose computing, office automation and a dramatic decline in the price of hardware.

Banking and Securities

The acceptance of UNIX based products in the banking and securities market closely parallels the pattern of acceptance in the insurance market. Initially restricted to isolated corporate dp environments at firms like **Wells Fargo** and **Crocker Bank**, UNIX is finding wider application for distributed, network office automation, and specialized applications in investment tracking, trust accounting and commodities and foreign exchange trading.

Sales to these market segments will be derived from high end database oriented machines in the $50-300K category networked to smaller <$50K systems targeted for specific applications. The market for UNIX based systems in the banking and securities industry is still forming. Selected targets have been identified by **Bunker Ramo, Quotron Systems** and **Computer Consoles**. It is expected that these companies will pioneer the UNIX penetration in the banking and securities sectors.

Figure 2
COMMUNICATIONS
PROJECTED NEW UNIX SALES
(NUMBER OF UNITS)



Market Projections

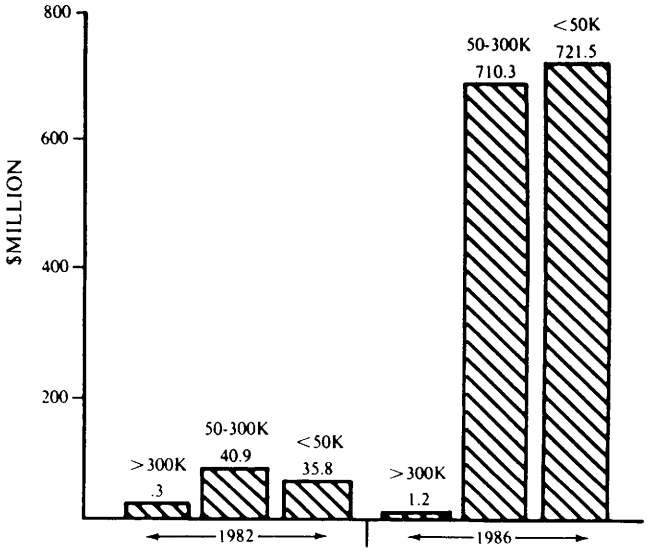
Shown on the following pages are Figures 3 and 4, which illustrate annual dollar and unit sales for the years 1982 and 1986. The greatest unit market growth will occur in the <$50K market segment, from 1900 units in 1982 to 68000 units in 1986. By 1986 the dollar value in the $50K and $50-300K segments will be approximately equal. Caution is in order, however, as YATES VENTURES projects that the influence of IBM at the high

continued on page 24

Vertical Markets continued from page 23

end, particularly in the insurance market, will limit opportunities for today's UNIX vendors. Specialized computers, such as fault tolerance with networked database software will have the greatest degree of success in this segment. In the under <$50K area, vendors must address specific enduser needs through systems integrators if they intend on penetrating the vast, unfocused financial services industry.

Figure 3
FINANCIAL SERVICES
PROJECTED NEW UNIX SALES
($M)



Government/Military

Currently, the UNIX government/military market falls into the following categories:

- Federal Non-Commercial—Military, R&D related;
- Federal Commercial—Civilian Agencies, non R&D related;
- State and Local.

Federal Non-Commercial

Traditionally UNIX acceptance in the government/military markets has been restricted to selected Department of Defense (DOD) sites engaged in research and development. Very few systems were utilized for commercial or non R&D use. Prior to 1983, many, if not all sites were source licensees with high end *VAX* and *PDP-11* systems representing the largest share of the installed base.

Applications in this market segment are typically engineering/scientific in nature and often require virtual memory support. Opportunities for continued UNIX penetration in the Federal Non Commercial market will most likely consist of sales of high end UNIX computers offering virtual memory, distributed networking and massive data crunching capabilities.

