UNIX* Micros for Students Majoring in Computer Science and Personal Information Retrieval†

Edward A. Fox
Sandra Birch

Department of Computer Science
Virginia Tech
Blacksburg, VA 24061

In the fall of 1985, over 230 freshmen majoring in computer science at Virginia Tech acquired microcomputers running the UNIX™ operating system. Purchases have also been made by other students, and by faculty and staff. This event had been planned since the spring of 1983, at which time the faculty of the Department of Computer Science voted to institute a requirement of personal computers for majors. A Request for Proposal for such systems was distributed to vendors in the fall of 1984, and selection took place early in 1985.

The first part of this article reviews the planning and decision making process, and explains why UNIX was chosen as the most suitable system to use in the training of computer science students (and for many other purposes as well). Implementation of the microcomputer program is discussed, including revisions to the freshman curriculum and the distribution of required hardware and software.

The latter part of this article continues with a discussion of research and development efforts aimed at giving users of these systems special software for personal information retrieval. ELSI (Enhanced Library System Interface) will provide a front-end with extra capability for those wishing to access the Virginia Tech Library System (VTLS). MIR (Microcomputer Information Retrieval) will automatically index documents and allow natural language queries instead of menu or Boolean forms.

INTRODUCTION

In the fall of 1985, more than 230 freshmen beginning their studies in the Department of Computer Science at Virginia Tech purchased personal computers running

*UNIX is a Trademark of AT&T Bell Laboratories, Inc.

†Project funded in part by grants from the National Science Foundation (IST-8418877) and the Virginia Center for Information Technology (INF-85-016). An earlier version of this article will appear in *Proceedings of the Seventh National Education Computing Conference*.

the UNIX[1] operating system. This event was the result of extensive planning by the Department of Computer Science which decided in the spring of 1983 to require the purchase of microcomputers.

Virginia Tech is the first major U.S. public university where students have been required to buy microcomputer systems running UNIX System V. This article traces the history relating to the requirement, explains key features of UNIX, explores the rationale for the decision, and describes the implementation of the program. It also describes research and development efforts to develop personal information retrieval tools for these systems.

HISTORY

There are two aspects to the decision to require students to buy computers running UNIX: first, the microcomputer requirement, and second, the choice of the UNIX operating system. Since these two matters were considered and decided consecutively, an historical account is in order.

The computer science faculty voted in the spring of 1983 to require students to buy personal computers. This would assure students of better access to computing facilities as the University's public facilities were fast becoming saturated. Students would have maximal opportunity to learn to use computers as tools for problem solving and also to use computers for other coursework, e.g., word processing in English classes. Determination was made that almost all of the courses in the computer science curriculum could be taught relying on reasonably powerful microcomputer systems. Reports of successful efforts at other universities further encouraged the planners.

At that time, UNIX systems were too expensive to be purchased by students. A standard Request for Proposal (RFP) was prepared in the fall of 1983, but was not distributed. The University had just completed a similar process for the College of Engineering, with the IBM PC family of systems selected. The assumption was also made that the PC Junior would be adequate for engineering students. The PC Junior was considered to be too limited and too slow for computer science students. Thus, the department chose to wait a year before preparing another RFP.

In the fall of 1984 an initial draft of the RFP was prepared, calling for personal computers running UNIX. Based on trends in the industry, it was projected that by 1985 such a system might become more reasonably priced for students, given suitable quantity discounts for educational institutions. The RFP was refined and distributed late in 1984. Certain hardware features were required, such as main memory size (RAM) of 512 KB. However, the most stringent stipulations were in the area of software and price. The RFP stated that all hardware and software costs must total no more than $2500, and that a package price for two years of maintenance be $500 or less. These figures were determined by the Virginia Tech administration, and reflected the philosophy that the cost of an education at a state university should be kept as low as possible. Software requirements stipulated full

[1]See also one of many UNIX books such as Bourne (1982) or Kernighan and Pike (1984).

microcomputer versions of UNIX, either Berkeley's Standard Distribution 4.1 or AT&T's System III, along with compilers for C, FORTRAN, and Pascal. The three languages required were important because C is commonly used with UNIX, FORTRAN is needed in numerical analysis courses, and Pascal has been and will continue to be the primary high level language used by Virginia Tech students during the first two years of their undergraduate training. See Feuer and Gehani (1982) for a comparison between C and Pascal, to better understand this choice.

