The more important characteristics of real-time systems are listed, discussed in an historical context, and illustrated by remarking on relevant features of typical applications. Because the intent of the paper is to provide a general survey, no attempt is made at a truly rigorous definition of the term "real-time." The point of view taken is a functional one, viz., that the distinguishing properties of most real-time systems stem directly from the distinctive needs of five different classes of applications: control, command and management information, time-shared computing, remote batch computing, and data acquisition.

A number of general references are included for the reader who is interested in more detail on the various aspects of real-time systems.

Real-time systems in perspective*

by J. D. Aron

The concept of real-time systems began to attract attention in the data processing field in the 1950's.[1] At that time, the generally accepted definition of a real-time system implied a data processing system with a processing cycle determined by the user's deadlines for receiving the system outputs. Such a definition embraced several applications that today would probably not be considered real time at all. One such case involved the payment of government checks by the U.S. Treasury. In 1956, the difficulty of satisfying a statutory 24-hour payment deadline led the Treasury Department to look on the resulting data processing as a real-time problem. A similar, more stringent, deadline shaped the weather forecasting system of the U.S. Weather Bureau.[2] That problem in 1958 was based on teletypewriter inputs from weather stations all over the country that arrived daily before noon. This data had to be edited and processed in a complex solution of partial differential equations within 90 minutes, so that the results could be sent out in the form of weather charts over a facsimile system. Since the early meteorological forecasts were only good for 12 hours, the elapsed time from receipt of the input until maps were in the hands of local forecasters was critical to the

* *Originally delivered at the* IBM-*sponsored Real-Time Systems Seminar in Houston, Texas, November 2–4, 1966.*

economic justification of the system. To satisfy the elapsed-time requirement, both special equipment and procedural shortcuts were designed into the system. Payroll processing also had real-time connotations since, if the paychecks didn't reach the employees on the date they were expected, severe personnel problems could be anticipated. Today, however, a system with response deadlines measured in hours is fairly routine and is generally not included in the class of real-time systems.

Contemporary with the check reconciliation and weather forecasting applications, several other projects were developed in which response requirements were measured in minutes and seconds. Many of these were analog computer systems for controlling processes, such as the distribution of power over an electrical network, or the aiming of single weapons, or even the sequencing of aircraft landing on an aircraft carrier. Such applications are widespread today, but because the trend toward higher-speed control is leading to digital replacement of some analog components, only ground-based digital computer systems are discussed here.

The first major real-time system that set the pattern for many years was the SAGE air defense system.[3,4] (It is no coincidence that the best examples of major applications of new technology are found in the government and particularly in the military services. Because the urgency of military requirements outweighs the economic limitations of commercial enterprise, and because of the size and complexity of government operations, the government tends to lead commerce and industry by two to five years in the development of new computer applications.) SAGE was originally conceived as a large real-time command and control system. It included direct digital inputs from radar to a local computer that formatted the data and transmitted it, via direct line, to the huge central computers. These inputs were received on a fixed cycle measured in seconds. Other data inputs came at less frequent and less predictable intervals from aircraft, airbases, other SAGE sites, and weather stations. The central computers built up a dynamic data base from which the computer displayed the key information to military officers. These men could follow the air defense situation on their display screens and give commands to the computer through keyboards, light pens, and switches. The commands could, in turn, lead to outputs from the computer directly to aircraft or missiles to initiate defensive actions.

The significance of SAGE is that its designers planned it as a real-time system. They did not simply add features to a standard data processor and hope to get the job done. We now recognize that these are elements of the management of large data processing systems that we do not understand. The formalism of hardware/software system management is just beginning to take shape. SAGE is a landmark because it worked in spite of its immense size and complexity.

Characteristics of real-time systems

SAGE also helps us characterize the variables that discriminate among real-time systems. The diversity of inputs to SAGE resulted in not one, but many processes. The previously given definition of a real-time system is applicable to SAGE, but it is not very informative. A better definition would invoke a data processing system in which at least one process is critically defined with regard to elapsed time measured in fractions of a minute.[5] This definition does several things: real-time systems are narrowed down to those with fast response times, multiple deadlines are covered, and a mixture of critically and non-critically timed processes is covered.

This definition, however, does not include some characteristics that may also apply to real-time systems:

- Requirement that the aggregate of all subprocesses satisfy a critically defined deadline
- Assumption that the critically timed process within the real-time system produces control outputs
- Requirement for continuous system availability, implying some sort of backup equipment

Rather than attempt to define real-time systems rigorously, this paper only discusses some of their characteristics. Illustrative applications are classified quite arbitrarily into five groupings:

- Real-time control systems (including process control)
- Real-time command and management information systems
- Time-shared computing systems
- Remote batch computing systems
- Data acquisition systems

Each application class is described in terms of its constraints, both hardware and software. The emphasis is on those points that discriminate between real-time systems and other systems.

Some significant parameters characteristic of real-time systems are now presented.

real-time system parameters

Input rate Real-time inputs may arrive at fixed intervals, either in a continuous stream or in bursts of one or more messages. Inputs may also arrive at variable intervals. Variable inputs are either supplied by the source on demand from the computer or they interrupt the computer. In the latter case, the input must identify itself.