Federal Commercial

The Federal Commercial market represents the single largest, most focused target for vendors of UNIX based products. A UNIX spec has appeared on nearly every major RFQ released by the federal government civilian agencies (e.g. the Forest Service, Justice Department, Social Security Administration, IRS, etc.). Recent UNIX wins include the **MA/Com Sigma Data** bid to supply over 120 **Zilog** *System 8000s* to the IRS valued at over $18 million. MA/Com competed against **Plexus**, **Altos**, and **DEC**.

State and Local

Although the greatest opportunities for UNIX vendors exist in the federal government and military segments, system integrators serving the state and local governments are gradually turning to UNIX. Our survey results showed a number of system integrators selling to the state and local segment to be actively evaluating UNIX or in the process of rewriting applications so they could be transported to a UNIX environment. In sheer numbers, the state and local government has the potential to match the federal commercial segment in unit volume, although this market is less focused and will require a greater investment in system integrator sales to yield profitable results.

Manufacturing

The manufacturing industry is widely diversified. For the purposes of this study, we have subdivided it into two major submarkets:

- Computer/Electronics Manufacturing
- Process Manufacturing

For example, market projections in the first category would include the internal R&D consumption of **Onyx** and **Zilog** computers by software and hardware vendors to develop UNIX based products. The latter category includes companies in the traditional manufacturing industries as well as any firm with specialized manufacturing (MRP related) applications.

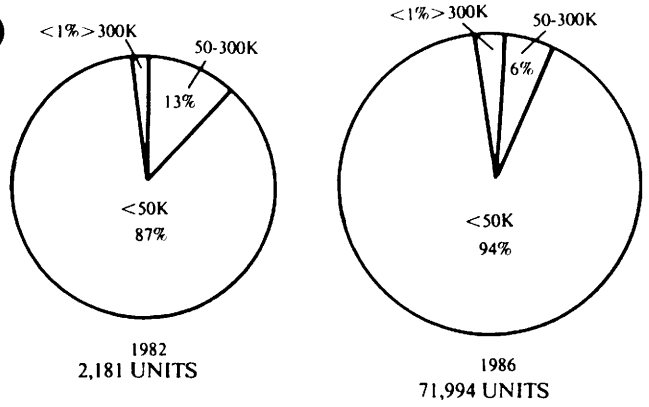
Computer/Electronics Manufacturing

This segment represents a diminishing opportunity for vendors of UNIX based products. As the market matures, and UNIX based products are released to market, the consumer or process manufacturing segment will outstrip the demand for development oriented machines. Once a development house has a UNIX based computer, additional sales are more likely to consist of add-on software products rather than new hardware. Practically all the major hardware vendors have already staffed and outfitted their UNIX projects. If they are indeed using UNIX to develop a UNIX based computer, then whatever future internal needs develop they will probably be fulfilled by their own internal production. As for computer vendors which are considering a UNIX project and may not yet have pulled together all the necessary resources, they will most likely encounter a VAX or PDP environment that could be retrofitted with UNIX to provide the appropriate UNIX environment.

continued on page 25

Vertical Markets continued from page 24

Figure 4
FINANCIAL SERVICES
PROJECTED NEW UNIX SALES
(NUMBER OF UNITS)



Process Manufacturing

This market segment contains businesses as diverse as garment assembly and petrochemical processing. Potential applications include everything from accounting to sophisticated product tracking to actual process control. A key characteristic of this market is that, until recently, factory automation has taken place on a piecemeal basis with different systems addressing different application requirements. However, in attempting to connect each component to another, users encounter a compatibility problem. As a result, the standardization UNIX offers is increasingly viewed as the solution to diverse manufacturing networks. The desire among system integrators and others offering factory automation and manufacturing turnkey systems to migrate to C and UNIX is stronger in this market than in any other. However, an established presence by **Hewlett-Packard** and **IBM** may prevent smaller firms from entering the market, particularly as these companies release their UNIX based products.

