TO: C. Clingen
F. Corbató
J. Gintell
N. Morris
R. Roach
J. Saltzer ✓
T. Van Vleck
S. Webber

FROM: V. L. Voydock, R. J. Feiertag

DATE: October 7, 1971

SUBJECT: A proposal for a distributed accounting system

Enclosed is a sketchy design overview of a distributed accounting system for Multics. It is in a very rough form. We feel, however, that the basic ideas of the design (e.g., that accounting should be distributed) should be approved before expending the substantial amount of effort it will take to produce a more detailed design, and indeed to determine whether such a design can be efficiently implemented. The document, therefore, concentrates on the external interface of the accounting system, that is, what features it offers, and discusses implementation only to give some indication of the feasibility of the design.

In the course of designing a new system control for Multics it has become clear to us that the nature of this design will be influenced by the nature of the Multics accounting system. Therefore, we feel that the basic properties of the accounting system should be decided upon before the design of system control is completed. One important basic property is that the accounting system should be independent of the rest of system control.

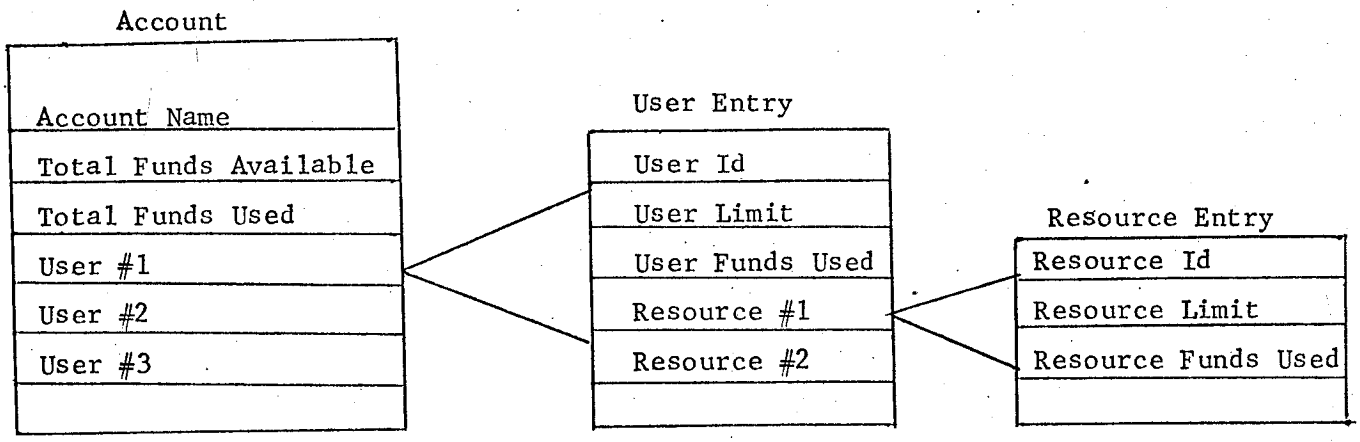
Currently, accounting is done by the user control process which places information into a transaction segment at regularly scheduled intervals. This scheme has two major disadvantages both of which are related to limit stops. First of all the information in the transaction segment is not cumulative, but merely describes what the process has done in a given interval. Thus, one cannot easily determine how much of a given resource a given user has used. This makes resource limit stops quite difficult to implement.

The second major problem is that the scheme does not generalize well. It is difficult to account for new resources. This comes from the fact that accounting is performed by a dedicated process rather than being distributed, and the fact that in order to correctly implement resource limit stops there must be two way communication between the resource manager and the accounting system. That is, when a user exceeds a resource limit stop, the resource manager wants to prevent him from using any more of that resource; in order to do this it must be told by the accounting system that the limit has been exceeded. Since the accounting system is in a separate process, this two-way communication is both awkward and inefficient. One could argue that each resource manager should maintain its own limit stop but this spreads knowledge of accounting throughout the system. It also makes it difficult, if not impossible, to enforce an overall limit stop on a user. That is, a limit stop on his total expenditures, irregardless of resource. We have, therefore, concluded that the Multics accounting system should be distributed and will now propose a design for such a system.