Input length Input messages may vary according to message type. Some inputs, such as punched cards, can be treated as fixed length. Other inputs must be described in a way that permits the system programmers to plan storage areas and input timings. Acceptable descriptions include:

- Length of next record is predicted by current data
- Length of records fits a known statistical distribution
- Maximum length of any record is known

Input types The number of input types affects the number and kind of formatting and editing programs, as well as computer storage allocation. It is possible for inputs to be unformatted, as in a library information retrieval system, but this is not common in real-time systems.

Input device response Most inputs automatically deposit their data in temporary storage buffers to give the computer some leeway in servicing an interrupt (or to permit simultaneous interrupts to be serviced one at a time). Other inputs, such as the following, are designed to be serviced immediately:

- Slow inputs that do not steal much computer time and for which buffers are uneconomical
- Alarms
- Redundant data that can be ignored if the computer is not available to service it immediately

Priorities A simple real-time system ordinarily has a built-in conventional priority scheme:

- Prearranged sequence — the computer has a finite number of tasks, each with a fixed, known priority and processing sequence
- First-In First-Out (FIFO) — the oldest transaction has the highest priority as indicated in a table or chained list
- Last-In First-Out (LIFO) — the most recent transaction has the highest priority as controlled by a pushdown list.

Convention may not suffice in all cases. For instance, military communications give each message a priority label that must be evaluated when the message arrives at the computer. In this situation, the computer arranges all messages by priority level and applies a priority convention to each level. To ensure that low priority items eventually are serviced, the computer may increase the priority after a certain elapsed time, or it may interleave the processing in some proportional manner with higher priority items.

Response time The time required to process an input and deliver the desired output is the response time. Response time is usually specified as a system constraint, and all elements of the system must be designed so that when they are integrated, the response time will be achieved. Typical requirements are:

- Fixed-precise response — the response time, usually in small fractions of a second, must be achieved within a tight tolerance
- Fast-average response — the response time, usually in seconds or minutes, must be achieved within a tolerance defined by a distribution function
- Delayed response — the response time must not exceed some time measured in minutes or hours

A distinction should be made between "real" and "apparent" response times. The real response time can be measured with a

timer. The apparent response time is a subjective evaluation of system performance made by a console operator in a man-machine system. Extensive human-factors tests are required to ensure that an acceptable real response time is, in fact, an acceptable apparent response time.

Operating characteristics Some sort of terminal input/output (I/O) device is found in a real-time system. The system may be dedicated to servicing only the terminal inputs. More likely, the system has more than one multiprogrammed capability. One of these capabilities may be for normal batch computing; this is called a "background program." Real-time services are "foreground programs." The numbers and kinds of allowed programs affect the computer capacity available to service a single terminal.

Output characteristics The output is controlled by the computer; therefore, it presents fewer distinctive problems than the input. The system is dependent on the rates of the output devices, since it must store output messages during the output transmission time. The system is also constrained in that the output is meant for immediate use at its destination. This leads to an increased requirement for error control that is represented by increases in computer time and storage.

Software considerations[6] Programs for real-time systems are constrained by the system deadlines. To keep the execution time low, some form of optimum programming is needed. Higher-order languages, such as FORTRAN and COBOL, although useful for some processing routines, are seldom acceptable for writing control programs because of compiler inefficiency. Standard vendor operating systems are often inappropriate because of their unnecessary generality. Real-time systems, therefore, incur the extra expense of machine-language coding for large portions of their programs.

Also, time constraints result in program segmentation problems. When programs plus data requirements exceed main-storage capacity, the programs are broken up and stored in external storage units. This creates an access-time delay that must be added to the execution time of the program in estimating real-time throughput. Therefore, simulation is used extensively to evaluate performance during the development cycle. When multiprogramming and segmentation coexist, storage protection is required so that one program will not accidentally be erased when another program relocates an overlay segment in memory.

Reliability Modern data processors have reliability characteristics (and repair times, in case of failure) that are good enough to permit some real-time systems to use normal equipment in the normal way. More commonly, the urgency that justifies the real-time system implies that interruptions due to system failure cannot be tolerated. The system specifications, therefore, may call

for "fail-safe" operation (where no system interruption is allowed) or for "fail-soft" operation (where some degradation of system performance is allowed, but total outages are not allowed). The goal of uninterrupted operation is achieved in various ways, all of which involve hardware.

- Configuration control registers are used to signal component failures.
- Parallel operations are specified so that either one of two computers operating in parallel can fail without losing application time. In space support operations, this approach is feasible since an orbiting vehicle is highly predictable; therefore, the second computer can rapidly catch up even if not in step with the first.
- Duplex computers are specified with the spare and active computers sharing I/O devices and key data in storage, so that the spare computer can take over the job on demand.
- Multiprocessing is specified, with several computers sharing the load in such a way that a single failure degrades but does not interrupt the system performance.[7]
- Manual backup procedures are specified by which operators can temporarily assume manual control at significantly degraded but safe levels of performance.