Several companies met the requirements of the RFP. A committee of computer science faculty, staff, and students thoroughly tested a number of hardware and software packages. A best and final offer was allowed, and the selection of a special system from Apple Computer, Inc., and Unipress Software, Inc., was approved by the computer science faculty. Two years of AppleCare maintenance, supported by University personnel in Communications Network Services (CNS), was offered. The selected system included a Mac-L (i.e., Mac-XL configuration with an external Apple Profile 10 MB disk instead of an internal drive) with 1 MB of RAM (user memory with UNIX running is over 750 KB), one 400 KB diskette drive, and a mouse. Figure 1 indicates how the system appears.

Apple software includes MacWorks, MacWrite, and MacPaint. UNIX System V, Assembler, and compilers for C, FORTRAN, and Pascal were provided by UniPress Software, Inc., as the distributor for several other companies. A separate purchase of UNIX System V (source code) for Motorola 68000 processors was made from Western Electric by the Department of Computer Science. Access to this software allows use of text processing programs and the addition or fixing of other UNIX routines. This follows a long tradition of computer science departments having the ability to adapt UNIX to special needs and requirements (McKusick, 1985).

Apple Computer, Inc. and UniPress Software, Inc. donated a number of systems to the University. Most of those are operating in two of the University's public microcomputer laboratories. Non-computer science majors taking computer science courses, as well as upperclassmen and graduate students, utilize these facilities. Figure 2 shows one section of the laboratory in McBryde Hall, the building where most computer science courses are taught. All students have access to printers in these 24-hour laboratories and can use the microcomputers for small editing tasks. In addition, more than a dozen of the donated systems were assigned to the Department of Computer Science so that faculty, staff, and graduate assistants could learn UNIX or become familiar with this particular version, and prepare for their teaching responsibilities.

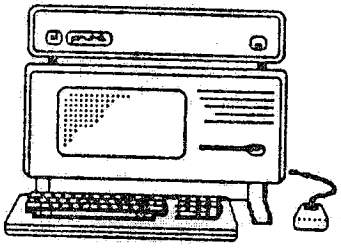

Figure 1. Mac-L System

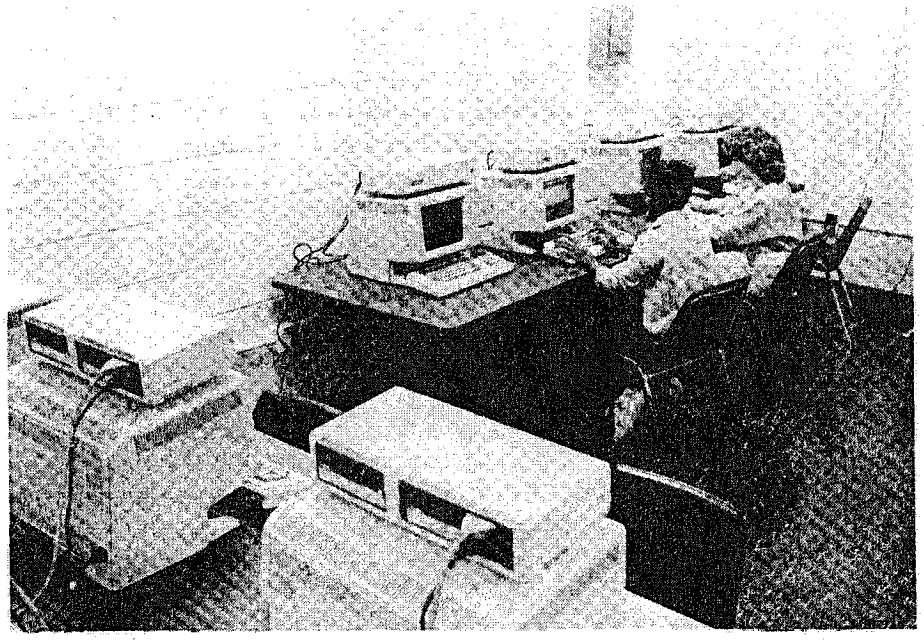

Figure 2. Microcomputer laboratory for graduate and undergraduate users

UNIX HISTORY AND FEATURES

UNIX is an operating system together with a large amount of software tools aimed at supporting program development, text processing, and other activities. Before giving specific reasons as to why we feel UNIX is the proper software environment for training computer science majors, it is of value to review the history and features of UNIX.

