An Operational System for Computer Resource Sharing*

B.P. Cosell, P.R. Johnson, J.H. Malman, R.E. Schantz,
J. Sussman, R.H. Thomas, and D.C. Walden

Bolt Beranek and Newman Inc.
Cambridge, Massachusetts

Abstract

Users and administrators of a small computer often desire more service than it can provide. In a network environment additional services can be provided to the small computer, and in turn to the users of the small computer, by one or more other computers. An operational system for providing such "resource sharing" is described; some "fundamental principles" are abstracted from the experience gained in constructing the system; and some generalizations are suggested.

Keywords and phrases: computer network, distributed operating system, computer resource sharing, distributed data base management, protocols

CR Categories: 4.0, 4.3

1. The Problem

As has been described previously [Ornstein 72], the Terminal Interface Message Processor (TIP) is a terminal concentrator device which supports terminal access to the ARPA Computer Network [Roberts 70, Heart 70]. Figure 1 is a schematic diagram illustrating the TIP's role in the ARPA Network. The units labeled IMP are communications switches connected together with wide-band leased circuits to form a communications subnetwork. Host computers such as a PDP-10, an IBM 360/91, and a TIP are connected to the network through IMPs, thereby enabling the hosts to communicate with each other. Several hosts can be connected to each IMP. As this is being written (in September 1975), the ARPA Network contains about sixty IMPs and over 100 hosts, of which about one quarter are TIPs.

The problem we wish to address stems from the basic fact that users (and administrators) of a small computer, in this case the TIP, will almost always desire more services than the small computer can provide. In particular, because of memory limitations, the TIP is incapable of providing its users with a sophisticated command language. The TIP has no space to hold tables of passwords or statistics on its usage; thus, the TIP has no capability for access control or accounting. The TIP cannot distribute operational information to its users, such as announcements of system changes. Further examples of the TIP's limitations are readily available [Mimno 73]. What the TIP does provide is a relatively transparent, simple, flexible, and high performance interface between a terminal and the network. However, if access control, accounting, and other operational capabilities were to be provided, it was necessary to devise a mechanism to obtain these capabilities elsewhere.

In the following sections we sketch a system of computer resource sharing which is able to effectively provide the TIPs in the network with a set of advanced capabilities. We also discuss the fundamental structures upon which our computer resource sharing approach rests, and we describe some of the capabilities which the system currently provides. Finally, we consider some ramifications and deficiencies of our solution.

*This work has been supported by the Advanced Research Projects Agency of the U.S. Department of Defense under contract numbers F08606-73-C-0027, F08606-75-C-0032, and DAHC15-71-C-0088.

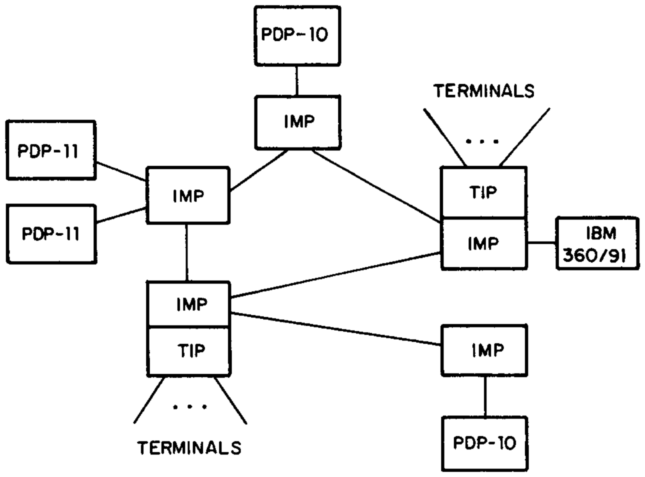

Figure 1 -- Small Example of an ARPA-like Network

Our system of computer resource sharing represents a substantial achievement in the following senses: 1) altogether some twenty-five computers are involved; 2) the system is capable of being used operationally "around the clock"; and 3) the components of the system span the globe from Hawaii to Oslo. Further, the system of resource sharing which we have developed is broader than just the provision of TIP functions; the same concepts can be generally used to permit a computer or collection of computers to enhance the capabilities of another computer or collection of computers. Thus, today we have more than just a demonstration, we have an operational system; yet, this operational system only begins to illustrate the potential of such resource sharing computer systems.

