"Time Sharing" is presented in its most general sense as any application of a computer system that involves simultaneous users. Concepts and equipment of time-shared systems are defined and described and criteria for system configurations are given in terms of application requirements.

FUNDAMENTALS OF TIME-

This is the second and concluding section of the article by C. Gordon Bell which appeared on pages 44 through 59 of the February issue of Computer Design. The first section discussed the hardware of time-shared computers and suggested advantages of time sharing. This section discusses operating system software and user components and includes an extensive bibliography on time sharing.

OPERATING SYSTEM SOFTWARE

Operating system, monitor, supervisor, and *executive* are names given to those processes that supervise and control the operation of the system for all users.

Unlike conventional operating systems that are static, a Time Sharing Operating system is growing and dynamic. New procedures may be added continuously.

The additional languages and facilities have a structure that may have a rather complex operating system as a major part of the language. For example, consider the administration of a teaching program. The program would undoubtedly schedule its users (pupils), and the hierarchy of the whole system would be: the operating system for the entire computer managing a central teaching program to manage all courses managing a course teaching program which would manage all individual users taking the particular course.

The objectives of the system software are:

1. Provide many user functions or facilities with easy-to-use processes.
2. Effective or efficient hardware utilization. Perhaps allow users to utilize the hardware directly. Provide special user services which utilize special hardware.

The criteria for the design might:

1. Meet the requirements for Time-Sharing (computer time and memory space) per user.
2. Provide for flexibility in the operating system using modular construction. Individual components can be independently designed, tested, and modified (or improved). If possible, the system components should be written as user processes.

In general, all systems are constrained by cost considerations. A special system may concentrate on a single objective, while a general system is forced to find a balance between many objectives.

The system software contains:

1. System data base, or information necessary for system management, and management procedures.
2. Resource allocation, control, and management procedures.
3. Common procedures or processes for the users, the library.
4. Miscellaneous elements: System initialization and shut-down; error recovery; file backup; creation of new system; and system debugging.

OPERATING SYSTEM DATA BASE

The operating system requires a large data base that is retained in primary memory and in files. Back-up files (copies of files) must be regularly written so that the system can be restarted in a correct state in the event of system failure.

The data for a user include: his memory map or process location, generally found in primary memory while running or active; the processor status (the location counter, processor flags, accumulators, index registers, etc.); identity information (name, number, project numbers, etc.); the time used, allotted, last run, etc.; the run state (e.g., presently running, waiting to run, requiring special service, waiting for file transaction, terminal action, additional memory, etc.); permanent user data to allow the assignment of terminals and file space; accounting information; system temporary storage to enact user requested procedures; and active terminal and file buffering storage.

SHARED COMPUTERS

by
C. GORDON BELL

Associate Professor of Computer Science and
Electrical Engineering, Carnegie Institute of Technology.
Formerly manager of Computer Design, Digital Equipment
Corporation, Maynard, Massachusetts.

In addition to the data base associated with each user there are inherent data associated with system components and resources. These include: hardware status and availability information; terminal names; file directories including descriptors of abilities, modes, etc.; primary memory free space; and file memory free space.

Historical, statistical, and accounting information are also kept, and historical or activity data provide tools for system improvement. They especially aid scheduling and memory allocation as well as indicate the system balance and load.

RESOURCE ALLOCATION, CONTROL AND MANAGEMENT

This responsibility includes: processor time or scheduling; process space (primary memory allocation) and assignment of a process to secondary memory or files; file space; and terminal/process/user allocation and assignment.

The two extreme philosophies that determine the number of users a system can have are "denied access" and "degraded service." "Denied access" provides for a fixed number of users, each of which will obtain a known or worse case response. "Degraded service" provides for more users and the service is at least inversely proportional to the number of active users.

Scheduling

The assignment of processors to processes is *scheduling*. The scheduling algorithms that compute the time a process is to run usually use the following input parameters: previous time used; memory space occupied; status of terminal or file data transmission; expected response time for the user; user information; and number of users.

The priority information available includes the user, his urgency, and willingness to pay. As economically realistic systems that charge for their actual uses come into existence, users will be able to get a broader range of service.

The round robin algorithm runs each user, in turn, for a fixed quanta of time, and when all users have been served, the process is repeated. If any user cannot run because he is waiting for input or output, or halted, he misses a turn. On completion of input or output the user is put at the head of the queue and run (subject to his allotted time).

The scheduling algorithm is a most subjective system component, and, therefore, might be written in a form that can be easily modified. How, when, and which components call the scheduler is also important.

Memory Allocation

Primary/secondary memory allocation occurs as users make demands for more space the system activates user processes. The hardware memory allocation scheme of Table 2 constrains the user map organization, and the process organization. This hardware constrains the user procedure with restrictions ranging from writing in interpretive languages; writing at particular addresses or using a convention determined index register as a base register; writing with no restrictions (over the basic machine); and finally providing a two-dimensional addressing space.

The memory paging-memory segmentation hardware will drastically influence future program structure and design. With two-dimensional addressing, the user is not required to manage primary memory, and is free to address data by two logical numbers rather than by physical numbers. (With such freedom, and ability one might expect a proportional cost.)

File Allocation and Control

File allocation and control are generally subject to extra-system constraints on the basis of user-size-restriction tables.

File allocation cannot easily be separated from detailed file management. The management includes the service of detailed user requests for data, while allocation

is concerned with broader control of all file space.

Hardware's View of Files. The hardware parameters that affect file organization are: the hardware access time for words or sectors of the file; the word or record transfer time; the size of the records transferred; the total file size; and the file failure rate.

Operating System's View of Files. The apparent file parameters are: the size of files; the number of users and number of files per user; the access time to segments of a file; the nature of addressing the file information (sequential or random accessing); the file index; and the file data buffering.

File activities can be divided into operations: naming, or declarations, inter-file manipulation, intra-file utilization, and file closing.

User's View of Files. Parameters associated with the directory or index of files for users provide a means of controlling a file's activity, flexibility, general usage, name, users, record of its activity, and actual location of the file components. File accessibility control for the user is on the basis of the originator (owner), group, and public. The modes of file activity include read/write, read only, execute only (a procedure), and denied access. Other information about file access includes creation date, number of times used, last time used, times modified, etc. The user requests of functions for utilization include: reading, writing, naming, re-naming, deleting, appending, inserting, providing access restrictions, obtaining statistical information, or in general, any operation that can be done with the data in or about a file.

Terminal Allocation

Terminal allocation in general systems is either on a first-come-first-served basis or on a completely reserved basis. Requests for terminal reservations are via a control terminal, and as a job is initiated, the terminals required for job completion are requested. The terminal is the means by which a process is

TABLE 2. MEMORY ALLOCATION METHODS

Hardware Designation	Method of Memory Allocation Among Multiple Users	Limits of Particular Method
Conventional computer — no memory allocation hardware	No special hardware. Completely done by interpretive programming.	Completely interpretive programming required. (Very high cost in time is paid for generality.)
1 + 1 users. Protection for each memory cell	A protection bit is added to each memory cell. The bit specifies whether the cell can be written or accessed.	Only 1 special user + 1 other user is allowed. User programs must be written at special locations or with special conventions, or loaded or assembled into place. The time to change bits if a user job is changed makes the method nearly useless. No memory allocation by hardware.
1 + 1 users. Protection bit for each memory page.	A protection bit is added for each page. (See above scheme.)	No memory allocation by hardware.
Page locked memory	Each block of memory has a user number which must coincide with the currently active user number.	Not general. Expensive. Memory relocation must be done by conventions or by relocation software. A fixed, small number of users are permitted by the hardware. No memory allocation by hardware.
One set of protection and relocation registers (base address and limit registers). Bounds register.	All programs written as though their origin were location 0. The relocation register specifies the actual location of the user, and the protection register specifies the number of words allowed. (See Fig. 7.)	As users enter and leave, primary memory holes form requiring the moving of users. Pure procedures can only be implemented by moving impure part adjacent to pure part.
Two sets of protection and relocation registers, 2 pairs of bounds register.	Similar to above. Two discontiguous physical areas of memory can be mapped into a homogeneous virtual memory.	Similar to above. Simple, pure procedures with one data array area can be implemented.
Memory page mapping*	For each page (2^6-2^{12} words) in a user's virtual memory, corresponding information is kept concerning the actual physical location in primary or secondary memory. *If the map is in primary memory, it may be desirable to have "associative registers" at the processor-memory interface to remember previous reference to virtual pages, and their actual locations. Alternatively, a hardware map may be placed between the processor and memory to transform processor virtual addresses into physical addresses. (See Fig. 8.)	Relatively expensive. Not as general as following method for implementing pure procedures.
Memory page/segmentation mapping	Additional address space is provided beyond a virtual memory above by providing a segment number. This segment number addresses or selects the page tables. This allows a user an almost unlimited set of addresses. Both segmentation and page map lookup is provided in hardware. (See Fig. 9.) May be thought of as two dimensional addressing.	Expensive. No experience to judge effectiveness.

intiated and requests for additional terminals, primary memory, time, etc., are made through it. It is the medium for job control.

