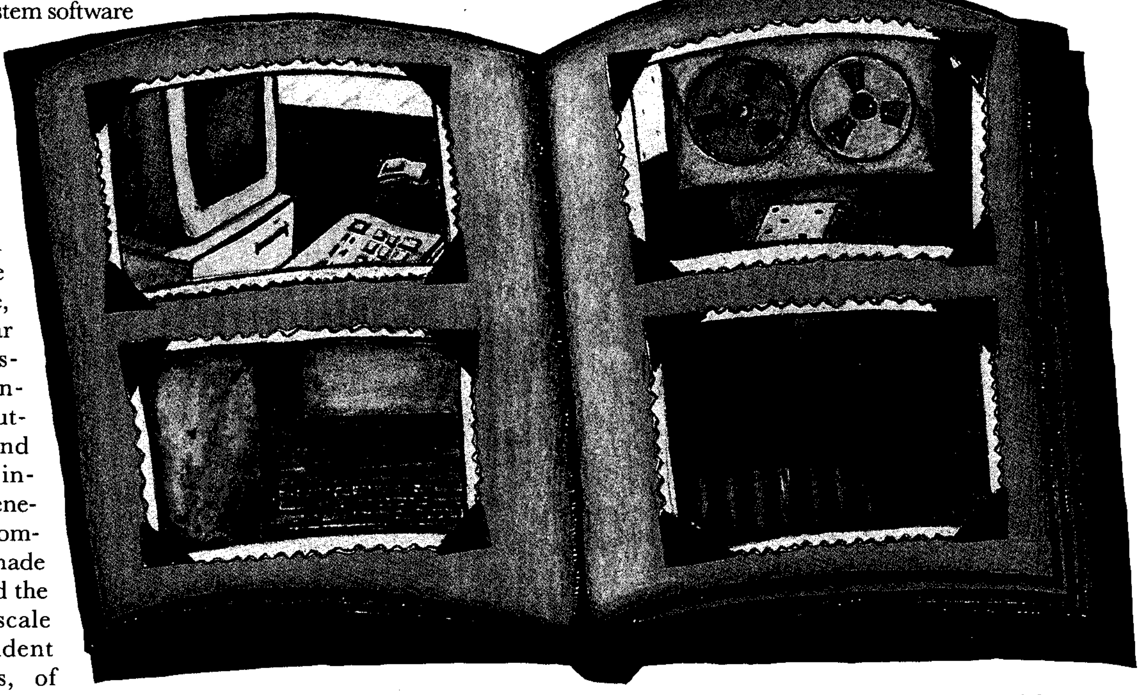
The Long-Term Future of Operating Systems



Major changes in system software are slow in coming. A recent landmark has been the emergence of Unix as the first major machine independent operating system. In 1967, when I gave my Turing lecture, it was already clear that operating systems would eventually follow computer languages and become machine independent. Some penetration of the minicomputer world was made in the late 1970s and the 1980s by small-scale machine independent operating systems, of which TRIPOS written in BCPL was perhaps the best known. Unix, which has come to the front in the workstation world, is the first machine independent operating system to impact the industry in a major way.

Workstations were not a development of the personal computers that came before them. The engineers who developed workstations were from the minicomputer side of the industry and few of them had had anything to do with PCs. It is for this reason that workstations have Unix for their operating system rather than MS/DOS, or something compatible with it.

In any case, MS/DOS would have lacked features essential to high-end users—for example, file protection and multithreading—and they would have been infuriated by the restriction to eight-letter file names. On the other hand, it was the very simplicity and lack of frills that constituted the real strength of MS/DOS in its own proper sphere.

Other operating systems for desktop computers are available or becoming available. OS/2 has existed for some time and a new one, Microsoft NT, is in beta test. Microsoft NT is intended for the business user who is expected to have at least an Intel 386 processor with 2MB of memory.

A wide variety of smaller machines are in use. Many of them were bought for use at home. Some are compatible with an IBM PC and some are not. The smaller ones have no fixed disk. Most of these small computers run MS/DOS. The same is generally true of the laptop and even smaller computers now on the market. Their power is in many cases remarkable but, compared with desktop models, they will always be limited by weight, size, and battery power.

In spite of their different evolutionary histories, workstations, PCs, and portable computers now form a continuous range. The use of different operating systems divides this range in an awkward way. As the major operating systems become available on a wider range of computers, the situation will improve as far as the larger PCs and workstations are concerned. However, we can hardly expect that any of these operating systems will run on the entire range, from laptops to well-endowed workstations. Nevertheless, as a long-term aim, we might hope for a range of operating systems that are in some sense compatible, or at least friendly to one another. Here is a challenge for those interested in operating system research.