2. A Solution

Because the TIP functions in a network environment, it was natural to consider the possibility of using another host on the network to provide some of the capabilities missing from the TIP. Our first experiment in this direction was to provide a TIP "news" capability through which users could be notified of events which affected their use of the TIP (such as changes in the way a TIP command worked or the release of a new TIP system). The TIP was given a new command named @NEWS ("@" is a character normally reserved to delimit TIP commands). When a TIP user executed this command, a logical connection* was established between the TIP user's terminal and a process in a particular PDP-10 host on the network. This process was programmed to send the latest TIP news over the connection to the TIP upon receiving a connection from a TIP terminal. After sending all of the news, the process would break the connection to the TIP. Alternately, the TIP user could explicitly break the connection at any time. Either case freed the user's terminal for communication with other hosts for other purposes. While no special effort was made to hide the fact that another host was being called on to provide the TIP news function, the user did not normally have to be concerned with the fact that another host was involved; he had only to execute a TIP command and, in effect, the TIP printed the news. Thus, we had implemented a rudimentary example of resource sharing.

At the time of this initial experiment, the Resource Sharing Executive (RSEXEC) system [Thomas 73b] also began to come into existence. The RSEXEC is an experimental, distributed, executive-like** system which acts to couple the operation of some ARPA Network hosts. RSEXEC is designed to provide an environment which allows users to access network resources without requiring attention to network details such as communication protocols and without even requiring users to be aware that they are dealing with a network. RSEXEC is currently used both as an operational service facility and as a vehicle for exploring the technical problems of realizing an effective environment for resource sharing.

Development of RSEXEC was motivated initially by the desire to pool the computing and storage resources of the individual TENEX [Bobrow 72] hosts on the ARPA Network. At the time, the TENEX virtual machine was becoming a popular network resource (at present there are fourteen TENEX systems in the network). Further, it was becoming clear that for many users, in particular those whose access to the network was via TIPs or other non-TENEX hosts, it should not actually matter which host provides the TENEX service so long as the users could do their computing in the manner to which they had become accustomed. A number of advantages would result from such resource sharing. The user would see TENEX as a much more accessible and reliable resource. Because he would no longer be dependent upon a single host for his computing, he would be able to access the TENEX virtual machine even when one or more of the TENEX hosts were unavailable. Of course, for him to be able to do so in a useful way, the TENEX file system would have to span across host boundaries. The individual TENEX hosts would see advantages also. For example, some sites, because of local storage limitations, do not provide all of the TENEX subsystems*** to their users. Because the subsystems available would, in effect, be the "union" of the subsystems available on all TENEX hosts, previously limited hosts would be able to provide access to all TENEX subsystems.

During the development of the RSEXEC system two observations were made: first, since many of the features planned for the RSEXEC were well matched to the desires of TIP users, it became clear that with some additional effort the RSEXEC system could provide TIP users with a sophisticated command language and other features they desired; second, because the RSEXEC was to be run on several PDP-10 TENEX systems, RSEXEC could potentially provide capabilities to the TIP very reliably. With a single host providing a function, such as the news service discussed above, there would be times at which that host would be down when some TIP user required the function. Thus, it would be possible through TIP use of the RSEXEC to obtain TIP capabilities superior to any the TIP could provide itself or that could be provided with the help of any single other host. Our attempt at resource sharing was becoming less rudimentary.

3. Current TIP/RSEXEC Capabilities

A service program called TIPSER (for TIP SERver) currently runs (alongside other user programs) on three ARPA network TENEX hosts. TIPSER allows TIPs to make direct use of certain features of RSEXEC as a "virtual executive". Development of the TIPSER-RSEXEC system has been guided by the general philosophy that the TIP should be a transparent front-end component supporting only terminal-device-specific functions and that access control, accounting, command language interpretation, and other "large host operating system-like" functions should be handled by other more capable (larger) network machines [Mimno 73].

