0685-76

76 685

Network Working Group
Request for Comments: 714
NIC: 35144

Alex McKenzie
BBN-NCC
21 April 1976

A Host/Host Protocol for an ARPANET-type Network

Recently we have been involved in the planning of a network which, if implemented, would use ARPANET IMPs without modification, but would allow re-specification of Host/Host (and higher level) Protocol. The remainder of this document is a slightly edited version of our recommendation for Host/Host protocol; we thought that it might be of interest to the ARPANET community.

1. INTRODUCTION

The Host/Host Protocol for the ARPANET was the first such protocol designed for use over a packet-switched network. The current version has been in existence since early 1972 and has provided for the transportation of billions of bits over tens or hundreds of thousands of connections. Clearly, the protocol is adequate for the job; this does not mean that it is ideal, however. In particular, the ARPANET Host/Host protocol has been criticized on the following grounds (among others):

(1) It is specified as a simplex protocol. Each established connection is a simplex entity, thus two connections (one in each direction) must be established in order to carry out an exchange of messages. This provides great generality but at a perhaps unacceptable cost in complexity.

(2) It is not particularly robust, in that it cannot continue to operate correctly in the face of several types of message loss. While it is true that the ARPANET itself rarely loses messages, messages are occasionally lost, both by the network and by the Hosts.

(3) Partly because of the simplex nature of connections, the flow control mechanisms defined in the ARPANET protocol do

not make efficient use of the transactional nature of much of data processing. Rather than carrying flow control information (in the form of permits, or requests for more information) in the reverse traffic, a separate channel is set up to convey this information. Thus, for transactional systems, up to twice as many messages are exchanged (half for flow control information and half for data) as would be needed for data alone.

(4) Prohibition against the multiple use of a connection termination point makes the establishment of communication with service facilities extremely cumbersome.

The International Federation for Information Processing (IFIP) working Group 6.1 (Packet-switched Network Interworking) has recently approved a proposal for an internetwork end-to-end protocol. The IFIP Protocol is based on experience from the ARPANET, the (French) Cyclade Network, and the (British) NPL Network, as well as the plans of other networks. Thus, one would expect that it would have all of the strengths, and few (or none) of the weaknesses, of the protocols which are in use on, or planned for, these networks.

In fact, the IFIP Protocol avoids the deficiencies of the ARPANET protocol mentioned above. Connections are treated as full-duplex entities, and this decision permits flow control information to be carried on the reverse channel in transaction oriented systems where there is reverse channel traffic occurring naturally. In addition, the IFIP Protocol is to some extent self synchronizing; in particular, there is no type of message loss from which the protocol does not permit recovery in a graceful way.

The IFIP Protocol makes a minimal number of assumptions about the networks over which it will operate. It is designed to permit fragmentation, as a message crosses from one network to another, without network reassembly. It anticipates duplication, or non-delivery, of messages or message fragments and provides ways to recover from these conditions. Finally, it permits delivery of messages at their destination Host in a completely different order from the order in which they were input by the source Host. Unfortunately, it achieves these advantages at a relatively high overhead cost in terms of transferred bits. The complete source and destination process addresses are carried in every message, 24-bits of fragment identification are carried with each fragment and 16-bits of acknowledgement information are also carried in every message.

When considering channel capacities of hundreds of kilobits (or more), message overhead of a few hundred bits is a modest price to pay in order to achieve great flexibility and generality. However, for a "stand-alone" network of the type under consideration, and especially in view of the anticipated use of many circuits of 10kbs capacity, the IFIP Protocol offers far more generality than is needed, for which a fairly severe overhead price is paid.

The virtual circuit protocols currently being debated within the International Telegraph and Telephone Consultative Committee (CCITT) are a step in the opposite direction. Virtual circuit protocols attempt to make a packet switching network indistinguishable (from a customer's point of view) from a switched circuit network, except possibly in regard to error or delay characteristics. Thus, virtual circuit protocols generally place responsibility for end-to-end communications control within the network rather than within the Hosts. For example, when a receiving Host limits the rate at which it accepts data from the network, the network in turn limits the rate of input from the Host which is transmitting this data stream. Host protocols which are designed for virtual circuit networks can be quite simple, if somewhat inflexible. For example, the Host might give the network a "link number" or "index" and ask the network to set up a virtual circuit to some other Host to be associated with this number, and report back if and when the circuit is established. However, significant development would be required to add a virtual circuit capability to the ARPANET IMP software; the required changes would seem to be more expensive and carry greater uncertainty than they are worth.

In light of the above, our approach in defining this proposed protocol has been to start with the ARPANET Host/Host protocol and modify it according to some of the concepts of the IFIP Protocol in order to remedy its major deficiencies. The remainder of this document specifies the protocol which we have designed for this purpose.

II. COMMUNICATION CONCEPTS

The IMP subnetwork imposes a number of physical restrictions on communications between Hosts. These restrictions are presented in BBN Report No. 1822. In particular, the concepts of leaders, messages, padding, message ID's and message types are of interest to the design of Host/Host Protocol. The following discussion assumes that the reader is familiar with these concepts.

The IMP subnetwork takes cognizance only of Hosts, but in general a host connected to the network can support several users, several terminals, or several independent processes. Since many or all of these users, terminals, or processes will need to use the network concurrently, a fundamental requirement of the Host/Host Protocol is to provide process-to-process communication over the network. Thus, it is necessary for the Host/host Protocol to provide a richer addressing structure than is required by the IMP subnetwork.

Processes within a Host are envisioned as communicating with the rest of the network through a Network Control Program (NCP) resident in that Host, which implements the Host/Host Protocol. The primary functions of an NCP are to establish connections, break connections, and control data flow over connections. A connection couples two processes so that output from one process is input to the other, and vice versa. The NCP may be implemented either as part of the Host's operating system or a separate user process, although it must have the capability of communicating with all of the processes or routines which are attempting to use the network.

In order to accomplish its tasks, the NCP of one Host must communicate with NCPs of other Hosts. To this end, a particular communication path between each pair of Hosts has been designated as the control connection. Messages transmitted over the control connection are called control messages, and must always be interpreted by an NCP as a sequence of one or more control commands. For example, one kind of control command is used to initiate a connection while another kind carries notification that a connection has been terminated.*

The maximum size of a message is limited by the IMP subnetwork to approximately 1000 8-bit bytes, and in fact may be further limited by the receiving Host for flow control reasons,

----

*Note that in BBN Report No. 1822, messages of non-zero type are called control messages, and are used to control the flow of information between a Host and its IMP. In this document the term "control message" is used for a message of type zero transmitted over the control connection. The IMPs take no special notice of these messages.

as described later. Accordingly, the transmitting process, or its Network Control Program, must take responsibility for fragmenting long interprocess messages into messages of a size conforming to the Host/Host and Host/IMP protocols. For this reason, it is impossible for a sending Host to guarantee that any significance should be attached to message boundaries by receiving processes. Nevertheless, message boundaries will occur naturally, and should be used in a reasonable way wherever possible; that is, a sending process or its NCP should not act arbitrarily in deciding to fragment messages. For example, this protocol specifies that each control message must contain an integral number of control commands and that no single control command will be split into two pieces which are carried through the network in separate messages.

A major concern of the Host/Host Protocol is the definition of a method for references to processes in other Hosts. In order to facilitate this, a standard name space is used, with a separate portion of the name space allocated to each Host. Each Host, therefore must map internal process identifiers into its portion of this name space. The elements of the name space are called sockets. A socket forms one end of a connection and a connection is fully specified by a pair of sockets, one in each Host. A socket is identified by a Host number and a 16-bit socket number. The same 16-bit socket number in different Hosts represents different sockets. In order to avoid the transmission of a pair of 16-bit socket numbers in each message between these sockets, the process of connection establishment allows each Host to define a mapping, valid for the lifetime of the connection being established, from the 32 bits which specify the socket pair to an 8-bit number.

No constraints are placed on the assignment of socket numbers; however, since a pair of socket numbers defines a unique connection, it is clear that in assigning socket numbers, a Host must insure that for each new connection at least one of the socket numbers is unique. For example, a Host which supports many terminals might choose to use a terminal's physical interface number as a portion of the socket number involved in any connection established on behalf of that terminal. This would insure uniqueness at the terminal end. Thus, no conflict would occur if several terminals attempted to access a common resource (identified by its own unique socket number).

From the foregoing it should be clear that the Host/Host Protocol allows a single socket to participate in several connections simultaneously. This is quite similar to what happens in the telephone system, where a company, as well as an individual, can be identified with a phone number. As seen from the outside, the phone number of a company is sharable, since several conversations can proceed at the same time and the caller does not have to worry about the already existing conversations. Conversely the phone number of an individual is not sharable, since he can process only one conversation at a time; the same is generally true of a connection to a terminal which might be using the network.

A final major concept which should be explained is the "windowing" concept which is used for flow control. This concept is adapted from the IFIP Protocol with some appropriate modifications for use in an ARPANET-type network. When a connection is established, a sequence number is initialized to some specified starting point and the receiver allocates a certain number of credits to the sender. Each credit entitles the sender to transmit one message; that is, the receiver agrees to provide buffering for the number of messages specified by the number of credits granted. If one thinks of sequence numbers advancing from left to right, the initial sequence number defines the left edge of a window into the entire sequence number space and the credit, when added to the initial sequence number, defines the right edge of the window. The transmitting process is permitted to send as many messages as would fill the window, but not more.

When a receiver receives a message whose sequence number is at the left window edge (or several consecutive messages extending rightward from the left window edge) the receiver returns an acknowledgement for the rightmost such message, along with a new credit, and advances his own window; its new left edge immediately follows the last acknowledged message and its new right edge is at the location defined by adding the new credit to the new left window edge. Similarly, when a sender receives an acknowledgement he advances his own left window edge to the location in the sequence number space specified by the acknowledgement and his own right window edge to the location specified by adding the new credit allocation to the new left window edge. Fields are reserved in each data message to carry an acknowledgement and a credit for traffic flowing in the reverse direction. Thus in the case of interactive or transactional exchanges, no control messages need to be sent.

In the event that a sender does not receive acknowledgements for previously transmitted messages within some timeout period, the messages are transmitted again, using the same sequence number as was previously assigned. This allows straightforward recovery from the situation of lost messages. On the other hand, if it is the returning acknowledgement which is lost, the fact that the retransmitted message carries an identical sequence number allows the receiver to discard it. However, the receiver should notice that at the time of retransmission the sender had not received an acknowledgement; therefore the receiver should reacknowledge this (and any subsequently received messages) by transmitting an acknowledgement bearing the current left window edge. Thus, in both the case of lost data messages and the case of lost acknowledgements the protocol remains synchronized.

The primary difference between this protocol and the IFIP Protocol is in the size of the sequence number field. The IFIP Protocol is designed for interconnections of many networks with huge variabilities in delay and with a strong possibility that messages will not be delivered at the destination in the same order in which they were transmitted by the source. Thus, the IFIP Protocol uses a 16-bit sequence number field which, even at megabit per second rates cannot be completely cycled through in less than several hours. However, the proposed ARPANET-type network has the characteristic that delays are typically short, messages are rarely lost, and they are always delivered in the same order in which they were sent if they are delivered at all. Therefore this Host/Host Protocol uses only a 4-bit sequence number field which, of course, is cycled through every 16 messages. This imposes the constraint that a window may never be larger than eight messages. Since the sequence number is contained in a 4-bit field it is also possible to use only four bits for each of the credit and acknowledgement fields; thus, this protocol uses only 12 bits in each message header rather than the 40 bits used under the IFIP Protocol.

III. NCP FUNCTIONS

The functions of the NCP are to establish connections, terminate connections, control flow, transmit interrupts, and respond to test inquiries. These functions are explained in this section, and control commands are introduced as needed. In Section IV the formats of all control commands are presented together.

Connection Establishment

The command used to establish a connection is the RFC (request for connection).

```
    8*       16             16            8        16        8
-------------------------------------------------------------------
!  RFC  ! my-socket ! your-socket !  index  !  size  !  credit  !
-------------------------------------------------------------------
```

The RFC command either requests the establishment of a connection between a pair of sockets or accepts a previously received request for connection. Since the RFC command is used both for requesting and accepting the establishment of a connection, it is possible for either of two cooperating processes to initiate connection establishment. Even if both proceses were to

----
*The number shown above each control command field is the length of that field in bits.

simultaneously request the establishment of a connection, each would interpret receipt of the RFC sent by the other as an acceptance of its own RFC, and thus the connection would be established without difficulty. The my-socket and your-socket fields in the RFC identify the sockets which terminate the ends of the connection at each Host. The index field of the RFC specifies an index number which will be contained in each data transmission sent over this connection from the "my-socket" to the "your-socket" end of the connection. The size field of the RFC specifies the maximum number of 8-bit bytes which are permitted to be sent from the "your-socket" to the "my-socket" end of the connection in any one message. The credit field of the RFC specifies the initial size (in the range 0-7) of the window in the "your-socket" to the "my-socket" direction of the connection. A pair of RFCs exchanged between two Hosts matches when the my-socket field of one equals the your-socket field of the other, and vice versa. A connection is established when a matching pair of RFCs has been exchanged.