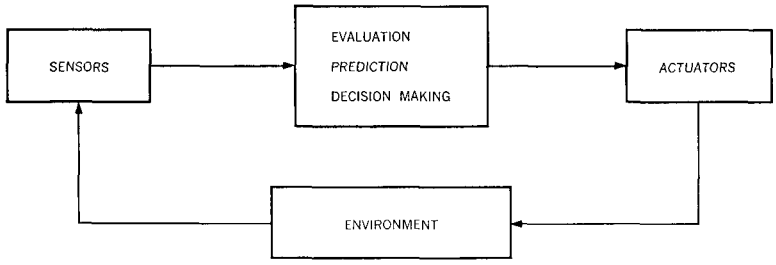
Real-time control systems

A real-time control system is one "which controls an environment by receiving data, processing them and returning the results sufficiently quickly to affect the functioning of the environment at that time."[6] The timely output that controls the external world is the key to the control system. SAGE fits the class. So too do systems that control paper mills, petroleum refineries, steel mills, and other continuous processes. In a control system, the algorithms for handling each transaction are well defined, and their execution times are precisely known. The priority sequence of inputs may or may not be fixed, but the inputs themselves are likely to have known rate and length. Response times are very short in both the real and apparent sense. To achieve the response times, many restrictions on function and sophistication are accepted. Where possible, the control system is "closed loop" in that its outputs are used immediately. This is true even when the output is displayed to human operators who close the loop as the result of their decisions.

The control system appears to the user to be an integrated complex of equipment that monitors the environment, evaluates the environmental parameters, predicts environmental changes, and then makes optimal decisions for controlling the environment so its changes stay within desired limits (Figure 1).

It is of relatively little significance to the user that there is a computer somewhere within this complex; the computer is simply one functional component of the system. Because of this

Figure 1 Control system



attitude, control-system computers are usually specified to exactly fit the initial system objectives with little room for expansion of the computing function. This approach succeeds in process-control systems where the number of components may expand, but the number of functions tends to remain constant. It is a risky system design in operational-control systems where functional growth is common.

process control

Many continuous processes, such as chemical manufacturing, petroleum refining, nuclear reactor control, and flow control of water resources and electric power, follow reasonably well-known laws that can be stated mathematically.[8] These processes can vary, due to changes in flow characteristics, input constituents or economic factors, but generally the control action to be exercised can be determined from the mathematics of the problem. Consequently, process-control systems can permit the control computer to make decisions to select and implement control alternatives. To do this, and at the same time reduce the requirement for human operators in the system, the environment sensors and control actuators are automated and tied directly to the computer. The computer must have a flexible interface connection to accommodate the many different input and output devices, but each device can be designed to provide a single simple message format for the computer.

Both the time constraints and the completeness of the control algorithm relieve the control computer from storing a large data base needed to help make decisions. Computer storage is primarily used to hold computer programs. A history of the job is maintained, but it is recorded outside the computer storage as soon as possible. Similar characteristics apply to such discontinuous control systems as communications switching (where each connection in the network is a discrete control activity), weapons control (where target and weapon position data are analyzed with weather and ballistic data in a continuous process to control a mission, but where the missions occur only occasionally), and in the ground checkout of rockets and missiles (where the checkout involves testing both continuous and discrete signals for a short period

of time). In all cases, the purpose of the computer is to perform an action that human operators could not reasonably perform in the allowed time. The function is often an optimization computation (linear programming, say, to determine the most profitable petroleum cracking policy), or a large analytical problem (determining how to control a multitude of switch interconnections for maximum efficient communication system throughput), or a precise guidance function (numerical machine tool control or missile guidance involving very fast, very precise feedback loops).

operations control

An increasing number of real-time control systems are being employed to improve the performance of government and business operations. The air-traffic control system is an example in which data from many sources, most of which are remote from the control center, are analyzed by human controllers who are responsible for safe traffic management.[9] The computers do not make air traffic control decisions, but they do organize and summarize the inputs from pilots, radar, weather teletypewriter, and other control centers so they can be displayed in a form the human controller can use. At the minimum, the computers relieve the controller of many clerical tasks that formerly interfered with his ability to concentrate on making control decisions. In addition, the computers increase the accuracy and readability of the data displayed to the controller. Because of the emphasis on the human decision maker, "apparent" response time is significant in this and in other operations-control systems.

The input and output units of an operations-control system tend to be standard devices, such as teletypewriters, in large numbers and at great distances. Except for such inputs as a local radar, most messages arrive at low speed due to the limitations of the communications line. Each input may have several message formats, but these are all designed so that the computer need do very little syntactic analysis. The human operator requires a variety of visual and audible outputs. Display units are built to user requirements and may be equipped to superimpose dynamic data and computer output.