Since the first version of UNIX was developed in 1969 by Ken Thompson of Bell Laboratories Research Group, many enhancements have been made, leading to a host of products. The name UNIX is a pun on MULTICS, the system developed over more than a five-year period through joint efforts at the Massachusetts Institute of Technology, General Electric, and Bell Laboratories (Organick, 1972). UNIX was strongly influenced by MULTICS, but was originally implemented by a much smaller group in a much shorter time frame.

The C programming language (Kernighan & Ritchie, 1978) was developed at Bell Laboratories to support UNIX, and was used almost exclusively in the third version of that system, completed in 1973. Most system development work undertaken in the UNIX environment has been in C, and C is now used for similar efforts under other operating systems such as MS-DOS and VMS.

UNIX Version 7 was distributed in 1978 for PDP-11 and Interdata 8/32 systems. AT&T later released UNIX System III and V, and the University of California at Berkeley prepared other versions: 4.1 and 4.2BSD for VAX, and 3BSD for PDP-11

series. Many other names have been given to licensed "ports" of UNIX to a variety of computers, including XENIX, VENIX, and ULTRIX.

UNIX systems have a kernel, the central core of the operating system, which supports a set of system calls so that user programs can have desired tasks carried out. UNIX kernels manage memory, processes, and input/output (I/O); recent efforts have aimed at standardizing UNIX at the kernel interface level. Interactive users, however, typically communicate with a "shell" program that surrounds the kernel; typically they use the Bourne shell (sh), developed at Bell Labs, or the C shell (csh), developed at Berkeley. The shell understands commands and can be programmed (Bourne, 1978).

At the shell level, UNIX provides many capabilities. A hierarchical file system, where all hardware elements are included as "special files" in the "/dev" directory, contains all system and user data. Device-dependent I/O simplifies many programming tasks. Managing processes is straightforward, leading to an elegant style of computer use whereby tasks are broken into subtasks which function concurrently. Pipes are easily set up so that one process can transform input to an intermediate form which is then handled by a sequence of one or more "filters"; if each process performs a single function, then the pipeline is a composition of those functions. Processes can be submitted for background execution in parallel with foreground (interactive) commands, so users have the ability to learn about and control simultaneous job processing.

There are hundreds of UNIX routines available as commands which can be executed interactively or through files of commands, called "shell scripts." System commands display and manipulate files or directories. Some do sorting, comparing, or looking for patterns. There are editors, compilers, and text formatters. Since most UNIX systems have storage space for different users, there are also facilities for mail handling.

By 1983, the two most complete versions of UNIX were System III, developed at Bell Labs, and 4.1BSD from Berkeley. Both provided very good environments for program development, text processing, and other computing activities. System III was emerging as the most common form for commercial applications, while 4.1BSD was the typical UNIX system used in computer science research. Either was considered fully adequate for our students. While these versions have been improved with releases of System V and 4.2BSD, respectively, they were not widely available on personal computers until recently. Current efforts of AT&T and Sun Microsystems Inc. indicate that System V will eventually be enlarged and improved to have all important features now in 4.2BSD (Chandler, 1985).

RATIONALE FOR A UNIX REQUIREMENT

UNIX, an operating system that has become well known in recent years, has been hailed by some and criticized by others. This section summarizes many of the reasons the department selected this system.

A. Familiarity with UNIX is likely to directly benefit students entering the job market:

1. UNIX is becoming a standard operating system for micros, minis, and large systems. While UNIX first became popular as an operating system for Digital Equipment Corporation minicomputers such as the PDP-11 and VAX series, it now operates on personal computers and supercomputers as well. AT&T has made a firm commitment to support UNIX System V, and Digital Equipment Corporation supports UNIX 4.2BSD under the name of ULTRIX. Because of this support by leading computer manufacturers, the availability of versions of UNIX on most types of computers, and the almost fanatical enthusiasm of many UNIX users, UNIX is becoming the closest approximation to a standard operating system that has yet emerged. It already is the *de facto* standard in computer science departments in the United States and in several other nations (Stoneback, 1985). Students studying UNIX and/or C will have an obvious advantage in the job market.