Resource management deals with servicing user demands after resource allocation has occurred. It is imperative to provide users with a system that requires little or no knowledge of particular device or terminal idiosyncrasies. Even though terminals have differing characteristics it is desirable for the system to provide users with a single basic set of characteristics. More flexible terminals would, of course, leave abilities in access of the common characteristics which could be utilized. On the other hand, it is important to allow users the freedom to control special terminal activity directly. This is particularly necessary in mixed experimental-production systems involving terminals that differ widely. For example, in flight simulation systems, the usage may range from program debugging, new terminal hardware-software debugging, and simulation.

The terminal characteristics are: speed or data rate of the terminal; amount of primary memory used for buffering and the location of the buffers; system overhead time for data requests, including processing time required for the data; and device data acquisition modes, and terminal data usage. Detailed terminal management includes the process that buffers data from the terminal and synchronizer user demands with terminal performance.

SYSTEM-PROVIDED PROCEDURES AND PROCESSES

In addition to providing the software framework within which users operate the hardware, the system also supplies many of the processes for a user. That is, the system includes a library of procedures for arithmetic function evaluation, special and procedure oriented language translations, computer aided instruction, file data conversion, text editing, program debugging, fact retrieval, simulation, etc. In fact, the difference between a user and a system process is that a user process can be altered.

The method of calling these procedures (or job setup) and the ability to have a hierarchy of procedure calls is important. A system-supplied procedure can be considered an extension of the system and called with the same mechanism with which a user would request file or terminal activity. In fact, the hardware instructions that provide communication between the system and the user should also be used for procedure calls. In this fashion, the system can conserve memory space by not providing duplicate copies of routines that are in use by multiple users. The data or temporary storage required by the system while enacting a procedure on behalf of a user is part of the user's memory. This structure conserves space both for users of small subroutines (e.g., arithmetic, data conversion, etc.) and large programs (translators, text editors, etc.).

A set of commands might include programmed floating point arithmetic (for a small system), common arithmetic functions, complex arithmetic, string processing, data conversion and operating libraries for the language translators, translators, editors, loaders, etc. Also desirable is the facility for a user to define and call his own functions in the same hierarchy and framework.

MISCELLANEOUS SYSTEM FUNCTIONS

These processes include record keeping, the periodic recording of the system state for backup, error detection, error recovery, error handling for a device, and communication with the user terminals for system requests.

The system clock is a part of the operating system that provides the actual time base and is used by the scheduler and the accountant, for example, to carry out their functions.

System start-up and shut-down procedures are necessary for initialization of system and the recording of history. Parts of the system can be written as pseudo users. This allows functions like data gathering and system analysis to go on by watching the system rather than being embedded in it. This operation is obtained by defining monitor instructions that allow a user to obtain behavioral characteristics on demand.

A debugging system for the operating system might have the following features: ability to examine or alter; ability to dump or save the complete system in the event of a "crash"; ability to control the substitution of a "new" system for the present one, etc. These features are extensions of a normal on line debugging program.

EXAMPLE OF TIME SHARING SYSTEM FOR THE DEC PDP-6

Figure 10 first presents a simplified view of the system in terms of the memory map of the user and operating system, together with terminals and files. The system runs either as a multi-programming or multi-programming/swapping system depending on whether a secondary memory device is available for program swapping.

A job for a user can be viewed as an area of memory which it occupies while running and I/O

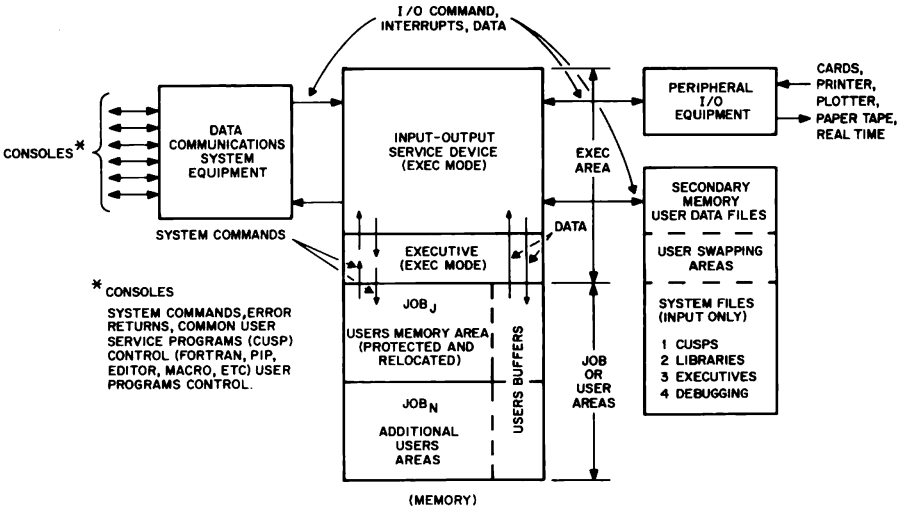


Fig. 10 PDP-6 Multiprogramming System Diagram. (Courtesy of Digital Equipment Corporation.)

equipment assigned to the job, including the user's files and terminals. The operating system software has four main modules: the system files (e.g. FORTRAN, assembler, language translators); terminal control; file control; and the main body of the executive.

Figure 11 gives a more detailed view of what a user program looks like. The user program (e.g., a common user program such as a Fortran Compiler) has its own executive system which communicates with the operating system. The user executive translates user commands from a console into operating system commands for file and terminal activity, while the actual Fortran compiler only accepts input data and produces output data. The user executive is responsible for making it possible for the compiler to read and write files.

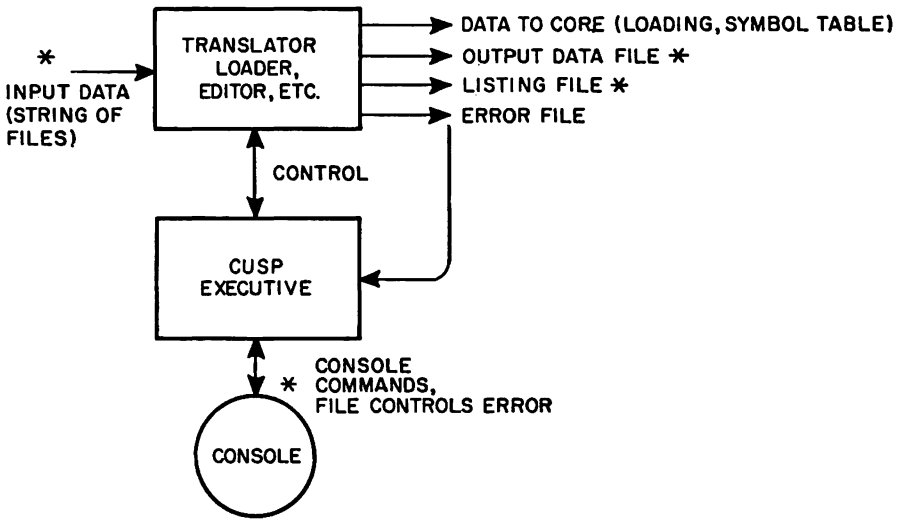


Fig. 11 General structure of common user service program (CUSP) for PDP-6. (Courtesy of Digital Equipment Corporation.)

Figure 12 shows a memory map of a user's program. The space can grow (and contract) as the program is running, since a user program may make requests to the operating system for space. The first main area, that reserved for operating system parameters, is 140_8 long and is available to both the user and the operating system, although special commands must be given to the operating system to change it. The other areas are a function of what programs are being run.

The system's part of the user's job area contains temporary registers which store the processor state while the job is not running. These include:

1. Two groups of 20_8 registers to store the accumulators or general registers (AC's).
2. The Program Counter (PC) and processor flags.
3. The program's location or boundaries.

The registers that hold the organization to a particular program include:

1. Starting address of the program.
2. Starting address of the debugging program, DDT.
3. Location of various blocks in the user's area, i.e., the symbol table, free storage space, etc.
4. Assignment of I/O device names to numbers, so that a device can be referred to by name rather than on an absolute basis ($2 \times 20_8$ locations).

The registers used as working storage for the system include:

1. The STACK, a pushdown area of temporary storage, and stack pointer.
2. Input-Output data Buffers.
3. Job number.

User requests to the monitor are handled via a defined set of instructions which are called the un-used operation codes, or Programmed Operators, or UUO's. Any time the user program makes a call to the system for service it is via these instructions.

The loader is a system routine that is placed in the user area initially and loads the various subprograms required into the user area. The loader links all symbolic references together and fetches needed library programs.

Figure 13 presents a memory map of the operating system which shows the kinds of program modules in it, together with some of the communication paths. The modules perform the following functions:

Job Status Table holds the state of each job in the system, whether a job is in core or residing within a secondary memory prior to running. The state is defined by several words and includes its condition for running, the time it is used, and the location of the job (which includes more status information).

IO Device Service exists for each peripheral device, and the module manages the transmission of data between primary memory and the device, the initiation of the device, and the processing of error or unusual conditions associated with the device (e.g., re-read trys for magnetic tape).

File Directory and File Free Storage Control is used with devices that have named files and directories. It provides the ability to enter new file names and delete files, and it manages the file's free storage.

