The Lawrence Livermore Laboratory's awesome collection of hardware, organized and connected to serve some thousand users

The Octopus Computer Network

by John G. Fletcher

This paper is a description of the Octopus computer network at the Lawrence Livermore Laboratory of the University of California (LLL).* It is intended that additional papers will follow describing certain aspects of the network in more detail.

Why should anyone outside of LLL be interested in the Octopus network? There are at least two reasons. First, the network consists of one of the largest concentrations of computing power and information storage capacity in the world; while other systems may possess capabilities similar to those at LLL, those capabilities usually have not been implemented on the same scale. Second, the experience at LLL has led to conclusions that in many ways differ from attitudes that seem to be most prevalent in the current literature. Thus, knowledge of that experience should be helpful in evaluating those attitudes.

The Lawrence Livermore Laboratory is operated by the University of California under a contract from the United States Atomic Energy Commission (AEC). It is engaged in research and development in various disciplines, including explosives, controlled thermonuclear reaction, and biomedicine. Actual experimentation and testing, particularly in the case of nuclear explosives, is often quite expensive, time-consuming, and heavily involved with political and legal considerations. Therefore, it is desirable that as much work as possible be done by simulation on large digital computers. To fill this need, LLL has acquired a considerable inventory of computer hardware.

*Work performed under the auspices of the U.S. Atomic Energy Commission.

Hardware

The LLL computer hardware is always in a state of more or less constant change. Faster and larger-capacity equipment is acquired as it becomes available, replacing older, slower, and smaller equipment. For example, new large computers arrive at intervals of about 12 to 18 months. In many cases (such as the CDC 6600 and 7600), LLL has been the purchaser of the first machine of a new kind.

At the present time, LLL has eight major processors: a Xerox Sigma 7, two DEC PDP-10s, two CDC 6600s, and three CDC 7600s. A CDC STAR-100 is on order. Each of these machines has, associated with it, on-line I/O equipment (e.g., tape transports, impact printers, film output), primary (core) storage averaging about 16 million bits per processor, and secondary (rotating) storage averaging about 5 billion bits per processor and having transfer rates up to 34 MHz. In addition to these major processors, there are many smaller processors, chiefly DEC PDP-8s and PDP-11s, but also others of various manufacture.

Supplementing the processors and their immediately associated peripherals are a number of other facilities shared among the processors by means of the network described below. These facilities include an IBM Data Cell (over 3 x 10^9 bits) and an IBM Photo-digital Store (over 10^{12} bits). There are also assorted interactive terminals (including Teletype units, television monitors, and Evans and Sutherland LDS-1 displays), an ultra-high-speed printer (currently off-line) that operates at 30,000 lines per minute, and a microfilm recorder (currently off-line).

This brief listing of LLL computer hardware raises certain questions:

1. Why does LLL need so much big, fast equipment? We need it because the scientific personnel of the laboratory can usefully employ it in accelerating their work and improving its quality.

Much of the calculation done consists of large numerical programs that solve partial differential equations by finite difference, Monte Carlo, or other such techniques. Typical problems will run several hours on a 7600, producing the equivalent of about 10 complete tape reels of output, which is then edited into a stack of printouts several inches thick. Astonishing as some may find it, the responsible scientist actually is interested in the result and benefits from scanning it.

2. Why does LLL buy from so many different manufacturers? We do this because whichever manufacturer builds the best (most capacity per dollar) hardware depends upon the kind of hardware and its purpose. The advantages of using a uniform brand of equipment are outweighed by the advantages of flexibility.

The network

We realized some years ago that the growing inventory of hardware could be best utilized if most of it were interconnected in some way. The benefits were chiefly in four categories:

1. All remote terminals, both interactive (such as Teletype terminals) and noninteractive (such as line printers), could communicate with all the major computers; there would not be one set of terminals for use with one computer, another set for use with a second computer, etc.
2. All major computers could use unique equipment (for example, the ultra-high-speed printer and the Photodigital Store).
3. A single data base could be shared among the major computers, eliminating the need for multiple copies and/or manual transport of information.
4. More sophisticated forms of cooperation among the major computers would be possible, such as their working jointly on a single problem.