By "distributed" we mean that accounting tables are stored in an inner ring and are updated by calls to accounting primitives made by the various resource managers. Each resource is separately accounted for and a running total is kept of its usage. Each resource has a limit stop which is enforced "as soon as possible". Funding and resource management are strictly separated. The accounting system deals with the former, the various resource managers with the latter. Thus, the accounting system does not enforce limits on the number of tape drives or the amount of secondary storage a user can use: the device manager does the former, the secondary storage quota manager the latter. On the other hand, the resource managers do not know about shifts or the price of their resource. Their interface to the accounting system is in terms most appropriate to them (e.g., "number of records written" for the tape manager). The primitives must convert these units to the common unit of the accounting system (probably dollars).

It is the responsibility of the resource manager rather than the accounting system to decide what to do if a limit stop is exceeded. The accounting system merely notifies the resource manager that such an event has occured. The usual action taken will be to cut-off the use of that resource.

The accounting system will keep track of everything for which one incurs a charge including secondary storage usage. This usage will no longer be kept in the directory (except possibly for redundancy). To minimize accounting mistakes due to system crashes, etc., some sort of redundant information will be produced -- possibly an audit trail of all transactions. Since they are entirely orthogonal the two accounting systems (current one and new one) will be able to run concurrently. This will greatly aid the debugging of the new system and will give us a measure of how accurate and bug free it is. With these preliminaries in mind we move now to more detailed description of the new system.



Simplified View of an Account as Seen From Outside Accounting System.

Acctng is [illegible] for [illegible]

Description of an account

The account is the basic billing entity of the accounting system. All charges incurred by an account are charged as a unit to a single funding source. From an external view an account contains three basic entities as shown in the accompanying diagram. These are the total charges incurred by this account so far in this billing period, the total funds allocated to this account which may not be exceeded by the total charges, and a set of subaccounts. Each subaccount represents the accounting for a particular user associated with this account. Each user subaccount contains the total charges incurred by this account on behalf of the user, a user charge limit which the user charges cannot exceed, and a set of resource usage entries. Each resource entry contains the total charges incurred by this account on behalf of this user for the use of this resource and a resource charge limit which the resource charge cannot exceed.

The account, therefore, provides three levels of detail: the individual resource level, the user level, and the overall account level. Each level has limit stops which gives each account administrator the ability to control his funds disbursement in as flexible or rigid a manner as he so chooses. Limit stops, of course, can be set to zero or infinity as well as intermediate values. Limit stops can be modified dynamically by the system administrator or possibly even by individual users subject to access checks. All three levels of usage, i.e., resource, user, and account are updated simultaneously and are checked against all three levels is this kept updated at all time? Very hard to implement!

of limit stop. The user level usage is the total of all resource usages associated with the user and the account total is the sum of all the user usages. In order that usage totals can be determined all usages will be recorded in terms of the same units, possibly dollars. For convenience, it may be deemed necessary that individual resources be kept in other units as well. Individual resources will report charges in terms of their own units and the accounting system will convert to the common unit in order that price rates be maintained at a central location where they are visible and easily modifiable as each installation sees fit.

Operation of the Accounting System

The usages are updated by a call to the accounting system by the resource manager for the resource for which the charge was incurred. For example, the traffic controller will invoke the accounting system to update the cpu time resource usage. The IO daemon will invoke the accounting system to update IO Daemon resource usage. The typewriter DIM and the user control mechanism will update the typewriter and registration usages respectively. Examples of other resources are core usage, tape usage, retrieval usage, secondary storage usage, and quota usage.

It can't call!

When a resource manager invokes the accounting system to record a charge it is notified as to whether or not the charge has caused a limit stop to be exceeded. It is the task of the individual resource manager to react to a limit stop overrun, not the job of the accounting system.

It is expected that the usual action by the resource manager upon being informed of a limit stop overrun would be to discontinue use of that resource. For example if tape usage is overrun the use of tapes would be discontinued by the tape DIM or if cpu time usage is overrun the user would be logged out. However, in some cases discontinuation of use of the resource would be inappropriate. For example, if secondary storage usage is overrun, rather than delete the segments involved, quite a drastic action, it would probably be better to simply notify the user and the system administrator, and dump the segments on tape before deletion.