Error Handling is a common routine that may be called whenever a job (or the monitor) detects an error. A notice of the error is passed on to the user at his console (or to his program), and the job status may be altered.

Run Control is called by other programs and is just concerned with starting and stopping a particular job.

Core Allocation is a common routine responsible for knowing the location of free core in the system and when told, it reserves core blocks.

Clock and Clock Queue are common routines that accept requests for future notification from other parts of the monitor. The clock (more correctly, a timer) notifies the caller at a specified future time.

(For example, the timer is called by the scheduling program so that the scheduler can be activated to schedule the next job.)

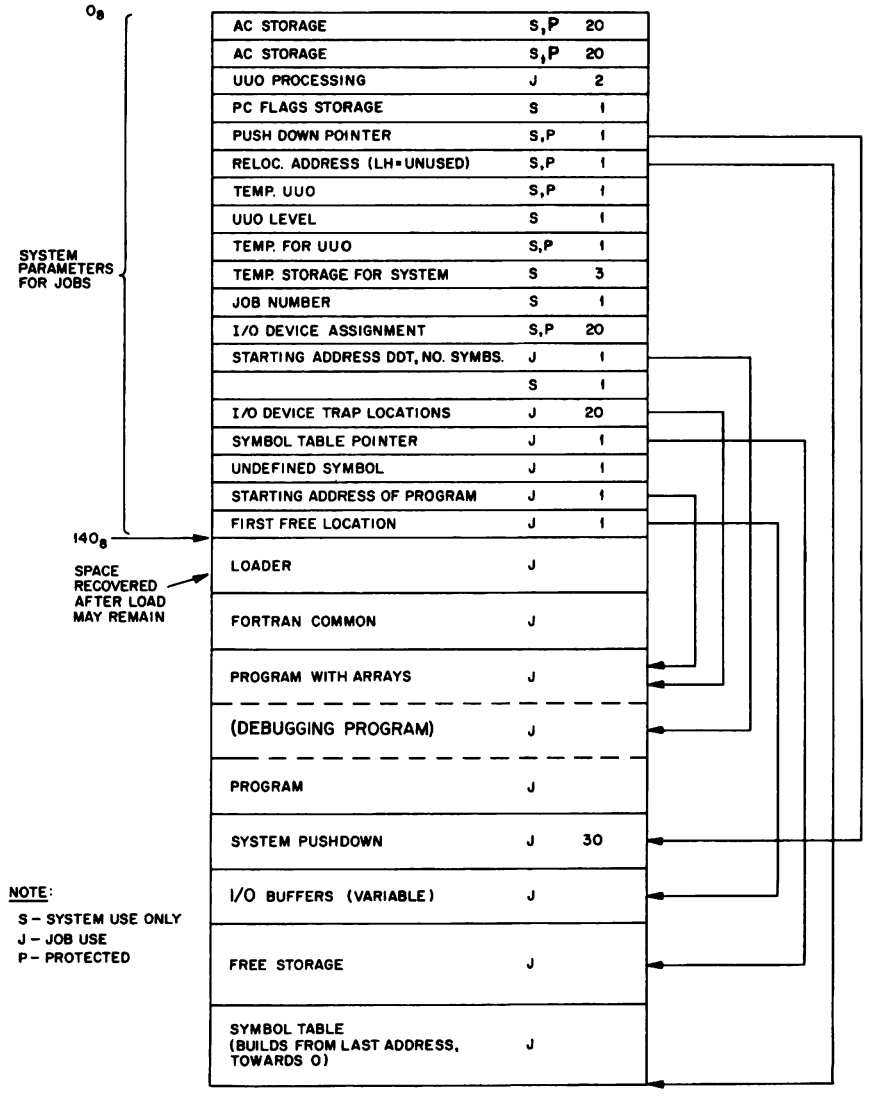
Scheduler makes a decision about the number of the next job to run, based on the variables associated with the system's state (each job status, time, core, etc.).

Programmed Operator Dispatcher processes the instructions that are given by the user program to the executive system. The dispatcher looks up the instruction in a directory, does common pre-processing, and passes control to the appropriate part of the monitor. Some of the instructions are defined by a mnemonic call name. A Call table is hash coded with the name, and corresponding monitor address for the processing.

Command Decoder processes console requests and decides the system routine to call.

Console Command Processors include the programs for actually processing the user console requests (or a user program request). These include programs for log in, save job, start, stop, assign a device, etc. Some programs may not be resident, in which case they are loaded and run in a fashion similar to that of a user program.

System Initialization starts the system just after it is loaded, and includes the freeing of devices and the initialization of all variables.



PDP-6 USER
JOB AREA STORAGE
(COURTESY OF DIGITAL EQUIP. CORP.)

Fig. 12 PDP-6 user job area storage. (Courtesy of Digital Equipment Corporation.)

System Debugging Program is a version of the debugging program, DDT, and may be loaded with the system. It can be used in the event of system failure, to interrogate the state of the system, and includes facilities for preserving the system for future examination.

System Maker allows a complete new monitor to be made as a user program, and when called will copy the new monitor into the area occupied by the old monitor and transfers control to the new monitor.

USER COMPONENTS

TERMINALS

Communication among the terminal, system software, and user process is very important because of process time, memory space, ease of use, and design modularity considerations. "Human engineering" design aspects include those that affect a user's apparent or actual response.

Although there are many aspects of terminals and their design, the following terminal unit groups will be used:

1. Typewriters.
2. Text—Keyboard Displays. (Text cathode ray tube displays with keyboard inputs)
3. General Graphic Displays or Consoles.
4. Direct Terminals.
5. Indirect Terminals.
6. Specialized Terminals.
7. Machine Links.
8. Peripheral Computers.
9. Other time-sharing systems or computer networks.

The parameters that are common to all terminals and that present the user with certain apparent characteristics have been discussed in the hardware section. The physical data transmission modes, character sets, speed, etc., and general appearance differ among terminals, but the "apparent" characteristics to a user program can be nearly constant, so that user programs can be written independent of their environment or terminals they use. The operating system software is responsible for translating basic user requests into common commands that operate the hardware.

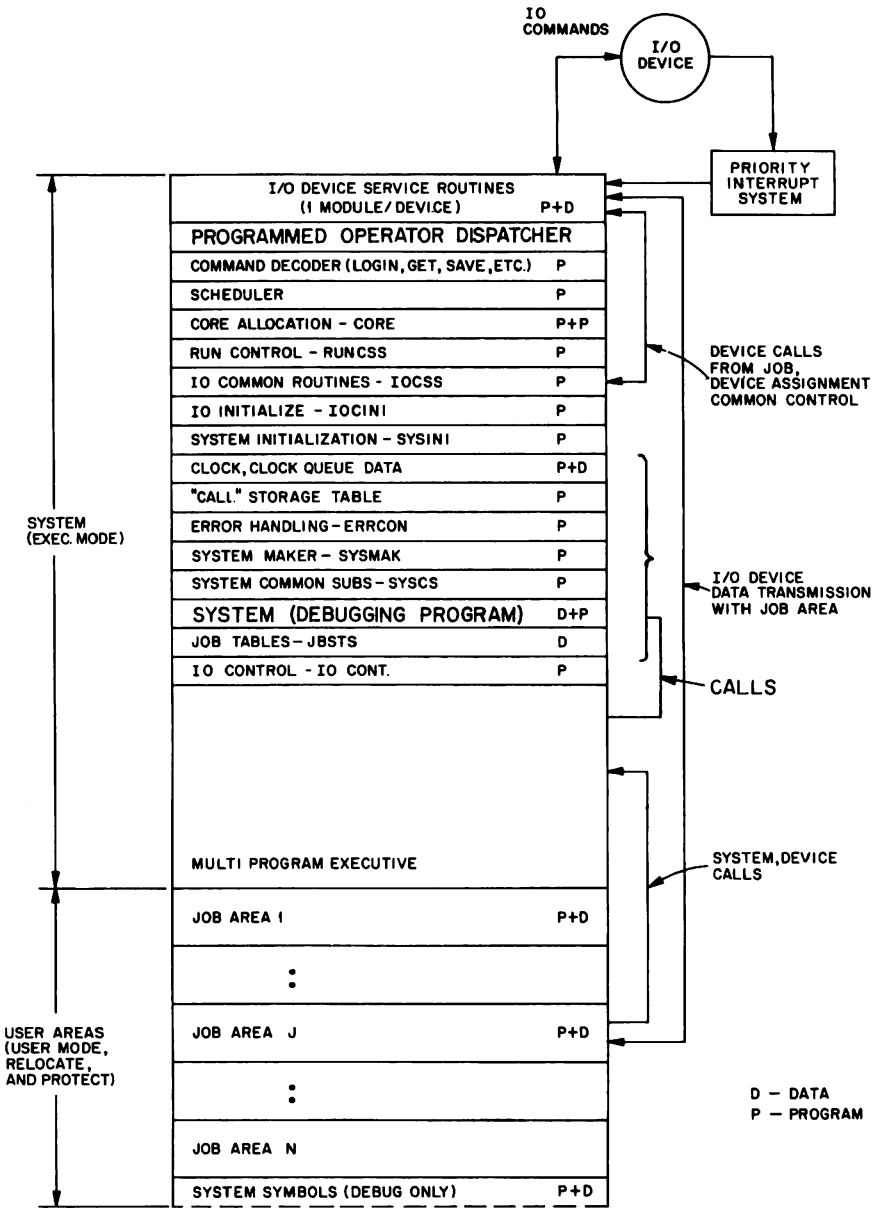


Fig. 13 PDP-6 Multiprogramming System Storage. (Courtesy of Digital Equipment Corporation.)