Maurice V. Wilkes

ILLUSTRATION: ANNA CICA

A Family of Operating Systems

The early operating systems were thought of as monolithic programs for scheduling jobs, guiding the computer operators in their work of mounting tapes, and so forth. We later came to think of an operating system as composed of a set of procedures for scheduling, file management, spooling, etc. Now, with the need for operating systems to be machine independent, we must learn to think, not in terms of code, but in terms of a set of interfaces on which working software can be based.

The challenge is to design a series of interfaces that can form the basis of a family of operating systems, with members suitable for use over the whole range of machines from PCs to workstations. The relationship should be a close one so that migration of files and programs would be possible.

There do not appear to be insuperable problems in providing for the migration of files. The full specification of a filing system might include file protection and the retention of a full updating history. In a lower-level system, these features might not be provided, but that would not prevent the files themselves from being copied; it would simply mean that some attributes would be lost. Similarly, when a file was being copied in an upward direction, default values for some attributes might have to be supplied.

Easy movement of programs in the downward direction would make life easier for the application programmer. They would be able to develop applications on a workstation equipped with a full operating system and then package them to run on smaller computers. It is perhaps fair to comment that, in the past, writers of operating systems have been more sensitive to the needs of system programmers like themselves than to the needs of application writers.

Renewed Problems with Paging

In the 1970s, efforts to improve memory management loomed large in the search for run-time efficiency, and elegant paging algorithms, based on the working set model, were devised. As larger high-speed memories became available, many of the problems which this research set out to solve went away. For a time it seemed that paging systems were both well understood and highly effective. However, with the continuing progress being made in the semiconductor industry, high-speed memories have become steadily faster, and the gap in access time between disks and high-speed memories has widened. For that reason, paging is now much less effective than it used to be.

A consequence is that users of high-end engineering workstations now demand very large memories—32MB or more—so that fewer disk transfers are necessary. However, this is not a complete solution, even at the top end. There is an urgent need to reexamine memory management at a fundamental level and to revisit a number of issues which it once seemed were finally settled. These include the choice of page size, the role of prepaging as compared with demand paging, and the exploitation of techniques for improving the locality of information, so that fewer pages need to be loaded into high-speed memory. Perhaps more radical approaches to memory management should be considered, especially those in which full advantage is taken of the very high speed at which blocks can be transferred when the words within them are accessed in the order in which they are stored.

Every time the feature size on a memory chip is halved, the capacity is quadrupled. If we assume a further shrinkage factor of eight—something that is not impossible in the long term—a high-speed memory composed of a given number of chips will hold 64 times as many words as at present. The ratio between disk capacity and high-speed memory capacity commonly found in today's PCs and workstations is of the same order. We may therefore see the role of the disk as a swapping device coming into question. However, we have a long way to go before memory chips of this density become available.

Need Operating Systems be so Large?

New features in operating systems are proposed because they are seen as conferring benefits. Unfortunately, too many features defeat their object. They increase the amount of material that the user has to master, and they increase the complexity of the code, with possible repercussions on the responsiveness of the system. At the present time the designer has no objective criteria to help in deciding what features to provide and what to reject. Until recently, a similar problem faced the designer of a processor. This situation has now been completely transformed by the development of powerful simulation techniques that enable an objective estimate of the performance of a processor to be made at the design stage. Simulation soon showed how poor a guide the designer's unaided intuition could be. Unfortunately, no comparable techniques are available for objective estimation of operating system performance. The difficulties in the way of developing such techniques are not to be underestimated. Nevertheless the subject is to be commended as one in which research could pay good dividends.

There have long been advocates of lean programming languages. There, the object is to keep down the size of the compiler and run-time system, and at the same time keep down the amount of information that a user must hold in his head. As a rough and ready way to achieve this, Tony Hoare made the practical suggestion that a limit should be set to the length of the programming manual; he suggested 50 pages. The designers of Modula-3 took up this challenge, and were able to provide a complete user description of the language practically within that limit. Could this idea be tried for an operating system?

However, it would be a mistake to say that the tendency for operating systems to become large and need a lot of memory is solely due to the use of faulty design and implementation techniques. There are also powerful technical forces at work. One of these

CONTINUED ON PAGE 112

CONTINUED FROM PAGE 24

products. In Cuba, a successful piece of software creates a problem for its authors, since they must furnish copies of the program, documentation and support free of charge—all of the headaches, but none of the benefits.

Whither Cuban Informatics?