Primitives to record resource usage will be provided by the accounting system for use by resource managers. These primitives will be available to resource managers in any ring. Special consideration will have to be given to certain resource managers, e.g., the traffic controller which is wired down. These special considerations will be discussed in greater detail below.

Account limit stops will be dynamically modifiable by primitives in ring 1. It is expected that the primitives available to account administrators and users to manipulate limit stops will be written in the administrative ring.

At regular intervals, e.g., monthly, the current charges to an account will be saved and the values reset to zero,in order that the charges may be collected and billed.

As well as updating usages when a charge is incurred the accounting system will keep records of transactions separately in order that consistency checks can be made at regular intervals and discrepancies resolved when errors occur.

Implementation

The following discussion indicates some of the problems which are foreseen in the implementation of the above accounting system and describes means by which some of these problems can be overcome. However, the implementation should be considered as an appendix to this document, present only to indicate the feasibility of the system and should not be allowed to reflect on the basic ideas presented above.

An account can be most easily represented as a segment containing the above described information. Account segments could be kept in a predefined place in the file system hierarchy. When the account is to be updated, the accounting system must initiate the account and modify the affected values. However, under many circumstances initiating the account is both inefficient and in some cases not possible. As mentioned above, certain resource managers require special consideration. The page control mechanism and the traffic controller cannot update an account which is a paged data base. For these reasons certain data in accounts must be placed in segments where they are more readily accessible to resource managers.

Look in MSPM at the AMT

Two new segments will serve this purpose: the Active Account Table (AAT) and the Wired Account Table (WAT). When a resource manager is about to place a resource in service it calls the accounting system to activate the account to which the resource is to be charged. Activating the account implies placing parts of the account in the AAT. In the case of a particular resource,the account funds used, account funds available, relevant user funds used, user limit stop, particular resource funds used, and resource limit stop would be placed in the AAT. Of course, some of all these six items might already be in the AAT. If the particular resource manager involved cannot take page faults the data would be placed in the WAT. The accounting system must keep track of which copy, the account copy, the AAT copy, or the WAT copy, is the current one.

Once the account is activated all charges are updated into the active copy. When use of the resource is completed the resource manager calls upon the accounting system to deactivate the account. This implies that the information is placed back into the account.

For the interest of the reader, the following list includes some problem areas that can be foreseen in the implementation of the above accounting mechanism:

1. Updating and Crashes - As stated above, when an account is deactivated the information is updated from the AAT into the account. However, if the account stays active for very long periods of time it is quite possible that the system will crash

during this time, therefore, losing the information. For this reason accounts can be updated from the AAT at regular intervals in order to insure that the account information is not too far out of date. Also the AAT can be a permanent hierarchy segment which is retained through a system crash or shutdown. This would mean that forced updates could be done quite infrequently. Information from the WAT can be updated into the AAT at regular intervals because it could be a relatively simple operation thereby avoiding the necessity for retaining the WAT or the WAT could be part of the same segment as the AAT.

? ?? (no way to do it)

There is also the problem of reverse updating, i.e., if a user changes a limit stop on a resource that is currently active this information should be updated in the AAT or WAT copy, therefore each copy will have to be aware of the existence of the other copies.

2. Shifts - Certain resources, e.g., cpu time, are charged in shifts. From a logical point of view each shift of each resource can be charged as a separate resource. However, the accounting system, in its implementation, can make use of the fact that two shifts cannot be charged to simultaneously.

This is vital

3. Secondary Storage Usage - Secondary storage is a special problem because it is a passive resource, i.e., a charge is incurred for it even if no one is doing anything. It is complicated by the fact that unlike core usage, also a passive resource, its turnover rate is low, i.e., it is possible that a segment

on secondary storage will exist for months without being referenced. This makes the updating problem extremely severe. Such passive resources may have to be handled in a special way such as keeping a resource usage -- time product and the current resource usage in the account.

Requires monthly scan

4. Bootstrapping - There are problems of how to bootstrap the accounting system and exactly what should be accounted during system initialization. For example, should we account for reading the system tape during bootload and to what account do we charge the account segments to, if any.