Professional/Health Services

The single largest opportunity for UNIX based vendors exists in the professional and health services markets. Approximately 75 percent of this market remains unautomated. Marked by great diversity in professions and desired applications, one common theme runs across all: most are small businesses either contemplating purchasing their first computer products and services or switching from a time sharing, service bureau approach for specialized accounting applications to purchasing a general purpose computer to handle office automation and specific turnkey applications.

The majority of the professional/health services market can be divided into:

- Professional--Lawyer, CPA, Consultant, Architect, etc.
- Health--Doctor, Dentist, Small Clinics, Hospital

Professional

Historically, computer vendors have entered the professional market wearing a word processing or an office automation banner. **Wang**, in particular, has penetrated a significant number of legal firms by offering a solution to the paper bottleneck and clerical burnout many firms experience. However, this is changing. The impact of the PC has changed the way professionals view automation. Rather than restricting computers to word processing tasks, many are now looking beyond a dedicated **Lanier** or **Wang** word processor into a more generalized system offering mutliuser capabilities for a wider range of applications including profession specific or vertical market, accounting and financial planning, in addition to word processing.

Health

Although marked by great potential, UNIX acceptance in the health services market is just beginning. Traditionally dominated by time sharing service bureaus, the health services market is looking for less expensive, more generalized solutions to provide word and data processing. Remotely connected terminals are rapidly being replaced with PCs and multiuser micros to provide greater flexibility. Initial forays by system integrators and software developers writing medical software indicate that general market acceptance of UNIX based products won't occur before mid 1984. However, UNIX has been accepted to a limited extent in several hospitals including Henry Ford in Michigan, and St. Georges Health Center in Toronto, where the need for database oriented applications like patient record and equipment tracking is creating a demand for computers in the $50-300K category.

Retail Trade

In general the retail market is expected to provide limited revenue potential for UNIX vendors in comparison to the other nine markets. This market is almost as diverse as the professional/health services market covering florists to auto parts stores. For the purposes of this study we have segmented the retail market for UNIX into the following areas:

- Point of Sale;
- Accounting and Inventory Control Systems.

Point of Sale

The real time, transaction processing and high reliability expected from a point of sale system will restrict the acceptance of UNIX by most retail stores. Sales that do take place will most likely consist of high performance computers enhanced for real time transaction processing. The inclusion of fault tolerance will be a key differentiating factor. However, only large retail establishments will be able to afford the current average price tag of $100,000 on fault tolerant machines.

Accounting/Inventory Control

Retail market acceptance of UNIX will be greatest for accounting and inventory control systems where transaction processing and fault tolerance are not critical requirements. Assisted by **Fortune Systems** and others, several system integrators have developed portable versions of their proprietary retail trade applications. Yet, the requirements for support among small, computer na-

continued on page 26

Vertical Markets continued from page 25

ive retailers and the premature release of products will be responsible for slow market acceptance and growth. The majority of sales will be derived from <$50K unit purchases by small retailers.

Transportation

Transportation covers many diverse submarkets. Opportunities for UNIX vendors will exist mainly in addressing the following application:

- accounting, inventory control and scheduling for the transit, trucking and warehousing industry;

due to the lack of an established presence by specific hardware vendors.

Trucking to Lead Industry

With hundreds of small-to-medium sized trucking firms, intra and inter-city transit companies, a potentially large volume market exists for UNIX vendors. De-regulation of the trucking industry, in particular, is forcing companies to streamline costs and improve general productivity. As automation comes to the front office and shipping and receiving, UNIX based systems are sought to provide a cohesive, standardized, cost effective solution.

Software Availability Creates Market Potential for Low-end Systems

Although system integrators offer transportation, transit and warehouse management systems based on **IBM** and **Basic Four** equipment, the expensive, high ticket price has kept the majority of the industry from automating. However, because their software has been written in C or other high level languages, the system integrators are evaluating UNIX based products as the most efficient vehicle to move their software to and address lower price point customers. Our interviews showed that several system integrators have already brought their trucking applications to UNIX based micros. By doing so, these system integrators will assist the penetration of UNIX vendors into this market segment.