At the start of a TIP user's session, the TIP has the capability of automatically connecting the user's terminal to the most responsive TIPSER-RSEXEC available. After the user correctly identifies himself, he is granted access to the network and to the TIPSER-RSEXEC as a network command language interpreter, preparatory to logging in to a particular network host. In addition, at any time throughout his TIP session, a user is free to execute the TIP's @NETEXEC command to instruct the TIP to connect him back to an RSEXEC. The annotated typescript in Figure 2 illustrates the process of connecting and logging into the TIPSER-RSEXEC and listing the services available to TIP users. The services include inter-site user interaction features and a number of information services as listed in Figure 3.

```
                                    <user dials TIP>
BBN10X TIP 337 #: 25                <TIP answers with its herald>
Wait...                             <TIP attempts to make
 Open                                connection to RSEXEC>
 RSEXEC 3.8.01 (361)                <RSEXEC answers with its herald>
 LOGIN Please (type ? for help)     <user is asked to login>
-login                              <user logs in, giving a
 (name) Walden, D.                   portion of his unique name
 (password)                          and his password>
 Checking...                        <RSEXEC accepts user, and
                                     prints next three lines>
 DAVID C WALDEN of BBN-DIV6 logged onto BBN10x-TIP 25-MAY-75 13:11-EDT
 Type QUIT <cr> to return to TIP
 Latest NETNEWS: 15-MAY-75 -- NEW TIP RELEASE SCHEDULED
-?                                  <user requests RSEXEC command list>
 BREAK                              <RSEXEC prints list>
 DESCRIBE
 FULLDUPLEX
 GRIPE
 HALFDUPLEX
 HELP
 HOSTAT
 LINK
 LOGIN
 NETNEWS
 NETSTAT
 QUIT
 RECEIVE
 REFUSE
 SCHEDULES
 SERVERS
 SITES
 SNDMSG
 TENXSTAT
 TIMECONSTANT
 TRMINF
 WHERE
 WHO
-quit                               <user types quit command>
 Quitting...                        <RSEXEC breaks connection
 Closed                              with TIP terminal>
```

Figure 2 -- TIP Connection and Login to RSEXEC

----

*Most of the hosts in the ARPA Network have implemented a conventional set of procedures which they use to communicate with each other. These conventional procedures have come to be called "protocols." At the base of all the standard protocols is a protocol which provides logical connections between hosts desiring to communicate (actually, a logical connection is between processes in the hosts).

**In our terminology, an "executive" is that program or command language interpreter which a user uses to communicate with an operating system.

***In TENEX terminology, a subsystem is a program which runs in user mode but which is available to all users as if it were a basic part of the operating system.

The LOGIN and QUIT commands allow the user to log into the RSEXEC and to leave it.

The HELP, DESCRIBE, and SERVERS commands give the user information on the available functions how each function works, and which sites run RSEXEC.

The LINK, BREAK, REFUSE, and RECEIVE commands allow the user to "link" his terminal to terminals of other users to engage in on-line conversations, to break links from other users, to refuse links from other users, and to accept links from other users.

The FULLDUPLEX, HALFDUPLEX, and TIMECONSTANT commands allow the user to set various parameters of the system operation.

The NETNEWS command allows the system operations staff to announce information of interest to users; the GRIPE command lets users register suggestions and complaints with the system operations staff.

The SNDMSG command lets users send messages to other users.

The WHERE, WHO, and SITES commands let a user find the site at which a particular active user is running, list the active users at a set of sites, and find the sites at which a particular user is known.

The NETSTAT, HOSTAT, SCHEDULES, and TENXSTAT commands let a user ascertain such information as which hosts are up or down, the future down time schedules of IMPs and TIPs and various hosts, and the instantaneous loads on various of the network TENEX systems.

The TRMINF command allows the user to determine certain information about the TIP port he is using.

Figure 3 -- TIPSER-RSEXEC Command Functions

The redundant implementation of the TIPSER-RSEXEC serves to distribute the load among the machines providing the service and to increase the accessibility of the service by guaranteeing that the service is available whenever at least one TIPSER-RSEXEC site is up. The relationship of users, TIPs, TIPSER processes, and RSEXEC processes is illustrated schematically in Figure 4.

Two mechanisms were developed to support the redundant implementation. The first is a "broadcast" initial connection protocol (ICP). This enables a TIP to connect to an available and responsive RSEXEC rather than to a particular one at a specific site. Using this mechanism, a TIP broadcasts requests for service to the known TIPSER-RSEXEC sites and then selects the site that responds first as the one to provide the service.

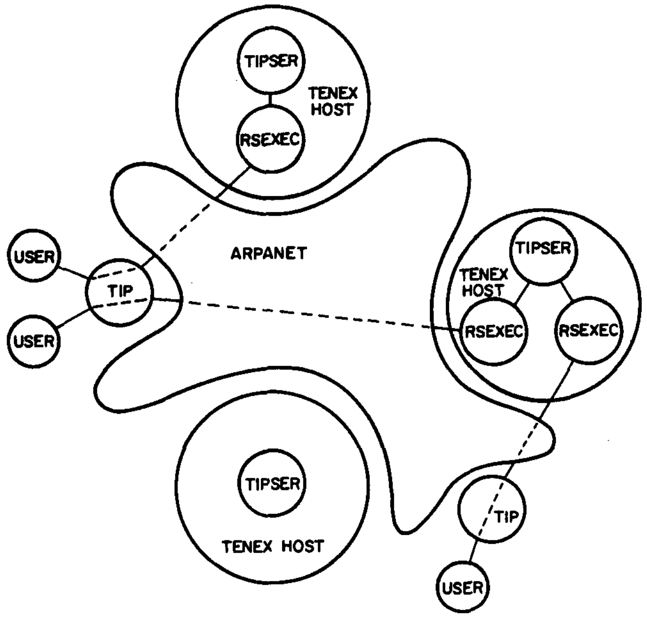

Figure 4 -- Components of the TIP/RSEXEC System

Mechanisms similar in spirit to the broadcast ICP have been developed for use in other distributed systems [Farber 73].