2. There is a tremendous demand for people experienced in UNIX and C. In the spring of 1985, a petition was circulated by the department's graduate students requesting that a graduate level course on UNIX and C be offered. Nearly 30 enrolled in the course, some having postponed taking other required courses so as not to miss this opportunity. Clearly, computer science graduate students feel UNIX and C are important topics to master. Many have explained this interest as stemming in part from the awareness that these skills will help them obtain desirable jobs. Programmers experienced in UNIX are indeed in short supply, and this trend will continue during the next several years as the number of UNIX systems steadily grows. The C language is being used with other operating systems as well, as a highly effective implementation language, so the need for good C programmers is also rapidly growing.

B. Learning C and UNIX helps one learn computer science methods, and vice versa:

1. Regular expressions are utilized in the editors, in filters (e.g., grep), and in the shells, so students learn to actually use ideas that are often taught only in the abstract in formal language theory classes.

2. Work with the file system helps to develop understanding of tree structures and hierarchies.

3. Structured programming of the shell helps to teach good software development practices; many other command languages lack such elegance (Bourne, 1978).

4. The C language is a good vehicle for systems programming because of its handling of data types (including casts, structs, typedefs), its use of pointers, the ability to manipulate bits, and the storage flexibility of unions (Gehani, 1985).

5. Extensions like C++ can be taught as well, once they are more fully developed (Stroustrup, 1985).

6. UNIX tools such as LEX and YACC are useful in compiler courses and motivate students to use what they have learned about language theory to perform useful tasks.

7. UNIX text processing software increases student exposure to handling textual data in elegant ways—pattern matching capabilities, relational algebra operations (e.g., join), report preparation (e.g., AWK), and macro processing.

8. The style of UNIX development, where prototypes are quickly constructed by composing tools, is important to learn as an example of good software engineering. Tools like MAKE (Feldman, 1979) help with medium-size development efforts that are possible in university settings.

9. UNIX use requires understanding of the notion of process, which helps students understand the functioning of an operating system.

10. Students must become system administrators, and thereby learn even more about operating system support functions, such as backup procedures, and handling different devices.

11. Textbooks such as Comer (1984) are about systems very similar to UNIX, and are widely used in operating systems courses. Clearly the popularity of UNIX was one of the motivating factors of many authors' efforts to build UNIX-like systems and to use them to teach students about the theory and practice of operating systems. A number of textbooks (e.g., Peterson & Silberschatz, 1983) have chapters about UNIX to illustrate some of the important concepts of modern operating system design.

C. Using UNIX and C allows interchange of software developed in research projects:

1. All of the major software research and development efforts of the Computer Science Department are being done with C and/or UNIX. Students will be given software resulting from some of those efforts. For example, the first author of this article is supervising students developing an adaptation of LOGO for the Mac-L, a microcomputer version (MIR) of part of the SMART information retrieval system, and a front-end (ELSI) to the Virginia Tech Library System. Another faculty member is supervising development of a Prolog compiler that will also be ported. Database software will be developed, and the second graduate course in that area, CS5662, has for the spring of 1986 made knowledge of the C language a prerequisite.

2. UNIX software is relatively complete, so students need not buy a number of expensive packages. UNIX itself is expensive but provides a large number of services. While we had to make special arrangements to obtain compilers for Pascal and Fortran, a typical UNIX software package provides languages such as C and Snobol. There are games, text processing routines, source code maintenance facilities, and macroprocessors among the available packages.

3. UNIX software from other computer science departments can be utilized. UNIX is available in a great many computer departments around the world, so the base for development is very large.

Some negative factors considered in the decision were also addressed:

A. Running UNIX requires expensive hardware and software. However, the price/performance ratio is actually lower than that of less expensive systems and having a 10-MB hard disk and 1 MB of RAM ensures rapid response even on large problems. The functionality is also sufficient so that students can comfortably use their system for almost all of their computer science classwork as well as for work in other disciplines.

B. UNIX systems are not able to run the many commercially available software packages, such as are available under MS-DOS. However, these systems support the Macintosh operating system, which many students prefer. Students also have a monetary incentive to do their own application development for simple tasks, or else will learn how to combine UNIX tools to accomplish their goal.