The success of government real-time systems has encouraged commercial real-time systems for the control of business operations. The airline reservations systems reflect this.[10] The competitive success of the airline depends on its ability to fill all of its seats. This ability is achieved by providing fast, responsive service to airline customers. By maintaining a data base of all reservations for all scheduled flights, the reservation system performs the inventory control function of checking each input (from a terminal at the reservation office) against the data base and making a reservation when space is available. Similar services are being installed by banks and brokerage houses whose profit position is based on holding down interest payments and credit risks. More and more keyboard terminals are appearing in banks, stock brokerage offices, and check cashing locations.[11] Whenever a transaction is entered, it triggers a search of bank balances, credit

Table 1 Real-time control systems

	Critical response time	*Input rate*	*Number of input types*	*Number of simultaneous inputs*	*Priorities*	*Reliability*
Process control						
Physical process optimization	real	low to high	medium	high	fixed	fail safe
Natural resource control	real	low	high	high	fixed	fail soft
Electrical network flow	real	low	low	high	convention	fail safe
Communications line switching	real	low to high	low	high	convention	fail soft
Weapons control	real	high	low	medium	fixed	fail safe
Automated ground checkout	real	high	high	high	convention	fail safe
Operations control						
Air traffic control	real and apparent	low to high	medium	high	convention	fail safe
Ground support of space mission	real and apparent	low	medium	medium	convention	fail soft
Airline reservations	apparent	low	low	high	convention	fail soft
Banking transaction and credit control	apparent	low	low	high	convention	fail soft
Brokerage order processing	apparent	low	low	high	convention	fail soft

ratings, or margin account balances before completing the transaction. This protects the financial institution without delaying or annoying the customer.

Table 1 shows that within the class of control systems there are variations of characteristics among the applications. This table and the others in this paper are not intended to be precise or quantitative; they only show relative features.

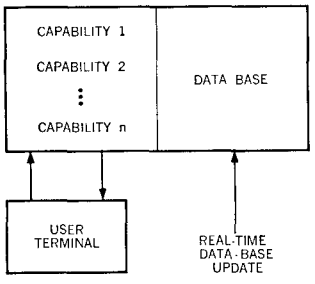
Command and management information systems

Real-time command and management information systems differ from control systems in that they supply information rather than control. Their inputs are queries from user terminals. Their outputs are answers to the queries obtained by analyzing the contents of a large data base. The inputs vary widely in rate, length, and type. Each message has a priority label and, in some systems, a security label that restricts access to individuals with the appropriate security clearance. Because of the wide variety of inputs and the need for extensive syntactic analysis, command and management systems may require large programs. It is quite normal for the user to be vague in his specifications for such a system; until he uses it, he cannot really verbalize his needs. As a result, the system is always in a state of flux that further increases its size. Since this flexibility is important to

the user, he is willing to pay for it by relaxing real-time constraints on everything but the apparent terminal rseponse times; this he will never relax because, if his people will not use the system because it seems too slow, he will have no system at all.

Those systems that retain tight deadlines at some loss of flexibility are called command systems or management information systems.[12,13] Their purpose is to provide information needed for management decision making. They are characterized by a finite, well-structured data base on a predetermined number of subjects, a limited number of terminals, and a limited number of capabilities that can be drawn upon by the use of a limited vocabulary query language. Thorough training of the user personnel gives them all the apparent flexibility they need, since their job is defined within the scope of the system. As indicated in Figure 2, the user looks on the system as a collection of processing capabilities, any one of which is available for his use. He does not expect to use the computer in the system for anything other than these capabilities. He may call on the system to list the names of company personnel with certain skills, but he knows the system is not programmed, for example, to solve a set of simultaneous equations. The system is dedicated to the main task, but there may be periods, such as third shift, when other applications displace the main task. Duplexing is common but not essential, since manual backup is usually practical.

Figure 2 Command management information system



The inputs to a command or management information system include messages that are ultimately intended for some other destination. The information system is often used as a communications message-switch responsible for routing all incoming messages to their destinations, since it can extract data for the data base as the messages pass through.[14,15] In one type of existing management system, the performance of a communications system is the subject being managed: to ensure the existence of connection paths in a long distance telephone network, long-lines managers monitor the status of all lines and make decisions as to routine policies and priority assignments.

computer-assisted instruction

Command and management systems may take advantage of "conversational mode" programs to improve the apparent response time.[16] This involves fast responses to inquiries, usually displayed on a Cathode Ray Tube (CRT). The response provides the best answer to the query, with cues as to where additional information is likely to be found. This mode is obviously an advanced and difficult capability to provide but, since it makes the user feel that he is talking effectively to the machine, it substantially increases his desire to use the system. A special case of conversational real-time systems is Computer-Assisted Instruction (CAI), in which the student is guided through his studies by a computer program.[17] Here, the visual information transfer, perhaps supplemented by audio, has been shown to be an effective teaching aid. One advantage of CAI of interest to government and industry is its ability to let each student learn at his own

speed. The individualized instruction is well suited to training new employees and upgrading experienced employees without the need for rigid classroom schedules.