Figure 5
WHOLESALE TRADE
PROJECTED NEW UNIX SALES
($M)

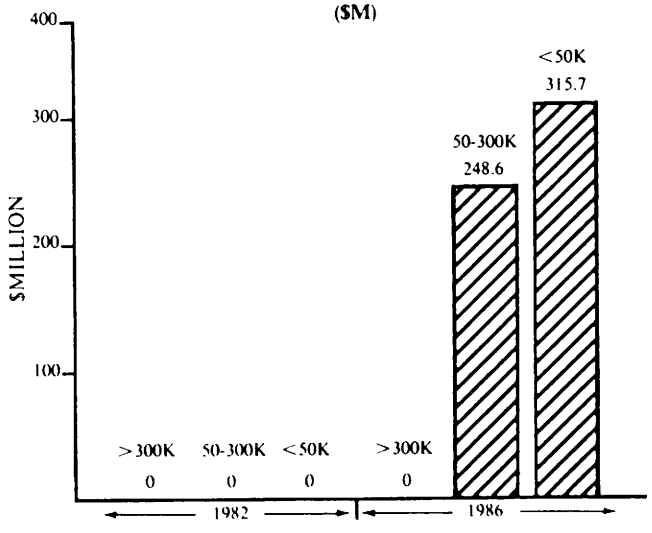
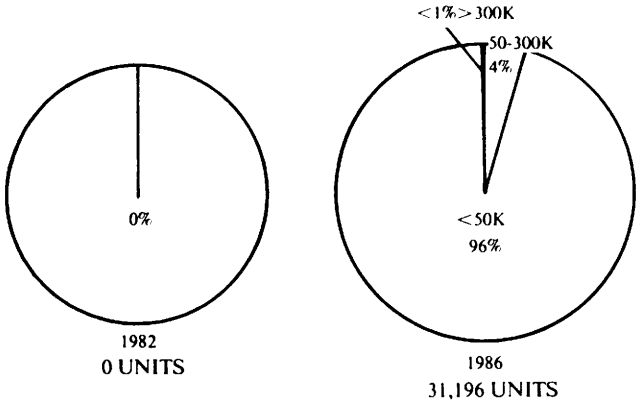


Figure 6
WHOLESALE TRADE
PROJECTED NEW UNIX SALES
TOTAL (NUMBER)



Wholesale Trade

Wholesale trade is concerned with purchasing and distributing products for resale. Like retail trade, wholesale trade covers many diverse businesses ranging in size from small local distributors to nationally based firms. Our surveys indicate that customer demand for software and turnkey systems in this market segment is driven by a need to automate accounting and inventory control functions. Most customers are first time users, with fairly generic software requirements.

The greater customer and system integrator awareness about UNIX means that the vendor wishing to sell into this market has less of a learning curve to cover in promoting his product.

Market Projections

Illustrated in Figure 5 are annual dollar sales of new UNIX systems. The wholesale trade market did not register any new UNIX sales in 1982, since system integrators addressing the wholesale trade market were still in their UNIX R&D stage. Products, particularly the lower price systems, began shipping earnestly in 1983. However, the market will grow quite rapidly, generating over $550 million in annual revenue by 1986.

Annual unit sales are compared in Figure 6. As expected, by 1986, the <$50K segment dominates as the majority of sales will be to small firms. Also, the growth in small wholesaling establishments will dominate the growth of larger firms.

The majority of market growth will occur among smaller firms automating for the first time. In the past, these firms have remained unautomated because they have not been able to afford the expensive turnkey products that adequately met their needs. However, with today's multiuser, microprocessor based systems, greater system performance and functionality is available at prices starting as low as $16,000 for a four user system.