The typical commands or instructions a user program gives that deal with a terminal include:

1. Assignment of terminal to a process (including the ability to change the name of a terminal, so that programs do not have to address terminals in an absolute sense).

2. Initialization of the terminal to begin transmission, including the declaration of data buffering (number and size), specification of transmission modes, etc.

3. Actual transmission of data (a character, word, buffer, etc., at a time).

4. Termination of transmission, and relinquishing terminal.

Typewriters

Typewriters include both typewriters and Teletypes. The typewriter is the most important because people have been trained to use them. Although harder to use, Teletypes are a common system terminal because they can be used remotely (low bandwidth communication lines), hard copy oriented, low cost, and are available.

Although they are inherently character oriented, it is sometimes desirable to buffer terminal data on a page text line at a time basis or until a special data delimiting key has been struck by the user. (This requires less overhead time from the system to process the

characters, since processing is done for each separate line of text rather than for each character of the text.)

It is necessary to allow some form of simultaneous input and output in order that a user can communicate with the system while it is printing, so that a user can stop or change the process. Full duplex Teletypes easily provide this; half duplex Teletypes can accomplish this by a form of "echo checking" during output. Most typewriter consoles must be supplied with special switches or keys to "break" the information output flow so that the user can stop runaway programs, for example.

Keyboard-Text Displays

These devices are similar to the typewriter in principle. The keyboard-text display does not have the hard copy provided by the typewriter (unless the terminal or console also has a printer), but it does provide the viewing of almost a full page of text, together with the ability to "point" anywhere on the page. These displays also require a higher output data rate from a computer in the form of "page turning" requests. This is the principal terminal for systems requiring simple graphical results or rapid scanning of text.

A small cursor, which is controlled by the terminal allows the user to "point" to any character on the page. The data associated with a single page of text is associated with the display.

The control of text displays requires more information processing than other terminals, since data can be randomly addressed by blocks both for input and output, rather than on a strictly sequential basis.

General Graphical Displays

These displays are similar to the text display, but have the added ability to display data by points, characters, lines, circles, etc., and in general have better resolution and are faster.

The information forming the picture may exist in primary memory (as a process or as data for a process) or within the display's own storage. The human eye requires a complete refresh or regenerate cycle about every 30 milliseconds, in

which the data forming the picture must be sent to the display. This may impose a high data transmission rate on the memory system, interfering with processing, unless the display has an independent data memory to hold the picture.

For graphical input, a light pen is used to "point" to displayed information. The light pen can be used to "draw" on the scope face. The control and data structure problems of the text display are present to a much higher degree in general graphical displays.

The RAND Tablet is a very simple graphical input device. It allows one to draw on a 10″ × 10″ tablet with a stylus, and it can allow free hand drawing, printed character input, or curve tracing (through paper). It may be used independently or in conjunction with a graphical display. The resolution or number of electronically independent points over the 10″ × 10″ area corresponds to 1024 × 1024 points.

Plotters

These devices provide hard copies of general graphical data. Typically, a plotter operates on an incremental or discrete basis (0.01 inches/increment) at a rate of 300 points/second over a plotting area of 12-30 inches by several hundred feet.

Direct Terminals

The above terminals are special cases of direct terminals, but in them most of the problems of terminal hardware and software design can be seen. Namely, problems of providing continuous two-way dialogue, response time, and the other human engineering problems.

Indirect Terminals

These terminals include most terminals used by other systems, i.e., peripheral card readers and line printers. The interface from a user's viewpoint can be identical to the above terminals. The logical difference, for example, between a line printer and a typewriter printer may just be the number of allowable characters on a line; thus, a page output on a line printer would appear identical to that of a typewriter (but not vice versa).

Specialized Terminals

These terminals are used for special time-sharing systems such as airlines reservations, etc. They include: banking teller windows, airline reservation stations, stock quotation inquiry keyboards, production line data acquisition terminals, etc. They provide the best possible coupling between the user and his system and are designed to minimize the number of errors and the time required as data is entered and extracted from the terminal by restricting the format and by encoding the information.

Inter-Machine Links

The link to specialized "non-human user" devices imposes the highest performance requirements on the design because the data transmission rate is high and is determined by the device characteristics, rather than the system. That is, these devices have to be served in real time, at the demands of the device. Devices of this type include those used in process control applications, simulation equipment (aircraft or aerospace cockpits), film reading devices or scanners, hybrid linkages, etc.

By providing for this equipment in a system, hardware protection may also be required. A very complete interrupt or trap system may also be necessary in the hardware so that a job can be rescheduled rapidly to serve the device.

Peripheral Computers

These form a most necessary class of terminals by distributing terminal data transmission or loading to the system periphery. The peripheral computer provides the ability to lower the data rate for a larger system by providing local storage and processing capability. For example, display computers with the ability to detect light pen position and track the pen, and perform

TABLE 3. TERMINAL INPUT REQUESTS TO SYSTEM SOFTWARE

MESSAGES TO THE OPERATING SYSTEM:

1. Log in and log out. (Includes presentation of name, number, password, data, etc.)
2. Resource requests (assignment of terminals, primary memory, file space).
3. Setup of the job, or process.
4. Start, stop, and continuation of a process.
5. Examination and modification of elements of the primary memory process. (Presentation of a storage or memory map.)
6. Information requests:
   a. Run time, time of day
   b. Files used or space available
   c. Facts about system use.
7. Communication with other users or human operators.
8. Saving and restoring the complete state of a process.
9. Transmission of a job to a queue for batch processing.

MESSAGES TO EDITORS:

1. File name declarations including specification of access restrictions, formats, etc.
2. Transmission of data among files and/or terminals.
3. General file editing including creating, appending, inserting, modifying, deleting, etc.

MESSAGES TO TRANSLATORS:

1. File specifications including:
   a. Control statements.
   b. Source language inputs.
   c. Object output.
   d. Object listing.
   e. Object linkage information (if separated from output).
   f. Errors and diagnostics.
2. Control switches (e.g., what to do in case of errors).

MESSAGES FOR PROGRAM DEBUGGING:

Command messages to system debugging routines are similar to the system commands, except that they are in terms of the source language program. They include:

1. Start, stop, and continuation of the process.
2. Examination and modification of the process in terms of the source language. Insertion of program patches. Display of data in any format.
3. Data set searching.
4. Program tracing.
5. Conditional tracing via breakpoints which are executed only if program reaches a specific state.

MESSAGES TO SYSTEM OPERATORS (HUMAN) AND MANAGEMENT (HUMAN)

1. Equipment availability or status information.
2. Configuration specification.
3. Accounting and system status requests.
4. Appending user availability, cost, facility, priority lists.
5. Message broadcasts.
6. Manual instructions for tape mounting, card removal, etc.
7. System diagnostic reports.
8. Control of back-up or archival storage.

MESSAGES TO CONVERSATIONAL LANGUAGES

1. Language or Text Edit commands. Creation, modification, and deletion of programs is provided.
2. Direct Statement Commands Execution. For languages which allow arithmetic statements to be written, the ability to have a statement executed immediately (e.g., 2 + 2 = ?) is provided.
3. Commands for Control of the Programs.
4. Data entry and data output from the program.

some coordinate transformations on the display data may be desirable.

In process control applications, data sampling, limit checking, and data logging can be done by peripheral computers, on a more economical basis, since they do not require the generality of a large machine. Also, since the overhead time to switch to another program may be high, the high data rates associated with these processes would degrade the large machine.

External Time-Sharing Systems

These terminals form the link with other time-sharing systems. This form of intercommunication is new, but may be significant in total problem solving systems by allowing programs in one system to call on other systems.

Message switching centers with some local file storage might form the immediate link with users. As users require more advanced services, the switching centers would likely call either large, general systems or systems specializing in a particular service. Because of our geographical time zones, inter-system load sharing is possible in a fashion similar to that in which utilities share electrical generation capacity.

TERMINAL COMMUNICATION WITH THE OPERATING SYSTEM

In addition to the terminal connection with the process, a terminal must connect with the operating system software for the control of the job. All of the programs (translators, editors, loaders, etc.) that form the system also require control words or statements. Table 3 lists the information required from the user to specify tasks for the system.

Communication Dialogue

The format used for control information is an important design consideration, and it is important to have a "forgiving system," or one which does not affect a user too adversely when a wrong command is given.

It may be important that the user react (type in, observe output, etc.) as little as possible to specify a given situation. Abbreviated commands might be permitted in place of longer words (e.g., LOGIN = LI),

although longer commands would also work. For example, two interesting possibilities are: a user types a command that has enough information to make the command unambiguous, and, the user types enough information to make the command unambiguous, followed by the system typing the rest of the command in a "ghost-like" fashion. When commands are given that irrecoverably affect files, the system might require some sort of verification that the command specified is actually desired.