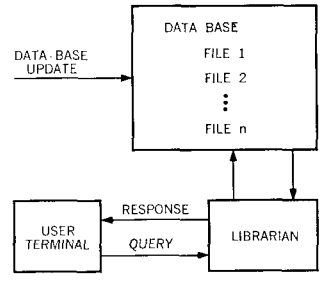
information storage and retrieval

A potential for real-time systems can be found in Information Storage and Retrieval (ISR) applications. As the technology for information retrieval (as opposed to document retrieval or report generation) grows under the impetus of military-intelligence users, the demand for real-time response will grow. Current capabilities indicate that very complex inquiries can be handled at relatively high cost. In the next decade, however, costs should come down to a level that will make it economically feasible to build multi-terminal ISR systems for general use.[18]

There are several reasons for wanting an ISR system to operate in real time.

- The apparent response time determines whether a scientist or engineer will use a library. If service is slow, he will reinvent a solution that may be available in the literature.
- ISR effectiveness requires a comprehensive data base or library file. Updating this file automatically with real-time inputs that are combined with editing and abstracting consoles may improve the quality of the current portion of the data base.
- The utility of ISR is improved when remote users can obtain information as easily as a user whose office is next to the library. This requires a capability for the user to browse through the data base. The mechanism that permits a remote user to browse is a real-time conversational mode terminal.

Figure 3 Information storage and retrieval system



Whereas the command-system user looks on his system as a set of capabilities tailored to his decision-making needs, the ISR user considers his system to be a fairly complete library that contains broad, loosely structured data and requires some skill to search. See Figure 3. Rather than a black box with one or more clearly defined functional capabilities, the ISR system appears to be a huge library in which there is a computer that has all the experience and analytical ability of a good librarian. With this view, the user expects to be able to ask complex questions in free English. He expects to get fully responsive replies; ranging from a brief statement, to a clearly arranged summarization report, to a complete book. Although these expectations complicate the input translation process and the output-display hardware, they are by no means insurmountable problems. Table 2 classifies information systems in the same general terms as Table 1.

Time-sharing systems

It is customary to optimize the efficient use of expensive resources by centralizing them. This has been the case in computing as is seen in the large number of closed-shop batch-processing facilities. Centralization has increased the turnaround time to the user; i.e., the apparent response time (which includes all the time from the user's readiness to make a computer run until he gets the results

Table 2 Real-time information systems

	Critical response time	*Input rate*	*Number of input types*	*Number of simultaneous inputs*	*Priorities*	*Reliability*
Command and management						
Business management information	apparent	low	medium	medium	convention	normal
Military command	apparent	low	medium	high	labelled	fail soft
Message switching	real	medium	low	high	labelled	fail soft
Long-lines status control	apparent	low	low	high	convention	fail soft
Computer-assisted instruction	apparent	low	low	high	convention	fail soft
Information storage and retrieval	apparent	low	low	high	labelled	normal

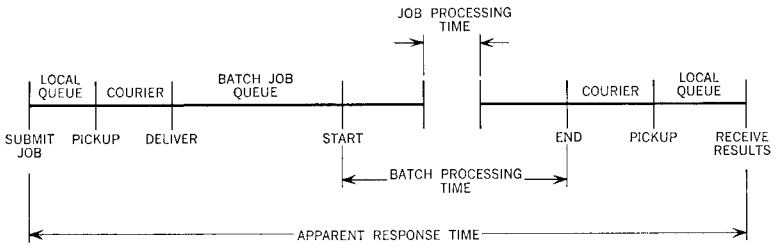
Table 3 Time-sharing and remote-batch systems

	Critical response time	*Input rate*	*Number of input types*	*Number of simultaneous inputs*	*Priorities*	*Reliability*
Time-sharing systems						
Engineering design	apparent	high	low	medium	convention	normal
Program development	apparent	low	low	medium	convention	normal
Computer service center	apparent	low	low	high	convention	fail soft
Remote-batch system	apparent	high	low	low	convention	normal

of the run) is quite long (Figure 4). In industry, the turnaround time ranges from two hours to two days. The user either adjusts his work plan to mesh with this turnaround cycle, or he attempts to bypass the central facility. The significance of this problem in some environments has prompted a technological solution to improved turnaround. One approach, called "time sharing," provides the best apparent response time at the expense of computer overhead.[19,20]

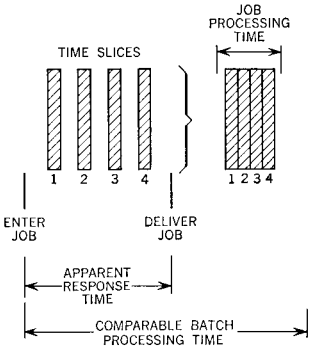
A time-sharing system consists of a computer system with many user terminals. Each terminal is permitted to use all of the available resources for a short slice of time. A control program assigns time slices to each terminal according to a priority convention and ensures that work in progress at the end of each time slice is properly saved. Since the control program itself uses part of every time slice, the total work required to complete a set of jobs in a time-sharing system is greater than in a centralized batch system. On the other hand, time sharing eliminates the