Thus was born the concept of the Octopus network.

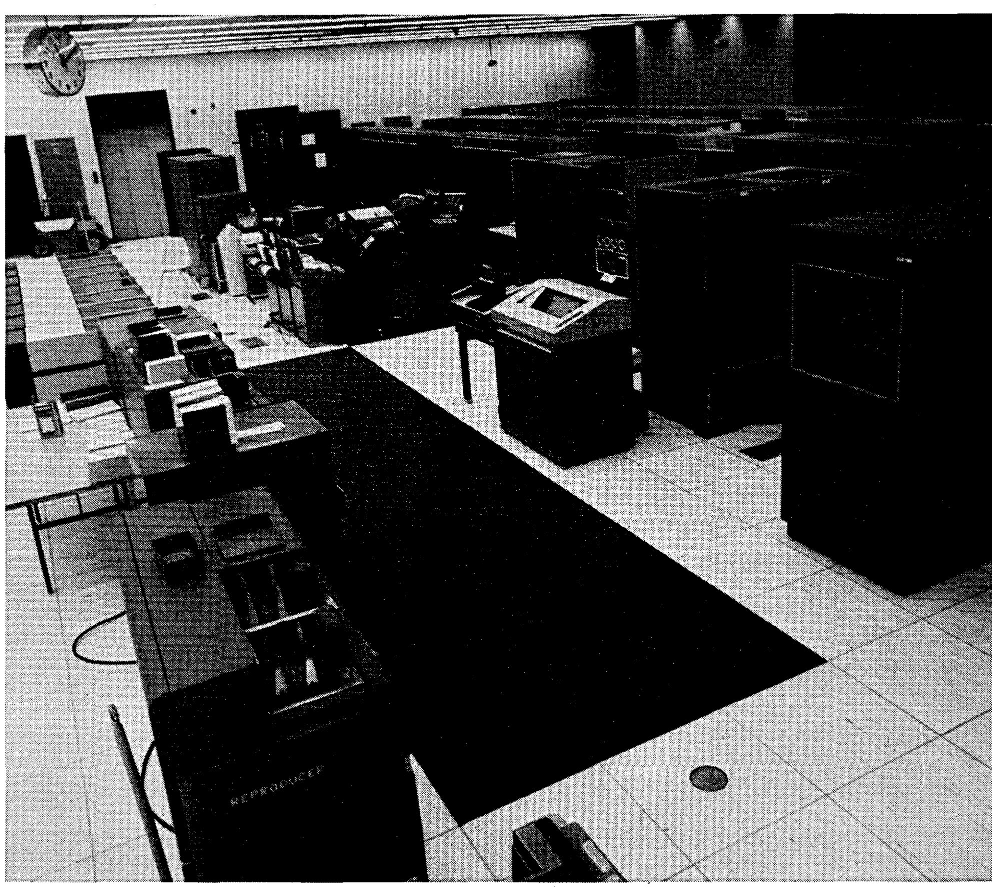
As the name implies, the original conception of Octopus was of a single, moderately large computer (the "head") that would manage the information flow among its "tentacles," namely, the large computers ("workers") that execute user programs, the remote terminals, and the central storage devices. One important thing that has been learned is that this was a bad idea. Such a scheme makes the network excessively vulnerable to difficulties with a single component, the head. The likelihood of difficulties with the head is increased by the fact that each new kind of equipment must be attached to it, necessitating a period of hardware and software debug and shakedown. The disruption caused by a single addition to the network might thus affect all existing components.