Connections are uniquely specified by the sockets which terminate the connection; thus a pair of socket numbers cannot be used to identify two different connections simultaneously. Similarly, the index is used to specify which connection a data message pertains to; thus an index value cannot be reused while the connection to which it was first assigned is still active or in the process of being established. For example, consider an RFC sent from Host A to Host B whose my-socket field contains the value X, your-socket field contains the value Y, and index contains the value Z. Until the requested connection has been closed (even if it is never established) or reinitialized, Host A is prohibited from sending a different RFC to Host B whose my-socket field and your-socket fields are X and Y, or whose index field is Z. Note that the prohibition against the reuse of the values X and Y treats them as a pair; that is, another RFC may be sent from Host A to Host B, whose my-socket field contains the value X so long as the your-socket field contains some value other than Y.

In general there is no prescribed lifetime for an RFC. A Host is permitted to queue incoming RFCs and withhold a response for an arbitrarily long time, or, alternatively, to reject requests immediately if it has not already sent a matching RFC. Of course, the Host which originally sent the RFC may be unwilling to wait for an arbitrarily long time so it may abort the request.

The decision to queue or not to queue incoming RFCs has important implications which must not be ignored. Each RFC which is queued, of course, requires a small amount of memory in the Host doing the queueing. If the incoming RFC is queued until a local process takes control of the local socket and accepts (or rejects) the RFC, but no local process ever takes control of the socket, the RFC must be queued "forever". On the other hand, if no queueing is performed, the cooperating processes which may be attempting to establish communication may be able to establish this communication only by accident.

The most reasonable solution to the problems posed above is for each NCP to give processes running in its own Host two options for attempting to initiate connections. The first option would allow a process to cause an RFC to be sent to a specified remote socket, with the NCP notifying the process as to whether the RFC was accepted or rejected by the remote Host. The second option would allow a processs to tell its own NCP to "listen" for an RFC to a specified local socket from some remote socket (the process might also specify the particular remote socket and/or Host it wishes to communicate with) and to accept the RFC (i.e. return a matching RFC) if and when it arrives. Note that this also involves queueing (of "listen" requests) but it is internal queueing which is susceptible to reasonable management by the local Host.

Connection Termination

The command used to terminate a connection is CLS (close).

```
       8          16           16
   ----------------------------------
   !  CLS  ! my-socket ! your-socket !
   ----------------------------------
```

The my-socket field and your-socket field of the CLS command identify the sockets which terminate the connection being closed. Each side must send and receive a CLS command before the connection termination is completed and prohibitions on the reuse of the socket pair and index value are ended.

It is not necessary for a connection to be established (i.e. for both RFCs to be exchanged) before connection termination begins. For example, if a Host wishes to refuse a request for connection it sends back a CLS instead of a matching RFC. The refusing Host then waits for the initiating Host to acknowledge the refusal by returning a CLS. Similarly, if a Host wishes to abort its outstanding request for a connection it sends a CLS command. The foreign Host is obliged to acknowledge the CLS with its own CLS. Note that even though the connection was never established, CLS commands must be exchanged before the prohibition on the reuse of the socket pair or the index is completely ended. Under normal circumstances a Host should not send a CLS command for a connection on which that Host has unacknowledged data outstanding. Of course, the other Host may have just transmitted data so the sender of the CLS command may expect to receive additional data from the other Host.

A Host should quickly acknowledge an incoming CLS so that the foreign Host can purge its tables. In particular, in the absence of outstanding unacknowledged data a Host must acknowledge an incoming close within 60 seconds. Following a 60 second period, the Host transmitting a CLS may regard the socket pair and the index as "unused" and it may delete the values from any tables describing active connections. Of course, if the foreign Host malfunctions in such a way that the CLS is ignored for longer than 60 seconds, subsequent attempts to establish connections or transmit data may lead to ambiguous results. To deal with this possibility, a Host should in general "reinitialize" its use of connection parameters before attempting to establish a new connection to any Host which has failed to respond to CLS commands. Methods for reinitializing connection parameter tables are described below.

Acknowledgement

As described in the previous section, flow control is handled by a windowing scheme, based on sequence numbers. Credits and acknowledgements can be piggybacked on data traveling over the reverse channel. Thus, in general, acknowledgement of the receipt of messages will take place over the data connection rather than over the control connection. However, there are some cases when it may be desirable to pass acknowledgements over the control connection (for example, when there is no data to be returned in the reverse direction). In addition, for efficiency it may be desirable to negatively acknowledge data transmissions known not to have been delivered, rather than waiting for the timeout and retransmissions mechanism to cause such messages to be retransmitted. [Note that such negative acknowledgement is not required, since timeout and retransmission is always sufficient to guarantee eventual delivery of all data, but may be used to increase the efficiency of communication.] Since the frequency of use of the negative acknowledgement system over an ARPANET-type network will be extremely low, it is undesirable to leave space for negative acknowledgements in the header of every data message. Thus, negative acknowledgement can be most conveniently handled by control messages.

There are two commands dealing with acknowledgements.

```
      8        8        4       4
    ----------------------------------
    !  ACK   ! index !  seq   !  crd  !
    ----------------------------------
```

The ACK (acknowledgement) command carries three data fields. The index value is the index used by the sender of the acknowledgement to identify the connection. The sequence ("seq") field contains the sequence number of the highest-numbered sequential data message correctly received over the connection. [The very first data message to be transmitted over a newly established connection will have the sequence number one; until this data message is correctly received, any acknowledgement commands transmitted for this connection (for example, to change the credit value) will have the sequence field set to zero. This applies whether the "acknowledgement" is carried by an ACK command or is contained in data messages being sent to the foreign Host over the connection.] The credit ("crd") field contains a number, in the range 0-7, which gives the size of the receive window. This number, when added to the "seq", gives the sequence number of the highest numbered message which is permitted to be transmitted by the foreign Host. Thus, a credit of zero says that the Host transmitting the ACK command is currently not prepared to accept any messages over the connection; and a credit of 7 says that the Host is prepared to accept up to 7 messages over the connection. Of course, since the sequence number is contained in a 4-bit field, the addition of the sequence number and the credit value must be performed modulo 16 (sequence number 0 immediately follows sequence number 15).

As noted above, the ACK command is intended for use with data connections where there is no data flow in one direction, for example, the transmission of a file to a line printer. In fact it should be clear that, since transmission of control messages is not synchronized with transmission of data messages (either in the network or, more importantly, in the transmitting NCP), ACK commands should not be sent for any connection over which data is flowing in the same direction. Thus, if an ACK command is generated, the NCP which transmits it must insure that the control message which contains it is transmitted prior to the transmission of new data messages for the same connection.

```
        8        8        8
   --------------------------
   !  NACK  ! index !  seq  !
   --------------------------
```

The NACK (negative acknowledgement) command contains two data fields. As with the positive acknowledgement command described above, the first field is the index number assigned to this connection by the sender of the NACK. However, the second field contains only the 4-bit sequence number, right justified in an 8-bit field, of the data message for the connection in question which is being negatively acknowledged. As previously noted, the NACK serves no vital function in the protocol but may occasionally allow more efficient communication. The NACK is intended to be used when the window width is greater than one, the message at the left window edge has not been correctly received, and messages toward the right of the window have been correctly received. A timeout will eventually cause the retransmission of the missing message, at which point the left window edge can be moved forward several messages. Use of the NACK, however, could trigger the immediate retransmission of the missing message and thus reduce the delay. Of course, if more than one message is missing it may be desirable to send several NACKs for one index in a single control message; the protocol permits this, although it is extremely unlikely to occur.

Reinitialization

Occasionally, due to lost control messages, system crashes, NCP erors, or other factors, communication between two NCPs will be disrupted. One possible effect of any such disruption might be that neither of the involved NCPs could be sure that its stored information regarding connections with the other Host matched the information stored by the NCP of the other host. In this situation, an NCP may wish to reinitialize its tables and request that the other Host do likewise. This reinitialization may be requested for a particular index and/or socket pair, or globally for all connections possibly established with the other Host. For these purposes the protocol provides three control commands as described below:

```
        8        16           16          8
   ------------------------------------------
   !  RCP  ! my-socket ! your-socket ! index !
   ------------------------------------------
```

The RCP (reinitialize connection parameters) command contains three data fields. The my-socket and your-socket fields contain a pair of socket numbers which define a connection; the index field contains a value which would identify data messages over a connection. When this command is received by an NCP it should purge its tables of any reference to a connection identified by the socket pair or any reference to a connection for which received data would be identified by the specified index value; of course, only connections using these values with the Host sending the RCP would be purged. In effect, the Host sending the RCP command is saying: "I am about to send you an RFC using this socket pair and this index to identify a data connection which I hope we can agree to establish. I do not believe that use of this socket pair or this index conflicts with any previous use, but if you believe it does, please record the fact (for later examination) as an error and then delete from your tables the conflicting information so that we may proceed to establish the connection."

In case more global difficulties or loss of state information are suspected, the protocol provides the pair of control commands RST (reset) and RRP (reset reply).

```
     8
---------
!  RST  !
---------

     8
---------
!  RRP  !
---------
```

The RST command is to be interpreted by the Host receiving it as a signal to purge its tables of any entries which arose from communication with the Host which sent the RST. The Host sending the RST should likewise purge its tables of any entries which arose from communication with the host to which the RST was sent. The Host receiving the RST should acknowledge receipt by returning an RRP. Once the first Host has sent an RST to the second Host, the first Host should not communicate with the second Host (except for responding to RST) until the second Host returns an RRP. If both NCPs decide to send RSTs at approximately the same time, each Host will receive an RST and each must answer with an RRP even though its own RST has not been answered.

A Host should not send an RRP when an RST has not been received. Further, a Host should send only one RST (and no other commands) in a single control message and should not send another RST to the same Host until either 60 seconds have elapsed or a command which is not an RST or RRP has been received from that Host. Under these conditions, a single RRP constitutes an answer to all RSTs sent to that Host and any other RRPs arriving from that Host should be discarded.

Interrupts

It is sometimes necessary in a communication system to circumvent flow control mechanisms when serious errors or other important conditions are detected. For example, the user of a timesharing terminal who creates and begins the execution of a program which contains an erroneous infinite loop may need to "attract the attention"" of the operating system to ask it to cancel the execution of his program, even though the operating system may normally "listen" to the terminal only when the program in execution asks for input. Similarly, in a computer communication network, where flow control may prevent the transmission of data from one process to another, under certain extraordinary conditions it may be necessary to pass a signal from one process to another. Since the channel between the NCPs of two Hosts is not subject to the flow control mechanisms imposed on the data connections, it is possible to transmit such an "out-of-band" signal over the control connection, and for this purpose the INT (interrupt) command is provided.

```
        8       8       8
    -------------------------
    !  INT  ! index !  seq  !
    -------------------------
```

The INT command contains two data fields. The index field identifies the data connection to which the "interrupt" pertains; the sequence number ("seq"), which is four bits right-justified in an eight-bit field, gives the sequence number of the first data message which should "come after" the interrupt. In other words, the INT command notifies the receiving NCP of an exception condition which must be synchronized with the data stream, and the sequence number provides the necessary synchronization. Any data messages with sequence numbers to the left of the specified sequence number were generated before the exception condition arose.

An NCP which receives an INT command should advance the right window edge of the specified data connection so that the window contains at least the sequence number specified in the interrupt command. (It may be necessary to acknowledge data messages which were not correctly received or were not buffered in order to be able to advance the window to this point; justification is provided by the assumption that the INT was sent only because the flow control mechanisms were preventing the transmission of important information.) Of course, the interrupt or exception signal itself is subject to the interpretation of the Host receiving the signal, but should have a meaning equivalent to: "notify the process in execution, or that process' superior, that something exceptional has happened and that the data now buffered is an important message."

Test Inquiry

It may sometimes be useful for one Host to determine if some other Host is carrying on network conversations. The control command to be used for this purpose is ECO (echo).

```
     8       8
  -----------------
  !  ECO  !  data  !
  -----------------
```

The data field of the ECO command may contain any bit configuration chosen by the Host sending the ECO. Upon receiving an ECO command, an NCP should respond by returning the data to the sender in an ERP (echo reply) command.

```
     8        8
 -------------------
 !  ERP  !  data  !
 -------------------
```

A Host should respond (with an ERP command) to an incoming ECO command within a reasonable time, here defined as sixty seconds or less. A Host should not send an ERP when no ECO has been received.

IV. DECLARATIVE SPECIFICATIONS

Message Format

All Host-to-Host messages which conform to this protocol shall be constructed as follows:

Bits 1-96: Leader - This field is as specified in BBN Report No. 1822, with the following additional specifications.

Bits 38-40: Maximum Message Size - This field should be zero for all control messages. For messages sent over data connections, the value of this field should be calculated from the size received in the RFC which established the connection.

Bits 65-76: Message-id - This field is subdivided into eight bits giving the index of the connection of which the message is a part, and four bits giving the sequence number of the message. The index is contained in bits 65-72, and the sequence number in bits 73-76.

Bits 97-100: Acknowledgement - This field contains the four-bit sequence number of the highest-numbered data message to the left of the window for this connection; that is, the sequence number identifying the highest-numbered of the sequence of consecutively numbered (none missing) data messages which have been correctly received over this connection. If no data messages have been received since the connection was established, this field must contain the value zero. This field is not used (i.e. may have any value) in control messages.

Bits 101-104: Credit - This field contains a number in the range 0-7. Adding this number (modulo 16) to the sequence number in the acknowledgement field (bits 97-100) gives the highest sequence number which the foreign Host is permitted to send over this data connection. Thus, a value of zero in this field indicates that no new data messages should be sent, and a value of seven indicates that the foreign Host may send up to seven messages beyond the message whose sequence number is specified by the acknowledgement bits. Since flow control does not apply to messages sent over the control connection, this field may have any value in control messages.