The second mechanism was developed to maintain multiple copies of the various information files (e.g., network news and host schedules) at the TIPSER-RSEXEC sites. This mechanism allows additions to these distributed information files to be initiated from any TIPSER-RSEXEC site and guarantees that the additions are incorporated into each file image in a consistent manner.

Having now briefly mentioned the capabilites currently available to the TIP through use of the TIPSER-RSEXEC, in the rest of this section we describe in detail the most recent pair of functions (TIP access control and accounting) to be supported within the environment for resource sharing we have developed.

In order to solve the problem of controlling access to the network and the related problem of accounting for TIP usage, a distributed, multi-computer access control and accounting system based on the TIPSER-RSEXEC and the RSEXEC distributed file system was developed*. This system consists of three distinct, but related, components: network login server processes (TIPSER-RSEXEC processes), data collection server processes, and data reduction software.

Whenever a user activates a TIP port, the TIP uses the broadcast ICP mechanism to connect to an RSEXEC which acts as a network login server. If the user successfully supplies a valid name and password, he is granted continued access to the TIP, the network, and to the standard TIPSER-RSEXEC functions. In addition, the RSEXEC transmits the user's network ID code (which serves to uniquely identify the user for accounting and subsequent authentication purposes) to the TIP and makes a "login" entry into an "incremental" TIP accounting data file. If the user fails to supply a valid name and password within the allowed time, he is denied further access to the TIP.

After the TIP receives the user's network ID code it activates "connect time" and (outgoing) message counters to accumulate usage data for the user's session. These counters remain active until the user terminates his TIP session. Periodically the TIP executes an "accounting checkpoint" procedure whereby it transmits usage data for its active users, accumulated since the last checkpoint, to a data collection server process. The data collection server stores the checkpoint data in an incremental TIP accounting file for later processing.

Like the TIPSER-RSEXEC login servers, the data collection servers are redundantly implemented to insure high availability and to achieve load sharing [Schantz 74b]. The TIP uses a request mechanism similar to the broadcast ICP to select one of the servers to accept its checkpoint data. The protocol used for this purpose is quite general and can be used for the collection of data other than that for TIP accounting. Furthermore, the protocol is designed to allow considerable flexibility in the choice of a server. For example, a TIP can switch from one data collection server to another after initially choosing one in the event that the chosen server can not complete the transaction (for example, because of network or host failure).

The collection of incremental accounting files created by the data collection servers is a large,

---

*The TIPSER-RSEXEC system was by design an evolutionary system. The plan was to implement a system with limited capabilities and then to let it evolve, expanding its capabilities as experience and understanding of the technical problems permitted. The TIP access control and accounting capability represents the most recent addition to the system. The access control and accounting mechanisms are developed to the point of being operational and have been used operationally in the system for a period of several weeks. However, because of a number of broader (beyond the TIP itself) administrative issues within the ARPA Network, TIP access control and accounting are not currently being used.

distributed and segmented data base. We note that some checkpoint data for a given TIP session may be collected by a data collection process at one site while other parts of the data for the same session are collected at other sites. The reduction of data in this distributed data base to produce periodic accounting summaries is accomplished by software which executes within the environment provided by the RSEXEC distributed file system [Thomas 73b, Thomas 75a]. This software performs a series of data management and network access operations in response to simple commands. When the "TIP accountant" (a person) issues the proper commands, the software automatically connects to the data collection sites and selectively retrieves and processes remote (and previously unprocessed) accounting data. The data reduction software was designed to be consistent with the RSEXEC philosophy: to allow a user to deal with resources (in this case accounting data) distributed throughout the network while relieving him of the complexities of dealing directly with the network itself.

We reiterate that the significance of the TIPSER-RSEXEC system exceeds the utility of the particular functions it currently supports. It has served to demonstrate the feasibility of having small hosts share the resources of larger hosts to reliably support features that exceed the small hosts' own capacities. Users of a small host obtain these services automatically in a network transparent manner.