The present design of Octopus is more complex and decentralized. (See Fig. 1.) It is a superposition of several subnetworks, each having a specific function (or functions). A typical one of these subnetworks consists of a small computer called a "concentrator," which is the head of the subnetwork. The concentrator is joined to each of the large worker computers and also to whatever I/O gear, terminals, or storage devices are appropriate to the function of the subnetwork. In practice, each of the subnetworks departs in some ways from this typical plan, but the basic idea is maintained. Present subnetworks include the file transport network (for moving files among the worker computers and between those computers and the central storage facility), the teletypewriter network, and the remote I/O network (for operating remote card readers and line printers).

The interfaces that join the various components of Octopus are designed, installed, and maintained by LLL engineers. One reason for not relying on computer or peripheral manufacturers to provide this equipment is that we have found that they are not at their best when interfacing their own equipment to that of another manufacturer. Another reason is that in-house engineers are more willing to tailor their designs to software needs and to make changes in those designs as experience suggests the possibility of improved performance. Most importantly, in-house design provides for a degree of uniformity throughout the network, which largely overcomes the difficulties arising from the use of equipment made by various manufacturers.

Each new concentrator added to the network is provided with a multichannel bit-parallel interface (or sometimes several interfaces transferring at about 10 MHz.) These interfaces are called either "adaptors" or "line units" (for obscure reasons). They communicate with one another by an LLL standard protocol. Thus the interface on a new concentrator can be plugged into an unused channel of the appropriate adaptor or line unit on each of the worker computers and, with minimum disruption, can become part of the network. The connections between the concentrators and remote terminals are LLL-designed serial interfaces, either asynchronous (for example, to Teletype terminals) or synchronous (for example, to remote printer/card reader stations), as appropriate.

The present Octopus organization,



Some of the equipment at the Lawrence Livermore Laboratory computer complex: right center (with window) is part of the photodigital storage unit, left center are eight disc drives controlled by a PDP-10, behind the teletypewriters, and to the right are PDP-10 memories, PDP-8 concentrators, and assorted interfaces, including those for other computers in the network.

just described, has proven quite satisfactory and flexible. Adding a component identical with an existing one usually requires no changes except the making of new entries into some software tables. On the other hand, an entirely new facility is first brought up and debugged off-line. It is next attached to one worker and the communication link is debugged; any disruption this occasions is thus limited to a single worker. When all is satisfactory, the new facility is then fully joined to the network.

Worker computers

The worker computers execute programs selected (and very often written) by the network users, who are scientific, clerical, and administrative personnel of LLL. The computing needs of these users are quite eclectic, involving not only numerical calculation, but also program generation, information retrieval, editing, etc. Accordingly, the worker computers (and the entire network) are organized essentially as a computer utility. The user may run any program that is available to him, and his behavior is predictable only statistically. The worker operating systems are designed for time-shared interactive use, together with a batch-like facility used mainly at night. (We assume that the advantages of interactive computing are so obvious as to require no justification here.)

At LLL a clear distinction is always maintained between "system" programs and "user" programs. System programs include all programs outside the worker computers and those programs in the worker computers that operate with special privileges. The functions of system programs are to store, locate, and retrieve files; direct the flow of messages through the network; schedule the execution of, allocate storage for, and treat the requests of, user programs; manipulate I/O devices; and handle accounting and security matters. In short, they do anything that, if a user program were allowed to do it, could gain special privilege for the user and/or disturb other users.

User programs run only under the limitations allowed by the system programs and are restricted in the core they can access and in the instructions they can execute. Any requests they make to the operating system (for file access, I/O, etc.) are never trusted and are checked for legality. The integrity of the system is not based upon any assumptions about the kinds of programs that the user will choose to run or about his degree of ignorance regarding how the system operates. Adherence to these rather obvious principles solves the software security problem. It should be noted that certain programs (called "utility" programs) that are often thought of as being system programs—such as compilers, interpreters, editors, information retrievers, and debuggers—are, in fact, user programs, although they are often written by the same people who write system programs.

The worker computers in the Octopus network consist of a CDC 6600 and all three 7600s. The Sigma 7 is soon to be added, while the other 6600 is, for various reasons, stand-alone. As is discussed below, the functions assigned the two PDP-10s are quite varied and include the worker function.