Bits 105 - ... : Text and Padding - A sequence of 8-bit bytes of text, followed by padding, as specified in BBN Report No. 1822.

Index Assignment

Index values must be assigned (in bits 65-72) as follows:

Number	Assignment
0	Identifies a control connection
1	Reserved for revisions to this protocol
2-191	Identify data connections
192-255	Reserved for expansion or for other protocols

Sequence Number Assignment

Every data message contains a sequence number in bits 73-76. The sequence number is used by the receiver to detect the fact that a transmitted message has been lost, to identify the correct location in the data stream to insert a retransmitted (and therefore probably out of order) message which was previously lost (or to detect the retransmitted message as a duplicate) and to identify acknowledged messages (or sequences of messages) to the sender. The sequence number is also used by the flow control mechanism. Since the IMP subnetwork itself contains elaborate mechanisms to achieve these same goals, it is not anticipated that the error-recovery mechanisms based on the sequence numbers will be called into play frequently, and thus their efficiency is not of primary importance.

Sequence numbers are assigned to the two directions of a connection independently. For a given direction of a connection, the first data message transmitted after the connection is established must have sequence number one. Subsequent messages are assigned sequentially increasing (modulo 16) sequence numbers; that is, sequence number zero is assigned to the message following message number 15.

Sequence numbers are not assigned to control messages, since the protocol is designed to permit these messages to be delivered out-of-sequence without ill effect, and since flow control cannot be applied to the control link.

Control Messages

Messages sent over the control connection have the same format as other Host-to-Host messages, with the exceptions noted above. However, control messages may not contain more than 120 8-bit bytes of text. Further, control messages must contain an integral number of control commands: a single control command must not be split into parts which are transmitted in different control messages.

Message Transmission and Retransmission

Control messages may be transmitted whenever they are required. Data messages, however, may be transmitted only when permitted by the flow control mechanism; that is, whenever the sequence number assigned to the message is within the "window" for the appropriate direction of the given connection. The "left window edge" (LWE) is defined by the highest sequence number (modulo 16) which has been acknowledged (or zero, if no messages have been acknowledged). The "right window edge" (RWE) is defined by adding (modulo 16) the most recently received credit to the left window edge. [Note that LWE=RWE if the most recently received credit is zero.] A message with sequence number SEQ may be transmitted only if, prior to the (possible) reduction modulo 16 of SEQ and/or RWE, it is true that

LWE less-than SEQ less-than-or-equal RWE

Messages should be retransmitted whenever any of the following conditions occur:

- The IMP subnetwork has returned an "Incomplete transmission" (type 9) or "Error in Data" (type 8) response to the message (identified by having bits 41-76 of the response equal to those bits of the transmitted message). Note that this condition applies to control messages as well as data messages.

- The sequence number of this message is equal to (LWE + 1), and it has been more than 30 seconds since the message was last transmitted.

- The sequence number of the message is specifically identified in a NACK command for this connection from the foreign Host.

Since messages may occasionally have to be retransmitted, it is clear that they should not be discarded by the transmitting NCP until they have been acknowledged. A message is considered to be acknowledged when its sequence number, or the sequence number of any message to the right of it in the same direction of the given connection, is returned in the acknowledgement field of a data message transmitted in the other direction over this connection, or is returned in an ACK command for this connection from the foreign Host.

Control Commands

Control commands are formated in terms of 8-bit bytes. Each command begins with a one byte opcode. Opcodes are assigned the sequential values 0, 1, 2, . . . to permit table lookup upon receipt. The conditions underlying the design and anticipated use of the control commands are described in Section III.

NOP - No Operation

```
        8
    ---------
    !  NOP  !
    ---------
```

The NOP command may be sent at any time and should be discarded by the receiver. It may be useful for formating control messages.

RST - Reset

```
        8
    ---------
    !  RST  !
    ---------
```

The RST command is used by one Host to inform another that all information regarding any previously existing connections between two Hosts should be purged from the NCP tables of the Host receiving the RST. Except for responding to RSTs, the Host which sent the RST should not communicate further with the other Host until an RRP is received in response. When a Host is about to begin communicating (e.g. send an RFC command) to another Host with which it has no open connections, it is good practice to first send an RST command and wait for an RRP command.

RRP - Reset Reply

```
      8
  ---------
  !  RRP  !
  ---------
```

The RRP command must be sent in reply to an RST command.

RFC - Request for Connection

```
    8       16          16           8      16       8
 ----------------------------------------------------------
 ! RFC ! my-socket ! your-socket ! index ! size ! credit !
 ----------------------------------------------------------
```

The RFC command is used to establish a connection. The "my-socket" field specifies the socket local to the Host transmitting the RFC; the "your-socket" field specifies the socket local to the Host to which the RFC is transmitted. The "index" field specifies the index value which will be given in bits 65-72 of each data message sent from "my-socket" to "your-socket". The "size" field specifies the maximum number of 8-bit bytes which may be transmitted in any single message from "your-socket" to "my-socket". The "credit" field specifies the size of the initial sequence number window (in the range 0-7) in the "your-socket" to "my-socket" direction.

CLS - Close

```
           8        16            16
         ------------------------------------
         !  CLS  ! my-socket ! your-socket !
         ------------------------------------
```

The CLS command is used to terminate a connection. A connection need not be completely established before a CLS is sent.

RCP - Re-initialize Connection Parameters

```
           8        16            16          8
         ---------------------------------------------
         !  RCP  ! my-socket ! your-socket ! index !
         ---------------------------------------------
```

The RCP command is used by one Host to inform another that all information regarding a possibly previously-existing connection between "my-socket" and "your-socket" AND all information regarding a possibly previously-existing connection identified by "index" (between these Hosts) should be purged from the tables of the Host receiving the RCP. The "my-socket", "your-socket", and "index" fields are defined as in the RFC command.

ACK - Acknowledgement

```
        8       8       4       4
       -------------------------------
       !  ACK  ! index !  seq  !  crd  !
       -------------------------------
```

The ACK command may be used to acknowledge received data, or to assign credit, without sending a data message. The value in the index field identifies the data connection which uses the same index value (in the direction from the sender of the ACK to the receiver of the ACK). The eight bits following the index field (the "seq" and "crd" field) have the same meaning as bits 97-104 of the data message identified by the index value.

NACK - Negative Acknowledgement

```
        8        8       8
       -----------------------
       !  NACK  ! index !  seq  !
       -----------------------
```

The NACK command informs the receiver of the NACK that it should immediately retransmit the data message identified by the remaining fields. The index field is defined exactly as for the ACK command. The "seq" field gives the 4-bit sequence number (right-justified) which should be immediately retransmitted. Note that the data message to be retransmitted does not have an index value equal to "index", but instead is transmitted over the other direction of the data connection which the Host sending the

NACK identifies by "index". No Host is ever required to transmit or act upon a NACK command; however, use of the NACK may occasionally permit a decrease in retransmission delay.

INT - Interrupt

```
     8       8       8
  ------------------------
  !  INT  ! index !  seq  !
  ------------------------
```

The INT command is sent over the control link to provide an "out-of-band" (and hence not subject to flow control) signal for the data connection denoted by the index field. The index value is the value which would appear in bits 65-72 of a data message sent from the sender of the INT command to the receiver of the INT command. The means of synchronizing this signal with the data being transmitted over the data connection is the inclusion of a 4-bit sequence number (right-justified) in the "seq" field. The number specified by this field denotes the first data message which "follows" the out-of-band signal.

ECO - Echo Request

```
     8       8
  ----------------
  !  ECO  ! data  !
  ----------------
```

The ECO command is used only for test purposes. The data field may be any bit configuration convenient to the Host sending the ECO command.

ERP - Echo Reply

```
      8       8
 -----------------
 !  ERP  !  data !
 -----------------
```

The ERP command must be sent in reply to an ECO command. The data field must be identical to the data field in the incoming ECO command.

Opcode Assignment

Opcodes are defined to be 8-bit unsigned binary numbers. The values assigned to opcodes are:

```
NOP  =  0
INT  =  1
RFC  =  2
CLS  =  3
ACK  =  4
NACK =  5
RCP  =  6
RST  =  7
RRP  =  8
ECO  =  9
ERP  = 10
```

NIC/QUERY, A Novice User Interface Program

E. J. Feinler and D. S. Maynard

SRI, Augmentation Research Center, Menlo Park, Calif.

ABSTRACT

This paper describes the NIC/QUERY program and the data bases that run under it. NIC/QUERY permits novice or sometime users to access a data base made up of structured files. It uses a few simple commands and is designed to be self-explanatory. The query program is a subsystem of the SRI Augmentation Research Center Online System (NLS). NLS utilizes heirarchically structured files. It also supports a linking capability which allows special strings of text, called links, to reference any node in any NLS file. NIC/QUERY takes advantage of these two NLS features. Users do not need any knowledge of NLS to use the query program, even though the program draws upon sophisticated NLS procedures.

NIC/QUERY is being used by the ARPA Network Information Center at SRI to enable network users to browse through information about the ARPANET. This application of the program is described in detail. It shows how an NLS user, without programing experience, can construct a data base which can in turn be easily browsed by a novice user through use of the NIC/QUERY subsystem.

INTRODUCTION

In 1972 the Network Information Center (NIC), a part of the Augmentation Research Center at SRI, was asked to participate in a demonstration of the ARPANET at the International Computer Communication Conference (ICCC). A large amount of information had already been gathered describing the ARPANET, but a method was needed for presenting this information to the conference attendees, many of whom were not computer oriented. The information was to be accessed through the network, since that was the emphasis of the meeting. We also wanted to draw heavily upon useful features of the SRI Online System (NLS), particularly the heirarchically structured file system and linking capability. However, we felt that the

NLS command language and subsystem access were too sophisticated for random users. It was this need for a simple subset of the full NLS capability that would allow novice users to enter NLS and view data files with little or no knowledge of the system, that triggered the concept and design of NIC/QUERY.

NLS FILE SYSTEM

To understand how the Query program accesses data it is necessary to have an overview of the NLS file structure (4)(6).

NLS operates on a heirarchical, random file system with several unique features that provide advantages for entering, organizing, and accessing data. Storage of information within separate structure and data blocks permits rapid movement within and between NLS files.

Conceptually, an NLS file is a tree. Each node has a pointer to its first subnode and a pointer to its successor. If it has no subnode, the subpointer points to the node itself. If the node has no successor, the successor pointer points to the node's parent. Each node is currently represented by a ring element. These ring elements point in turn to the associated data block.

The basic unit of an NLS file is the 'statement' which is a string of text made up of from 1 to 2000 characters. A statement may be a character, line, sentence, heading, paragraph, table, or graphic. A file is composed of one or more statements that can be arranged at different levels in an outline or tree form. The text in an NLS file can be clipped and re-expanded, and the user's view of it may be altered by single-letter codes known as viewspecs.

For a more detailed description of the NLS file structure see (4). For the purpose of this discussion it is sufficient to know that NLS has a heirarchical or tree-like file system made up of multilevel statements. Statements, or text within statements, may be easily referenced, and level and view clipping are available.

STRUCTURE OF THE DATA BASES

In order to discuss NIC/QUERY in specific terms - what it does and how it works - an NLS file, which happens to be a short description of one of the Hosts on the ARPANET

(LLL-RISOS, dec 21), has been included as a sample data base (See Appendix 1). This is a typical file used to produce the hardcopy ARPANET Resource Handbook, a document published by the NIC under the sponsorship of the Defense Communications Agency (DCA) for distribution to ARPANET users (2). The same files used to produce the hardcopy handbook can with minor modifications serve as the data base for Query. This eliminates the need to recapture or reorganize the data extensively for hardcopy production.

The basic structure of this sample file is that of an outline. A heading or data element is followed, down a level, by the data pertaining to it. Note that some of the descriptive text is presented in paragraph form and some is presented as tables. Either is acceptable.

HOW NIC/QUERY WORKS

Essentially NIC/QUERY does the following: it creates a numbered menu from any headings or data element names at a given level and presents this menu to the user. The user then selects one of the menu items or specifies a keyword. Upon selection of an item or data element by the user, the program displays the text one level beneath that data element. The user can 'walk' down through the outlined file in a logical sequence opening up successive levels of information. He can also jump back to a previous menu or go back to a higher node in the tree structure and go down a different information path. This path can be a branch within a file or an entirely new file; thus the data base can be a single file or multiple files, whichever the builder chooses.

Although this is a very simple concept, its application creates a powerful data access tool which any NLS user can drop over a properly structured NLS file.

The simplicity of the program does mean that it has several limitations. These are important to note:

- NIC/QUERY is not primarily a data searching tool, it is rather a data displaying tool. It will not find all occurrences of a given subject or keyword, although it will find the first occurrence of a keyword within a given file. It does not have Boolean searching capability at present.

- The current NIC/QUERY is designed for use on a teletype terminal rather than a CRT display, although it could easily be adapted for either.

- NIC/QUERY is not suitable for very large data bases. The file size is not open-ended. NLS was not designed to be a data base management system in the traditional sense, and does not effectively support multiple parallel organizations on one data file. There is also a limit to the number of initial data paths that can be introduced to the user and still be meaningful.

- Although the user can go down many successive levels within a file, there is a limit to how many levels he wants to go down before the number of menu selections becomes impractical.

As presently implemented, NIC/QUERY is a useful tool for displaying moderate amounts of data, but it is not suitable for searching or for accessing large data bases. (The ARPANET Resource Data Base contains approximately 150 files and represents about 1500 pages of written text. This is what is meant by 'moderate', as opposed to, for example, large bibliographic data bases with many thousands of records.)