4. Fundamental Structures

In addition to the standard communications protocols used by hosts for communication among themselves, structures providing several additional functions were necessary to support TIP/RSEXEC resource sharing. In the following subsections we discuss these structures, two of which have already been alluded to in the previous section.

4.1 Broadcast Service Requests

To enable a TIP to conveniently discover and use an instance of the RSEXEC required a mechanism other than having it try to connect to each TIPSER-RSEXEC site in turn until it finds an available one. This is an example of a general problem in accomplishing resource sharing -- that of finding and selecting resources. Two techniques for supporting the selection function are apparent:

1. Maintain up-to-date status information about the various network resources and machines, and use it to select the machine best suited for a task. The server processes that support the RSEXEC system exchange status information for this purpose. Although automatic job assignment has not yet been implemented, the status information is currently available to users who may use it to manually select a machine and it is, in principle, available to programs for automatic resource selection purposes.

2. Dispatch "requests for service" to the appropriate machines, allowing them to respond with status information if they choose, and then make a selection on the basis of those machines which have responded as willing to accept a new task. This is the technique TIPs use when it is necessary to select a responsive RSEXEC.

The first technique involves a fixed overhead, that of exchanging and maintaining the resource status information, which is independent of the frequency of resource selection. For the second technique, the overhead is incurred on a per transaction basis and is, therefore, proportional to the frequency of selection. Although the frequency of service requests was expected to be relatively high in the TIP/RSEXEC case, the second technique was chosen because it does not require TIPs to allocate limited storage resources for maintaining status information. Another basic difference in these two techniques is that the second allows the constituent machines to retain a higher degree of autonomy in managing their own resources. Each machine can choose whether or not to respond to particular requests for service.

4.2 Reconnection

Another area where users can be relieved from attending to network details is that of establishing and breaking connections with various service machines. Our experience with the TIP/RSEXEC has suggested the use of a dynamic "reconnection" mechanism [Thomas 73a, Schantz 74a] in order to transfer a user from the "virtual executive" to a service-providing machine after he logs into the TIP and network, and also subsequently from one service-providing machine to another as his computing requirements change. Reconnection should be accomplished in a transparent manner that requires no manual intervention by the user. Furthermore, it should include the transfer of his authentication and accounting identity from machine to machine. That is, moving a user from service to service should require no explicit disconnects, connects or logins on the user's part after initial connection to and authentication with the TIP/RSEXEC. Mechanisms to support such reconnection have been designed and a prototype implementation is planned within the TIP/RSEXEC context to validate them.

4.3 Distributed Data Base Management

Multi-computer systems introduce a new class of data base management problems which result from the distributed nature of the data. These problems occur at all levels of system design and implementation, ranging from low level system primitives to function oriented application software.

Experience with the ARPA Network indicates that data tends to be distributed for a variety of reasons.

1. To insure reliability. The accessibility of critical data can be increased by redundantly maintaining it. The data base of network user IDs used by the TIPSER-RSEXEC to authenticate users [Johnson 74] is an example of a data base which is redundantly distributed to achieve highly reliable access.

2. To insure efficiency of access. Data can be more quickly and efficiently accessed if it is "near" the accessing process. A copy of the network user ID data base is maintained at each of the TIPSER-RSEXEC sites to insure rapid, efficient access. Reliability considerations dictate that this data base be redundantly maintained, and efficiency considerations dictate that a copy be maintained at each authentication site.

3. As a consequence of the naturally distributed manner in which the data is generated or collected. The data base represented by the collection of incremental TIP accounting files is an example of a data base generated in this way. Individual data items are stored at the data collection site best prepared to handle them at the time they were generated by some TIP.

There are two fundamentally different types of distributed data bases. The first is one which is maintained "identically" at a number of sites. The second type consists of distributed, non-overlapping segments; that is, the data base is a collection of segments, each of which is singly maintained at a (possibly) different location. It is important to recognize that these two types represent extremes and that applications may call for "intermediate" types -- for example, a data base consisting of a collection of segments of which some, but not all, are redundantly maintained.

The emphasis of our work within the TIPSER-RSEXEC context with the first type of data base has been to develop techniques for consistently and automatically maintaining redundant data base copies. Below we cite two applications of such data bases and describe the techniques used in their implementation:

1. The TIPSER-RSEXEC maintains a copy of the TIP news file at each of the TIPSER-RSEXEC sites. Updates to the news file are limited to addition of news items. The system allows additions to the data base to be initiated at any TIPSER-RSEXEC site and insures that all such updates are transmitted to and incorporated into all copies of the data base.