Need more information on this important subject? Yates Ventures offers a more detailed analysis of market trends in our just-released multiclient report on vertical markets for UNIX based systems. The report includes profiles of more than 200 software designers and systems integrators. Contact Pamela Pasotti, Account Executive at Yates Ventures, at 415/964-0130.

Future of UNIX continued from page 19

as enduser and vendor alike attempt to provide a cohesive method of connecting personal computers to back end file servers running the UNIX operating system.

Application Software

Only a few specialized application packages are available from **AT&T**. There is a demand for a large number of applications, particularly on under $25,000 systems. A UNIX equivalent to IBM's "Charlie Chaplin" library of MS-DOS applications for the **IBM** *PC* is needed. AT&T must encourage third party software developers.

Today, no standard System V microcomputer UNIX is in the hands of endusers. The **Motorola** port is undergoing quality control at AT&T, and will probably be first out the door. Hardware vendors and endusers are taking a wait-and-see attitude until the product is released. A 16 bit micro small-disk-oriented UNIX is needed for office microcomputers. Will the standard microcomputer System V product meet this need? The market will wait and see.

Today, the $, #, > or @ signs are used to indicate the UNIX prompt, and these symbols are the totality of the user interface. A standard cross vendor menu/windowing/graphic representation, such as the one on the Apple Lisa, is required for the success of UNIX in the office. This product should be available in late 1984 to meet the demand for easy to use UNIX systems.

AT&T has defined some sample UNIX system tool kits, which may be refined further before the final standard is set. An operating system communications tool kit containing basic networking, an X.25 facility, local area networking, and an RJE facility has been proposed. This tool kit will hopefully address the issue of the MS-DOS or concurrent CP/M connection to UNIX.

Source Code Control System and Programmers Workbench were unbundled from UNIX in Version 7, and bundled back into the system in System III. Unbundling of PWB and SCCS again is a natural direction for UNIX to take.

Unbundling of languages is another tool kit that could be interesting to users. Fortran 77 and the C language could be unbundled and offered as separate products.

Instructional and Documentors Workbenches

With System V.2, AT&T introduces some new Workbenches that accompany *Writers Workbench*. Writers Workbench is a writer's tool that checks spelling, sentence structure, word usage, etc. *Instructional Workbench* is a training tool for UNIX students. *Documentors Workbench* is new with System V.2, and unbundles the vi, nroff, troff, and related macros from UNIX as a whole. The troff is device independent.

Along with unbundling of the operating system the low-end PC market requires a smaller UNIX, designed for less than 256K memory, and containing only a subset of the total system calls. The latest rumblings from the press have it that AT&T will introduce a home computer with UNIX. Obviously, 10 megabytes of utilities will not be included in an under $1000 system. But even with everything unbundled, UNIX is inherently a 32 bit product. A desirable accompaniment to MS-DOS or CP/M 86 would be an AT&T home/personal UNIX. If this product, compatible with big UNIX in networking and file transfer comes to market, we should see an expanded UNIX market place. If UNIX becomes a standard for personal computers to connect to large data bases on high end micros and minis, the use of UNIX on personal computers will make all YATES VENTURES forecasts expand rapidly in response to the huge ship rates.

Lastly, in this preliminary overview of challenges that AT&T must meet, the terminal speed connection, which is currently narrow bandwidth and transaction oriented, must be expanded to allow bit mapped, high band width terminal connections. Personal computers are fundamentally graphics/high band width fast screen response systems that take advantage of the entire screen. If PCs are to act as terminals to UNIX or run native mode UNIX, UNIX must provide the proper terminal protocols.

The AT&T Response

AT&T has designed some UNIX system architecture objectives. They are responding to customer requests for increased flexibility and a modular definition of UNIX, and are providing definitions of the UNIX system. AT&T plans to maintain a consistent compatible system between System V.1 and V.2.