Another limitation that should be mentioned is that of insufficient online access. A computer query system is useful only if the user has enough access, and the response time is adequate for his needs. An inherent problem in any online query system is the time required for finding and displaying information compared with looking the same information up in a well-organized book. On the other hand, online data is easier to keep current than that contained in a book, and it is at the user's finger tips when needed. All other factors being equal, the amount of online access and speed of data display will probably have a strong influence on user acceptance of a fully automated query system.

THE COMMAND LANGUAGE

The original ICCC version of the query program had two commands: 'show' and 'quit' (8). It was obviously easy to learn, but severely limited movement within files. Also, the original version required the user to type in the name of the data element that he wished to 'show'. This was time consuming and introduced typing errors. Consequently, a few more commands and a numbered menu selection feature were added. It should be emphasized, however, that the point is NOT to have many commands, and indeed, one of the hardest design problems has been to keep new commands from creeping in when the present ones, or lack thereof, will do.

The command language for NIC/QUERY is very simple and consists of the following commands:

[@]NIC <CR>

where 'NIC' is a single program-access word, typed by the user immediately after login. It automatically loads NLS and the Query subsystem and presents the user with his first set of instructions. (This word can easily be changed for access to data bases other than those of the NIC.)

TYPEIN <CR>

where TYPEIN is usually a menu number corresponding to a chosen data element followed by a carriage return. In addition the user may at any time type in a keyword. Query will search the data tree below the current node for a corresponding data element, and it will display the first data element it finds which matches the search criteria. The user may also type in a phrase as a series of words separated by spaces. Query will perform the indicated word searches sequentially. This provides a user who is familiar with the data base, or a branch therein, with direct access to the data element he seeks. In essence it allows a more experienced user to 'come in from the side' of a file rather than down from the top.

As an optional feature the data base builder may also define additional "index" files which contain keywords and pointers to relevant data nodes. These index files will be searched automatically whenever the given keyword cannot be found within the user's current context.

^

takes the user up to the next highest level and displays the menu for that level.

-

takes the user back through previously displayed nodes, one at a time. The user can accept the choice displayed or continue jumping back. This facility might be thought of as a list of 'book markers' marking nodes where the user has already been and to which he may want to return.

?

displays a list of legal commands

CONTROL-O

aborts TYPEIN back to the prompt

CONTROL-A

backspaces a letter

CONTROL-W

backspaces a word

DEL (CONTROL-X)

aborts session but gives user the option of leaving the system, continuing or, beginning again, in case the abort was accidental.

QUIT <CR>

aborts session and returns the user to EXEC level.

BUILDING THE INSTRUCTION SET AND LINK TABLE

Before discussing the format and content of the instruction set and link tables, the concept of NLS links will be introduced. Links are strings of characters in a statement that name the address of any location in any NLS file along with a specified view. Links have the syntax <ADDRESS : VIEWSPECS> and may have several kinds of delimiters. Links may be automatic (that is, during a pointing command, if the program finds the link syntax within the specified text, it will automatically jump to the address specified by the link), or they may be manual (the user must issue a command 'Jump to link' and bug or address the link string.)

Query makes extensive use of automatic links which the user never sees. Thus a query data base may be simply a table of links off to multiple files or addresses within files. Appendix 3 gives a condensed version of the link table for the ARPANET Resource Data Files. This particular data base is composed of over 100 small files, and was chosen to demonstrate how easily multiple files can be included under one query 'umbrella'.

A Query data base can also consist of only one file, if the builder chooses. In fact, this would be the simplest to build. With only a single data file, links would be missing altogether or would point to addresses within the file.

When the user types the recognition word to enter Query, e.g. NIC <CR>, he enters NLS and the query subsystem automatically, and causes a file containing the initial menu of topics and the self-explanatory instruction set to be loaded and displayed. The contents of this file can be tailored to fit the data being viewed, and it is written by the data base builder.

The instruction set generally explains how to make a menu selection and states the purpose for which the data is intended. It then gives a beginning menu from which the user can begin branching downward. One item on the initial menu can be a verbose HELP file corresponding to a HELP document when printed out. Any node may have an instruction set of its own pertaining only to that node.

A LOOK AT THE ARPANET RESOURCE DATA THROUGH QUERY

Appendix 2 gives a brief scenario of a typical ARPANET user querying the ARPANET Resource Data Files.

QUERY AS AN NLS SUBSYSTEM

Originally NIC/QUERY was a separate user program, or what is commonly known as a 'hack'. It was designed for a demo and was thought of as transitory. However, as often happens with such programs, it was useful and refused to go away. In the first version the command language and instruction set were specified by the program and were specific to the NIC data base. The instructions and introductory menus could not be altered by the data base builder without an understanding of the code and without being able to modify and recompile the program.

As other applications for the same type of novice access arose, several similar NLS tools began to evolve. The NLS online HELP system was one of these (5). Implementation of this system provided many of the design features now incorporated into NIC/QUERY. To accomodate other NLS users who wanted to adapt similar programming concepts to their own data bases, NIC/QUERY was rewritten as a generalized data access tool. Now it is a subsystem of NLS available to all NLS users. The instruction set and introductory files can be written and edited by the data base builder. The command language is written in a high level language, so that commands as well as prompts associated with commands can be easily altered with a minimum of programming effort. This means that the Query program has achieved a high degree of flexibility, and can be used by NLS users to display a variety of data bases of their own design and choosing.

FUTURE POSSIBILITIES

As stated above NIC/QUERY is not a searching tool at present. We would like to add searching capability including Boolean logic. Since NLS is not a data base management system, we would like to interface NLS with an already existing database management system in such a way that information could be captured, edited, and viewed in NLS, but stored and searched in the data base management system. Data files could be queried through the query program and the results displayed at the user's console. The user would not need to know that more than one system was invoked to produce an answer, nor what the hand shakes were among them.

It would be useful to be able to store data elements sequentially within statements as well as in separate statements. This is possible now but needs to be simplified for the data base builder. We would like to have an easily constructed data input prompting system. This would allow the data base builder to specify data input elements for which the system would construct suitable input prompts, e.g., for a data base consisting of personnel data, the builder might specify that NAME, ADDRESS, and TELEPHONE NUMBER are data elements. The input system would then prompt a clerk for these data elements while the question-mark facility would supply added instructions.

NLS is a sophisticated, extensible, modular information processing system. A growing online user community uses

NLS to maintain a wide variety of information. To the systems builder NLS provides a unique environment for modular programming. A Command Meta Language (CML), which is a high level language for specifying the user interface of a modular extension or subsystem, is available (1). A Command Language Interpreter (CLI) parses the user input according to the grammar specified by the CML (3). The CLI performs the functions of command recognition, command completion, command feedback, prompting, response to ?, and limited editing and validation of user entered parameters. NLS also provides a rich set of primitives for manipulating information within NLS files (7).

Many application subsystems for managing particular data bases have been built using these NLS tools (9). Typically these subsystems perform the tasks of data entry, data verification, data update, interactive query, and report generation. Implementation time for producing such a subsystem within the NLS environment is on the order of a few man-months from design to completion.

A general methodology of data manipulation and display is evolving out of the work already done on these various subsystems, of which NIC/QUERY is only one piece. The goal is to provide a system which allows any NLS user to effectively become a systems builder and data base administrator. In other words, the average NLS user will be able to define, create, and maintain a data base easily, using only his existing NLS capabilities. Query will provide an accessing mechanism for his data base suitable for a wide range of users.

SUMMARY

NIC/QUERY is a program that permits nonprogrammers to present data to a novice user population. Combined with the text editing and data handling features of the SRI Online System (NLS), it constitutes a useful approach to providing data access in a networking environment. The interesting features of NIC/QUERY are: 1) the data base and the instruction set can be written by a nonprogrammer, 2) the program is self explanatory, 3) files used for the data base can be readable, outlined files usable for other purposes, 4) the command language and prompts can be changed with simple programming modifications, and 5) access to NLS, Query, and the data files is automatic. Query is one of a collection of data-handling tools provided by the SRI Augmentation Research Center NLS system.

REFERENCES

1. Dornbush, C. F., Victor, K. E. and Irby, C. H. A Command Meta Language for NLS, ARC 22130, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., Jan. 10, 1975.

2. Feinler, E. J. ARPANET Resource Handbook, NIC 23200, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., Sept. 1975.

3. Irby, C. H. The Command Meta Language System, ARC 27266, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., Jan. 6, 1976.

4. Knowledge Workshop Development, ARC 22133, Final Rept. on SRI Proj. 1868, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., Jan. 30, 1976, p. 17.

5. Ibid, p. 15.

6. Lehtman, H. G. NLS File Structure: Documentation, ARC 17069, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., June 6, 1973.

7. Smith, D. C., Maynard, D. S., and Cornish, J. A. Information Retrieval Primitives for NLS, ARC 27356, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., May 13, 1976. (An SRI-ARC Internal Working Paper)

8. Vallee, J. F. QI: A Simple Retrieval Tool for TNLS Structured Files, NIC 11841, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., Sept. 19, 1972.

9. White, J. E. QUERY2, L10 Source Code, NIC 28001, Augmentation Res. Center, Stanford Res. Inst., Menlo Park, Calif., May, 7, 1976. (Not available for distribution)

APPENDIX 1

SAMPLE OF AN NLS FILE

```
(LLL-RISOS)  UNIVERSITY OF CALIFORNIA
             LAWRENCE LIVERMORE LABORATORY

   (FUNCTION)

      SERVER   COMPUTER: PDP-11/45   HOST ADDR. 21

   (ADDRESS)

      University of California
      Lawrence Livermore Laboratory
      RISOS Project, L-307
      Box 808
      Livermore, California 94550

   (PERSONNEL)

      PRINCIPAL-INVESTIGATOR
        Robert P. Abbott (ABBOTT@ISI) (415) 447-1100

      LIAISON
        James E. Donnelley (JED@BBN)  (415) 447-1100

      SOFTWARE-CONTACT
        Charles R. Landau (RISOS@ISI) (415) 447-1100

      HARDWARE-CONTACT
        Doyle R. Hopkins (RISOS@ISI)  (415) 447-1100

   (HARDWARE)

      (COMPUTER)

         TYPE        CORE AMOUNT   CORE SPEED     WORD
         LENGTH
         PDP-11/45   56K           1.0 microsec   16 bits

      (PERIPHERALS)

         HOW MANY    TYPE                 MAKE      MODEL

         DISKS
           1         1.2M word platter    DEC       RK11
           1         20M word pack        DEC       RP11
```

DRUMS None

TAPES

1	9 track, 1600 bpi	DEC	TU66F-9/45
1	dual Dectape	DEC	TU56

PRINTERS

1	96chr, 7x9 matrix	Versatec	LP1150

CARD-READERS

1	300 cpm	DEC	CR11

DATA-COMMUNICATIONS

5	300 bd auto ans	Vadic	VA305D
3	Auto dial	Vadic	VA801AD
1	2000 bd synch	ICC	2200/20

(TERMINALS)

HOW MANY	TYPE	MAKE	MODEL
7	96 chr 5x7 matrix	TI	733 KSR
3	96 chr 5x7 matrix	TI	725
1	96 chr CRT w/edit	Hazeltine	2000
6	128 chr CRT w/edit	Beehive	SB2

(OPERATING-SYSTEM)

The RATS (RISOS Arpanet Terminal System) is a capability list, virtual memory, multiprogramming system. RATS allows controlled communication between mutually suspicious processes in separate environments and permits naturally extended use of all programs through a user definable capability mechanism.

(USER-PROGRAMS)

RATS currently has very few user programs. Most of the existing programs are utility programs or programs for using the communication equipment on the system. All available programs are kept in the RATS public directory and the available program documentation is kept online (see HELP below). Anyone that is interested in writing programs for a C-list system are encouraged to try their hand at RATS. Contact the Liaison for more information.

(SYSTEM-USE)

Currently undergoing rapid change. Users interested in writing software are welcome. Look for online documentation or contact the Liaison.

(HELP)

The command language processor and most subsystems respond to "?" with a list of available commands and to "help <CR>" with more complete information.

(PROTOCOLS)

(SERVER)

The ARPA Network server protocols currently implemented are:

1. TELNET (new server only. Socket 23)

(USER)

The ARPA Network user protocols currently implemented are:

1. TELNET

2. FTP

(INTERESTS)

The RISOS (Research In Secured Operating Systems) project at the Lawrence Livermore Laboratory is interested in the ARPA Network largely to connect to remote computers on the network for testing operating system integrity and for transporting files (e.g., operating system listings) and general communication (e.g., network mail). The heavy communications orientation of the RATS system has demanded a continued active interest in communication protocols.

The LLL IMP has been eyed by several other groups at the Laboratory for applications from file transport of data at LLL for specialized processors on the ARPANET to possible connection of the Laboratory's unclassified CDC-6600 computer. Such applications are currently only speculation.

The Laboratory is also currently designing and building a computer network for use by the ERDA laboratories doing research on controlled thermonuclear fusion. (This has been dubbed the CTR subnet).

(DOCUMENTATION)

For a description of RATS as of March 1, 1974, see:

1. An Introduction to RATS (RISOS/ARPA Terminal System): An Operating System for the DEC PDP-11/45, UCRL 51582, Lawrence Livermore Laboratory, Livermore, Calif. (Mar. 1, 1974)

Available from the Technical Information Department of the Lawrence Livermore Laboratory, Box 808, Livermore, Calif. 94550.