C. Administering a UNIX system is difficult for new computer scientists. However, learning these techniques helps in the training of students and helps to instill confidence; the only problem is during the initial learning period when this burden weighs most heavily. The time to become reasonably proficient in UNIX may well be as much as six months, but this is offset by the fact that students will own their systems for four years.

D. Only computer scientists are likely to have a UNIX requirement in the near future, so students will be isolated from their classmates. However, this can be viewed as a way that students learning UNIX now will get a head start on their colleagues as UNIX continues to rapidly proliferate.

E. Students transferring out of computer science may have difficulties selling their systems. While this is a possibility, there are many more students trying to transfer *into* the curriculum, so students should be able to sell their systems to incoming students.

IMPLEMENTATION OF THE MICROCOMPUTER REQUIREMENT

Selection of a microcomputer and software package was only the beginning. No less than fifteen departments in addition to the Department of Computer Science have been involved in the implementation of the micro requirement. The Personal Computer Auxiliary, a unit of the University's Communications Network Services (CNS), is playing one of the most active roles.

First, financial arrangements had to be determined. Since Virginia Tech is the land-grant state university, the cost of an education should be within the reach of most of its citizens. An arrangement with the Virginia Tech Education Foundation, a non-profit organization, allowed students the option of purchasing their microcomputers over a two-year period through monthly payments. The Admissions Office, anticipating that the cost of the computer system would be a deterrent to prospective incoming freshmen, issued more offers of admission than in previous years. Surprisingly, a *greater* percentage of students actually accepted, and the 1985-86 freshman class is 60 students larger than previous freshman classes. The requirement appears to have *encouraged* more students than usual to accept admission to the department.

All entering freshmen were sent a letter from the department head in April explaining the program. This preliminary contact was followed in May by detailed commitment and order forms mailed by the Personal Computer Auxiliary of CNS. By mid-July, orders for 230 systems were mailed to Apple Computer, Inc. and UniPress Software, Inc., with delivery expected in August. Three staff members from the Department and CNS spent three days "burning-in" all the hardware and loading the UNIX software and compilers on each of the hard disks. To serialize in-

dividual software packages, three code numbers were retrieved from each boot diskette and sent to UniPress Software, Inc. for authorization numbers. This process saved a great deal of time and trouble as students did not have to contact UniPress individually to serialize their own systems.

Next, a team of upperclass students and graduate students were trained as installers and "trouble-shooters." Popularly known as the "swat-team," the students were headquartered in the Computer Science Consulting Lab, where a "hotline" telephone was installed. Distribution of the systems was conducted during the three days preceding the start of classes.

Following a distribution schedule they had received in early September, students first attended a two-hour seminar during which the "care and handling" of a Macintosh L was discussed by computer science faculty and Apple Computer, Inc., staff. Examples of the UNIX and MacWorks operating systems were briefly demonstrated. Special Apple T-shirts were distributed before the students moved on to the actual distribution site in the student union building. No system was issued unless the student had paid the proper fees and showed evidence of seminar attendance (a series of labels, coded with the student's name, which were used to identify the student's equipment boxes after pick-up).

Appointment times for equipment pick-up were determined by dormitory assignments. To facilitate delivery to students of their systems, all students in a single dormitory received their equipment at approximately the same time and rode together in special vans to their dormitories. Members of the University's ROTC were available to carry equipment to the awaiting vans. Swat-team members were then waiting at the dormitories to move the equipment into the students' rooms and install the microcomputers, as can be seen in Figures 3 and 4.