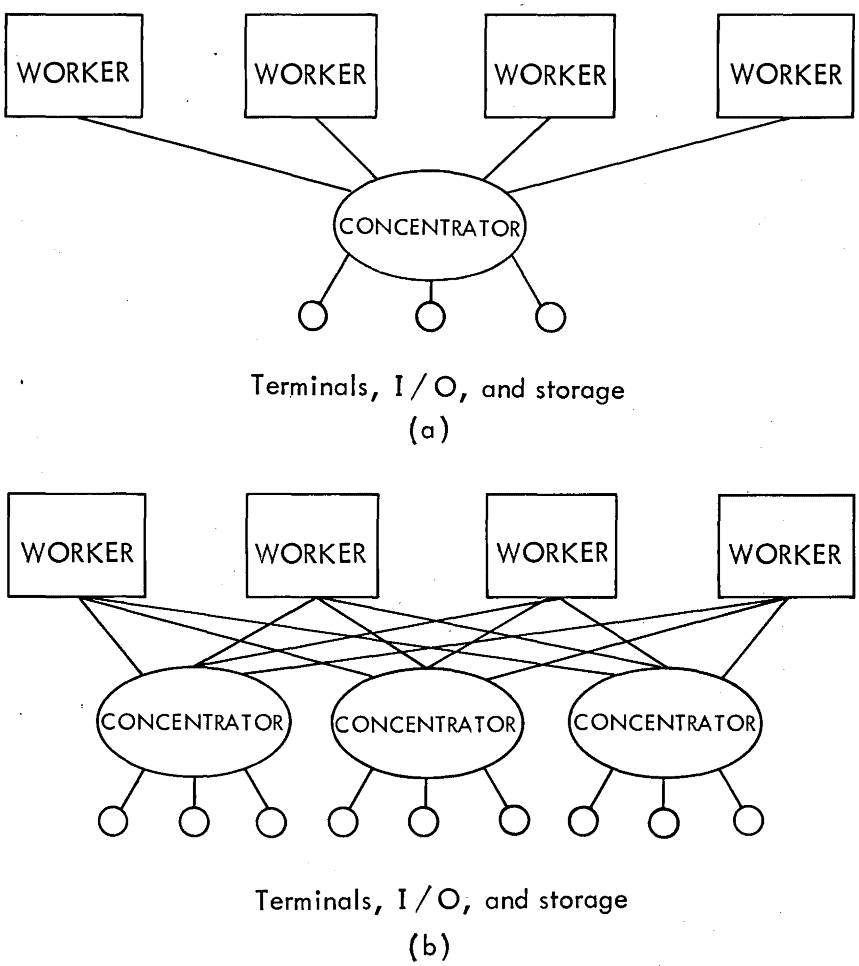


Fig. 1. (a) A typical Octopus subnetwork. (b) The complete Octopus network, composed of superposed subnetworks (only three of which are shown).

Interaction

A user communicates with the workers through interactive terminals. With appropriate input, he selects the programs he wishes to run and supplies directives and (perhaps) data to those programs. Anything he wishes to do (calculate, compile, manipulate files, etc.) is done through the agency of a suitable user program, and there are simple rules that distinguish messages intended for an executing user program from those intended for a worker operating system. The interaction might be viewed as acting on three levels: the user, the user program, the system. In many cases, an input from the user is interpreted by his program, which then generates a request to the system to perform the desired activity. While the format of the system request is necessarily standardized and rigid, the format typed at the terminal depends upon the design of the user program and is infinitely flexible.

Currently, most interactive terminals are Teletype Model 33s. The teletypewriter network has four concentrators. Each concentrator is a PDP-8 (or PDP-8L) with 8K of core that multiplexes 128 Teletype terminals. About 400 of the 512 Teletype terminals thus provided for are in use.

