*Reprinted from –*
**AFIPS – Conference Proceedings, Volume** 36


# Computer resource accounting in a time sharing environment

*by* LEE L. SELWYN*

*Boston University*
Boston, Massachusetts

## INTRODUCTION

The past several years have witnessed major stages in the evolution of time sharing service suppliers toward the (perhaps) ultimate establishment of a computer utility or utilities that will, presumably, resemble other public utilities in many ways. This paper is concerned with the development of managerial accounting techniques that will enable suppliers to broaden their range of services and allow some of them to evolve into vertically integrated information service organizations.

### *Background*

The early time sharing suppliers provided a relatively narrow range of services; the general-purpose systems usually provided access to but a small number of languages and virtually no proprietary software. Indeed, if the latter was available, access to it was provided to customers of the service at no additional charge. Other services provided *only* access to some proprietary applications package, and usually did not offer the generality of access to a programmable service. However, with respect to this latter type of supplier, a charge was indeed imposed for access to the applications software, although it was embedded in the overall cost of the service.

The particular pricing policy established by any one supplier was, perhaps as often as not, forced upon it by some limitation in the computer resource accounting mechanism associated with the time sharing operating system. Hence, many firms "lived with" some schedule of charges that were almost certainly suboptimal. This, in many instances, resulted in substantial limitations upon the overall variety and type of computing services they could provide.

In an earlier paper,[1] the author, along with D. S. Diamond, considered the implications of various types of services upon the pricing policies of a time sharing service supplier. Several possible strategies were considered at that time. These included:

- Transaction-based charges for access to some applications package or data base, where the approximate quantity of computing power required for a given transaction was either (a) reasonably predictable, (b) an insignificant part of the total costs associated with provision of the service, or (c) negligible with respect to the "value" of the service to its end-user.
- Prices based upon resource usage (e.g., cpu time, connect time, core residence, etc.) for (a) general-purpose programmable services, (b) for access to a proprietary applications package where the quantity of computing resources consumed is unpredictable, highly variable, and a major component of costs. (Of course, the resource prices for access via the proprietary package could be higher than for general access to the system.)
- Flat rate for unlimited access to the system, or perhaps a variation on flat rate, such as elapsed connect time. This may be appropriate in cases where the nature of use of the system was sufficiently limited such that this type of rate structure would not result in abuse, or in cases where the charge for a unit of service was so small that the cost of accounting for service was prohibitive. (It should be noted that, with the exception of a poorly designed operating system, such conditions are rather difficult to imagine in practice.)

Whatever pricing structure is eventually adopted

---

* The author was a research participant at Project MAC, Massachusetts Institute of Technology, which provided partial support for much of the work reported here. He is presently Assistant Professor, College of Business Administration, Boston University.

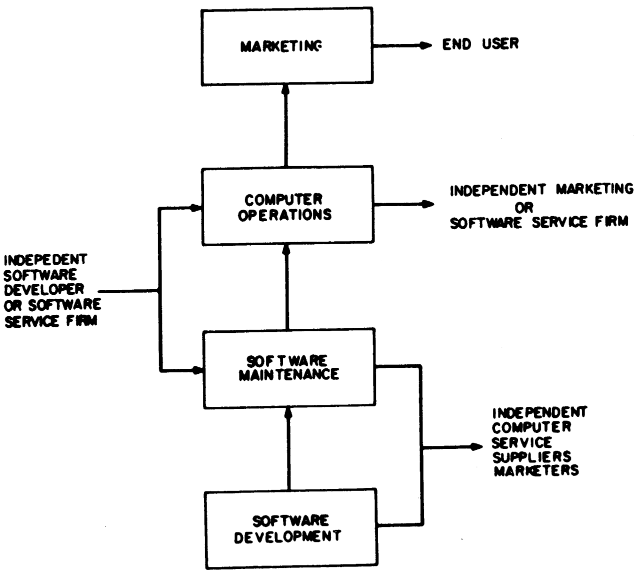


Figure 1—Organization of an information service firm

must be the result of consideration along a number of dimensions. The pricing structure must be "market oriented" so as to satisfy the needs and objectives of customers. The user must be made to feel as if he is paying only for his share of the computer, that activities of other users do not affect his charges. He wants to be able to predict, with some degree of accuracy, what his costs will be. He will base his purchase decision, presumably, upon the value of the proposed computing service to his organization. A second dimension is "operations oriented." The pricing structure must induce users to behave in a manner that is consistent with the best interests of the operators of the facility. If it is found desirable to have a relatively level load on the system, then lower prices must prevail in the less desirable times of the day or week. Users must not be permitted to "hog" the machine, even if they have the funds to pay for the service, since this could cost the computing service other customers who balk at the (perhaps temporary) degradation of service they receive.

Thus, a pricing structure for a computer utility must be flexible enough to handle a variety of service types, must be accurate to a point that satisfies customers' needs for fairness and predictability, must encourage the use of proprietary software with royalties or other payments accruing to the owner, and must be consistent with the requirements of a well oiled operating procedure. The present paper considers the requirements of a managerial accounting and control mechanism to support such a pricing structure, with the stated requirements.