Figure 4 Centralized batch processing



keypunch, courier, and queuing delays that make up a large part of the batch response time. The average user sees a considerably shorter apparent response time in a time-sharing system than in a centralized batch system, as suggested by Figure 5. This is particularly true of the user who has a short job that can be easily submitted at typewriter speed and that produces answers suitable for printing on the same typewriter. Limitations of time sharing include a lack of facility for dealing with long programs or large data bases that cannot readily be typed in. In addition, trade-offs must be made in the cost and size of the control program versus the size of the computer available to each user. In a system with 512,000 characters of high-speed storage, 10,000,000 characters of bulk storage, and 40 terminals, it may be necessary to partition each user's resources into, say, 32,000 characters of main memory and 250,000 disk blocks in order to avoid time-consuming relocations of programs and data in between time slices. If much program relocation is necessary, the system throughput will be significantly degraded in comparison to batch processing. The trade-off decision must be made in terms of the applications planned for the time-sharing system. Table 3 identifies three types of time-sharing to be discussed.

Figure 5 Time sharing



engineering design

The first impetus toward time sharing came from universities and industrial organizations when it was found that their centralized batches contained hundreds of jobs that used less than a minute of computer time. Yet these short jobs were not delivered to the user until the whole batch was completed. Analyzing the nature of these jobs, the users identified a large number of problem-solving jobs submitted by engineers and scientists who needed the answers before they could continue with their projects. One might call these jobs "engineering design" because they are steps in a sequential technical design process that directly affect a designer's productivity. By placing a value on engineering manpower resources, it is possible to show that the economic utility of speeding up the computing time justifies the cost of the time-sharing system. Fortunately, the processing characteristics of the jobs involved are optimal for time sharing. The designer generally wants to evaluate a complex function of a few variables. The

task is too big for a desk calculator, but it can be done on a relatively small general-purpose computer; therefore, a partitioned time-sharing system is adequate, particularly if a language that is easy for the engineer to use is acceptable.

Once the time-sharing system is available, major additional benefits accrue. The engineer who formerly waited half a day or more for answers now finds that immediately available answers stimulate him to find the next logical step in the design. In effect, time sharing turns his intellectual activity into a more continuous process. A reinforcement is achieved, similar to the "conversational mode" mentioned earlier, that accelerates the design process. This is tangibly evident when the design requires subjective visualization as in designing automobile body shapes or magazine page layouts. Consequently, advances in graphic I/O are closely related to time sharing.[21,22]

program development

A second large category for short machine runs is program debugging. Progress toward time sharing for programmers is slower than for engineering design for several reasons.

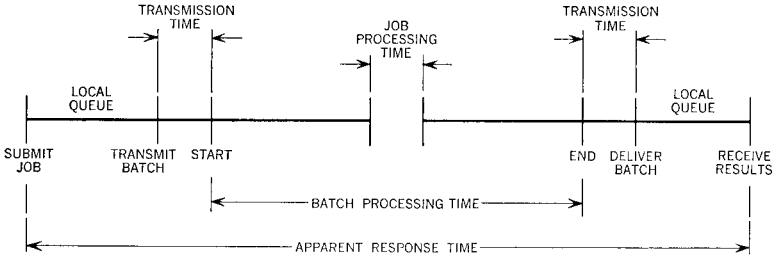
- Programs and data used to test them are voluminous
- Program interfaces are not always defined clearly enough for one programmer to debug independently of other programmers
- The value of different phases of programming is difficult to establish since the phases are not clearly defined

These problems will be overcome. The initial high-volume inputs can, of course, be handled by batch procedures. Thereafter, small changes to correct bugs can be entered through a terminal. The man-machine symbiosis important to engineers can be expected to apply equally well to programmers. Technical and procedural standards will clarify interface problems, so debugging a single program module or subroutine will be a clean-cut task. There is research underway that indicates special time-sharing systems will be able to automatically control interface specifications, so that several components of a program system can be separately tested. Such capabilities will permit the clarification of programmer tasks that is necessary to determine the effect of time sharing on programmer productivity.

computer service center

Engineering design and computer program development are candidates for time-sharing systems because the man at the terminal has a high economic utility that justifies the effort to optimize the use of this time.[23,24] A different type of time-sharing system satisfies the small-business user who would like to have the advantages of a computer, but cannot afford it. Perhaps he currently uses a service bureau, but that is not the same as having an in-house capability. This type of user would be happy to time-share a large-scale computer with similar users. The first problem to solve is getting him a terminal that he can afford. Having done that, the next step is to design a time-sharing control program that will serve hundreds of terminals, each having a diversity of jobs. System integrity is more critical here than

Figure 6 Remote batch processing



in other types of time sharing because the business records of the user are in the machine. System failures may result not only in service delays, but also in serious business losses. There seems to be little question that the computer service center will become a reality, but current indications are that severe limitations must be placed on the flexibility of early systems in order to guarantee system integrity.