2. The TIP login system requires that the network user ID data base be maintained in a consistent manner at all TIPSER-RSEXEC sites. Each copy of this data base is a collection of mutually independent user entries. Allowable updates to this data base include the addition, modification, and removal of individual user entries. We have designed a data base management technique which allows updates to be initiated at any site and guarantees that they are consistently incorporated into all copies of the data base [Johnson 75]. By "consistently incorporated" we mean that if all updating activity were to cease, all copies of the data base would eventually be identical.

The techniques used to maintain the NETNEWS and the user ID data bases each consist of two independent parts:

1. A reliable, data-independent, update transmission and distribution mechanism which uses persistent processes at the update entry sites to guarantee that all updates are eventually delivered (once, and only once) to all data base sites

2. A data-dependent update action procedure which is activated at data base sites whenever update commands arrive.

For the NETNEWS, the update procedure is a relatively simple one in which updates are appended to the data base as they arrive. For the user ID data base a more sophisticated update procedure is required. The nature of the user ID data base and the operations permitted on it are such that recent updates to an entry override (rather than interact with) older updates. For example, when a user password is changed, the old password is simply replaced with the new one. The update procedure is based on the use of a time stamping mechanism to enable each of the different data base sites to reconstruct and then act upon the (identical) sequence of update events. Furthermore, each entry (and modifiable subfield) in the data base retains the time stamp of the update which resulted in its current value. When most update commands arrive at a data base site, the command can be incorporated or rejected simply by comparing its time stamp with that of the data base entry to which it refers. The deletion and creation of entries require slightly special treatment. For example, if create and delete commands for a single entry are initiated at separate sites, normal network communication delays or network or system malfunction could cause the creation command to arrive at a third site after the deletion command. To properly handle such cases the data base update procedure defers "final" action on a deletion command until it is a certainty that all update commands for an entry which were initiated prior to the deletion have arrived. Only at that point is it safe to remove the entry from the data base.

The operation of the TIP accounting system results in the creation and manipulation of segmented data bases. The primary concern in the accounting application was with data base organization and convenient data access. The specific data base issues that required attention were:

1. Cataloging. It is obviously important to know where the various data segments (incremental accounting files) reside so that they can be accessed. This cataloging function is provided by the RSEXEC distributed file system.

2. Insuring that no duplicate entries occur in the data base. Because the entries contain accounting information, it is critical that any redundancy does not cause duplicate charging. The data collection protocol was carefully designed to prevent the occurrence of duplicate data entries in spite of the the fact that data is broadcast to all of the servers [Schantz 74b].

3. Insuring that each data base entry is processed exactly once when accounting summaries are produced. It is interesting to note that time stamping can also play a fundamental role in guaranteeing "once only" processing.

5. Discussion

Despite the fact that the system has attained operational status, there are some clear deficiencies, and we have learned some important lessons. We also see some ramifications of the system on technical and operational aspects of the network and network hosts. Finally, we see almost unbounded potential for the use and growth of our system and systems like it. We discuss these issues in the rest of this section.

In a number of situations the existing ARPA Network host-host protocol [McKenzie 72] forces difficult or clumsy implementations in support of functions which are conceptually quite simple. These difficulties are largely due to the complexity of the protocol. Among the situations which pose such difficulities are those which can be characterized as involving brief, transaction oriented interactions. The TIPSER-RSEXEC broadcast connection mechanism is a good example of such a situation. The mechanism requires the transmission of a short message from a process to one or more remote processes. The standard host-host protocol requires that the processes participate in an elaborate exchange of protocol commands, carefully remembering the exact state of each exchange, in order for the first process to transmit its simple message to the other processes. For large hosts this exchange is wasteful. For small hosts it is often impossible to implement correctly. In this regard, it is interesting to note that the data collection protocol used in the TIP accounting system was designed to be separate from (and exist in parallel with) the host-host protocol in order to make implementation feasible for memory resource-limited TIPs.

The presence of multiple components in a distributed system, together with the potential for their redundancy, makes it possible to achieve reliability by constructing systems from modules most of which are kept relatively simple. By using simple modules, component failure due to malfunction of non-essential features can be reduced. Complex components are redundantly supported in an effort to enhance their reliability. The evolution of the TIP and TIPSER-RSEXEC is a good example of this approach. Use of redundantly supported "logical" front-end servers allows the network access machine to be simple and reliable without loss of function. The more complex "front-end-like" features can be provided reliably by multiple network service machines rather than within the network access machine itself. Such an approach takes full advantage of both the heterogeneity and homogeneity of various network components. The important issues in designing a system of this type are the assignment of functions among the various machines, the degree of redundancy required, and the protocols used to bind the system modules together.