The next section considers the management information requirements of time sharing firms, in light of the current evolution of the industry. Specific design objectives of such an accounting mechanism are then examined in the following section. Finally, the paper concludes with discussions of implementations of such systems by the author—one on the IBM 7094 Compatible Time Sharing System at Project MAC, MIT, and the other on the PDP-10 system operated by Codon Computer Utilities, Inc.

## EVOLVING NATURE OF THE TIME SHARING BUSINESS

*Services offered by time sharing firms*

Where the early time sharing service suppliers provided only a limited range of services, it is becoming increasingly clear that the suppliers of the future will offer a much wider range of services at a much broader level. In effect, a time sharing firm may be thought of as a vertically integrated information service organization with activities ranging from the production of the raw computing power, the development of applications programs and other software, the maintenance of such software, and the retail marketing of its product to the end-user. Figure 1 illustrates a possible organization of an information service firm that provides all of these types of services.

One of the more significant developments in the time sharing industry has been the entry of firms that specialize in some subset of these four major activities. Thus, a software developer may only write an applications package, and may then turn over its maintenance, marketing and operation to an information service organization. Alternatively, the same software developer may perform his own maintenance and marketing, and use the time sharing service only as a source of computing power. The time sharing service, on the other hand, may choose to establish its own retail marketing outlets, or may instead sell its services to a retailer who will assume the marketing risks and rewards within, perhaps, a particular geographical region.

As the industry continues to develop along these lines, the nature of particular arrangements made between its members will grow increasingly more complex. The software developer that chooses to do his own maintenance and marketing will perhaps pay the time sharing supplier for the computing resources he has consumed and then go on to charge his own customers at whatever rate is appropriate. Alternatively, he may license a package to a time sharing house either on a flat rate or a royalty basis; in either case the two firms must somehow keep track of the use of the subsystem by the ultimate customer.

The necessary record keeping associated with these various levels of activity could, of course, be done at each level; the computer operations area (or firm) could simply measure the quantity of computing resources consumed by each of its customers, which could be end users but might also be software owners' packages, independent marketing departments or firms, etc., and render statements accordingly. The software owner would then develop his own accounting system to measure the use of his subsystem by each of his customers; the marketing organization would similarly have to account for resources used by each of its end-users, and so on. As an alternative, a single information system may be constructed that provides for appropriate managerial and financial control at all levels.

The author began work on the development of such a system while at Project MAC at the Massachusetts Institute of Technology, and has since designed a more complete information system structure for the DEC PDP–10 while serving as a consultant to Codon Computer Utilities, Inc., of Waltham, Massachusetts. The principal design objectives, features and capabilities of these systems are described in the following section.

## DESIGN OBJECTIVES

We have already suggested a rationale for the development of an information system for time sharing services that takes account of the vertically integrated nature of the business. Such a system must be based upon several key design objectives, which will be discussed presently.

### *Computer resource allocation*

The system must provide a mechanism whereby it is possible to allocate access to computer resources among the various end-users, subsystems, retailers, and in-house software development efforts.

Even the largest time sharing computers in operation today can support but a mere handful of simultaneous users; in most cases under 50 and in virtually no instance over 100. As a result, the actions of any one user on the system can, and often do, have a noticeable impact upon all of the other users in the community. In an economic sense, the users are "oligopsonists," i.e., there are relatively few buyers (of time sharing service) in the (captive) market associated with the single "monopolistic" machine. Of course, this description of the market structure does not hold for even the medium run, let alone the long run. There are now numerous time sharing service suppliers, and numerous customers, such that something more closely resembling a competitive market exists. However, in the very short run, a given user is, in effect, captured by a given system. As an oligopsonist, his actions affect both the system and the other users.

Hence, his access to the system must somehow be controlled, irrespective of his ability to pay. As an example, a user desirous of obtaining the entire machine for a period of, perhaps even five minutes would create much unrest among the other users. As a result, the time sharing monitor will normally use some sort of round-robin or related method of scheduling jobs so as to prevent a user from obtaining this much service in this short a time. But what of the user who requires some large number of simultaneous lines and many connect hours, plus perhaps some very large quantity of mass storage, over a relatively short period of time, perhaps a week or two. This could place an unnecessarily heavy load on the system and once again cause some unrest among the other users. This would, of course, be perfectly reasonable, from the point of view of system management, if this new heavy demand were permanent; but if it is only a very temporary thing, then the system, and the remaining customers, must be protected. In effect, some rationing scheme is indicated. By such a mechanism, a user reserves, in advance, that quantity of system resources he is likely to require during some time interval, perhaps a month. By this technique, system management may allocate available resources among its customers in an attempt to even the load on the system. For example, the day may be divided into several periods, such as peak and off-peak, and different rations might apply to each such period for any customer. By this technique, the user can be assured of more even levels of service at all times, and system management may more accurately forecast its facilities requirements.