The current RATS Handbook is available online as <RISOS>RATS-HANDBOOK.TXT from BBN using the Anonymous login convention under BBN's FTP server.

APPENDIX 2

SCENARIO OF ACCESS TO A DATA BASE THROUGH NIC/QUERY

The following is a brief scenario of a typical ARPANET user querying the ARPANET Resource Data Files. ("USER: xxx" = what the user types; "QUERY: xxx" = what the program replies; <CR> = a carriage return typed by the user)

USER: nic <CR>

QUERY:

The NIC/QUERY system lets ARPANET users view files that describe the ARPANET. Each list of topics is presented to the user as a numbered menu of selections.

To see more detail on one of the topics below, type its corresponding number followed by a carriage return <CR>.

1. HELP and additional Query commands
2. GENERAL-INFORMATION about the ARPANET
3. PERSONNEL-INFORMATION
4. HOSTS
5. ACRONYMS
5. PROGRAMS
6. PROTOCOLS
7. DOCUMENTATION

< for back, ^ for up, or TYPEIN:

USER: 1 <CR>

QUERY:

(HELP) and additional query commands

? Gives a list of legal commands.

^ Takes you up to the next highest level and redisplays the menu for that level.

_ Takes you back to previous views. For each previous view the system will show you the first line of the view

```
                 and ask "Y/N?" If the view shown is
                 the one you want, type:  y <CR>.  If
                 it is not, type:  n <CR> to see another
                 view.

      TYPEIN     You can type a keyword followed by a
                 <CR> instead of typing a menu number.
                 The system will search for this keyword.

      DEL        Aborts session but gives you the option
                 of continuing, beginning again, or
                 leaving the system in case the abort
                 was accidental.

      CONTROL-O  Aborts TYPEIN back to the last prompt

      CONTROL-A  Backspaces a character

      CONTROL-W  Backspaces a word

      quit <CR>  Aborts session and returns you to "@".
              (NOTE:  You must type out the word 'quit'.)

      < for back, ^ for up, or TYPEIN:

USER:  4 <CR>

QUERY:

   (HOSTS)
     1. SERVERS
     2. USERS
     3. TIPS

   < for back, ^ for up, or TYPEIN:

USER:  1 <CR>

QUERY:

   (SERVERS)
     1. AMES-67       2. ANL            3.ARPA-DMS
     4. BBN-11X       5. BBN-TENEX      6. BBN-TENEXA
     7. BBN-TENEXB    8. BBN-TENEXD     9. CCA-TENEX
     10. CHII         11. CMU-10A       12. CMU-10B
     13. HARV-10      14. HAWAII-500    15. I4-TENEX
     16. I4B-TENEX    17. LBL           18. LL
                      ---- ETC ----
```

```
   < for back, ^ for up, or TYPEIN:

USER:  17 <CR>

QUERY:

   (LBL)   UNIVERSITY OF CALIFORNIA
           LAWRENCE BERKELEY LABORATORY
           MATH AND COMPUTING DEPARTMENT

1. FUNCTION       2. ADDRESS                 3. PERSONNEL
4. ACCOUNTING     5. SERVICE-SCHEDULE        6. LOGIN
7. LOGOUT         8. CONTROL-CHARACTERS      9. HELP
10. NETWORK-COMMANDS 11. PROTOCOLS          12. HARDWARE
13. OPERATING SYSTEM 14. USER-PROGRAMS 15. INTERESTS
16. DOCUMENTATION

< for back, ^ for up, or TYPEIN:

USER:  3 <CR>

QUERY:

   (PERSONNEL)
     MANAGERS
       James A. Baker                    (415) 843-2740
       David F. Stevens                  (415) 843-2740
     LIAISON
       Dennis E. Hall (DEHall@SRI-AI)(415) 843-2740
     ACCOUNTS
       Thomas P. Hitchcock               (415) 843-2740

                        ---- ETC ----

   < for back, ^ for up, or TYPEIN:

USER:  2 <CR>

QUERY:

   (ADDRESS)
     University of California
     Lawrence Berkeley Laboratory
     Math and Computing Department
     Building 50B, Room 1143
     Berkeley, California 94720

USER:  111-risos <CR>
```

```
   (NOTE:  Here the user 'came in from the side' by
   typing a keyword, since he knew the name of the host
   whose write-up he wished to view.

QUERY:

   (LLL-RISOS)  UNIVERSITY OF CALIFORNIA
                LAWRENCE LIVERMORE LABORATORY

   1. FUNCTION     2. ADDRESS            3. PERSONNEL
   4. HARDWARE     5. OPERATING-SYSTEM   6. USER-PROGRAMS
   7. SYSTEM-USE   8. HELP               9. PROTOCOLS
   10. INTERESTS   11. DOCUMENTATION

   < for back, ^ for up, or TYPEIN:

USER: 1 <CR>

QUERY:

   (FUNCTION)
     SERVER     COMPUTER: PDP-11/45     HOST ADDR 21

   < for back, ^ for up, or TYPEIN:

USER: ^ <CR>

QUERY:

   (LLL-RISOS)   UNIVERSITY OF CALIFORNIA
                 LAWRENCE LIVERMORE LABORATORY

   1. FUNCTION     2. ADDRESS            3. PERSONNEL
   4. HARDWARE     5. OPERATING-SYSTEM   6. USER-PROGRAMS
   7. SYSTEM-USE   8. HELP               9. PROTOCOLS
   10. INTERESTS   11. DOCUMENTATION

   < for back, ^ for up, or TYPEIN:

USER: 5 <CR>

QUERY:

   (OPERATING-SYSTEM)
     The RATS (RISOS ARPANET Terminal System) is a
   capability list, virtual memory, multiprogramming
   system.  RATS allows controlled communication between
   mutually suspicious processes in separate environments
```

and permits naturally extended use of all programs through a user definable capability mechanism.

< for back, ^ for up, or TYPEIN:

USER: <DEL> <CR>

QUERY:

Type C to continue, Q to quit, or B to begin again:

USER: b <CR>

QUERY:

(HELP-Q)

The NIC/QUERY system lets ARPANET users view files that describe the ARPANET. Each list of topics is presented to the user as a numbered menu of selections.

To see more detail on one of the topics below, type its corresponding number followed by a carriage return <CR>.

1. HELP and additional Query commands
2. GENERAL-INFORMATION about the ARPANET
3. PERSONNEL-INFORMATION
4. HOSTS
5. ACRONYMS
6. PROGRAMS
7. PROTOCOLS
8. DOCUMENTATION

< for back, ^ for up, or TYPEIN:

USER: programs <CR>

QUERY:

(PROGRAM)
1. BY-NAME
2. BY-TYPE

< for back, ^ for up, or TYPEIN:

USER: 1 <CR>

QUERY:

(BY-NAME)
Typein a program name followed by a <CR>, e.g. ALGOL <CR>

This will give a definition of the program, and list the hosts on which it is available.

< for back, ^ for up, or TYPEIN:

USER: zog <CR>

QUERY:

ZOG
CMU query language. An interactive guide for novice users which provides an introduction to many available systems.
CMU-10A
CMU-10B

< for back, ^ for up, or TYPEIN:

USER: <<< <CR> (NOTE: User has backed up through 3 nodes)

QUERY:

"(BY-NAME)" Y/N: "(PROGRAMS)" Y/N: "(HELP-0)" Y/N

USER: n <CR> [to first two]; y <CR> [to final node]

QUERY:

(HELP-0)
[displays beginning menu again.]

< for back, ^ for up, or TYPEIN:

---- Here user decides to stop ----

USER: quit <CR>

QUERY: Leaving Query

APPENDIX 3

NIC/QUERY BEGINNING INSTRUCTION FILE AND TABLE OF LINKS

The Scenario above shows what text the user sees. This sample file shows everything in a beginning instruction file and link table, much of which the user does not see. Hidden links are recognizable by the
<DIRECTORY,FILENAME,STATEMENT-NUMBER:VIEW> ## syntax.
Also, the NOTEd explanations are not part of the actual file.

(HELP-Q)

The NIC/QUERY system lets ARPANET users view files that describe the ARPANET. Each list of topics is presented to the user as a numbered menu of selections.

To see more detail on one of the topics below, type its corresponding number followed by a carriage return <CR>.

(HELP) and additional query commands

(NOTE: This menu item has no link since the information to be displayed to the user occurs one level below the element name 'HELP')

? Gives a list of legal commands.

^ Takes you up to the next highest level and redisplays the menu for that level.

\- Takes you back to previous views. For each previous view the system will show you the first line of the view and ask "Y/N?" If the view shown is the one you want, type: y <CR>. If it is not, type: n <CR> to see another view.

TYPEIN You can type a keyword followed by a <CR> instead of typing a menu number. The system will search for this term.

DEL Aborts session but gives you the option of continuing, beginning again, or

leaving the system in case the abort
was accidental.

CONTROL-O Aborts TYPEIN back to the last prompt

CONTROL-A Backspaces a character

CONTROL-W Backspaces a word

quit <CR> Aborts session and returns you to "@".
(NOTE: You must type out the word 'quit'.)

(GENERAL-INFORMATION) about the ARPANET
<NETINFO,HELP-GEN,1:g>

(NOTE: This is a hidden link off to another file
that has its own instructions)

(PERSONNEL-INFORMATION)
<*,IDENTS,1:g>

(HOSTS)

(SERVERS)

(AMES-67)
<netinfo,ames-67,1>

(ANL)
<netinfo,anl,1>

(ARPA-DMS)
<netinfo,arpa-dms,1>

---- ETC ----

(NOTE: ---- ETC ---- indicates that the
list of links is actually much longer.
Only enough examples are included here to
show the structure of the link table.)

(USERS)

(ADR)
<netinfo,adr,1>

(AMES-11)
<netinfo,ames-11,1>

```
      (ARC-RD)
      ## <netinfo,arc-rd,1> ##

         ---- ETC ----

   (TIPS)

      (AFWL-TIP)
      ## <netinfo,afwl-tip,1> ##

      (ALOHA-TIP)
      ## <netinfo,aloha-tip,1> ##

      (AMES-TIP)
      ## <netinfo,ames-tip,1> ##

         ---- ETC ----

(ACRONYMS)
## <pi,bigboy,1a:xrryr> ##

(PROGRAMS)
## <netinfo,PROGRAMS,1:g> ##

(PROTOCOLS)
## <NETINFO,HELP-PROTOCOLS,1> ##

(DOCUMENTATION)
## <nic,rfc-index,1> ##
```

0686-76

76 686

NWG/RFC #716
NIC #35534

David Walden
Joel Levin
May 24, 1976

Interim Revision to Appendix F of BBN Report 1822

Over the past few months we have become aware that there has been some confusion as to how to operate a Host connected to an IMP as a Very Distant Host (or VDH). Therefore, next time BBN Report 1822 ("Specifications for the Interconnection of a Host and an IMP") is revised, we will include additional information on how the IMP side of a VDH connection works and how the Host side may operate most efficiently. As an interim measure, we are distributing this RFC which takes the form of a (logical) update to Appendix F of BBN Report 1822.

On page F-6 on Appendix F, delete the second footnote.

On page F-7, find the phrase "... and the odd/even bit is complemented." on line 17 of the page. Delete the rest of the page and insert the following text:

> In a standard Host to IMP interface, messages are delivered in a specific order and received in the same order. A Very Distant Host interface operates similarly in that messages are passed, for example, from the IMP to its RTP in order; the Host's RTP then delivers them to its receiving process in the same order. It is important to note, however, that between these two software interfaces there is nothing said about ordering. In particular, if the special interface detects an error in a packet, for example, the receiving RTP will discard the packet. The next packet may arrive on another logical channel before the sending RTP retransmits the discarded and unacknowledged packet, and the receiver should be prepared to accept this packet out of order. The protocol described above explicitly permits such out-of-order behavior between the RTPs, requiring only that the transmit portion of the RTP fill its channels in sequence (one to channel zero, one to channel one, one to channel zero, etc.), and that the receive portion of the RTP empty its channels in sequence. In addition, to insure correct sequencing, the first channel filled or emptied after initialization must be channel zero. Null packets use neither a channel nor a channel number when sent and are not acknowledged when received.

When packets must be retransmitted until acknowledged, processing and transmission delay may cause acknowledgement to be delayed for more than one transmission time. Unnecessary retransmission may interfere with new transmissions, as well as placing an added burden on both receiver and transmitter. Therefore, we recommend a program delay before deciding to retransmit an unacknowledged packet. This amount of delay should be adjustable, but we recommend a trial value of 100 msec. Additional efficiency may be gained if the RIP can notice that the next packet has been acknowledged while the previous one has not: in this case, it is clear that the first packet was not correctly received and it may be retransmitted immediately without waiting for the programmed delay to expire. This option has not, however, been implemented in the IMP at this time.

BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BB
BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BB
BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BB

FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER

(JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)3571
(JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)3571
(JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)35710 (JAKE)3571

Thursday, June 17, 1976 19:11:13-EDT Thursday, June 17, 1976 19:11:13-EDT Th
Thursday, June 17, 1976 19:11:13-EDT Thursday, June 17, 1976 19:11:13-EDT Th
Thursday, June 17, 1976 19:11:13-EDT Thursday, June 17, 1976 19:11:13-EDT Th

< LJOURNAL, 35710.NLS;1, >, 11-Jun-76 22:38 XXX ;;;; Title:
Author(s): N. Dean Meyer/NDM; Distribution: /SRI-ARC([INFO-ONLY]) ;
Sub-Collections: SRI-ARC; Clerk: NDM; Origin: < MEYER,
TEMPLATE-DOC.NLS;5, >, 12-MAY-76 16:22 NDM ;;;;####;

I sincerely hope you will release this to the KWAC soon. --D

Introduction 1

The new TEMPLATE program prompts users for information, and then creates a file in a given format. 1a

The format is determined by a "template," a branch in some NLS file which is easy to make. The program looks at the template, and builds the new file based on the template, asking the user for information whenever the template says to. It plugs that information into blocks provided in the template. 1b

Using the TEMPLATE Program 2

The TEMPLATE program is a user attachable subsystem. It currently resides as <xprograms, template.subsys,> (with the appropriate .cml). Use the Load Program command in the PROGRAMS subsystem. Then you may Goto Letter. 2a

The TEMPLATE subsystem has two commands: 2b

"Insert" will insert the resulting group after a statement chosen later. 2b1

"Create" will create a new file or new version of an existing file with the name given later. 2b2

You may then choose one of three standard templates, or give a link to your own template file. You must choose one of four keywords: Letter format, Blocked letter format, Memo format, or Format in template branch. 2c

Then the command requests the destination and a level adjustment or name of the file to be created. 2d

Note: if you are creating a file, it is best not to use the same name as (i.e. new version of) the file which holds text you intend to use (e.g. the body which you may have typed in before). When you wish to address that text, it becomes confusing to specify that previous version number. 2d1

At this point, it will scan the template, request information based on prompts in the template (either text or structure). If it wants text, you may type it in or address it. When it requests structure, you must first choose: Statement, Branch, Group, Plex, or Text, then specify the appropriate structure or text. 2e

You may not back up and redo text or structure already specified! 2e1

If you must abort the command (with a Command Delete part way through), note that it either already created the file (Create command) or may have already added statements to the existing destination file (Insert command). 2e2

When it is done, it will Jump you to the destination statement or the new file. 2f

Building Templates 3

Templates are branches in NLS files. They determine the format of the file created by using the TEMPLATE program. Templates are easy to build. 3a

Templates are a series (plex) of NLS statements (any number). In general, each statement in the template file produces one statement in the resulting file. 3b

The top statement in the branch is ignored. 3b1

The structure of the template branch determines the structure of the resulting file. Statements just below the top statement in the template branch will create level one statements in the resulting file. (Many templates will not use structure.) 3b2

Each statement may hold any number of four types of things: 3c

1) Actual text to be put in the resulting file. 3c1

2) A place for text to be gotten from the user. 3c2

3) A place for continuing previous text on the next line. 3c3

4) A place for structure to be gotten from the user. 3c4

Literal Text 3d

Any text in the template file that is not one of the three types of fields (described below) will be copied into the resulting file exactly as it is in the template file. This lets you put in carriage returns, spaces, standard headings, output processor directives, etc. 3d1

Text from User 3e

When you want to leave room for text to be gotten from the user, put in: 3e1

An exclamation point (!). 3e1a

Immediately following, a prompt that will go in the command feedback line. The prompt must begin with an upper-case letter or a digit, followed by up to 50 characters. It is terminated by a colon (:). (Thus you may not have a colon in the prompt.) 3e1b

The prompt may be followed by a space. Then you must have at least one "x". Put in as many x's as you want to make the field as long as you want. I.e. the number of x's determine how much space is left for the user's input. Be sure to leave enough room for any text your users might type. If the user types more than will fit, his input will be truncated. 3e1c

Note: the length of the field includes the exclamation point, the prompt, and the x's. 3e1c1

For example: 3e2

```
!Address of recipient (with zip code): xxxxx
```
3e2a

Continuation of Text 3f

If you want to have a text field spread over more than one line (like an address), you may continue it on the next line: 3f1

Put an "x" directly below the exclamation point which began the field. Follow it with at least two more x's (a total of three), then as many x's as you want the field to be long (on this line). 3f1a

New lines are formed with Carriage Returns in the text of the template statement. Lines must not be longer than 120 characters. Continuations must be in the same statement. For example: 3f2

```
!Address of recipient (with zip code): xxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```
3f2a

If the user types more text than will fit in a field and its continuations, it will fit what it can and discard the rest. 3f3

If the user puts Carriage Returns in the text he types, it will break the line there. But sure to leave enough lines for all purposes. 3f4

Structure from the User 3g

When you want structure from the user (e.g. for the body of the letter): 3g1

Put in two exclamation points, immediately followed by the prompt with its colon. 3g1a

Any previous text in that template statement will be inserted at that point. The structure specified by the user will begin a new statement at the same level. 3g1b

Everything in the template statement after that colon will be ignored. I.e. you should start a new template statement after a structure field. 3g1c

For example: 3g2

!!Body of letter: 3g2a

Special Features 3h

Justification 3h1

Usually the user's text will be put in a field, then the field will be filled out with spaces (up to the length allowed by the x's, so that other fields or text in that line will line up vertically). This will not happen when: 3h1a

1) the field is the last thing in a line, 3h1a1

2) when the field is immediately followed by punctuation (for example: Dear !Salutation: xxxxxxxxxxxxxxx,), or 3h1a2

3) when the last "X" in the series of x's is upper-case (X). 3h1a3

You may have the user's input right justified within the space left for that field by the template by making the next-to-the-last "X" upper-case. (This may be handy for columns of numbers.) This will occur even if the last "X" is upper-case. E.g. 3h1b

!1975 Amount: xxxXx
or
!1975 Amount: xxxXX 3h1b1

Warning: if you want a short field and only leave two x's, making the next-to-the-last "x" upper-case will both right justify AND force the blank line (see below). 3h1b2

Blank Lines 3h2

If a text field (new or continuation) is the only thing on that line and there's no text to put in it, that line will not be put in. 3h2a

If you wish a line to be put in whether or not there is any text from the user to put in it, make the first "x" upper-case (X). 3h2b

Contingent Text 3h3

You may have TEMPLATE put literal text in a line only when the following field is filled. Put the text (up to 1000 characters) in square-brackets [] immediately after the exclamation point that starts a text field; the upper-case letter that starts the prompt should appear immediately after the second square-bracket. E.g. 3h3a

![Attachments:]Attachments (if none, type <CTRL-N>): xxxxxxxx 3h3a1

The literal within the square-brackets (Attachments:) will only be put in if the user types something into that field. 3h3a2

Note: the literal and its square-brackets do not count as part of the length of the text field. 3h3a3

Similarly, if a new text field is immediately followed by a string of text in square-brackets, it will only be put in if that field has been filled. E.g. 3h3b

!Your name: xxxxxxxxxxxxxxX[, Stanford Research Institute] 3h3b1

Note: if the last "x" were not upper-case, the name field would be filled out with spaces before the contingent text is inserted. 3h3b2

Warning: the square-brackets are characters in the template that will not appear in the resulting file; thus they will upset the vertical alignment of the template -- the actual line will be two characters shorter than the template line. 3h3b3

Examples of Templates 4

The Standard Letter Format 4a

```
                                              !Your address or <CTRL-N>:
      xxxxx

      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

                                              !Date:
      xxxxxxxxxxxxxxxxxxxxxxxxx                                          4a1




      !Name of recipient: xxxxxxxxxxxxxxxxxxxxxxxxxx
      !Title of recipient or <CTRL-N>: xxxxxxxxxxxxx
      !Address of recipient: xxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx                     4a2

      ![Dear ]Salutation (e.g. Sir) without colon: xX[:]                 4a3

      ![Re:  ]Re (Subject) or <CTRL-N>: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx     4a4

      !!Body of letter:                                                  4a5
                                     !Closing (Sincerely) no comma:
      xxxxxxxxX[,]




                                     !Your name:
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                                     !Your title or <CTRL-N>:
      xxxxxxxxxxxxxxxx                                                   4a6
```

!Typist (ABC/def) or <CTRL-N>: x
![Enclosures:]Enclosures or <CTRL-N>: xxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx 4a7
![cc]CC or <CTRL-N>:
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx 4a8

The Standard Blocked Letter Format 4b

!Your address or <CTRL-N>: xxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

!Date: xxxxxxxxxxxxxxxxxxxxxxxxx 4b1

!Name of recipient: xxxxxxxxxxxxxxxxxxxxxxxxxx
!Title of recipient or <CTRL-N>: xxxxxxxxxxxxx
!Address of recipient: xxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx 4b2

![Dear]Salutation (e.g. Sir) without colon: xX[:] 4b3

![Re:]Re (Subject) or <CTRL-N>: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx 4b4

!!Body of letter: 4b5

!Closing (Sincerely) no comma: xxxxxxxxX[,]

!Your name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxx
!Your title or <CTRL-N>: xxxxxxxxxxxxxxxx 4b6

!Typist (ABC/def) or <CTRL-N>: x
![Enclosures:]Enclosures or <CTRL-N>: xxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx 4b7

![cc]CC or <CTRL-N>:

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx 4b8

The Standard Memo Format 4c

To: !To:
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx 4c1

![From:]From:
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx 4c2

![Date:]Date:
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx 4c3

![Subject:]Subject:
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxx
4c4

!!Body: 4c5

![ ]Sender's initials or <CTRL-N>:
x 4c6

!Typist's initials or <CTRL-N>: x 4c7

Another Letter Format (Note the use of the "Text" Output Processor
directive.) 4d

!City for dateline: x,
!Date: xxxxxxxxxxX
!Our
file: x
!Reply to: x
!Your
file:x 4d1

!Label (COMPANY PRIVATE) or <CTRL-N>:
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
4d2

!Inside Address:
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxx
4d3

!!Body of letter: 4d4

!Header for successive pages: xxxxx - 2 - !Date:
xxxxxxxxxxX
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

!Title:
xxxxxxxxxxxxxxxxxxxxxxx 4d5

!Typist (ABC/def): xx
!Attachment/Attachments/<CTRL-N> if none:x
![Copy to]Copy to (names) or <CTRL-N>:xxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx 4d6

```
OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFF
OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFF
OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFFICE-1   OFF

TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES
TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES
TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES   TORRES

MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL
MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL
MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL   MTRIAL