User defined macro commands compose the most general method to provide users with the commands they want, and what they call the commands, because users define, name, and write them in terms of standard sets of system commands.

FILES

It is desirable to consider the file and terminal structure in a similar fashion from both a user and system software viewpoint; that is, the access, method of transmitting data, and data formats may be nearly identical for both files and terminals.

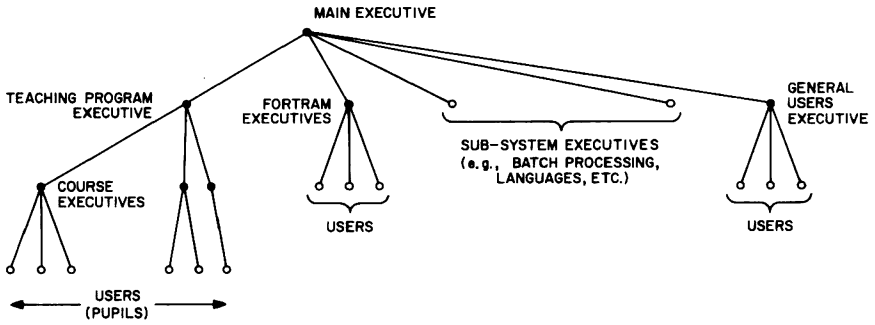


Fig. 14 Hierarchy of executives with a general purpose time-sharing system.

The file characteristics have been previously discussed as part of the operating system software in terms of what the hardware is, what the operating system provides, and what the file looks like to a user.

USER PROCESS

The user process or procedure includes: a memory map locating the process, the actual process, and user status information (terminal and file assignments).

Occasionally, a guaranteed service must be made available to a user both for specialized devices, and processing. For example, a user may have a particular terminal that requires service at regular intervals. A protected, assignable command subset to control the particular device may be required. Alternatively, control can sometimes be provided by incorporating the device in the normal system peripheral or input-output service programs. Scheduling of users now becomes more complex, since the device anomalies constrain the scheduling algorithm.

Guaranteed processing capabilities are provided by treating the total processing capacity as a resource. Thus, a guaranteed capacity at a guaranteed time can be scheduled according to request. Users of systems may get degraded service rather than be denied access because of poor service. With a supply of unattended jobs to process in a batch queue, or compute-bound problems to run as background, a combination denied/degraded service may be provided which balances the system's capacity.

The methods of communication with the system through a hierarchy of higher level operating systems pose the questions: "What is the user process?" and "What is the system?" A user's procedure may be appended to the system and become a system function or common user service procedure. This ever expanding set of program segments which form the system present the problems of segment naming, file location within the system, and protection while they are being run. Nevertheless, the ability to run normally while creating and testing other parts of a system, or to have a portion of the system removed and another one substituted gives rise to very powerful tools in the graceful creation of the system. As a minimum, a new system should be able to be created on a general purpose system, with the substitu-

tion for the existing system occurring at a time when the system is inoperative. We can look forward to complete systems that allow subsystems that do their own scheduling of time, etc., and allocate some resources. Thus, a completely general purpose system might allow complete freedom to incorporate any of the systems described in Table 1 in an efficient manner. Figure 14 shows the relationship processes might have to one another in a general purpose system.

CONVENTIONAL VERSUS CONVERSATIONAL LANGUAGE PROCESSING

Conventional processing or translation of a language occurs in the sequence:

1. Creation of a text format source file (cards or system file) which describes the process.
2. Translation of source files into object files with linkage, relocation, subroutine, listing, and error information.
3. Loading the object file together with library files to form the process.
4. Process execution.

In contrast, conversational language processing provides nearly simultaneous creation and execution of procedures. The input language can be checked at the time of entry at the terminal and is translated, being immediately available for execution.

The data may be transformed into an interpretive form with all sub-routines, linkages, etc., occurring directly on input with no intermediate files. The insertion of additional statements or program steps is done directly, and debugging is through the run time diagnostics and user abilities to examine variables directly and execute statements conditionally. The conversational system may require a slightly longer execution time, but is most effective because of its combined editor, translator, loader, library and debugging system. Clearly, for problems involving little computation, the turn-around time is very short for solving problems in this fashion. The main structure of programs is such that this interactive approach may be the common method in a few years.

Batch Processing

This is one of the most efficient methods of controlling the execution of a large number of programs, since jobs are always run to completion. In a time-sharing system which is principally serving on-line users, the batch process can be used as a background job or to absorb spare capacity. A fixed or guaranteed amount of processing can be allocated to batch processing. The batch must be able to be loaded by either external users with card decks or users who defer jobs that can be done anytime (or at batch convenience).

The handling of a batch need not be incorporated within the system, but rather a batch process can

TABLE 1. CAPACITY REQUIREMENTS FOR TIME-SHARING SYSTEM APPLICATIONS

Specialized System Service, or Application	Primary Memory for Process (in bits)	Primary Memory for User Data (in bits)	Processing Capacity/ User (in operations*/ interaction†)	File Organization and Size (10^6-10^9 bits)	Direct Terminals
Desk calculator	very small	very small ($<10^3$)	very small ($>10^4$)	none	typewriter, input keyboard, strip printer, scopes, audio output, or special console.
Stock quotation	small	small ($<10^4$)	very small ($>10^4$)	one (small-medium)	see above, stock ticker tape or transactions input, telephone.
Airline reservations	medium	small ($>10^4$)	small ($>10^5$)	approx. 6 (medium-large)	special consoles, typewriters, scopes.
On line banking	medium	small ($>10^4$)	small ($>10^5$)	approx. 10 (medium-large)	see above, special bank teller consoles.
General conversational computational languages (JOSS, CULLER-FRIED System)	medium	small-very large (10^3-10^5)	small-large unbounded (10^4-$>10^8$)	multiple files per user, with few file types (medium-large)	typewriter, printer, scope, plotter. (Culler-Fried consists of scope, keyboard, and tablet.)
Specialized computer aided design, engineering, problem solving languages (COGO, etc.)	medium-large	small-very large (10^3-10^5)	small-very large (10^4-$>10^8$)	see above	see above
Process control	medium-large	medium ($>10^5$)	small-very large (10^4-$>10^8$)	few (small)	physical quantity transducers, general user terminals.
Text editing (Administrative Terminal Service)	medium	small ($>10^4$)	small (10^4-10^5)	multiple single purpose files/user. (medium)	typewriter, printer, scope.
On line information retrieval of periodical headings, bibliographies, keywords, abstracts	medium-large	medium ($>10^5$)	medium (10^5-10^7)	one (very large)	see above. telephone (dial in, audio out)

*assumes a fairly sophisticated processor and instruction set
†maximum interaction intervals for user requests are $\simeq$ 10 sec.

be regarded as a special user. Thus, a common service program (the batch manager) would permit any user to "batch process."

CONCLUSIONS

PRESENT PROBLEMS

Before widespread time-sharing systems and system networks can be formed, standardization of data and file format descriptions will have to occur. Simple conventions must be established to control the actual format of the bits transmitted between computers. This will enable the transmission of problems, data, and procedures between systems. Present intersystem communication experiments should provide a framework for the standardization of information interchange formats, and detailed data representation.

Once a data representation for higher speed lines is established, it will be possible to remove the terminals we presently associate with the computer outside the computer's periphery. This will enable the cross-use of terminals among computers. It will also allow software that is more independent of the peripheral and computer to be written.

Current data transmission costs for the remote typewriter user (with an average input rate of ten bits per second) do not reflect the true cost-capacity (2400 bits per second for a voice grade line) or use of the line.

Although good, low cost computers (processor, memory, and minimum peripheral equipment) are available, the higher costs associated with file storage for smaller systems do not permit the design of low cost time-shared computers.

Present time-sharing structures for computers are extension organizations of the basic computer. Present systems were not initially designed for time-sharing, but were modified slightly to accommodate potential users. Hence, these systems create almost as many problems as they solve. A more reasonable approach for a system's design is an initial specification that includes Time-Sharing as a goal. A solution might take on the form of a network. For example, the very large computing machines that are built by computer manufacturers have: taken a long time to build (and technology has changed, invalidating industry's extrapolations before the computers were operational); required longer than expected to become operational; failed to meet initial design goals, have been uneconomical from a production standpoint; and only a few systems have been built. The current large, very general systems also suffer from the same kind of design thinking.

Each component of a general purpose time sharing system is constrained to supply such general service that the system as a whole may be so inefficient (and expensive) as to make the system impractical. The issue is similar to an organization consisting of either highly trained specialists or generalists. An organization of generalists is very flexible; but, on the other hand, it may not be economical to have people who are capable of being the president doing all the tasks within an organization. The general purpose systems just now becoming operational are constructed in such a flexible fashion as to probably be uneconomical. Each system component is so general (for example, the filing system) that, although it can perform any task (given enough time), the act of doing very trivial operations requires a great deal of time. Perhaps a better approach is to divide the systems's resources by allowing several independent operating systems to care for them (e.g., editing, assembling, filing, translating, and running).

FUTURE SYSTEMS

Future computers will be equipped with hardware to allow some form of time-sharing. For smaller com-

puters, the additional hardware greatly enhances a system's utility, especially when being used in process control and in research requiring the direct links with other machines or to experimental equipment.