### *Flexibility in pricing*

Although the resource management system will use the firm's pricing structure as a basis for its record-keeping, it should by no means be bound or limited by the specific pricing mechanism that has been selected by management. There are several reasons for this. Perhaps the most obvious is that the marketing strategy of the firm, and hence its pricing policy, may be subject to some modification as time goes by. Moreover, if in

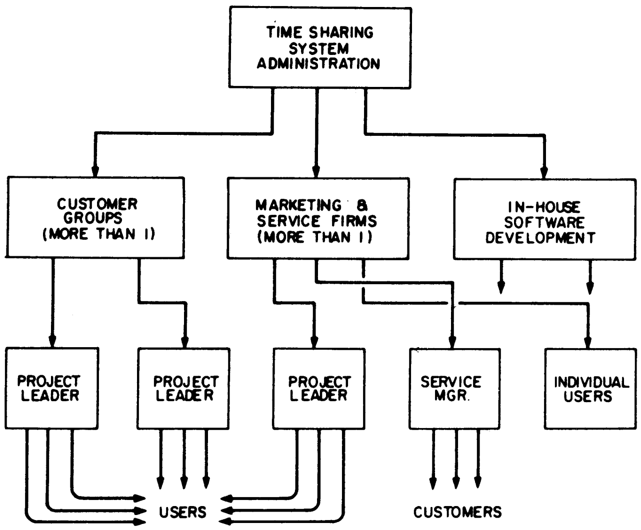


Figure 2—Resource management structure

fact the firm is going to deal with independent software developers, independent retailers, or both, it will not want to impose its pricing structure upon these firms in the latters' dealings with their own customers. Hence, the system should be capable of supporting a variety of pricing policies at the same time. It should have the capability of charging the intermediary at prices established by mutual negotiation or any other means, and then permit the intermediary to impose virtually any sort of charge upon the latters' customers. This separation of wholesale and retail types of charges should be reflected in all other parts of the system, from resource allocation to billing.

*Decentralized management*

Since the time sharing supplier will be dealing with some intermediaries, it is necessary that the latter be provided with some resource management and administration tools, thereby enabling control over the activities of the intermediary's customer. Moreover, the actual customers of both the time sharing firm and its intermediaries can greatly benefit from such local resource management facilities. Thus, a customer's project leader should be able to directly manage use of the computer among the several participants in his project. He should be permitted to allocate some set of resources given to him among users in his project. Similarly, an intermediary should be allocated some pool of resources and should be permitted to directly allocate these among its customers according to whatever method it wishes.

In effect, the time sharing supplier need deal only with intermediaries, its own marketing organization, and customers. Each may then manage its own use of the computer and allocate the ration of system resources among those users and activities subject to its control. Such a resource management structure is illustrated in Figure 2.

*Access to system resources and services*

The foregoing implies a fairly tight method of controlling access to the computer. This is especially true where price differences apply to different classes and types of services. Thus, a user of a proprietary subsystem who pays a higher price for system resources than does a general time sharing service user must be restricted to use the system only through the account specifically established for this purpose. Similarly, a general purpose time sharing service user must be prevented from gaining access to any proprietary service for which some unusual charge applies, without first establishing an account for the use of such services. For any end-user, the operating system must know (1) who is responsible for charges (i.e., a customer or an intermediary), (2) what type of service this user has subscribed for, (3) what resources have been allocated to this end-user, and (4) which price shcedule applies to the user and which to the responsible account (which may, of course, be the same).

*Detail of system usage*

It is possible for a time sharing system to keep a detailed log of all transactions that take place within it. It is, for example, possible for every system command to create a message, stored somewhere in the system, that indicates the time, date, name of the command, the identity of the user, and the quantity of resources used in the execution of the command. This would, however, be a fairly costly procedure, in that system overhead would be increased to handle each of these message creation and storage. From a managerial point of view this may be quite unnecessary. However, there are certain types of individual services which must be accounted for in great detail, both from the standpoint of billing and collection of revenues, and from the standpoint of operations analysis and control. Thus, services provided by proprietary software that is to be charged on the basis of transactions performed rather than computer resources consumed must be maintained in great detail so that the customer may know how he has spent his money and that the software developer may examine the relative efficiency of his programs. Moreover, the time sharing system management will want this information to compute that portion of the subsystem's total revenues to retain for providing service.

*Subsystem usage accounting*

Proprietary subsystems that offer services to be charged on a "value of service" or transaction basis necessarily have pricing structures radically different from any resource-based rate schedule. In fact, just about the only similarity between any two such subsystems is that they both compute their rates according to some unique set of rules determined by the developer of the application package. The resource accounting mechanism must not only cope with such a variety of structures, but should additionally encourage innovation on the part of software developers to use any arbitrary mechanism that they desire. Hence, the accounting system should permit the subsystem to compute its charges and to report to the accounting mechanism these charges together with some identification of the nature of the service provided to the end-user. The accounting system should be able to record these subsystem-imposed charges in the end-user's account, retain the details of the transactions for billing and analysis purposes, and to impose upon the subsystem itself a charge based upon the resources consumed in completing the user-initiated transaction. In effect, the subsystem becomes the customer of the time sharing system, the end user is charged by the time sharing system on behalf of the subsystem. (That is, the time sharing firm offers, as a service to software developers, a means by which it will handle all of the paper work associated with the software developer's customers' accounts.)