The teletypewriter network operates in store-and-forward mode. A concentrator collects an input message character by character until it is complete and then forwards it to the appropriate worker. (A complete input message is in most cases the same as a single typed line.) Similarly, entire output messages are accepted from the workers and then printed character by character. The teletypewriters operate full duplex;

input is printed only because it is "echoed" by the concentrator. The user can begin typing while output is in progress, in which case the echoing is delayed until the output finishes; in the same way, output that arrives while input is taking place is held and then printed after the input is complete. Erroneously typed input may be corrected (before the message is ended) by canceling single characters or an entire message.

Since worker computers do not know of an input message until it is complete, a user cannot interact character by character with a worker program. While undoubtedly there are many situations in which such interaction is valuable, we decided that those situations are not numerous enough and/or the performance improvement not great enough to justify the added burden on the system. Character-by-character interaction would increase message-handling activity manyfold and therefore would impair other service and perhaps require investment in additional message-handling facilities.

A second interactive terminal network is just now being put into use: KIDS (keyboard interactive display system). The concentrator is a PDP-11/45. Each of the terminals, which are built to LLL specifications, consists of a keyboard for input, a character and vector generator to convert output into a raster (which is stored on a scan converter and then used to refresh a monitor), and a joy-stick for positioning a cursor. Operation, except for cursor movement, is full duplex. This network will offer services beyond those of the teletypewriter network, including greater speed, vector as well as character output, and more comprehensive local text-editing. The initial order is for 48 terminals.

The two interactive terminal networks have connections to one another and to other subnetworks in such a way that there are always at least two paths from a given concentrator to a given worker; this reduces vulnerability to hardware failure. Each input or output message includes an Octopus standard heading that gives source, destination, and other information about the message.

It is also a network standard that all characters be represented as ASCII contained in 8-bit bytes. ASCII is used not only for communication but also for storage, since there seems no reason whatsoever to pay for software or hardware conversion or to suffer from the confusion of multiple codes. ASCII was chosen because LLL assumes that government and industry mean what they say when they define a standard; it would be unfortunate if this view were naive. The 8-bit unit was chosen on the basis of the laboratory's estimate that the trend toward computer word sizes divisible by eight will continue. Of course, certain older items of hardware do not use ASCII and must be serviced by software conversion, but these devices are being gradually phased out.

Files

By "file" is meant a quantity of information kept by the system for a user and known to that user by a symbolic name. All editing and information retrieval operations accessing portions of files are done by user programs rather than by system programs. However, files are located, stored, and retrieved by the system in response to requests of user programs. In considering such a request, the system will look for the symbolic name only in lists of files private to the requesting user, files shared by that user and other users (by mutual consent), or public files available to all users; this it is impossible for a user even to describe to the system a file to which he has no right.

Each worker computer has its own filing system separate from the central "Elephant" filing system. The Elephant system is managed by one of the PDP-10 computers, and the Elephant files are kept on the rotating storage of that PDP-10 as well as on the Data Cell and the trillion-bit photostore.

The recording medium for the photostore is ordinary silver halide film.[1] Thus the storage is nonerasable and archival. Each piece of film (called a "chip") measures about 1½ by 3 in. and holds about 5 x 10^6 bits. The chips are kept in small plastic boxes (called "cells"). There are 32 chips per cell and nearly 7000 cells in the photostore, giving a grand total of over 10^{12} bits on-line. The photostore includes facilities for writing on the chips (with an electron gun), developing the chips, moving the cells pneumatically to and from the storage area, and reading the chips with a "flying spot," all under computer control.