The form of Time-Sharing Computers will be:

1. The system with a single general user or batch process, *plus* one fixed job or a fixed multi-terminal community of special users (1+1, or 1+n special users). Process control and on-line special business data processing systems take this form.
2. Dedicated special systems which service a particular user community. These provide little or no communication with other systems. (E.g., library, airlines reservations, etc.)
3. Dedicated systems with switching ability so that a problem that requires other aids can be referred to other systems. More general systems may refer problems to them.
4. Message switching for other systems. These may have file processing, editing, and limited calculation capability, or message buffering; such a system would communicate with other systems for most demands from users.
5. Peripheral computers that service special terminals and control small local processes. Processing capacity for general purpose problem solving, file storage, program translation, and diagnostics for the peripheral system would be derived from a higher level system.
6. The totally general system with a large community of users. The general system would undoubtedly communicate with other systems.

Although the author has attempted to be objective, it is felt that the technique of computer Time-Sharing is a significant advance toward an effective use of computers. Time-Sharing removes one more restriction in computer usage — that of allowing only a single use of a machine. As such, the additional generality creates opportunities, as well as countless problems.

BIBLIOGRAPHY

1. Adams, E. N., "Reflections on the Design of a CAI Operating System," *AFIPS Conference Proceedings, Spring Joint Computer Conference* Vol. 30, 419–424 (1967).
2. Allan, Pryor T. and R. Warner Homer, "Time Sharing in Biomedical Research," *Datamation* Vol. 12 (4) (April 1966).
3. Amdahl, G. M., "New Concepts in Computing Systems Design," *Proc. IRE* Vol. 50 (5), 1073–1077 (Memory Protection) (May 1962).
4. American Management Association, "Advances in EDP and Information Systems," *AMA Report* No. 62.
5. Anderson, J. P., S. A. Hoffman, J. Shifman, and R. J. Williams, "D-825 — A Multiple Computer System for Command and Control," *Proc. Fall Joint Computer Conference* Vol. 25, 86–96 (December 1962). See also *D-825 Manual* — Burroughs Corporation.
6. Arden, B. W., et al, "Program and Addressing Structure in A Time-Sharing Environment," *Journal of the Association for Computing Machinery* Vol. 13 (1), 1–16 (January 1966).
7. Aschenbrenner, R. A., M. Flynn, G. A. Robinson, "Intrinsic Multiprocessing," *AFIPS Conference Proceedings, Spring Joint Computer Conference* Vol. 30, 81–86 (1967).
8. Avakian, Emik A. and F. Walter Jenison, Jr., "Voice Response and Visual Display Techniques for On-Line Information Handling Systems," The Bunker-Ramo Corporation, Stamford, Connecticut.
9. Bachman, C. W. and S. B. Williams, "A General Purpose Programming System for Random Access Memories," *Proc. Fall Joint Computer Conference* Vol. 26, 411–422 (1964).
10. Baldwin, F. R., W. B. Gibson, and C. B. Poland, "A Multiprocessing Approach to a Large Computer System," *IBM Systems Journal* Vol. 1, p. 61 (Sept. 1962).
11. Bauer, Walter F., "Why Multi-Computers?," *Datamation* Vol. 8 (9) (Sept. 1962).
12. Beckman, F. S., F. P. Brooks, Jr., and W. J. Lawless, Jr., "Developments in the Logical Organization of Computer Arithmetic and Control Units," *Proc. IRE* Vol. 49 (1), 53–66 (January 1961).
13. Bell, G. and M. W. Pirtle, "Time-Sharing Bibliography," *Proc. IEEE* Vol. 54 (12), 1764–1765 (December 1966).
14. Belluardo, R., R. Gocht, G. Paquette, "A Time Shared Hybrid Simulation Faculty," UAC, East Hartford, Conn., *AFIPS* Vol. 28 (1966).
15. Bitzer, P. L., and H. G. Slottow, "The Plasma Display Panel — A Digitally Addressable Display with Inherent Memory," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 541–548 (1966).
16. Boilen, S., "User's Manual for the BBN Time-Sharing System," Bolt, Beranek, and Newman, 50 Moulton St., Cambridge, Mass.
17. Bolt, Richard H., "Man-Machine Partnership in Intellectual Pursuits: A Look Ahead," Publication No. 1191, Printing and Publishing Office, National Academy of Sciences.
18. Brillouin, Leon, "Science and Information Theory," Second Edition, Academic Press, Inc. (1962).
19. Brooks, F. P., Jr., "A Program Controlled Program Interrupt System," *Proc. Eastern Joint Computer Conference,* 128–132 (December 1957).
20. Buchholz, W. (Editor), "Planning a Computer System — Project Stretch," McGraw-Hill Book Company, Inc., New York (1962). (See also *IBM 7030 (Stretch) Manual*).
21. Burks, A. W., H. H. Goldstine and J. von Neumann, "Preliminary Discussion of the Logical Design of an Electronic Computing Instrument," (reprinted) *Datamation,* 24–31 (September 1962).
22. Burroughs Corporation, "The Descriptor," Burroughs Corporation (1961).
23. Burton, A. J. and R. G. Mills, "Electronic Computers and Their Business Applications," London, E. Benn (1960).
24. Bush, Vannevar, "As We May Think," *Atlantic Monthly* Vol. 176, 101 (July 1945).
25. Calingaert, P., "System Performance Evaluation: Survey and Appraisal," *Comm. ACM 10* (1), 12–18 (January 1967).
26. Carnegie, "Carnegie Institute of Technology Computation Center Users Manual."
27. Castle, C. T., "Planning the 3600," *Proc. Eastern Joint Computer Conference,* 73 (December 1964). See also CDC-3600, *Datamation,* 37–40 (May 1964).
28. Clippinger, Richard F., "Programming Implications of Hardware Trends," *IFIP Congress, New York* Vol. 1, 207–212 (1965).
29. Codd, E. F., "Multiprogramming Scheduling," *Comm. ACM* Vol. 3 (6) (June 1960).
30. Codd, E. F., "Multiprogramming Stretch: A Report on Trials," *Proc. IFIP Congress, Munich,* 574, North Holland Publishing Co., Amsterdam (Aug. 27 to Sept. 1, 1962).
31. Coffman, E. G., "A General Flow Chart Description of the Time-Sharing System," *SDC TM-1639/000/00* (Dec. 12, 1963).
32. Comfort, W. T., "A Computing System Design for User Service," *Proc. Fall Joint Computer Conference, Las Vegas, Nevada,* Vol. 27 (Nov. 30, 1965).
33. Computer Research Corporation, "Time Sharing System Scorecard, No. 1," Computer Research Corporation, 747 Pleasant St., Belmont, Mass.
34. Conway, M. E., "A Multiprocessor System Design," *Fall Joint Computer Conference* Vol. 24, 139–146 (1963).
35. Cook, P. A. C., "Real-Time Monitoring of Laboratory Instruments," *AFIPS Conference Proceedings, Spring Joint Computer Conference* Vol. 30, 779–782 (1967).
36. Coons, S. A., "An Outline of the Requirements for a Computer Aided Design System," *1963 Spring Joint Computer Conference,* 229–304.
37. Corbato, F. J., et al, "The Compatible Time-Sharing System: A Programmer's Guide," M.I.T. Press, Cambridge, Mass. (1963).
38. Corbato, Fernando J., M. Merwin-Daggett, and R. C. Daley, "An Experimental Time-Sharing System," *AFIPS Conference Proceedings* Vol. 21, 335–344 (Spring 1962).
39. Corbato, F. J., V. A. Vyssotsky, "Introduction and Overview of the Multics System," *Proc. Fall Joint Computer Conference,* Las Vegas, Nevada (Nov. 30, 1965).
40. Crisman, P. A., Editor, "The Compatible Time-Sharing System," *A Programmer's Guide,* 2nd edition, M.I.T. Press, Cambridge, Mass. (1965).
41. Critchlow, A. J., "Generalized Multiprocessing and Multiprogramming Systems," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 24, 107–126 (1963).
42. Culler, G. J. and B. D. Fried, "The TRW Two-Station, On-Line Scientific Computer: General Description," *Computer Augmentation of Human Reasoning,* Washington, D. C., June 1964, Spartan Books, Washington, D. C. (1965).
43. Daley, R. C. and P. G. Neumann, "A General Purpose File System for Secondary Storage," *Proc. Fall Joint Computer Conference, Las Vegas, Nevada* Vol. 27 (Nov. 30, 1965).
44. Dartmouth, "The Dartmouth Time-Sharing System," Computation Center, Dartmouth College (Oct. 19, 1964).
45. *Datamation,* "A Survey of Airline Reservation Systems," p. 53 (June 1962).
46. David, E. E., Jr. and R. M. Fano, "Some Thoughts About the Social Implications of Accessible Computing," *Proc. Fall Joint Computer Conference, Las Vegas, Nevada* Vol. 27 (Nov. 30, 1965).
47. Dearden, John, "Can Management Information Be Automated," Harvard Business Review (March–April, 1964).
48. Denning, P. J., "Effects of Scheduling on File Memory Operations," *AFIPS Conference Proceedings, Spring Joint Computer Conference* Vol. 30, 9–22 (1967).
49. Dennis, J. B., "A Multiuser Computation Facility for Education and Research," *Communications of the Acm* Vol. 7, 521–529 (Sept. 1964).
50. Dennis, J. B., "Segmentation and Design of Multiprogrammed Computer Systems," *IEEE International Convention Record, Institute of Electrical and Electronic Engineers,* New York, Vol. 13 (3), 214–225 (1965); and *JACM* Vol. 12 (4), 589–602 (Oct. 1965).
51. Dennis, J. B. and E. L. Glaser, "The Structure of On-Line Information Processing Systems," *Proceedings of the Second Congress on Information* Systems Sciences, 1–11, Spartan Books, Washington, D. C. (1965).
52. Dertouzos, M. L. and H. L. Graham, "A Parametric Graphical Display Technique for On-Line Use," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 210–210 (1966).
53. Desmonde, William H., "Computers and Their Uses," Englewood Cliff, New Jersey, Prentice-Hall (1964); "Real Time Data Processing System — Introductory Concepts," Englewood Cliff, New Jersey, Prentice-Hall (1965).
54. Digital Equipment Corporation, Maynard, Mass., "Multiprogramming System Manual for PDP-6," DEC-6-EX-SYS-UM-IP-PREOO.
55. Dudas, J. F., "Concurrent Processing of Teletype Message Switching and Order Entry at Westinghouse Tele-Computer Center," Westinghouse Electric Corporation.
56. Duffy, G. F. and W. D. Timberlake, "A Business-Oriented Time-Sharing System," IBM, SDD Poughkeepsie, *AFIPS, Spring Joint Computer Conference,* Vol. 28, 265–275 (1966).
57. Dunn, T. M. and J. H. Morrissey, "Remote Computing — An Experimental System, Part 1: External Specifications," — J. M. Keller, E. C. Strum, and G. H. Yang, Part 2, *Proc. Spring Joint Computer Conference* Vol. 25, 413–443 (1964).
58. Eckert, J. P., J. C. Chu, A. B. Tonik, and W. F. Schmitt, "Design of UNIVAC — LARC System I," *Proc. Eastern Joint Computer Conference* (16), 59–65 (1959).
59. Edwards, J. D., "An Automatic Data Acquisition and Inquiry System Using Disk Files," (Lockheed Missiles and Space Co.), Disk File Symposium, March 6–7, 1963 (Informatics, Inc. Culver City, Calif.)