## IMPLEMENTATION

Many of the features mentioned here were included in a resource accounting system implemented by the author on the IBM 7094 Compatible Time Sharing System developed at Project MAC at the Massachusetts Institute of Technology.[2,3] The "BUYTIM" resource allocation system was designed to operate wholly within the CTSS operating system, and required no monitor-level modifications to the CTSS software. The CTSS implementation is described below.

When Project MAC started charging for CTSS usage in January of 1968, a need arose for some changes in the mechanism by which users requested additions to or changes in their set of allotments of computer resources. Formerly, these resources were allocated directly—that is, by specific allocations to each user, of time, divided into five shifts, and disk records for storage of user files. Under the new scheme, users receive dollar budget allocations, from some sponsored project, and use these funds to purchase resources on the CTSS System. The mechanism described here serves to give the individual user more direct control over the means by which his available funds are spent.

The system provides for computer resource management at a number of levels, ranging from top-level administration down to the level of the individual user. Under the pre-existing resource management system this function was assumed, within the MIT Information Processing Center, at two levels: Administration and User Group Leader. The Administration held ultimate responsibility for computer resource allocation; it classified all users into one of about twenty user groups, and determined the particular mix of resources that were to be made available to each of the groups. The Group Leader was responsible for apportioning the resources allotted to his group among its members. This responsibility is not altered under the BUYTIM set-up. However, what is provided to the Group Leader is a means for further delegating his responsibility—and power—to individual members of his group at two principal levels: the Problem Number Leader and the Individual User.

The Problem Number Leader is afforded the capability of apportioning resources allotted, by the Group Leader, to his problem number among its members, a relationship vis-a-vis the Group Leader not unlike the latter's relationship to CTSS Administration. The Problem Number Leader is further afforded the capability of delegating some of his authority to individual members of the problem group, with several levels of decision-making capability. That is, the problem number leader may designate another user in the problem group to have identical capabilities as himself, or he may choose to limit these capabilities in some way. At the limit, the individual may be given the ability to make changes in only his own account, or may be denied even this capability.

User preferences for computer resource allocations are subject to two types of limitations: pricing and rationing. Resource prices have been established by CTSS administration and govern the allocation of time and disk, as well as of several other "commodities." Only the time and disk are covered by this system. In addition to the constraints of the costs of CTSS resources, users are further constrained by several restrictions which limit their ability to spend their allocation of funds as they might please. First, each user group is given a set of Group Limits containing the maximum amounts of each commodity that may be

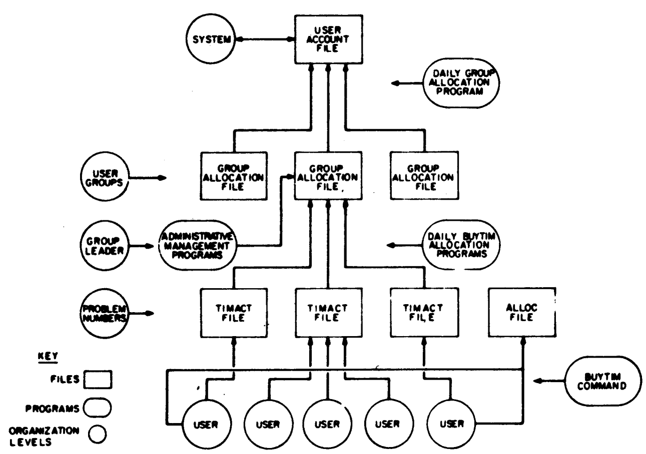


Figure 3—BUYTIM system files and programs

allocated among the group's members. Under the BUYTIM system, the group leader may further break this set of limits down among individual problem numbers in the group, or may have all problem numbers share the common pool of resources available to the group, or a combination of these techniques. To further restrict the user's freedom, the group leader may establish a maximum increment by which a user may increase his allocation over his actual usage to date. At the beginning of each calendar month individual user *time* allocations are reduced to nominal beginning of month levels, also established by the group leader for each user, thus requiring the latter to repurchase additional requirements for the new month.

The BUYTIM system communicates with the CTSS Administrative management programs by updating the Group Allocation File (GAF), a method used by all of the individual user groups within CTSS. As a result, it is not necessary for all user groups to subscribe to and implement the BUYTIM system; its use is totally transparent to the pre-existing allocation mechanism.

Two new file types were created for the purposes of effecting communication between the individual user and the GAF, and a number of programs were written to permit suitably formatted and protected modifications of these files. The principal file in the system is the 'TIMACT' file. There is one such file for each problem number in a subscribing user group. In addition, there is a file of the form "GRPXX ALLOC" for each user group. All of these files are maintained in a special directory in Private, Protected mode. Hence, these files are accessible directly to the Group Leader, and are made accessible to users only through a special privileged CTSS command which may modify these files. Figure 3 illustrates the file structure in this system.