THURSDAY, JUNE 17, 1976 16:17:13-PDT   THURSDAY, JUNE 17, 1976 16:17:13-PDT   TH
THURSDAY, JUNE 17, 1976 16:17:13-PDT   THURSDAY, JUNE 17, 1976 16:17:13-PDT   TH
THURSDAY, JUNE 17, 1976 16:17:13-PDT   THURSDAY, JUNE 17, 1976 16:17:13-PDT   TH
```

< MJOURNAL, 35780.NLS;1, >, 16-Jun-76 17:32 XXX ;;;; .HJOURNAL="JCN RLL 16-JUN-76 09:35 35780"; Title: .H1="DRAFT: Policy statement on User Programs classification"; Author(s): James C. Norton, Robert N. Lieberman/JCN RLL; Distribution: /KWAC([ACTION]) SRI-ARC([ACTION]) ; Sub-Collections: NIC SRI-ARC KWAC; Clerk: RLL; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES;

.PEL; .PN=PN-1; .GCR;Comments are expressly solicited. A preliminary classification of programs will be forth coming. No surprises are anticipated in assignment of program STATES. If you have any submissions, contact RLL. Please send comments as soon as possible since we expect to have this draft become final by the end of next week -- 26 June 1976. thanks Rob.

Introduction

Two of ARC's primary objectives in providing the Workshop Utility Service (WUS) to subscribers are:

- to stimulate the development of subscriber-oriented user applications and the new or enhanced system features that are needed to support them and
- to encourage the sharing of application techniques and system additions throughout the user community.

We were very pleased to see the results of increased activity in both the applications and programming areas in the user community that were demonstrated at the April KWAC meeting. In the programming area, there have been many L-10 and CML-based user-programs and subsystems added to the Workshop. Some of these were developed by ARC's Development staff, some by ARC's Applications staff, and some by subscribers themselves. Some are subscriber-specific, while others are much more general. Some are well-debugged, documented, and generally available, while others are in more primitive stages of development.

There are several potentially conflicting needs among the program designers and builders, development sponsors, and the various users of new features. Most of these come from each group's motivation to have users exercise some of the new programs before they are fully debugged, documented, or training is available. Rather than withhold all new programs from the user community until all levels of support are available and they are fully checked out, we are attempting to provide procedures that will permit testing (in that spirit), set user expectations at realistic levels, increase the communication between developers and users, and speed the process of getting new capabilities into the hands of our users.

This document presents the ARC Applications Group's approach to a methodology for the orderly and effective introduction of new user-programs and subsystems, their documentation, and necessary training to the user community, whether the programs are developed by ARC or by subscribers.

Program Classes and States

We will establish two classes of programs -- SUPPORTED and NOT SUPPORTED. Further, SUPPORTED PROGRAMS can be classified according to whether ARC or some other group provides the support.

In addition, there appear to be several states in which programs in each class can be. Some programs will move through all of these states; others will stop at some point due to funding, applicability, or other limitations.

State A: Program is as fully debugged as possible and meets NLS

coding standards; formal documentation and training are available
State B: Program is as fully debugged as possible and meets coding standards; informal documentation is available, but no training is available
State C: Code or design may not be well-debugged; informal documentation is available, but no training is available
State X: Code or design may not be well-debugged; incomplete documentation is available, and no training is available

Details on Classes and States

Each program must meet certain conditions cited above before it is considered in a particular STATE. A discussion of each condition follows below.

ERRORS - fully debugged as possible

The frequency of user encountered bugs should be sufficiently low to not be disruptive during the use of the program. In other words, there should be few known bugs. Also, the program should be 'failsoft' so that users will be unable to 'blow up' their files or any part of the NLS system even in the presence of some newly encountered bug.

DOCUMENTATION - formal

Documentation for the program must include a formal and ARC-Applications approved description of the program. This includes a user-oriented part that clearly and concisely tells people how to use the program and what it does. If this is not adaptable for the online Help database, then a suitable online version of it must be included in the documentation set.
Another part of the documentation must fully describe in detail the concept, logic, and operation of the program.

DOCUMENTATION - informal

The documentation must include a reasonably well written user-oriented part that clearly and concisely tells people how to use the program and what it does.

TRAINING AND CONSULTING - available

The supporting organization will have people trained and knowledgeable in the use and application of the program. This instruction and consulting will be available as part of the generally available training under the applicable contracts.

CODE STANDARDS

Because no program can be considered completely free of errors and because there is a very high probability that any program will be modified sometime in the future, certain coding standards MUST be adhered to for programs.
For L10 and CML languages the templates and standards set by ARC-Development for code produced by them will be used as the standard. Additionally, a strong emphasis will be placed on providing meaningful comments in the code.

CODE DEPENDENCY

In general, user programs and subsystems should not be directly dependent on the particular computer, operating system, or any 'low level code.' Although it is impossible to rigorously test for it, the ultimate dependency on these parts should be isolated in code developed and maintained by the system developers of NLS itself.

CONCEPTUAL COMPLETENESS

This condition is the most difficult to appraise and enforce. Its intent is to cause a reasonable analysis of the problem and careful design of the solution. If the program seems to be sufficiently incomplete as to preclude any ample use by people, it suggests that the problem and/or solution was not carefully thought out. Such programs are not in STATE A. This does not mean that the program must be entirely general or flexible, but encourages the collaborative design of programs so that a wider perspective is considered. This will aid in preventing design decisions that might later severely restrain the development of the program or badly alter the method by which it is used.

Implementation

The ARC-Applications User Programs Coordinator will act as the focal point for submission of new programs and determination of their state.

There will be two directories, called

PROGRAMS - the supported programs in STATE A

XPROGRAMS - all other generally available programs

Only ARC-APP designated people will be allowed to add or delete files to any of these directories. Each addition/deletion must be reflected in the library of all programs.

For all programs in XPROGRAMS, a message at initialization should be printed stating in what STATE the program is. This can be done either by CML code that prints it as noise words or by printing it in the TTY window.

It is also recommended that announcement of new features, correction of bugs, etc. should be briefly mentioned in a message at initialization for programs in XPROGRAMS directory.

A notice at this time is the best way for users to know what is happening to a program that is going through different STATES or is experimental (and thus likely to change rather rapidly at times).

The documentation of programs in directory XPROGRAMS should be made available in the same directory with the name "programname-doc.nls". Documentation of programs in PROGRAMS directory should be available via the HELP command/subsystem.

Library

Introduction

The need to have one place for listing all available programs and associated information seems quite apparent. This library of programs will be maintained by ARC with submissions to be sent to the coordinator of user programs. A small form will be available for submitting programs.

Elements to be listed in the library for each program are the following:

Name of program

Author or organization responsible

Class and state of program (to be filed in by SRI-ARC)

Brief description

Status of documentation and any link to it

Location where source and executable code exists

Contact point

Status of maintenance

Use restrictions

Policy

All program directories at O-1, BBNB, and ISIC (as well as any other supported, even partially, NLS system hosts) must contain the same files.

Only those programs that are intended to be available to all Utility users should be in any of these directories.

Programs that have limited general use or were coded for a particular client may be in any of the above directories so long as they meet the criteria for the program STATE and there is no objection for users in general to use it. Of course, such programs will carry the necessary disclaimers.

Versions of programs undergoing revision by authors should be maintained in other directories.

Coordination

Robert Lieberman (RLL) will act as User-Program Coordinator for ARC Applications. We will make a further assessment of existing user-programs and subsystems classifying them into state and placing them in the appropriate directory.

Future additions to the program library maybe made by contacting RLL and making the necessary code and documentation available for integration into the directories.

< LAWRENCE, NICDIR.NLS;9, >, 9-JUL-76 11:42 TFL ;;;;
.HJOURNAL="JAKE 6-JUL-76 19:50 35934"; Title: .H1="Updates for the ARPANET Directory"; Author(s): Elizabeth J. Feinler/JAKE; Distribution: /TFL([ACTION]) SPS([ACTION]) DLS([ACTION]) RKW([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: JAKE; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES;

ROME AIR DEVELOPMENT CENTER (RADC-TIP) MEMBERSHIP LIST

ENTRIES ARE IN THE FOLLOWING FORMAT:

```
Name                    NIC Ident   Network Mail Address
  U.S. Mail Address                 Phone(s)
  City, State, Zip                  Organization Ident


BARNUM, Alan R.               ARB  BARNUM@OFFICE-1
 Rome Air Development Center       (315) 330-2204
 IS                               RADC-TIP
 Griffiss Air Force Base
 New York 13441

BERGSTROM, Deane F.           DFB  BERGSTROM@OFFICE-1
 Rome Air Development Center       (315) 330-4654
 ISIM                             RADC-TIP
 Griffiss Air Force Base
 New York 13441

BUCCIERO, Thomas J.          TJB2  BUCCIERO@OFFICE-1
 Rome Air Development Center       (315) 330-4254
 IS                               RADC-TIP
 Griffiss Air Force Base
 New York 13441

CALICCHIA, Richard            RC3  CALICCHIA@OFFICE-1
 Rome Air Development Center       (315) 330-7011
 ISIM                              RADC-TIP
 Griffiss Air Force Base
 New York 13441

CARRIER, Roberta J.           RJC  CARRIER@OFFICE-1
 Rome Air Development Center       (315) 330-4254
 ISI                              RADC-TIP
 Griffiss Air Force Base
 New York 13441
```

CAVANO, Joseph P. JPC CAVANO@OFFICE-1
Rome Air Development Center (315) 330-4325
ISIS RADC-TIP
Griffiss Air Force Base
New York 13441

DECONDE, Agatha C. ACD DECONDE@OFFICE-1
Rome Air Development Center (315) 330-7688
IS RADC-TIP
Griffiss Air Force Base
New York 13441

FEMIA, Joe F. JFF FEMIA@OFFICE-1
Rome Air Development Center (315) 330-7834
ISIS RADC-TIP
Griffiss Air Force Base
New York 13441

GRIMES, Robert J. RJG GRIMES@OFFICE-1
Computer Sciences Corporation (703) 533-8877 ext 527
6565 Arlington Blvd. RADC-TIP
Falls Church, Virginia 22046

HILBING, Francis J. FJH HILBING@OFFICE-1
Rome Air Development Center (315) 330-2014
RADC-TIP
ISC
Griffiss Air Force Base
New York 13441

IUORNO, Rocco F. RFI IUORNO@OFFICE-1
Rome Air Development Center (315) 330-4325
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

JOHNSON, Robert W. RWJ BJOHNSON@OFFICE-1
Rome Air Development Center (315) 330-2558
(ISCA) RADC-TIP — ~~Change to AFSC~~
Griffiss Air Force Base ~~CPP = RADC-TIP~~
New York 13441

KENNEDY, Edmund J. EJK KENNEDY@OFFICE-1
Rome Air Development Center (315) 330-3857
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

KESSELMAN, Murray L. MLK2 KESSELMAN@OFFICE-1
Rome Air Development Center (315) 330-3461
RADC-TIP
ISCA
Griffiss Air Force Base, New York 13441

KLAYTON, Alan R. ARK KLAYTON@OFFICE-1
Rome Air Development Center (315) 330-2558
ISCA RADC-TIP
Griffiss Air Force Base
New York 13441

KRUTZ, Robert D. RDK KRUTZ@OFFICE-1
Rome Air Development Center (315) 330-2204
IS RADC-TIP
Griffiss Air Force Base
New York 13441

LAFORGE, Edward F. ELF LAFORGE@OFFICE-1
Rome Air Development Center (315) 330-7834
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

LAMONICA, Frank S. FSL LAMONICA@OFFICE-1
Rome Air Development Center (315) 330-7834
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

LANDES, Michael ML2 LANDES@OFFICE-1
(315) 330-2672
Rome Air Development Center RADC-TIP
ISIM
Griffiss Air Force Base
New York 13441

LAWRENCE, Thomas F. TFL LAWRENCE@OFFICE-1
Rome Air Development Center (315) 330-7746
ISCP RADC-TIP
Griffiss Air Force Base
New York 13441

LIUZZI, Raymond A. RAL LIUZZI@OFFICE-1
Rome Air Development Center (315) 330-7011
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

LOMBARDO, Larry M. LML LOMBARDO@OFFICE-1
Rome Air Development Center (315) 330-2672
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

MCLEAN, John B. JBM2 MCLEAN@OFFICE-1
Rome Air Development Center (315) 330-4875
RADC-TIP
ISIS
Griffiss Air Force Base
New York 13441

MCNAMARA, John L. JLM MCNAMARA@OFFICE-1
Rome Air Development Center (315) 330-3827
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

METZGER, Richard A. RAM4 RMETZGER@OFFICE-1
Rome Air Development Center (315)330-2846
ISCP RADC-TIP
Griffiss Air Force Base
New York 13441

NELSON, Richard RN2 NELSON@OFFICE-1
Rome Air Development Center (315) 330-3851
RADC-TIP
ISIS
Griffiss Air Force Base
New York 13441

PANARA, Roger B. RBP PANARA@OFFICE-1
Rome Air Development Center (315) 330-7834
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

PANAS, Leon R. LRP PANAS@OFFICE-1
Rome Air Development Center (315) 330-7746
ISCP RADC-TIP
Griffiss Air Force Base
New York 13441

PATTERSON, William W. WWP2 PATTERSON@OFFICE-1
Hq., Air Force Systems (301) 981-2364
Command/DLCEC RADC-TIP
Andrews Air Force Base

Maryland 20334

PETELL, Marcelle D. MDP PETELL@OFFICE-1
Rome Air Development Center (315) 330-3827
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

PREVITE, James L. JLP PREVITE@OFFICE-1
Rome Air Development Center (315) 330-3461
ISCA RADC-TIP
Griffiss Air Force Base
New York 13441

REIMANN, Oskar A. OAR REIMANN@OFFICE-1
(315) 330-4728
Rome Air Development Center RADC-TIP
ISCA
Griffiss Air Force Base
New York 13441

RIOLO, Antoinette M. AMR RIOLO@OFFICE-1
Rome Air Development Center (315) 330-7861
ISC RADC-TIP
Griffiss Air Force Base
New York 13441

ROBERTS, Donald F. DFR ROBERTS@OFFICE-1
Rome Air Development Center (315) 330-4325
ISIS RADC-TIP
Griffiss Air Force Base
New York 13441

ROBINSON, Richard A. RAR3 ROBINSON@OFFICE-1
Rome Air Development Center (315) 330-7746
ISCP RADC-TIP
Griffiss Air Force Base
New York 13441

RUPLE, Samuel L. SLR RUPLE@OFFICE-1
Rome Air Development Center (315) 330-3857
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

RZEPKA, William E. WER RZEPKA@OFFICE-1
Rome Air Development Center (315) 330-7834
ISIM RADC-TIP

Griffiss Air Force Base
New York 13441

SLIWA, Frank P. FPS SLIWA@OFFICE-1
Rome Air Development Center (315) 330-3857
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

STINSON, William F. WFS STINSON@OFFICE-1
Rome Air Development Center (315) 330-7009
RADC-TIP
ISF
Griffiss Air Force Base
New York 13441

STONE, Duane L. DLS STONE@OFFICE-1
Rome Air Development Center (315) 330-7746
ISCP RADC-TIP
Griffiss Air Force Base
New York 13441

SUTKOWSKI, Stephen P. SPS SUTKOWSKI@OFFICE-1
(315) 330-7746
Rome Air Development Center RADC-TIP
ISCP
Griffiss Air Force Base
New York 13441

TOMAINI, Frank J. FJT TOMAINI@OFFICE-1
Rome Air Development Center (315) 330-7507
RADC-TIP
ISI
Griffiss Air Force Base
New York 13441

VANALSTINE, Donald DVA VANALSTINE@OFFICE-1
Rome Air Development Center (315) 330-7011
ISIM RADC-TIP
Griffiss Air Force Base
New York 13441

VASSELLI, John J. JJV3 VASSELLI@OFFICE-1
(315) 330-4916
Rome Air Development Center RADC-TIP
ISCP
Griffiss Air Force Base
New York 13441

* WALKER, Robert K. RKW WALKER@RADC-MULTICS
Rome Air Development Center (315) 330-2501
ISF RADC-XPER
Griffiss Air Force Base
New York 13441

WEBER, Roger W. RWW2 WEBER@OFFICE-1
Rome Air Development Center
ISIM (315) 330-2672
Griffiss Air Force Base RADC-TIP
New York 13441

BBNB

WINGFIELD, Mike A. MAW WINGFIELD@OFFICE-1
Rome Air Development Center ~~(315) 330-7746~~
(ISCP) ~~RADC-TIP~~ org = BBNB
Griffiss Air Force Base gup = RADC-TIP
New York 13441 (617)491-1850

DUROSE, Barbara J. BJD2 DUROSE@OFFICE-1
(315) 330-2204
Rome Air Development Center RADC-TIP
IS
Griffiss Air Force Base
New York 13441

GIORDANO, John V. JVG GIORDANO@OFFICE-1
(315) 330-2205
Rome Air Development Center RADC-TIP
ISM
Griffiss Air Force Base
New York 13441

HENDERSON, Helen B. HBH HENDERSON@OFFICE-1
(315) 330-2846
Rome Air Development Center RADC-TIP
ISCP
Griffiss Air Force Base
New York 13441

PETERSON, Norman W. NWP PETERSON@OFFICE-1
TRW Systems Group (213) 535-2045
1 Space Park RADC-TIP
Redondo Beach, California
90278

SEMERARO, Richard C. RCS SEMERARO@OFFICE-1