Remote-batch systems

Any discussion of time sharing highlights the fact that time sharing is an expensive replacement for inefficient batch operations. Cannot the keypunching, courier, and queue services be made more efficient? In general, they can; if top management insists on getting good user service, they will place as much emphasis on apparent response time as on batch throughput. Nevertheless, the most efficient centralized batch processing system will have real delays due to the distance from the user to the facility. Time sharing eliminates queues and travel time by giving each user a terminal that communicates directly with the computer. But time sharing carries with it limitations on I/O volume and, to some degree, computer resource availability. When the user requirements are really better suited to batch processing, time sharing is less appropriate than remote batch processing (Table 3). Remote batch processing involves a computer at the user terminal, rather than simply a typewriter alone. This low-cost computer is used to prepare a local batch of programs for direct transmission to the large-scale central computer. There is still a local requirement for keypunching and queuing. However, since a small number of jobs are involved, the average delay is short. Then, by tying the remote terminal to the central computer over high-speed transmission lines, the courier delay is eliminated (Figure 6). Appropriate scheduling leads to relatively short batches being processed, so results can be transmitted to the user to achieve an improvement in apparent system response time. This type of real-time system is gaining in significance because low-cost terminal computers are now feasible, and because the remote-batch method is highly compatible with the existing procedures.

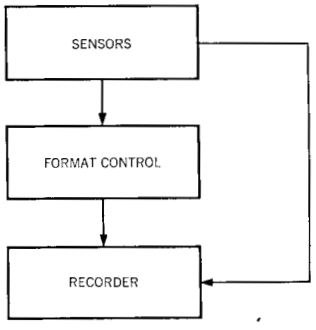
Table 4 Data acquisition systems

	Critical response time	*Input Rate*	*Number of input types*	*Number of simultaneous inputs*	*Priorities*	*Reliability*
Process industry	real	low to high	medium to high	high	fixed	fail soft
Biomedical instrumentation	real	high	high	medium	fixed	fail soft
Geophysics	real	low to high	high	high	convention	fail soft

Table 5 Summary of real-time system characteristics

	Critical response time	*Input rate*	*Number of Input types*	*Number of simultaneous Inputs*	*Priorities*	*Reliability*
Real-time process control	real	low to high	low to high	high	convention or fixed	fail soft or fail safe
Real-time operations control	apparent	low to high	low to high	high	convention or fixed	fail soft or fail safe
Real-time command and management information systems	apparent	low	medium	high	labelled	fail soft
Time-sharing systems	apparent	low	low	medium	convention	normal
Remote-batch systems	apparent	high	low	low	convention	normal
Data-acquisition systems	real	low to high	medium to high	high	fixed	fail soft

Figure 7 Data-acquisition system



Data-acquisition systems

With the definition adopted earlier, elapsed-time constraints are sufficient to classify an application as a real-time system. Previous examples have had a critical response time; i.e., the elapsed time from input to output is critical. There remains a class of real-time systems where no real-time outputs are involved, but where elapsed-time constraints on input govern. These systems acquire data in real time. Their inputs are similar to the inputs in real-time control systems, but their outputs consist of only an edited record of the inputs. The design approach for data acquisition is essentially the same as for control; in fact, data-acquisition systems are often the first step in a phased implementation of a real-time control system. They are simple, inexpensive, and easily adapted to a variety of input interfaces, formatting requirements, and editing procedures (Figure 7). Table 4 indicates that data-acquisition systems are used to continuously record high data rates for future analysis.

process industry

Certain processes[25,26] have such a slow reaction time that real-time control is not needed. Other processes have no known control algorithms, so closed-loop control is not feasible. Data acquisition is intended to build a history of the process, so at least a subjective evaluation of the process is possible. In the food processing industry, process monitoring permits managers to ensure that the ingredients satisfy standards of quality and profitability. However, the final evaluation of product quality is a taste test that, to date, has never been successfully automated. In each application area listed in Table 1, there is a corresponding data-acquisition application that is a low-cost solution involving manual control.

biomedical instrumentation

The diagnosis of disease is hampered by the lack of good data on cause and effect. Valid statistics on physiological processes require substantially more data than can be manually recorded. As a result, many separate biomedical instruments,[27-29] as well as integrated physiological monitors, are being built to collect useful data. An ultimate, but difficult, objective is the use of real-time data-acquisition systems to provide patient data that can be analyzed for immediate use by the attending physician.

geophysics

Current knowledge of weather and other geophysical phenomena is inadequate for control of weather. It has been determined, as mentioned earlier, that analysis of meteorological data permits good prognostication of short-term weather. The success of the weather forecasting system is based on real-time data acquisition that not only supplies data to the forecasters, but also provides historical data for scientists and climatologists who are interested in improving and extending the forecast validity. The National Meteorological Service System and the World Weather Watch are planning the real-time data-acquisition system that will ensure continued improvements in the future. Similarly, the Large Aperture Seismic Array (LASA) project is being planned as a real-time data-acquisition system for detecting and evaluating earth tremors (earthquakes or man-made tremors).[30,31] Other data-acquisition systems cover the ionosphere, the ocean, and deep space. All are aimed at improving our ability to predict the impact of our environment on our activities.