The use of these TIMACT files makes it possible for the Group Leader to subdivide the overall Group allocations of time and disk into blocks available to each problem number within the group. However, in some cases, particularly where the problem number contains only one or a few programmers, this feature may be undesirable, since the overall group limits would be segmented to a point where flexibility to meet individual user needs, within the pricing mechanism, would be seriously restricted. In such cases, the Group Leader may assign a particular problem number to a common pool of resources. This means that, instead of getting problem number limits from the TIMACT file, the limits come from the file GRPXX ALLOC.

The BUYTIM command provides an on-line mechanism for requesting changes in CTSS time and disk allocations. It permits a user to "spend" the funds allotted to him, at the prevailing prices for the several commodities as contained in the file SYSTEM PRICES, subject to availability, and also enables the user to change his password. In addition, BUYTIM provides varying capabilities for project leaders to make changes in funds and computer resource allocations of other users within the same problem number group. A wide range of access privileges are available for this purpose.

Although the individual CTSS user and his project leader are provided with the ability to trade-off among the various CTSS resources at the established prices, this capability is limited to the extent that such resources are available to the project group.

### *Classes of use of BUYTIM*

There are seven distinct classes of use of BUYTIM, each of which affords the classified user with certain privileges, in terms of what types of changes he may request via the command. The class codes for each user are contained in the TIMACT file, which also contains the other account information about the user. The class designations are as follows:

0. No access to BUYTIM whatsoever.
1. User may change his own time and disk.
2. User may change his own time, disk, and password.
3. User may change his own time, disk and password, and may change time and disk of other users in the problem number group.
4. User may change his own time, disk and password, and the time, disk and passwords of other users in the same problem number group.
5. User may change his own time, disk, and password, and the time, disk, and funds of other users in the same problem number group.
6. User may change time, disk, password and funds

allocations of himself and all other users in the same problem number group.

*Change codes*

There are nine change codes available, plus a termination code, as follows:

| *Code* | *Description* |
|---|---|
| *t*1 | Shift 1 time |
| *t*2 | Shift 2 time |
| *t*3 | Shift 3 time |
| *t*4 | Shift 4 time |
| *t*5 | Shift 5 time |
| disk | Disk records |
| pass | Password changes |
| funds | change in funds allocation |
| prpass | print user password (class 4 and 6 only) |

The '*' is used as a termination code, to denote the completion of a series of changes.

To change time, disk or funds allocations, a user types the appropriate change code followed by the amount of the change in integral minutes of time, disk records, or dollars of funds allocation. The amount of the change may be a signed or unsigned number. If signed (e.g., +25, −40) the present level will be changed by that amount. If unsigned, the number is assumed to be the new level of the allocation, and will thus replace the old one. For example:

TYPE CHANGES: *t*1 +10 *t*3 −5 *t*4 20
disk +50 funds +100 *

This will increase shift 1 time by 10 minutes, reduce shift 3 by 5 minutes, set the shift 4 allocation to 20 minutes, increase the disk allocation by 50 records and increase the funds allotment by **$100**.

Charges are levied by BUYTIM on the basis of allocation, rather than usage. An unused allocation may be returned for full credit at the prevailing PRICES at the time of the return. In the case of time, the charge imposed (or credited) is 1/60 of the prevailing hourly rate for each integral minute of time allocation purchased (or returned). In the case of disk space, the charge (or credit) is based upon 1/30 of the prevailing rate per disk record per month times the number of days left in the month. Thus, a disk allotment is paid through the end of the month, and credits are figured from the date the space is released until the end of the month. BUYTIM does not consider the other charges, such as those for the monthly account maintenance fee and the U.F.D. charge. These should be estimated by the Group Leader and subtracted from the funds balance appearing in TIMACT.

Because BUYTIM charges on the basis of allocation rather than usage, the dollar balances obtained from these alternate methods will usually differ somewhat. However, note that BUYTIM is provided for the convenience of users and Group Leaders, and where a discrepancy exists the figures based upon usage, as provided by CTSS charge statements, will prevail.

BUYTIM will reject several kinds of transactions:

Unauthorized use

1. Unauthorized use of BUYTIM (class 0 user).
2. Attempt to change account of another user (classes 1 and 2).
3. Attempt to change specific items of another user or of own account not permitted by class designation.

Allocation restrictions

4. Attempt to increase allocation of time or disk above maximum increment set by Group Leader or Problem Leader.
5. Attempt to reduce allocation below current usage.
6. Insufficient funds.
7. Increase in allocation exceeds available resources.

There are several capabilities available to the Group Leader that are not available to the individual user or problem number leader within the BUYTIM command. Several other programs were written to facilitate these functions by the Group Leader. For example, the group leader may add or delete users, may assign various types of access privileges and restrictions, and may apportion the group resource limits among the individual problem numbers in the group.