Figure 3. Van delivering student microcomputers

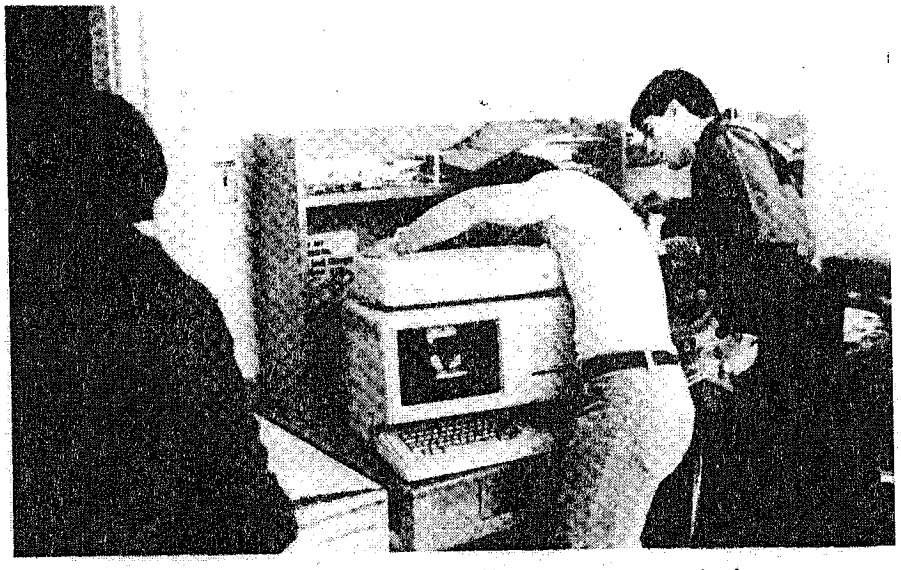

Figure 4. Member of "swat-team" installing microcomputer in dorm room

In less than three days, 230 systems were delivered and installed. By the end of the distribution period, more than 230 students were enrolled in a new one hour per week course on UNIX while also beginning their studies of Pascal in the first two introductory Computer Science courses. Whereas in previous years students used Pascal compilers on VAX 11/780 mainframes, they are now computing on their own Macintosh Ls. The Motorola 68000 assembler is being taught in the third course of the introductory sequence, instead of the VAX assembler. Each quarter the department plans to add at least one more course to the curriculum to be completed on the Macintosh L with the UNIX System V operating system.

Upperclassmen interested in UNIX and C can take the senior seminar, which will cover C during the winter quarter and UNIX during the spring quarter. These are only one-credit courses, but will convey the most important aspects of the subjects. Thus, starting this academic year, all graduates will have the opportunity to become familiar with C and/or UNIX.

An advanced graduate course in fall 1985 focused on UNIX and C. Twenty projects were undertaken to help with various UNIX activities. For example, several students worked with the Macintosh bitmap display and mouse so they could be better utilized through UNIX. These students are an example of the many individual efforts being made to help the overall UNIX effort in the Virginia Tech Department of Computer Science.

UNIX FOR INFORMATION RETRIEVAL

In the area of information retrieval, a number of research and development efforts have been underway at Virginia Tech to utilize the UNIX microcomputers. In-

deed, this work was first launched in 1983 when the Computer Science Department requirement was proposed.

In recent years it has become more and more obvious how valuable personal computers can be for information management. One exciting possibility is to transform the everyday encyclopedia into an exciting electronic service (Cook, 1984). This is indeed becoming feasible as laser optical disks are perfected that can provide low-cost access to large stored files (Fujitani, 1984). The information retrieval research community has eagerly followed this development (Goldstein, 1984). The greatest immediate potential, however, is that of integrating advanced information retrieval techniques with inexpensive compact disk laser read-only (CD-ROM) units (Fox, 1986).

The two leading United States information retrieval projects which led to development of radically different experimental systems in the 1970s were SMART and SIRE (Salton & McGill, 1983). In recent years SIRE, which allows the output of a search to be ranked according to presumed degree of relevance (Noreault, Koll, & McGill, 1977), has become a commercially marketed product that will even run on IBM PC/XT class machines. Efforts to adapt SMART to microcomputers, for non-commercial purposes, have been pursued largely at Virginia Tech (Fox, 1984).

Earlier work with SMART focused on selection of the proper data model to employ in an implementation (Fox, 1981). One version of SMART was developed for UNIX, but involved more than 20,000 lines of code and required support of a database management system for some of its functions (Fox, 1983). The challenge was to select the most important functions from that advanced system for inclusion in a microcomputer version. Five features of particular interest are:

1. Automatically indexing whatever form of document is provided. This involves identifying different "fields" of a document, locating words or other tokens, removing "stop words," and replacing normal text words with the stem of the word (Salton & McGill, 1983).
2. Allowing natural language queries to be similarly indexed and then used as queries (Salton & McGill, 1983).
3. Interpreting Boolean queries in an extended fashion, so that grades of "truth" can be determined for documents that do not "strictly" satisfy the logical query expression, and so that relative importance of terms in the document of query can be specified (Salton, Fox, & Wu, 1983).
4. Automatically constructing Boolean or extended Boolean queries from natural language statements, for cases where approach (2) above is not possible (Salton, Buckley, & Fox, 1983).
5. Automatically constructing new "feedback" Boolean or extended Boolean queries, by considering which documents a user has judged relevant after examining a small number of initially retrieved items (Salton, Fox, & Voorhees, 1985).