The PDP-10 is the concentrator of the file transport network. Requests for file movement or for other operations involving Elephant files are sent to the PDP-10 from the workers, either directly from their operating systems or via those systems from user programs. (These request messages and the replies to them are routed through the interactive terminal networks.) The PDP-10, queues these requests on the basis of algorithms intended to optimize the use of the Elephant storage media. When it is the appropriate time to move (a copy of) a file to or from a given worker, the PDP-10 enlists the aid of the operating system of that worker, and the transport takes place over the high-speed bit-parallel interfaces of the file transport network. The PDP-10 uses high-speed head-per-track rotating storage and very large core buffers to reduce the effects of the varied latencies and transfer rates of the storage media

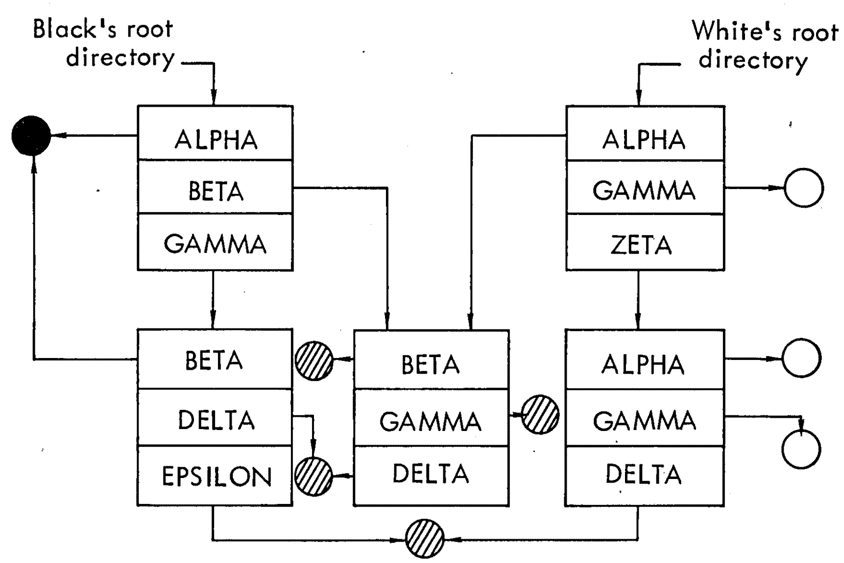


Fig. 2. Schematic representation of a file directory structure for two users (Black and White). Each rectangle represents a directory containing three entries. Each entry associates a symbolic name with a pointer to a file, which may be either another directory or a simple file, represented by a circle. Note that the directories on the left and right of the figure are private to Black and White, respectively, whereas the one in the middle is shared. Private data files are represented by black or white circles; shared data files are represented by shaded circles. Note that the same symbolic name may appear in different directories referring to different files, while the same file may be listed in two directories under different names.

[1] Kuehler, J. D., and Kerby, H. R., "A Photo-Digital Mass Storage System." *Proc. AFIPS 1966 FJCC*, Vol. 29, pp. 733-742.

in the network.

The Elephant system is organized around a directory structure similar to that used in the MULTICS system.[2] A directory is a special kind of file consisting of a number of entries; each entry associates a symbolic name with a pointer that locates the named file in Elephant storage. Since the file named by an entry in one directory may be another directory, a tree-like (more accurately, directed graph) structure is generated; the terminal nodes of this structure are simple (program or data) files. (See Fig. 2.) The files accessible to a given user are those that can be reached by a path through this structure starting at a particular directory, that user's "root directory." Means are provided for users to give pointers to parts of their structures to other users; this provides for very general and flexible file-sharing arrangements. This flexibility is enhanced by the fact that the kinds of access that may be made to a file can be a function of the path taken through the structure to access that file; for example, one user may have full access to a file, while another user can only read it but not write into it.

The PDP-10 is quite a bit larger than the other concentrators in the Octopus network. For economic reasons, therefore, it has been given a few functions other than central file management, even though this violates the ideal of one function per subnetwork. The PDP-10 manages a system of disc-refreshed output displays (not associated with keyboards) called the TMDS (television monitor display system). Control messages and data for the TMDS are moved either by the file transport network or by one of the interactive terminals networks, depending upon the quantity of information involved. User programs requiring a TMDS monitor must first send a request to the PDP-10 specifying the monitor desired. There are more monitors than there are refresh channels on the disc; PDP-10 software must allocate a disc channel and then operate a 16 x 128 (soon to be 32 x 128) crossbar switch to attach the channel to the monitor.