All changes made by users, problem leaders, and group leaders are recorded in the appropriate TIMACT file (although the group leader may occasionally modify the Group Allocation File directly). The modified TIMACT file records must be posted to the Group Allocation File in order for the CTSS resource allocation programs to recognize the changes. This is accomplished by means of a self-rescheduling Foreground Initiated Background job run each evening by the group leader. (A mechanism in CTSS permits such jobs to be scheduled and run automatically, without the presence of a user at the time the job is run.) Hence, changes made during any given day cannot be recognized by the time sharing system until they have, first, been posted to the Group Allocation File, and second, been copied from the Group Allocation File into the primary system accounting files. This latter activity usually occurs at midnight, also via a self-rescheduling job. Thus, changes made via the BUYTIM mechanism will usually take effect at or around midnight following the change request.

*PDP–10 implementation*

A major extension of the concepts developed in the BUYTIM system has been designed for the PDP–10 time sharing system in operation at Codon Computer Utilities, Inc.[4] The new features of this extension are described here. It should be noted, however, that unlike the CTSS version, several monitor-level system modifications were required for the PDP–10 design. As a result, the system is not transparent to the operating system, but forms an integral part of it. Besides extending his own concepts, the author wishes to acknowledge the work by Thomas H. Van Vleck of MIT, who developed the overall CTSS accounting mechanism (within which BUYTIM operated) for a number of the ideas incorporated into the PDP–10 version.

Unlike the CTSS case, the newer version was built directly into the time sharing monitor, replacing the manufacturer-supplied resource accounting mechanism, rather than simply operating within it. As a result, any alterations to the user accounts take effect immediately, rather than at some later time when the changes might be posted to the user accounts, as in the CTSS case. Several important new features were, additionally, added.

## Dynamic pricing

Under the CTSS version, the principal control on usage, during real-time operation of the time sharing system, was the central processor time consumed by an individual user. Thus, a user might have received an allocation of, say, 20 minutes of prime shift processor time for the current month. When that allocation is exhausted, the user is automatically logged off the computer by the monitor, and is prevented from logging back on during that shift for the duration of the month, or until he can secure from the group leader, or through the use of BUYTIM, additional allocation for the shift. In a commercial environment, the operator of a computer utility may desire to control other resources besides the amount of straight processor time employed by his customers. Moreover, his pricing mechanism may be non-linear, in that lower unit prices may apply for larger quantities of a given resource consumed. Further, depending upon the nature of the customer, (e.g., an end user or an intermediary) the nature of his application, and any special terms negotiated with him in contracting for service, it is conceivable that several different rate schedules may have to be devised and used simultaneously during the real-time operation of the computer. In the PDP–10 version, four distinct types of charges may apply to a user account during a console session.

Central Processor, or "Computation"
Transaction service usage
Connect time
I/O device usage

The computation charge is based upon the processor time and the core residence during execution of the user's job. The applicable rate may be non-linear, in that as core residence increases, the unit charge for a space-time unit (kilo-core-second) may vary.

Certain services of a computer utility may be marketed in terms of the "service" they provide, rather than the quantity of system resources they consume. Such "transaction" services are charged at varying rates, the exact charge being determined by the particular proprietary program that provides the service. The charge may be based upon what is being done, how much of it is being done, and, perhaps in some cases, who is doing it.

Connect time is the elapsed time between login and logout. It may vary according to the type of terminal (e.g., use of the system from a high-speed CRT display terminal may be charged at a higher connect-time rate), and perhaps at a different rate depending upon whether or not the job is in an "attached" state or in a "detached" state, wherein no console is physically connected to the computer for the detached job.

Use of I/O devices, such as line printers, magnetic tape drives, card readers and punches, etc., are charged for at rates applicable to each device. Further, a set-up or minimum charge may also apply in some instances.

In each of the four types of charges, the specific structure may vary among classes of users, as well as the time-shift in which the service is provided.

Under the dynamic pricing technique, each user is given money allocations for each applicable time shift, and is free to spend his allocated limits on any of the four types of services just described. Whenever some service is provided, except in the case of transaction services, the system computes the quantity consumed, determines the applicable rate structure for the customer in the current time shift, and proceeds to charge the account the money cost of the service. If this charge brings the balance for the shift below zero, the user may be logged off the computer, with an appropriate message explaining the reason for this action. In the case of transaction services, the transaction program itself computes the applicable charge and, by a suitable monitor call, conveys this information to the monitor. Specific resource usage (computation, connect time, and I/O device usage) is charged by the monitor to the transaction service, in a special account maintained for this purpose, and not to the user. No negative balance check is made against the transaction service account. When the transaction service informs the monitor of a charge to the user, the user's account is charged and a negative balance check is made. Control is returned to the transaction service in any event but, where a negative balance condition exists, the transaction program is so informed and is expected to take action. Thus, there is a very important assumption made about the nature of a transaction program. It is considered to be a well debugged program that is fully responsible for all accounting interactions with the monitor on behalf of the user. It must compute the charge, inform the monitor of its conclusion, and take appropriate action in the event the user's funds have been exhausted.

Under the PDP–10 time sharing system, it is possible for the same user to be logged on the system several times simultaneously. Thus, it becomes necessary to coordinate the charges incurred by each of the several possible jobs in simultaneous operation in a single funds balance. Thus, as soon as any one of the jobs logged in under the same project-programmer number does something that results in a negative balance, *all* of the jobs subject to this balance will be logged off the system.