AT&T is aware of the value added reseller market (VAR) and will provide a basic system from which VARs can build systems for a variety of price points and markets. A modular UNIX architecture will provide a flexible base for building applications. Isolating the operating system from the numerous UNIX utilities will clarify hardware dependencies and ease movement of application products to other machine configurations.

Trademark Acknowledgements

X,Y = X is a trademark of Y

Altos 586, 68000	Altos Computers
Lisa, Macintosh, Apple II, IIe, III	Apple Computers
UNIX	Bell Laboratories
Callan Unistar 200	Callan Data Systems
Codata 3300/84	Codata
Adam	Coleco
Commodore 64	Commodore
Corvus Uniplex	Corvus
VAX, PDP-11, DEC Professional	Digital Equipment Corp.
CP/M, CP/M 16, CP/M 86	Digital Research
Dual 80	Dual Systems
Eagle PC, SPirit, 1600, EagleNet I	Eagle Computers
Fortune 32:16	Fortune Systems
HP 9000/220, 9000/400, 9000/500	Hewlett-Packard Corp.
Intel 330/286	Intel
IBM PC, XT	International Business Machines
COHERENT	Mark Williams Company
MS-DOS, XENIX, Multiplan, Microsoft Word	Microsoft
NCR Tower	NCR Corp.
PLAN Series	Nestar
Onyx C8002	Onyx
Pixel 80	Pixel
Plexus P/35, P/40, P/60	Plexus
Pyramid 90x	Pyramid Technology
INGRES, MicroINGRES	Relational Technology
SMC Basic	Science Management Corp.
SunStation	Sun Microcomputers
VENIX, VENIX/86	VenturCom
Wangwriter	Wang
WICAT	Wicat Systems
Z-8000	Zilog

YATES VENTURES

Yates Ventures is a San Francisco Bay area based firm specializing in market research and documentation on standard system and application software. The founder, Jean Yates, is an author of *A User Guide to the UNIX System*, (1982, Osborne/McGraw-Hill). Yates Ventures has several more books under development on UNIX and 16 bit operating systems, and is a frequent contributor to the computer trade press.

The experience of the staff of Yates Ventures ranges from technical documentation to research expertise to technical backgrounds in mainframes, minicomputers and microcomputers.

Jean Yates, founder, is a featured speaker at NCC, COMDEX, UNIX User Group meetings, and IEEE, among others, and is regarded industrywide as a top expert on UNIX and small systems. Yates has managed numerous major studies on markets and strategies in the software and small system areas. She was previously manager of Gnostic Concept's UNIX Information Service and has performed custom studies on markets and strategies.

Kevin Gross serves as editor of the *Yates Perspective*. His background includes work in financial services and journalism. He has assisted in the preparation of research reports for the Department of Economics at Stanford University, and was previously engaged as a market research consultant at Yates Ventures. He is currently pursuing an economics degree at Stanford University.

The Yates Perspective

Publisher, Jean Yates; Editor, Kevin Gross; Managing Editor, William Franzosa; Production and Subscription: Linda Erickson; Contributors: Eileen Skrabutenas, Peter Marvit, Mike Dubrall, Kirsten Sanders, and Carolyn Peterson.

The Yates Perspective is a monthly newsletter produced by Yates Ventures on the software market. It includes competitive strategies, new product reviews, listings of interesting university publications and public domain software, and miscellaneous news on the UNIX-related hardware and software industries.

You can get on the mailing list for the newsletter or market research reports by calling or writing Yates Ventures, or by filling out and mailing the slip below.

_____ Yes, I want to subscribe to The Yates Perspective. Enclosed is a check or money order for $450.

_____ Yes, I would like a brochure on the 1984 Market Research Reports. Please send it to the address below.

Name _____

Title _____

Company _____

Street _____

City _____

State _____ **Zip** _____

Country _____

Phone Number _____

Address all correspondence to:

Yates Ventures
4962 El Camino Real, Suite 111
Los Altos, California 94022
(415) 964-0130