The second PDP-10 processor was originally intended as a spare and for use in engineering checkout of new equipment. However, it now also functions to operate the high-performance Evans and Sutherland LDS-1 (two-terminal) display, which is designed to interact with a PDP-10. This has required that the PDP-10 become in effect a worker computer with a time-shared interactive operating system. Actually, the two PDP-10s operate as a dual-processor computer, sharing a common memory and common software, although each processor has different duties.

[2] Daley, R. C., and Neumann, P. G., "A General-Purpose Filing System for Secondary Storage," *Proc. AFIPS 1965 FJCC*, Vol. 27, pp. 213-229.

Data collection

A PDP-8L serves as concentrator for the data-collection network. It is attached via asynchronous serial lines to other, remotely-located, small computers, each of which controls and monitors one or more experiments. Real-time data collected from these experiments is sent to the concentrator, from which it is relayed to the PDP-10 for storage in the Elephant filing system. Data may also move in the other direction. This network is joined only to the PDP-10 and not directly to the other workers; collected data can be moved to the workers by means of the file transport facility.

Remote I/O

The somewhat-inaccurately named remote job entry terminal (RJET) network uses a pair of PDP-11/20s as a concentrator. (An earlier acronym, RIOT—remote I/O terminal—met official opposition.) One of the PDP-11s is attached to the workers, and the other is attached to the remote terminals; the two PDP-11s are intimately joined, both in hardware and in software. A remote terminal is a PDP-8L computer that operates a line printer and a card reader. The connection from the concentrator to the remote PDP-8L is by a 4800-Hz synchronous line. Communication over this line conforms to ASCII communication control protocol. Input and output over this network are to and from files.

High-speed output

Hardware now on order will bring the ultra-high-speed printing facility (currently off-line) into the Octopus network. The new computer hardcopy output recording system (CHORS) subnetwork will include not only a pair of 15,000-line-per-minute nonimpact printers with graphical as well as character capability but also the microfilm recorder (currently off-line).

It seems clear that the future will see still more facilities added to the network, as dictated by growing need and technological advance. The flexibility of the Octopus is such that it should be able to accommodate all foreseeable additions. Ideas now being considered include a centralized (perhaps mechanized) magnetic tape facility and the acquisition of a 10^{13} (or more) bit store.

Atomic Energy Commission regulations are very fussy about "need-to-know," a concept essentially the same as privacy. That is, even though all LLL employees have passed a federal security check, each employee is permitted to access only that classified information necessary to his work. Thus, Octopus must see to it that each piece of information in the network is available only to the user (or users) authorized to access it. It should be clear that this is not difficult in view of treatment of user programs and of file access, as described above. The LLL system programmers generally are puzzled by the view that there is a serious "problem" in regard to software security. The only real difficulty, as in all programming, is the possibility of careless oversights and coding errors.

One issue not yet discussed is that of how the Octopus knows which user is using a given interactive terminal. The user of course "tells" the system by typing his "name" when he logs in. (His "name" is actually his six-digit employee number.) But how can the system be sure that it is not someone else typing the "name"? The user must also type a six-letter combination (or password) known only to him and the system. The concentrators do not echo the combination when it is typed, so it is not visible on a screen or teletypewriter paper. In fact, the user is forbidden (by government regulation) to write down his combination or to reveal it to another. Currently, each worker computer must keep a list of all combinations. In the near future a special (PDP-11/20) computer will be added to the network to do all verification of combinations; at that time it will become the only repository of the combinations.

Software

All the system programs and most of the utility programs used are written at LLL. In particular, all operating systems are LLL-designed and maintained. There are at least four reasons for this:

1. In some cases, LLL has obtained a computer and put it into use before the manufacturer has had time to create sufficient software. The CDC 6600 is a notable example.

2. Commercially supplied software does not exist for many of the needs at LLL. The small network concentrators are engaged in specialized tasks not foreseen by their manufacturers. In the case of the larger computers, the standard operating systems are ill-prepared for network connections, directory-oriented filing systems, and other unusual facilities.