Bills are rendered to a responsible account, rather than directly to the user. Of course, these may be the same person. However, by this mechanism, an independent retailer may render his own bills to his customers. Alternatively, the billing system may prepare such bills for the retailer. The flexible price structure mechanism enables the individual user to be charged, during real-time use of the computer, under a rate structure that may differ from the one for which services will be billed to the responsible account. Thus, an independent retailer may convey to the computer utility his own rate structure, based upon which the utility will charge and prepare bills for the retailer's customers. The wholesale prices charged by the utility to the retailer need not be the same, either in level or structure, as the latter's retail prices.

### Accounting for services used

The large variety of things that a user may be charged for in a computer utility system requires an accounting mechanism that can collect, maintain, display and summarize the detail of individual user activities. Moreover, in the case of proprietary transaction services, detail as to the nature of individual types of services provided, their quantities and applicable charges, is most desirable from the point-of-view of the customer. Further, system and application subsystem management will want such detail so as to best analyze the relative efficiencies of the various services offered, to perform market demand studies, and other management analyses.

The accounting system designed for Codon seeks to provide these features. Each user account record maintains a breakdown of the dollar value of resources used in each of the four principal categories and in each of the applicable time shifts. A user may obtain this information for himself from a logged in console by a suitable monitor command.

Thus, during a console session, the monitor maintains a record of the consumption of the four "temporal" resources and, upon logging off the system, reports these figures back to the user account record. Moreover, a record of the individual console session is created, containing the resource usage data, time and date, and other relevant information, and is maintained for later processing and auditing purposes.

In addition to the temporal resources (temporal in that they are consumed over time) the system accounts for use of "spatial" resources, such as mass storage occupancy. The technique here is quite analogous to the scheme employed within CTSS, as described earlier. Besides accounting for disk storage, the monitor will not permit a file to be opened for writing if the quota of disk blocks has been exceeded. When such a condition occurs, the user must either delete some files to free up some space, or have his allocation of disk space increased.

Specific transaction services are recorded as they are provided, by means of records that contain data on the user's identification, the type and quantity of transaction services provided, and the applicable charge. (E.g., preparation of 40 payroll checks @ 20¢ each, $8.00.) At the same time, the charge to be imposed to the user is also added to the transaction service's account for auditing purposes. At the end of an accounting period (e.g., a month) the transaction service records are sorted by user and summary statements are prepared showing the basis for the aggregate transaction service charge.

Usage of input/output devices is also handled by a similar detailed recording procedure. A record is maintained for each access to a particular device, indicating the duration of access and the applicable charge for the service. These records may also be summarized at the close of an accounting period and presented in a detailed statement to the user. System management may also be provided with a detailed picture of the relative demand for access to the various peripherals on the computer.

Thus, the billing process will generate a summary bill, which provides aggregate charges for each of the four temporal resources, the principal spatial resource, disk, and any other charges related to the account. Further, a detailed statement of specific transaction services supplied may be produced, as may a similar detailed breakdown of I/O device access. Finally, an historical record of the individual console sessions may be generated by the accounting system.

Applications subsystem owners, whose software is offered on a transaction basis by the computer utility, may be provided with a statement of the resources consumed by their respective programs, as well as the revenues generated by their subsystem's customers. A similar type of detail may be provided in the case of usage of peripheral devices.

Finally, it should be pointed out that the accounting system incorporates all transactions between the customer and the utility, and provides a convenient mechanism for posting miscellaneous charges and credits, such as a charge for consulting service or a credit for a system failure, directly to the user's account record maintained on the system.

### System access

The nature of access to the computer, by individual users, may be restricted by the system administrators in several ways. In the simplest case, it may be desired to restrict access to only some of the time shifts during which the machine is in operation. This may be handled quite simply by setting the funds allocations in the restricted shifts to zero. The system will not permit a user to log in during these times, since the balance remaining in his account, for the shift, is zero.

Where a user subscribes to the computer utility for a specific proprietary applications subsystem, it may be desired to restrict access to that subsystem. Moreover, access to the subsystem programs must be restricted to only those users who have subscribed to its service. These objectives may be accomplished by a fairly straightforward procedure. First, applications subsystems are accessed by special system commands which perform a login procedure for the user invoking them. If the user is to be restricted to a subsystem, the name, or some other unique identification, of that subsystem is placed in the account record. The normal system login procedure checks to see that no such restriction exists; if it does, login as a general time sharing user is not permitted. The login procedure in each subsystem must verify the equality of the subsystem restriction with its own requirements. For example, a subsystem might accept several legal subsystem identifications in the user account record, but assign different levels of service privileges to the user based upon the particular code that is present in his account. (Of course, a subsystem might not perform any such check, allowing any subscriber to access it and purchase its services.)