Concluding remarks

The premise of this paper is that real-time systems have started to proliferate and that it is desirable to recognize different kinds of systems to avoid over-generalization. Table 5 summarizes the five categories discussed in the paper. No attempt has been made to be mathematically precise in assigning applications to one and only one category. It is, of course, easy to show overlaps and redundancies in the tables. On the other hand, it is important to understand the features and capabilities of the various kinds of real-time systems in order to design a system for a particular

purpose. This paper suggests that five kinds of systems are available to satisfy five kinds of objectives:

- Control a process
- Provide information for decision making
- Improve user turnaround time for problem solving
- Improve system turnaround time for batch processing
- Collect data

No belief is held that these categories are the only pertinent ones—or even the best that could be defined. But a strong recommendation is made that the design of a real-time system can be improved by drawing on the experience of others who have built categorically comparable real-time systems.

GENERAL REFERENCES

1. *Proceedings of the Eastern Joint Computer Conference, 1957*, The Institute of Radio Engineers, New York, (December 1958).
2. G. P. Cressman, "Numerical weather prediction in daily use," *Science* **148**, No. 3668, 319–327 (April 16, 1965).
3. W. H. Tetley, "The role of computers in air defense," *Proceedings of the Eastern Joint Computer Conference*, American Institute of Electrical Engineers, New York, 15–18 (December 1958).
4. C. M. Seacat, "The SAGE system: a program approach for non-programmers," System Development Corporation TM-591, (April 1961).
5. F. P. Brooks and K. E. Iverson, *Automatic Data Processing*, John Wiley and Sons, Inc., New York, 459–460 (1963).
6. J. Martin, *Programming Real-Time Computer Systems*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, (1965).
7. P. A. Welch, "On the reliability of polymorphic systems," *IBM Systems Journal* **4**, No. 1, 43–52 (1965).
8. "Progress in automatic control applications," *IEEE 1965 International Convention Record* **13**, Part 6, Session 6, (March 1965).
9. J. A. Fusca, "Air traffic control," *Space/Aeronautics* **43**, No. 6, 56–61 (June 1965).
10. R. W. Parker, "The SABRE system," *Datamation* **11**, No. 9, 49–52 (September 1965).
11. R. V. Head, "Banking automation: a critical appraisal," *Datamation* **11**, No. 7, 24–28 (July 1965).
12. *Armed Forces Management* **12**, No. 10, 43–112, 151–158 (July 1966).
13. J. Dearden, "Myth of real-time management information," *Harvard Business Review* **44**, No. 3, 123–132 (May–June 1966).
14. J. Atwood, J. Volder, and G. Yutzi, "Data control message switching systems," *Datamation* **11**, No. 2, 26–31 (February 1965).
15. "Electronic switching: a score of years of organized attack," *Bell Laboratories Record* **43**, No. 6, 194–280 (June 1965).
16. S. L. Smith, "Man-computer information transfer," *Electro-Technology*, **72**, No. 2, 83–87, 94 (August 1963).
17. C. E. Silberman, "Technology is knocking at the schoolhouse door," *Fortune* **LXXIV**, No. 3, 120–125, 198–205 (August 1966).
18. National Information Center, Volumes 1 and 2, "Hearings before the Ad Hoc Subcommittee on a National Research Data Processing and Information Retrieval Center of the Committee on Education and Labor," Appendix to Vol. 1, Parts 1, 2, and 3, U. S. House of Representatives, (1963).
19. "Special report: time-sharing computers" *Electronics* **38**, No. 24, 71–89 (November 29, 1965).

20. E. E. David, Jr., "Sharing a computer," *International Science and Technology*, No. 54, 38–44 (June 1966).
21. J. C. R. Licklider, "Man-computer partnership," *International Science and Technology*, No. 41, 18–26 (May 1965).
22. D. Christiansen, "Computer aided design: Part I, the man-machine merger," Electronics **39**, No. 19, 110–123 (September 19, 1966).
23. R. S. Stein, "Computer power as a public utility—an evaluation," *On-Line Processing Symposium: Proceedings of Two Sessions on On Line Data Applications To Be Presented at the Winter General Meeting, New York, New York, January 27–February 1, 1963, IEEE Special Publication* S-143, 33–50 (January 1963).
24. S. Wernikoff, "Information services computer center," *Western Union Technical Review* **20**, No. 3, 128–137 (July 1966).
25. *IBM 1800 System Reference Manual*, A26-5918, IBM Data Processing Division, White Plains, New York.
26. *Principles of Data Acquisition Systems*, E20-0090, IBM Data Processing Division, White Plains, New York.
27. E. C. Greanias, "The computer in medicine," *Datamation* **11**, No. 12, 25–28 (December 1965).
28. R. L. Patrick and M. A. Rockwell Jr., "Patients on-line," *Datamation* **11**, No. 9, 57–60 (September 1965).
29. H. W. Mattson, "Future hospitals," *International Science and Technology*, No. 56, 30–37 (August 1966).
30. F. Press and W. F. Brace, "Earthquake prediction," *Science* **152**, No. 3729, 1575–1584 (June 17, 1966).
31. *Proceedings of the IEEE*, **53**, No. 12, 1816–2098 (December 1965).