60. Evans, D. C. and Leclerc, J. Y., "Address Mapping and the Control of Access in an Interactive Computer," *AFIPS Conference Proceedings, Spring Joint Computer Conference* Vol. 30, 23–32 (1967).
61. Fano, Robert M., "The MAC System: The Computer Utility Approach," *IEEE Spectrum* Vol. 2, 56–64 (January 1965).
62. Fine, G. H., C. W. Jackson and P. V. McIsaac, "Dynamic Program Behavior Under Paging," *Proc. ACM 21st Conference,* 223–228.
63. Flynn, Michael J., "Very High-Speed Computing Systems," *Proc. IEEE* Vol. 54 (12), 1901–1909 (December 1966).
64. Forgie, R. W., "A Time- and Memory-Sharing Executive Program for Quick Response On-Line Applications," *Proc. Fall Joint Computer Conference, Las Vegas, Nevada* Vol. 27 (Nov. 30, 1965).
65. Fotheringham, J., "Dynamic Storage Allocation in the Atlas Computer," *Comm. ACM* Vol. 4 (10), 435–436 (Oct. 1961).
66. Frankovich, J. M. and H. P. Peterson, "A Functional Description of the Lincoln TX-2 Computer," *Western Computer Proceedings,* 146 (1957).
67. Fredkin, Edward, "The Time-Sharing of Computers," *Computers and Automation* Vol. 12 (11) (Nov. 1963).
68. Gallagher, James D., "Management Information Systems and the Computer," *AMA Research Study:* No. 51 (1961).
69. Gallenson, L., "On-Line I/O Processor for the Command Research Laboratory," The PDP-1-C-30, SDC TM-1653 (Dec. 23, 1963).
70. Gallup, G., "The Miracle Ahead," Harper and Row, New York (1964).
71. Gass, S. I., Marilyn B. Scott, R. Hoffman, W. K. Green, A. Peckar, R. D. Peavey and J. E. Hamlin, "Project Mercury Real-Time Computational and Data Flow System," *Proc. Eastern Joint Computer Conference* Vol. 20, 33–78 (Dec. 1961).
72. Ginzberg, M. G., "Notes on Testing Real-Time Systems Programs," *IBM System Journal* 4 (1), 58–72 (1965).
73. Glaser, E. L., "The Structure of On-Line Information Processing Systems," *Proc. Second Congress on Information Sciences, Homestead, Va.,* 1–11 (Nov. 1965).
74. Glaser, E. L. and F. G. Corbato, "Introduction to Time-Sharing," *Datamation* Vol. 10 (11) (Nov. 1964).
75. Glaser, E. L., J. F. Couleur and G. A. Oliver, "System Design of a Computer for Time-Sharing Applications," *Proc. Fall Joint Computer Conference, Las Vegas, Nevada,* Vol. 27 (Nov. 30, 1965).
76. Greenberger, Martin, "The Computers of Tomorrow," *Atlantic Monthly,* 63–67 (May 1964).
77. Greenberger, Martin, "Management and the Computer of the Future," The M.I.T. Press and John Wiley and Sons, Inc., (1962).
78. Gruenbeyer, Fred, "Are Small Free-Standing Computers Here to Stay?," *Datamation* Vol. 12 (4) (April 1966).
79. Harris, R. P., "The PDP-6," *Datamation* Vol. 10 (11) (Nov. 1964).
80. Hastings, Thomas N., "Real-Time Computing with Time-Sharing," *Computers and Automation* Vol. 14 (10) (Oct. 1965).
81. Hittel, L. A., "Some Problems in Data Communications Between the User and the Computer," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 395–402 (1966).
82. Holland, J. H., "On Iterative Circuit Computers Constructed of Micro-Electronic Components and Systems, *Proc. Western Joint Computer Conference,* p. 259 (May 1960).
83. Holt, A. W., "Program Organization and Record Keeping for Dynamic Storage Allocation," *Comm. ACM* Vol. 4, 422–431 (Oct. 1961).
84. Hoover, E. S. and Eckhart, "Performance of a Monitor for a Real-Time Control System," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 23–26 (1966).
85. IBM "1800 Time-Sharing Executive System Specifications," File 1800-36, Form No. C26-5990-0.
86. Iliffe, J. K. and J. G. Jodeit, "A Dynamic Storage Allocation Scheme," *Computer J.* Vol. 5, 200–209 (Oct. 1962).
87. Johnson, T. E., "Sketchpad III: A Computer Program for Drawing in Three Dimensions," *Proc. Spring Joint Computer Conference,* p. 347, Detroit, Michigan (May 1963).
88. "The JOSS System, Time Sharing at Rand," *Datamation* Vol. 10 (11) (Nov. 1964).
89. Kemper, D. A., "Operation of CRL Teletype System," SDC TM 1488/000/00 (Sept. 18, 1963).
90. Kennedy, J. R., "A System for Time-Sharing Graphic Consoles," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 211–222 (1966).
91. Keydata, "Data Processing — On Line . . . In Real Time . . . The Keydata System," Keydata Corporation, 575 Technology Square, Cambridge, Mass.
92. Kilburn, T., R. B. Payne and D. J. Howarth, "The Atlas Supervisor," *Proc. Eastern Joint Computer Conference* Vol. 20, 279–294 (1961).
93. Kilburn, T., D. B. G. Edwards, M. J. Lanigan, and F. H. Sumner, "One Level Storage System," *IRE Transactions on Electronic Computers* Vol. EC-11 (2), 223–235 (April 1962).
94. King, Gilbert W. et al., "Automation and the Library of Congress," Washington, D. C.: Library of Congress (1963).
95. Kinslow, H. A., "The Time-Sharing Monitor System," *Fall Joint Computer Conference,* Vol. 26, Part 1, 443–454 (1964).
96. Kolsky, "Centralization vs. Decentralization," Tenth Annual Symposium on Computers and Data Processing (June 26–27, 1963).
97. Lampson, Butler W., "Time Sharing System Reference Manual," Working Document, University of California, Document No. 30.1030; issued Sept. 30, 1965; revised Dec. 30, 1965.
98. Lampson, B. W., W. W. Lichtenberger, M. W. Pirtle, "A User Machine in a Time-Sharing System," *Proc. IEEE* Vol. 54 (12), 1766–1774 (Dec. 1966).
99. Landis, N., A. Manos, and L. R. Turner, "Initial Experience with An Operating Multiprogramming System," *Comm. ACM,* Vol. 5 (5) (May 1962).
100. Lawless, W. J., "Developments in Computer Logical Organization," *Advances in Electronics and Electron Physics* Vol. 100, Academic Press, Inc., New York (1959).
101. Lehman, M., "A Survey of Problems and Preliminary Results Concerning Parallel Processing and Parallel Processors," *Proc. IEEEE* Vol. 54 (12), 1889–1901 (Dec. 1966).
102. Leiner, A. L., W. A. Notz, J. L. Smith and W. W. Youden, "PILOT Multiple Computer System (Manual)," National Bureau of Standards Report 6688. See also *Journal of ACM* Vol. 6 (3) (July 1959).
103. Lehrer, N. H. and Ketchpel, R. D., "Recent Progress in a High-Resolution, Meshless, Direct-View Storage Tube," *AFIPS Conference Proceedings,* Fall Joint Computer Conference Vol. 29 (7–10), 531–540 (1966).
104. Levine, S. et al, "A Fast Response Data Communications System for Airline Reservations," *Communication and Electronics* (Nov. 1961).
105. Lichtenberger, W. W. and M. W. Pirtle, "A Facility for Experimentation in Man-Machine Interaction," *Proc. Fall Joint Computer Conference, Las Vegas, Nevada* Vol. 27 (Nov. 30, 1965).
106. Licklider, J. C. R., "Man Computer Symbiosis," *IRE Transactions on Human Factors in Electronics* Vol. HFE-1, 4–11 (March 1960).
107. Licklider, J. C. R. and W. E. Clark, "On-Line Man-Computer Communication," *Proc. Spring Joint Computer Conference,* 113–128 (1962).
108. Lonergan, L. and P. King, "Design of the B5000 System," *Datamation* Vol. 7 (5) (May 1961).
109. McCarthy, J., "Time Sharing Computer Systems," *Management and the Computer of the Future* (M. Greenberger, Editor), M.I.T. Press, Cambridge, 221–236 (1962).
110. McCarthy, J., S. Boilen, E. Fredkin, and J. C. R. Licklider, "A Time-Sharing Debugging System for a Small Computer," Proc. *Spring Joint Computer Conference* Vol. 23, 355–363 (1963).
111. McClung, L. W., "A Disc-Oriented IBM 7094 System," Paper #3, Disk File Symposium, March 6–7, 1963, Hollywood, California (Sponsored by Informatics, Inc.)
112. Maher, R. J., "Principles of Storage Allocation in a Multiprocessor Multiprogrammed System," *Comm. of ACM* Vol. 4, 421–422 (Oct. 1961).
113. Malcom, Donald G. and Alan J. Rowe, "Management Control Systems," John Wiley and Sons, Inc.
114. Marcotty, M. J., F. M. Longstaff, and Audrey P. M. Williams, "Time-Sharing on the Ferranti Packard FP6000 Computer System," *Proc. Spring Joint Computer Conference* Vol. 23, 29–40 (1963).
115. Marill, T. and Roberts, L. G., "A Proposed Communications Network to Tie Together Existing Computers," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 425–433 (1966).
116. Mendelson, M. J. and A. W. England, "The SDS SIGMA 7: A Real-Time Time-Sharing Computer," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 51–64 (1966).
117. M.I.T. Digital Computer Lab., "Comprehensive System Manual — A System of Automation Codes for the Whirlwind Corporation," Memo M-2539-2 (Dec. 1953).
118. Nebel, B. E., "A Multiprogrammed Teleprocessing System for Computer Typesetting," *AFIPS Conference Proceedings, Fall Joint Computer Conference* Vol. 29 (7–10), 115–124 (1966).
119. Nelson, T. H., "A File Structure for the Complex, the Changing and the Indeterminate," *ACM National Conference* (Aug. 1965).
120. Nisenoff, N., "Hardware for Information Processing Systems: Today and in the Future," *Proc. IEEE* Vol. 54 (12), 1820–1835 (Dec. 1966).
121. Ochsner, B. P., "Controlling a Multiprocessor System," Bell Telephones Lab. Record (Feb. 1966).