On occasion, it may be desirable to limit the number of simultaneous jobs that may be active for a group of users, e.g., a number of individual users all associated with the same customer of the computer utility. This capability permits the utility to offer guaranteed access for a stated number of individual users to a large customer organization. This concept may be implemented in two ways: a guaranteed *minimum* number of lines, or a guaranteed *total* number of lines. In the former, the customer would be guaranteed that he could always have some specified number of lines; if he were using all of these, he could obtain additional lines on a contention basis with other customers who do not have any guaranteed access. In the latter case, the customer cannot exceed his guaranteed number of lines; additional users will be prevented from logging in until another user in the group has logged out.

We have already considered the case of multiple jobs for the same user account. The system has the capability of placing an upper limit on the number of times a user may be logged in. However, for a user in a guaranteed access group, the use of multiple logins on the same user account will count toward the access guarantee for every time a single user is logged in.

### Resource management

The preceding sections have described the control capabilities of the PDP-10 design. In order to administer the system, a mechanism has been provided for communicating with the accounting structure at a number of levels, similar, but with some significant extensions, to the BUYTIM system on CTSS described earlier. The program that enables such communication is implemented as an application subsystem that may be made available to virtually all users of the computer, but subject to several distinct levels of access. The same type of management levels that exist in the CTSS version are available here—utility administration, user group leader, project leader (analogous to problem number leader in CTSS) an individual user. In the CTSS version the group leader could allocate individual sets of resource limits to each problem number, or place some or all in a common pool. A similar capability exists here, except that any number of resource pools may be established within the same user group, instead of only the one available in CTSS. Generally, each level of management has greater direct control capabilities than were available within the CTSS version. The following table summarizes the principal capabilities of each level of resource management, under the PDP-10 version.

RESOURCE MANAGEMENT CAPABILITIES

| *Level* | *Capabilities* |
|---|---|
| User | May have no access to subsystem, or<br>May do any of the following, as determined by project leader:<br>Alter own resource limits, password, or funds.<br>Alter any or all of the above for other users in same project. |
| Project Leader | May do any of the above within his project.<br>May assign any of the above levels of permission to users in his project.<br>Add or delete users in his project.<br>Examine the accounts, including passwords, of users in his project.<br>Designate specific applications subsystem access to any user in his project, subject to restrictions imposed by Group Leader. |
| Group Leader | May do any of the above.<br>Designate project leaders for projects within his group.<br>Assign and remove individual project numbers to and from resource pools.<br>Change allocation limits for projects and resource pools within the group. |
| Utility Administration | May alter rate structure applicable to individual user accounts.<br>Alter guaranteed access group assignment for individual users, projects and customer groups.<br>Change overall resource limits for individual customer groups.<br>May add and remove project numbers to or from a customer group.<br>May create and destroy customer groups.<br>May alter the number of lines, as well as the nature of, a guaranteed access group.<br>May enter any extraordinary charge or credit to any individual user account.<br>May set, for each user, the number of times he may be simultaneously logged in.<br>May establish and modify the list of subsystems available to the user group. |

**Security of the System**

Several procedures and mechanisms have been incorporated into the design of the resource management system to provide protection from accidental or deliberate sabotage by users. All accounting files are maintained in a manner such that access to them is only possible by means of one of several system commands. Further, these commands are responsible for restricting access to the accounting files according to the level of authority of the user invoking the command. This permission information is maintained in the user account record. When executing the accounting commands, the user is prevented from returning to monitor level until all files have been closed and access has been terminated. Moreover, should a user accidentally be able to get to monitor level, the system will not permit him to do anything except log off the computer.

Similar restrictions apply in the case of use of an application subsystem. Once access has been gained, control cannot be returned to the time sharing monitor unless and until the program so desires. User-initiated interrupts are intercepted by the monitor and control is passed back to the proprietary subsystem then in execution. The subsystem is, of course, responsible for taking whatever action it considers appropriate. A normal exit from such a subsystem implies a logout. If a user wishes to use both the subsystem and the general purpose time sharing service, he must establish separate accounts for each purpose.

We have attempted to present a description of how a managerial accounting information system for a computer utility can significantly expand the scope of activities of such an organization. The approach to system design has been the result of actual administrative experience with such systems over a period of several years. This experience has shown the importance of such capabilities.

As time sharing service organizations become more complex in their structure and diversified in their activities, the need for a well-structured information management mechanism will no doubt become more critical. This implies that time sharing operating systems will, more and more, have to be designed with the necessities of system administration in mind. Operating systems deficient in this respect will find it difficult to provide the range of services necessary for survival in an increasingly more competitive industry environment.

## REFERENCES

1 D S DIAMOND L L SELWYN
*Considerations for computer utility pricing policies*
Proceedings of 23rd National Conference Association for Computing Machinery pp 189–200 Brandon/Systems Press Princeton New Jersey 1968

2 P A CRISMAN Ed
*The compatible time sharing system: A programmer's guide*
Second Edition MIT PRESS Cambridge Massachusetts 1965

3 L L SELWYN
*BUYTIM: A system for CTSS resource control*
Unpublished memorandum Project MAC MIT Rev July 1969

4 *Codon resource allocation system: User's guide*
Codon Computer Utilities Codon Corporation Waltham Massachusetts 1970