3. The security and privacy offered by standard software systems do not appear to meet AEC standards. As remarked above, the reason for this situation is a mystery.

4. It is much easier to innovate and

to develop new facilities suitable to LLL needs when not constrained by formats and protocols designed for other purposes.

There is, of course, the serious disadvantage that it is difficult to import and export programs. This problem should not lead to the conclusion that every operating system must be the same, since further development in system design would then surely stagnate. Perhaps the best hope lies in creating standard formats for requests to operating systems, although even this has the danger of freezing the state of the art.

To design and implement its software, LLL maintains a staff of about 30 system programmers. The same people do design and implementation; there is not a two-level structure of analysts and programmers. The laboratory has apparently decided that a system programmer who has to have everything charted out for him requires more input effort than he yields in output, while the designer who does not program resembles a mathematician who does not add. Typically, a programming group faced with a new machine to program breaks the job into as many parts as there are programmers. Small computers are usually programmed entirely by one person. Arguments that this approach should not work must contend with the fact that it does.

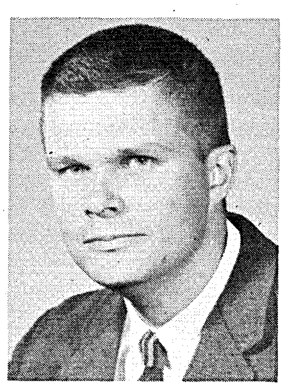
The CDC computer systems at LLL all have been programmed in an LLL-designed, FORTRAN-derived language called LLLTRAN[3], which has facilities and power comparable to PL/I. The other computers, including all the concentrators, have been programmed in assembly language. The choice between higher- and lower-level language has usually been based on the personal preferences of programmers. In spite of allegations in the current literature, LLL experience does not justify the view that the systems written in assembly language are somehow inferior. There is no evidence to suggest that they took longer to write, are more prone to error, or are more difficult to follow than if they had been written in another language.

Similar remarks can be made about another controversy now raging: the "go-to" issue. Until the question first appeared in the literature, it apparently never occurred to anyone at LLL that there might be something "wrong" with the go-to construction; and, now that the question has appeared, there is nothing in LLL experience to support the view that the go-to does lead to difficulty. No one can deny that go-to's can create problems for compiler designers and property-of-program provers. However, the contention that they lead to error and make programs difficult to understand and debug is a personal judgment that reflects more on the quality of programmers than on anything else.

Users

This exposition would be less than candid if it suggested that the thousand or so LLL users are uniformly ecstatic over the Octopus system. The fact is that, even at LLL, resources are limited. Therefore, when it has been decided to do one thing for the users, it quite often means that it becomes difficult or impossible to do another thing that some users would like to have done. For example, one cannot keep adding new facilities and at the same time not require the user to learn new ways of doing things. One cannot create complex, time-shared networks and completely preserve that intimate, "hands-on" feeling. One cannot offer hundreds of users the ability to output changing pictures at movie speeds without everything else grinding to a halt. And so forth. The best gauge of the success of the network is probably that no addition to it has ever been removed so that the older way of doing things could be reinstated.

Acknowledgments. Although the present author has done his fair share in designing and implementing Octopus, he is now playing the role of Boswell. There are so many people who have contributed to the development of the system over the years that it would be dangerous to select a few for special mention. I will therefore name only Dr. Sidney Fernbach, the head of the LLL Computation Department, who somehow has held the whole thing together for nearly a decade. □



Dr. Fletcher is the leader of the Lawrence Livermore Laboratory Computation Department's network group, which designs and implements the software for Octopus concentrators. He is also a lecturer in applied science at the University of California, Davis. He has a PhD in physics from Princeton University.

[3] Mendicino, S. F., Hughes, R. A. Martin, J. T., McMahon, F. H., Ranelletti, J. E., and Zwakenberg, R. G., "The LRLTRAN Compiler." *Comm. ACM 11*, November, 1968, pp. 747-755.