These features were each considered in planning the development of information retrieval tools that could be used on the UNIX microcomputers.

It was decided that the two most important uses initially possible with these microcomputers would be to serve as a front-end (ELSI) to the Virginia Tech Library

System (VTLS) and to act as a personal aid for microcomputer information retrieval (MIR). Students and other system owners could then have a flexible aid to help locate interesting library books, or to process their personal files for retrieval. Both ELSI and MIR were proposed by the first author of this paper in 1983. Figure 5 shows one of the systems where some of the recent testing of this software has taken place.

EXTENDED LIBRARY SYSTEM INTERFACE (ELSI)

In the winter of 1984 a group of computer science seniors drew up requirements, specifications, and design documents for the Extended Library System Interface (ELSI). The aim was to use a microcomputer as a front-end to the million-entry online library catalog (VTLS), both to make access simpler and to serve as a vehicle for experimentation. Whereas VTLS is a menu system, ELSI provides a command language interface that should be readily understood by computer science students. While the initial version would only carry out standard processing, a later version would utilize some of the advanced capabilities developed in SMART (see features 3, 4, and 5 listed in the previous section).

Coding proceeded in standard C, thereby making possible later use of the system even though a UNIX microcomputer had not yet been selected. Work continued in connection with a class project in a 1985 sequence of graduate courses in information retrieval, and developed into a Master's project. By the end of that project, all of the standard functions for accessing an online catalog will be imple-

Figure 5. MacL used to test MIR and ELSI

mented by Thomas McGuire. Subsequent work will focus more on experimental approaches. For example, it is planned that ELSI will be used to log and channel interactions of users accessing two systems to be compared: VTLS, and another computer using SMART to process the Humanities portion of the VTLS data.

Developing ELSI has been greatly aided by the multiprocessing capabilities of UNIX. Separate processes can do input and output between the user and the computer, and between that microcomputer and the library system—this is much easier to code than on more limited personal computer operating systems. Nevertheless, keeping track of the processing of a menu system is not simple when a computer can only issue commands that VTLS commonly supports. For users, however, it will be possible with one command to get a list of all books by any author without having to go up and down in the menu hierarchy. Users will also be able to save results in files, to get printouts, to receive assistance in response to help requests, and to access VTLS in "bypass" mode.

MICROCOMPUTER INFORMATION RETRIEVAL (MIR)

The Microcomputer Information Retrieval (MIR) system also began in 1984, with a like history. Completion of MIR is now being undertaken in connection with another Master's project, by John Gostel. Whereas ELSI will eventually provide enhanced Boolean logic capabilities, MIR is focused on automatic indexing of documents and queries. It employs automatic indexing stemming, term weighting, and relevance feedback in user-friendly form. While MIR was developed separately from SMART, many of the data and file structures and some of the processing modules were devised based on lessons learned with developing, adapting, and maintaining SMART during the last four years. Since both systems employ C and UNIX, this task was somewhat simplified, even though SMART was developed using 4.2 BSD while MIR will run also on System V UNIX.

It is expected that students will use MIR to maintain various databases that need to be regularly searched. While recipes are one possibility, it is more likely that bibliographies, letters, class reports, or library catalog entries (when used together with ELSI) will be the most common uses. Students need only create a properly formatted input file, and MIR will automatically index and store document representatives that can be later searched with English language queries.

SUMMARY AND CONCLUSIONS

In the fall of 1985, freshmen computer science majors at Virginia Tech were the first undergraduates in a public university required to buy their own personal computer running UNIX System V. These students will acquire, through the use of such powerful software and hardware tools, an excellent education in their major field.