As is the case with the former Soviet Union, there are now two parts to the Cuban computing community: the state-run sector that is rapidly falling apart; and a fledgling "mixed sector" of private, state, foreign, and black market activities [3]. Both Cuban sectors are in even worse shape than their problem-plagued ex-Soviet counterparts because the Cuban domestic market is incomparably smaller, Cuban government control on individual and organizational activities is more restrictive, and the Cubans are much more isolated internationally. But the potential exists for faster change in Cuba than has been possible in the ex-USSR, since: 1) tremendous authority rests in the hands of Castro and, for better or worse, the entire nation will or will not move depending on the decisions and fate of this one man; 2) the Cuban population is much smaller and homogeneous, and likely to obtain relatively more effective foreign aid, investments, and markets; and 3) much of the potential for such aid, investment, and market exists in a sizeable and fairly well-to-do (and sometimes resented) expatriate community 90 miles away.

Acknowledgments and Pointers

The Center for Automated Interchange of Information (CENIAI) of the Cuban Academy of Sciences has been linked to the Internet for roughly a year. At least twice a week, they receive a dial-up call from Toronto for two-way UUCP-based data transmission of files. Their Canadian partner is Web, part of the Association for Progressive Communications. In addition to its own use, CENIAI acts as a hub, receiving UUCP traffic from the University of Havana, the Ministry of Higher Education, and a Unix-based PC belonging to the main UJC computing club in Havana, which is supposed to become a collection point for traffic from other clubs. Unfortunately, the poor international telephone lines make this a fragile, slow, and expensive process. However, the email link through CENIAI (ceniai.cu) has been by far the most effective means we have had of communicating with the island.

CENIAI is also Cuba's connection to IASnet. IASnet is a joint network for (in many cases, now former) socialist countries operated by VNIIPAS (The All Union Scientific Research Institute for Applied Computerized Systems) in Moscow. VNIIPAS has multiple links to Western data networks using X.75. A link to Sprint (previously known as Telenet) has existed since at least 1988. When we visited CENIAI we were able to connect to one of our computers in Arizona—Havana to Moscow to Helsinki to Virginia to Ohio to Tucson—through four different networks.

CENIAI also provides electronic access to business, biomedical and other databases. A more detailed discussion of CENIAI's activities may be obtained by anonymous ftp from the global_net directory at dhvx20.csudh.edu [8].

Our trip would not have been possible without the help and hospitality of many Cubans who graciously accommodated our requests during Informatica '92 and during our visits to the Cuban informatics community at large. To all of those whose kindness and friendship made for an informative and memorable visit to Cuba, we extend our grateful thanks.

References

1. Cuba, sede de laboratorio contra virus informaticos. *Granma* (May 6, 1992), 16.
2. Geipel, G.L., Jarmoszko, A.T. and Goodman, S.E. The information technologies and East European societies. *E. European Politics and Societies* (Fall, 1991), 394–438.
3. Goodman, S.E. and McHenry, W.K. The Soviet computer industry: A tale of two sectors. *Commun. ACM 34*, 6 (1991), 25–29.
4. Gunn, G. Cuba's search for alternatives. *Current History* (Feb. 1992), 59–64.
5. Luxner, L. Cuban workers will be paid—if they produce. *J. Commerce and Commercial* (June 15, 1990), 1A.
6. Luxner, L. Out of order—indefinitely. *Telephony* (Jan. 7, 1991), 17–18.
7. Pearson, J. and DeGeorge, G. Mr. Castro goes to market. *Bus. Week* (Apr. 20, 1992), 46–47.
8. Press, L. and Snyder, J. A look at Cuban networks. *Matrix News, 2*, 6, Matrix Information and Directory Services, Austin, Tex. (June 1992), 1–2.

Readers are encouraged to send comments, suggestions, anecdotes, insightful speculation, raw data, and submissions for guest columns on any subject relating to international aspects of the information technologies. All correspondence should be addressed to:

Sy Goodman
MIS/BPA
University of Arizona
Tucson, AZ 85721 **or**
goodman@mis.arizona.edu **or**
fax: (602) 621-2433

CONTINUED FROM PAGE 29

I have already mentioned, namely the increasing unbalance between the access times of disks and high-speed memory. No doubt there are others.

Intensive research on operating system performance is badly needed. We cannot continue in a world in which the last ounce of speed is squeezed from the silicon, while operating systems are subjected to little critical scrutiny. Efforts made now to secure a better understanding of the underlying issues will begin to pay off in the first decade of the next century.

Maurice V. Wilkes *received the ACM Turing Award in 1967 and is the author of* Memoirs of a Computer Pioneer, *MIT Press, 1985.*