122. Ossanna, J. F., L. E. Mikus, and S. D. Dunten, "Communications and Input/Output Switching in a Multiplex Computing System," *Proc. Fall Joint Computer Conference, Las Vegas,* Nevada Vol. 27 (Nov. 30, 1965).
123. Parkhill, D. F., "The Challenge of the Computer Utility," *Addison-Wesley Publishing Company* LC-66-24245, (1966).
Addison-Wesley Publishing Company LC-66-24245, (1966).
124. Penny, J. D. and T. Pearcey, "Use of Multiprogramming in the Design of a Low Cost Digital Computer," *Comm. ACM* Vol. 5 (9), p. 473 (Sept. 1962).
125. Perlis, A. J., "A Disk File Oriented Time Sharing System," Disk File Symposium, March 1963 (sponsored by Informatics, Inc., Culver City, Calif.).
126. Peters, B., "Security Consideration in Multi-Programmed Computer System," *AFIPS Conference Proceedings, Spring Joint Computer Conference* Vol. 30, 283–286 (1967).
127. Proctor, James W., Jr., "The Voice Response System," *Datamation* Vol. 12, 43–44 (Aug. 1966).
128. Ramamoorthy, C. A., "The Analytic Design of a Dynamic Lookahead and Program Segment — System for Multiprogrammed Computers," *Proc. ACM 21st Conference,* 229–239.
129. Ramsay, Karl and J. C. Strauss, "A Real Time Priority Scheduler," *Proc. ACM 21st National Conference,* 161–166.
130. Reiter, A., "A Resource Allocation Scheme for Multi-User On-Line Operation of a Small Computer," *AFIPS Conference Proceedings, Spring* Joint Computer Conference Vol. 30, 1–8 (1967).
131. Roberts, L. G., "The Lincoln Wand," AFIPS Conference Proceedings, *Fall Joint Computer Conference* Vol. 29 (7–10), 223–228 (1966).
132. Rosenberg, A. M. (Editor), "Command Research Laboratories Users Guide," SDC TM-1354 (Nov. 19, 1965).
133. Ross, D. T. and J. E. Rodriguez, "Theoretical Foundations for the Computer-Aided Design System," *Computer Aided Design, Spring Joint Computer Conference,* p. 305 (1963).
134. Samuel, A. L., "Time Sharing on a Computer," *New Scientist* Vol. 26, 583–587 (May 27, 1965).
135. Scherr, Alan L., "Time Sharing Measurement," *Datamation* Vol. 12 (4) (April 1966).
136. Schwartz, E. S., "Automatic Sequencing Procedure with Application to Parallel Programming," *Journal of ACM* Vol. 8, 513–537 (Oct. 1961).
137. Schwartz, J. I., E. G. Coffman, and C. Weissman, "A General Purpose Time-Sharing Systems," *Spring Joint Computer Conference* Vol. 25, 397–411 (1964).
138. Schwartz, J. I., "Observations on Time-Shared Systems," *ACM Proceedings of the 20th National Conference,* p. 525 (1965).
139. Schwartz, Jules I., "The SDC Time-Sharing System Part 1," *Datamation* Vol. 10 (1), Part 2, (Nov. 1964); *Datamation* Vol. 10 (12) (Dec. 1964).
140. Scott, M. B. and R. Hoffman, "The Mercury Programming System," *Proc. Eastern Joint Computer Conference,* Vol. 20, 47–53 (Dec. 1961).
141. Sprague, Richard E., "On Line-Real Time Systems — 1964," *Management Services* (May-June 1964).
142. Stanga, D. C., "Univac 1108 Multiprocessor System," *AFIPS Conference Proceedings, Spring Joint Computer Conference* Vol. 30, 67–74 (1967).
143. Stotz, R., "Man-Machine Console Facilities for Computer-Aided Design," *Computer Aided Design, Spring Joint Computer Conference,* p. 323 (1963).
144. Strachey, C., "Time Sharing in Large Fast Computers," *Proc. of the International Conference on Information Processing, Paris, UNESCO,* 336–341 (1960).
145. Summer, F. H. and E. C. Y. Chen, "The Central Control Unit of the ATLAS Computer," *Proc. of IFIP Congress,* p. 657 (1962).
146. Sutherland, I. E., "Sketchpad: A Man-Machine Graphical Communication System," Lincoln Lab Technical Report No. 296, M.I.T., January 30, 1963, *Computer Aided Design, Spring Joint Computer Conference,* 329–346 (1963).
147. Teleregister, "200 Display System," The Bunker-Ramo Corporation, Stamford, Conn.
148. Teleregister, "On-Line Data Processing for Hotels," The Bunker-Ramo Corporation, Stamford, Conn.
149. Vyssotsky, V. A., F. J. Corbato, and R. M. Graham, "Structure of the Multics Supervisor," *Proc. Fall Joint Computer Conference,* Las Vegas, Nevada, Vol. 27 (Nov. 30, 1965).
150. Weil, J. W., "A Heuristic for Page Turning in a Multiprogrammed Computer," *Comm. ACM* Vol. 5 (9), p. 480 (Sept. 1962).
151. Weil, J. W., "The Impact of Time-Sharing on Data Processing Management," DPMA Quarterly 2, 2, 2–16 (Jan. 1966).
152. Wilkes, M., "A Programmer's Utility Filing System," *Computer Journal* 7, 180–184 (Oct. 1964).
153. Yates, John E., "A Time-Sharing System for the PDP-1 Computer," M.I.T. Press (1962).