Numerous individuals and departments at Virginia Tech have worked together to make this experience possible, and we believe that the atmosphere of cooperation will continue to benefit more students as they share their experiences

in using a powerful microcomputer and well-known operating system. Many lessons will be learned from this effort at Virginia Tech, and undoubtedly other universities will follow with similar requirements for UNIX in future years.

Furthermore, since UNIX has been made available, a wealth of software developed in connection with computer science research efforts will be usable. In particular, two packages to serve the information management needs of students and others in the university community have been under development. The Enhanced Library System Interface will allow flexible access to the online library catalog. The Microcomputer Information Retrieval system will provide tools for automatic indexing, search document ranking, feedback query construction, and display of results. It will be interesting to observe how students utilize the advanced capabilities of this information retrieval software and of their UNIX personal computers.

REFERENCES

Bourne, S. R. (1978). The UNIX shell. *Bell System Technical Journal, 57* (6), 1971–1990.

Bourne, S. R. (1982). *The UNIX system.* Reading, MA: Addison-Wesley.

Chandler, D. (1985). The monthly report. *UNIX Review, 3* (10), 8–18.

Comer, D. (1984). *Operating system design: The XINU approach.* Englewood Cliffs, NJ: Prentice-Hall.

Cook, P. R. (1984). Electronic encyclopedias. *Byte, 9* (7), 151–170.

Feldman, S. (1979). Make—A program for maintaining computer programs. *Software—Practice and Experience, 9,* 255–65.

Feuer, A., & Gehani, N. (1982). A comparison of the programming languages C and Pascal. *ACM Computing Surveys, 14* (1), 73–92.

Fox, E. A. (1981). Implementing SMART for minicomputers via relational processing with abstract data types. In *Joint Proceedings of SIGSMALL Symposium on Small Systems and SIGMOD Workshop on Small Data Base Systems, ACM SIGSMALL Newsletter, 7* (2).

Fox, E. A. (1983). *Some considerations for implementing the SMART information retrieval system under UNIX* (TR 83–560). Ithaca, NY: Cornell University, Department of Computer Science.

Fox, E. A. (1984). Information retrieval with undergraduate microcomputers, In *The micro revolution* (Collected papers of the American Society for Information Science 13th ASIS Mid-Year Meeting, May 20–23, Fiche 5:253–264).

Fox, E. A. (1986). Information retrieval: Research into new capabilities. In S. Ropiequet & S. Lambert (Eds.),, *The new papyrus: CD-ROM.* Bellevue, WA: Microsoft Press.

Fujitani, L. (1984). Laser optical disk: The coming revolution in on-line storage. *Communications of the ACM, 27* (6), 546–554.

Gehani, N. (1985). *C: An advanced introduction.* Rockville, MD: Computer Science Press.

Goldstein, C. M. (1984). Computer-based information storage technologies. *ARIST, 19,* 65–96.

Kernighan, B. W., & Pike, R. (1984). *The UNIX programming environment.* Englewood Cliffs, NJ: Prentice-Hall.

Kernighan, B. W., & Ritchie, D. M. (1978). *The C programming language.* Englewood Cliffs, NJ: Prentice-Hall.

McKusick, M. K. (1985). UNIX unleashed. *UNIX Review, 3* (10), 28–33.

Noreault, T., Koll, M., & McGill, M. J. (1977). Automatic ranked output from Boolean searches in SIRE. *Journal of the American Society for Information Science, 28* (6), 333–339.

Organick, E. I. (1972). *The Multics system: An examination of its structure.* Cambridge, MA: MIT Press.

Peterson, J., & Silberschatz, A. (1983). *Operating system concepts.* Reading, MA: Addison-Wesley.

Salton, G., Buckley, C., & Fox, E. A. (1983). Automatic query formulations in information retrieval. *Journal of the American Society for Information Science, 34* (4), 262–280.

Salton, G., Fox, E. A., & Voorhees, E. (1985). Advanced feedback methods in information retrieval. *Journal of the American Society for Information Science, 36* (3), 200–210.

Salton, G., Fox, E. A., & Wu, H. (1983). Extended Boolean information retrieval. *Communications of the ACM, 26* (11), 1022–1036.

Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval.* New York: McGraw-Hill.

Stoneback, J. (1985). The collegiate community. *UNIX Review, 3* (10), 24–48.

Stroustrup, B. (1985). *The C + + programming language.* Reading, MA: Addison-Wesley.