Experience with the ARPA Network has indicated the need for access controls above and beyond those supported by the constituent host service machines. For example, an access control mechanism has recently been implemented within the communication subnetwork to allow the set of network hosts with which a particular host can communicate to be administratively limited. The access controls applied to the TIP also fall into this category. In many cases the goals of network transparency and ease of access conflict with those of security and privacy. Each security or access check places a barrier between the user (or his program) and the desired resource.

If the TIP access control function were today actively enforced, then the use of a host from a TIP would require that the TIP user first authenticate himself to the TIP, next open a logical connection to the service host, and finally authenticate himself with the host before actually beginning to use the host's services. Although the actual time and effort required of the user to complete these steps is not large, many users, when forced to adhere to TIP access control, have had strongly negative reactions to this process of "double login". Rather than perceiving the two instances of authentication as providing additional security, many users perceive the process as forcing them to do the "same thing" twice. To cure this perceived problem, modifications to the TIP and the TIPSER-RSEXEC would be required to make it possible for service hosts to learn the identity of a TIP user based on the authentication data provided at the time of TIP login. This mechanism could be provided in such a way that only those hosts choosing to make use of it would be required to modify their software and only users choosing to make use of it would lose the extra security barrier [Thomas 75b].

As mentioned in Section 4, after a TIP user is connected to the TIPSER-RSEXEC, it would be convenient if the user could choose a service host and have the TIPSER-RSEXEC reconnect him to that host without requiring him to explicitly break his connection to the TIPSER-RSEXEC and then explicitly open a connection to the service host. Ideally, the user would request not a particular service host, but a particular service; and the TIPSER-RSEXEC would reconnect him to the site providing the desired service in the most responsive way or the most economical way or the way having some other desirable attribute. Finally, when finished with a service (or service host), the user could be reconnected back to an available TIPSER-RSEXEC. All of this reconnection back and forth could be transparent to the user, thus truly providing the appearance of a common (albeit virtual) executive.

Once such a virtual executive is conveniently available to TIP users, it becomes possible to think of additional features that can be added. For instance, the TENEX RSEXEC makes available to TENEX users a file system which spans machine boundaries. It is a simple technical step to provide the TIP users (who, unlike TENEX users, have never had a file system) with a virtual file system. Another example: while the TIPSER-RSEXEC has the capability to permit users to leave messages for other users, it does not provide the capability for TIP users to receive such messages. Yet, through the concept of resource sharing, the potential capability to provide virtual mailboxes through which users can receive messages exists. Furthermore, through the redundancy inherent in the system, these mailboxes could be provided in a way which would insure that a user's mail was accessible no matter which individual computers were down. A final example: once the TIP user is connected to the TIPSER-RSEXEC and is ready to use the services of some host, and once it is possible for the user to call for service independent of host, there is no need to retain in the user's view the concept of the host(s) from which service is obtained; rather, the virtual executive could be expanded to provide the virtual operating system from which all service is obtained.

To the extent that the virtual executive, the virtual mail service, the virtual operating system, and the like are made available to users, two changes in traditional computer operations are in order. First, the problem of unique user names arises. Traditionally, a user name had only to be unique to each local computer system. However, if users of many systems are to communicate through a single virtual mail system, keep their files in a single virtual file system, be authenticated by a single virtual authentication system, and so on, then there is a clear need for universal user names. Our system currently provides for such universal names by allowing the use of a person's full name (i.e. first, middle, and last) along with the person's affiliation (i.e. address), although only the minimum data for unique recognition is required.

The second necessary break with traditional computer operational practice is in the area of accounting and billing. Traditionally, each user makes arrangements with each center of computer service to which he desires access. With an integrated resource sharing system in which the existence of the individual hosts is of minimal importance, it is highly desirable to have a system-wide accounting and billing system. The user should not have to execute a large number of contracts with individual sites or receive a large number of bills for computer service each month, especially when his use of these individual systems was not apparent to him. Rather, the user will want to execute one contract for all his computer service, or at most one contract for each type of system he desires, independent of the sites from which the service is obtained. Our system contains prototype mechanisms to facilitate such global accounting practices (in particular, for invoicing a TIP user for all his TIP use in a month independent of the number of TIPs from which he received his TIP use).

It is interesting to note that the TIPSER-RSEXEC system need not be limited to TIP use. Any host needing similar functions, out of a desire for standardization or because the host is unable or unwilling to provide the services itself, could make use of the TIPSER-RSEXEC. In general, we believe that terminal concentrator hosts such as the TIP should make use of the TIPSER-RSEXEC, as we assert it is the proper function of such terminal concentrators to specialize in the handling of terminal I/O and to leave other functions to other hosts. We assert that the complement is also true: service hosts should generally specialize in the handling of application functions and leave the details of terminal I/O to a terminal concentrator. We believe our system properly supports such specialization of function, and that it is economically advantageous to make use of such a system whenever possible.

Acknowledgments

We are grateful to the Information Processing Techniques Office of the Advanced Research Projects Agency of the U.S. Department of Defense for its support for our work. J. Burchfiel and A. McKenzie of Bolt Beranek and Newman Inc. (BBN) offered helpful criticism throughout the development of the system described in this paper. Thanks also go to F. Heart (BBN) and A. Bernstein (SUNY at Stony Brook) for commenting on early drafts of this paper. TIP users throughout the ARPA Network, particularly at BBN, served as "guinea pigs" as we developed the system: to them we are grateful and we apologize for any inconvenience we caused them.

References*

[BBN 74]. Distributed Computation Research at BBN, Volume III, Final Report -- Natural Communication with Computers, BBN Report 2976, December, 1974.

[Bobrow 72]. D.G. Bobrow, J.D. Burchfiel, D.L. Murphy, and R.S. Tomlinson, "TENEX, A Paged Time-sharing System for the PDP-10," Communications of the ACM, Vol. 5, No. 3 (March 1972), pp. 135-143.

[Farber 73]. D.J. Farber, J. Feldman, F.R. Heinrich, M.D. Hopgood, K.C. Larson, D.C. Loomis, L.A. Rowe, "The Distributed Computing System," Proceedings of the Seventh Annual IEEE Computer Society International Conference, San Francisco, California, February 1973, pp. 39-43.

[Heart 70]. F.E. Heart, R.E. Kahn, S.M. Ornstein, W.R. Crowther, and D.C. Walden, "The Interface Message Processor for the ARPA Computer Network,"

----

*All references which are BBN reports, internal memos, or Network Working Group Notes are available from the authors at BBN upon request.

AFIPS Conference Proceedings, Vol. 36, June 1970, pp. 551-567.

[Johnson 74]. P.R. Johnson, "An Overview of the Network User Identification System," BBN Internal Memo, November, 1974.

[Johnson 75]. P.R. Johnson and R.H. Thomas, "The Maintenance of Duplicate Databases," ARPA Network Working Group Note No. 677, January, 1975.

[McKenzie 72]. A. McKenzie, "Host/Host Protocol for the ARPA Network," Available from the Network Information Center as NIC 8246 at Stanford Research Institute, Menlo Park Calif. 94025.

[Mimno 73]. N.W. Mimno, B.P. Cosell, D.C. Walden, S.C. Butterfield, and J.B. Levin, "Terminal Access to the ARPA Network -- Experience and Improvements," Proceedings of the Seventh Annual IEEE Computer Society International Conference, San Francisco, California, February 1973, pp. 39-43.

[Ornstein 72]. S.M. Ornstein, F.E. Heart, W.R. Crowther, S.B. Russell, H.K. Rising, and A. Michel, "The Terminal IMP for the ARPA Computer Network," AFIPS Conference Proceedings, Vol. 40, June 1972, pp. 243-254.

[Roberts 70]. L.G. Roberts and B.D. Wessler, "Computer Network Development to Achieve Resource Sharing," AFIPS Conference Proceedings, Vol. 36, June 1970, pp. 543-549.

[Schantz 74a]. R.E. Schantz, "A Note on Reconnection Protocol," ARPA Network Working Group Note No. 671, December, 1974.

[Schantz 74b]. R.E. Schantz, "A Multi-Site Data Collection Facility," ARPA Network Working Group Note No. 672, December, 1974.

[Thomas 73a]. R.H. Thomas, "Reconnection Protocol," ARPA Network Working Group Note No. 426, January, 1973.

[Thomas 73b]. R.H. Thomas, "A Resource Sharing Executive for the ARPANET," AFIPS Conference Proceedings, Vol. 42, June 1973, pp. 155-163.

[Thomas 75a]. R.H. Thomas, "JSYS Traps - A TENEX Mechanism for Encapsulation of User Processer," AFIPS Conference Proceedings, Vol. 44, May 1975, pp. 351-360.

[Thomas 75b]. R.H. Thomas, "Eliminating the Double Login," BBN Internal Memo, March 1975.