```
 Rome Air Development Center          (315) 330-7861
 ISC                                  RADC-TIP
 Griffiss Air Force Base
 New York 13441


SMITH, Yale                       YS  SMITH@OFFICE-1
 Rome Air Development Center           (315) 330-2382
 ISFA                                 RADC-TIP
 Griffiss Air Force Base
 New York 13441


TRZEPACZ, Irene G.               IGT  TRZEPACZ@OFFICE-1
 Rome Air Development Center         (315) 330-2382
 ISFA                                 RADC-TIP
 Griffiss Air Force Base
 New York 13441
```

ROME AIR DEVELOPMENT CENTER (RADC-MULTICS) MEMBERSHIP LIST

ENTRIES ARE IN THE FOLLOWING FORMAT:

```
Name                    NIC Ident     Network Mail Address
  U.S. Mail Address                   Phone(s)
  City, State, Zip                    Organization Ident


DOANE, Robert E.                 RED2 Network Address Not Known
 Rome Air Development Center          (315) 330-7628
 ISFF                               RADC-TIP
 Griffiss Air Force Base
 New York 13441


DRAPER, Melville J.               MJD Network Address Not Known
 Air Force Audit Agency               (714) 382-2814
 AFAA/DOY                             RADC-TIP
 Norton Air Force Base
 San Bernadino, California
    92409


JONES, William D.                 WDJ JONES@RADC-MULTICS
 Rome Air Development Center          (315) 330-2319
                                     RADC-MULTICS
 ISFO
 Griffiss Air Force Base
 New York 13441
```

OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF
OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF
OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF

MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD
MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD
MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD

(DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (D
(DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (D
(DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (DSM)35953 (D

FRIDAY, JULY 16, 1976 13:27:48-PDT FRIDAY, JULY 16, 1976 13:27:48-PDT FRIDAY
FRIDAY, JULY 16, 1976 13:27:48-PDT FRIDAY, JULY 16, 1976 13:27:48-PDT FRIDAY
FRIDAY, JULY 16, 1976 13:27:48-PDT FRIDAY, JULY 16, 1976 13:27:48-PDT FRIDAY

QUERY2 Prop

useful references

ORIGIN .H1="Proposal For Research No. ISU 76-185.GCR;Extensions to Query2 Subsystem"; .F="Part One -- Technical Proposal .Split; page .GPN;"; .SN=Off; .SNF=Off; .SNFShow=[1,1]; .RM=72; .BLM=-3; .BRM=68; .YBS=1; .PN=0;

TABLE OF CONTENTS.PBS;.H2=".Split;Table of Contents";.GYBS=6;.Center;.GYES=1;

I INTRODUCTION.PBS;.H2=".Split;Introduction";.Center;.SNF=76;

Stanford Research Institute (SRI) is pleased to submit this proposal for research to the Rome Air Development Center (RADC). The proposed work consists of technical services for the Naval Electronics Laboratory Center (NELC) in the area of programming, as described in Paragraph 4.1.2.4 of RADC Contract No. F30602-76-C-0230.

II BACKGROUND AND OBJECTIVES.H2=".Split;Background and Objectives";.Center;

As part of the Capabilities Assessment Task of its Project J732600, NELC has been investigating the potential of SRI'S oNLine System (NLS) for application to Navy command and control problems. To facilitate this investigation, SRI designed and implemented a demonstration system called Query in accordance with specifications provided by NELC.

Query is an NLS-based data management system that can be used to interrogate and manipulate a relatively small and simple, yet plausible facsimile of a Navy command and control data base. The initial version of Query, Query1, which accesses a static database of 70 ships and supports two commands, required several man-days to implement. The second version of Query, Query2, which manages a dynamic database of 250 ships and planes and 100 ports and supports 25 commands, required several man-months to implement.

In the proposed project, SRI will modify and expand Query2, according to specifications supplied by NELC, to produce a more powerful and realistic version of the system, Query3.

III METHOD OF APPROACH.PBS;.H2=".Split;Method of Approach";.Center;

The work requested of SRI by NELC falls into the following four major categories: improvement of existing Query2 capabilities, addition of minor new capabilities, addition of major new capabilities, and attachment of Query3 to the Datacomputer. This work is to be funded as two separate tasks. Task 1, a 3 man-month effort, consists of Items A and B below: improvement of existing Query2 capabilities and addition of minor new ones. Task 2, a 6 man-month effort, consists of Items C and D below: addition of major new capabilities and attachment of Query3 to the Datacomputer.

A. Improvement of Existing Query2 Capabilities

1. Opcon, Task Force, and Command Discrimination

Although Query2 supports the concepts of opcon, task force, and military command, it fails to discriminate properly between these important entities. To correct this error, task force and command fields will be added to the database (opcon is already present), and minor modifications will be made, as directed by NELC, to some or all of the following commands: ASSIGN and DEASSIGN, ATTACH and DETACH, ESTABLISH and DISESTABLISH, INCLUDE and REMOVE, and SHOW.

2. Current, Maximum, and Cruising Speed Discrimination

Query2's SHOW command permits display of either the platform nearest in hours to a specified reference point, or all the platforms within a specified number of hours of the reference point. The HOW command displays the distance in hours between a specified platform and reference point. For both these commands, Query2 arbitrarily uses maximum speed when computing a platform's distance from the reference point, although both current and maximum platform speeds are maintained in the database.

To enhance the HOW command and the "nearest" and "within" options of the SHOW command, Query will be modified to allow the user to specify whether current speed, maximum speed, or economical cruising speed is to be used for the calculation. To support this new capability, cruising speed will be added to the database. Current and maximum speed are already present.

3. Nearest and Within Options in FIND Command .PBS;

Query2's FIND command permits interrogation of the database for platforms satisfying complex criteria. For example, any of the following may be specified by the user as criteria for selecting platforms for display: .IOvr=4;

(a) Containing a constant, for the composite database field SYNONYMS.

(b) Equal to a constant, for the textual fields NAME, FLAG, POSITION, PLATFORM, CATEGORY, TYPE, CLASS, CO, CARGO TYPE, CARGO QUANTITY, and OPCON.

(,c) Less than, equal to, or greater than a constant, for the numeric fields BEARING, HOURS, RANGE, SPEED, MAXIMUM SPEED, and PERCENT FUEL.

(d) True, for the boolean field DOCTOR.

(e) Known or defined, for any field. .IOvr=0;

Arbitrarily complex, composite criteria can be constructed from the atomic criteria listed above using AND, OR, NOT, and parentheses.

At present, "nearest" and "within" options like those offered by the SHOW command are not offered by FIND and, therefore, will be added. Like Query3's SHOW command (see Item 2 above), these options will include specification by the user of whether current, maximum, or cruising speed is to be used in the calculations.

4. Knowledge of Task Organization Hierarchy

Query2's SHOW and FIND commands allow the user to display the platforms that constitute a specified task force. To identify such platforms, Query2 simply locates those whose task force field matches that specified in the command.

Query3's SHOW and FIND commands will be made to understand the task organization hierarchy--fleet, force, group, unit, element--and will provide command variants for displaying the member platforms of a particular fleet, force within a fleet, group within a force, and so forth.

5. Shorthand Terms for Collections of Related Platform Types .PBS;

Query2's SHOW and FIND commands permit display of platforms whose type field has a specified value. Although this capability is important, it is often desirable to display platforms whose type field has one of several related values. For example, it may be desirable to display all cruisers, whether heavy cruisers (CA), guided missile cruisers (CG), guided missile light cruisers (CLG), and nuclear powered guided missile light cruisers (CLGN).

Query2's FIND command can be used to generate such displays, for example:

```
   FIND ALL CRAFT WITH .LBS=0;
      TYPE=CA OR TYPE=CG OR TYPE=CLG OR TYPE=CLGN .LBS=1;
```

However, this procedure is cumbersome. Therefore, Query's SHOW and FIND commands will be expanded to recognize several "shorthand" platform types such as CRUISER, SUBMARINE, DESTROYER, and FRIGATE. NELC will specify the shorthand platform types to be recognized by Query3, their mneumonics, and the set of platform types to which each corresponds. These shorthand terms then will be built into the software, rather than defined in the database or specified by the user.

6. Support for New Database Fields

Several additional fields must be added to the database to support the Query3 work described in this proposal; these include economical cruising speed, task force name, and command. In addition to the explicitly described new software necessitated by these new fields, the following general changes will be made to Query to support them: .IOvr=4;

(1) The ADD command will be augmented to allow the new fields to be declared when a new platform is added to the data base.

(2) The CHANGE command will be augmented to allow the new fields associated with an existing platform to be changed.

(3) The FIND command will be augmented to support atomic criteria relating to the new fields. .IOvr=0;

These three changes constitute simple extensions to mechanisms already provided by the software.

7. Internal Improvements

The most important characteristic of the Query demonstration system is that its user interface, specified in SRI's Command Meta Language (CML), can be changed rapidly. Query is not a particularly efficient data management system, since it is built upon the NLS file system, which is designed primarily for other applications.

Although database management efficiency is not Query's central focus, inefficiencies detract from the system's operation. Consequently, Query will be modified to search its database more efficiently. In particular, the Query3 database will be inverted on platform names to drastically reduce the time required to locate platforms by name. Also, the mechanism by which fields are extracted from a record will be improved so that a single record will be scanned only once, regardless of the number of fields required or the number of times those fields must be examined. This work will result in no new user features, but it will significantly decrease the execution time of nearly every existing and future Query command.

In addition to the efficiency-related changes described above, several other minor modifications to Query2 will be made. For example, Query will be improved to solicit clarification from the user whenever an ambiguous platform name is specified, rather than aborting the command.

B. Addition of Minor New Capabilities

1. Rendevous Feasibility Testing

Query2 provides various commands for grouping platforms. The PUT command places a specifed craft at a named port. The ESTABLISH command forms a task force by placing a specified list of craft at the position of a named platform. Each of these commands effects the movement of craft by simply altering the position fields of the craft. The feasibility of the rendevous, in terms of the time and fuel required to effect it, is not considered in these commands.

To assist the user in determining the feasibility of a proposed rendevous, a new command (for example, RENDEVOUS) will be implemented. Given the coordinates of a rendevous point, the desired time (or time bounds) of rendevous, and the list of participating craft, this command will determine whether or not the rendevous is feasible and, if so, the minimum time required to effect the rendevous and the fuel status of each craft upon its completion. If the rendevous is feasible, RENDEVOUS also will optionally modify the position, bearing, speed, and percent fuel of the participating craft to effect the rendevous.

2. Determining and Displaying Threats to Platforms

The SHOW and FIND commands will be expanded to permit display of platforms that constitute an existing or potential threat to a named platform.

For each class of craft, NELC will define three types of threats: passive surveillance, active surveillance, and attack. These threat definitions will be of a type definable via the FIND command and will be built into the software, rather than being described in the database or specified by the user.

3. Static Class Characteristics

Naval ships are usually constructed in groups known as classes, and all craft in a given class have identical physical characteristics. Although Query2's database contains the class of each craft, only two of the physical characteristics associated with a class are maintained in the database: maximum speed and range. Therefore, Query will be expanded to deal with additional class characteristics to be specified by NELC.

Class characteristics will be isolated from the other platform characteristics in a special branch of the database. Variants of Query2's ADD, CHANGE, and SHOW commands will be provided for manipulating and displaying these special database records. Maximum speed and range, the values for which are currently maintained for each individual platform, will be moved to the class characteristics branch.

C. Addition of Major New Capabilities

1. Use of Ellipses to Minimize Searches

As already pointed out, Query2's FIND command permits extremely complex search criteria to be applied to the database. If the user can state precisely all the characteristics of the platforms he seeks, he can use FIND to locate them.

Often the user can only formulate his criteria only one step at a time, using feedback from previous steps as an aid in stating further criteria. The user can employ FIND to perform the first phase of his search, from the resulting display determine his next criterion, and then reissue the FIND command, restating his initial criterion and adding the new one to it. Appropriate repetition of this procedure will locate the target platforms (which will be displayed as the result of the final invocation of FIND). However, the process is cumbersome and time consuming, since the platforms located in previous searches must be relocated in each successive step.

To eliminate such inefficiencies, Query's FIND and SHOW commands will be modified so that the system remembers the most recently located set of platforms and allows that set to be used as the initial search domain the next time the command is invoked. Thus, it will be possible to ask questions such as the following:

SHOW ALL SPANISH CRAFT WITHIN 500 NM OF BARCELONA. .LBS=0;
OF THOSE, FIND ALL WITH PERCENT FUEL LEFT GREATER THAN 50.
OF THOSE, SHOW ALL WITHIN 2 HOURS OF ME. .LBS=1;

In this example, Query3 will be able to answer the second and third questions very rapidly, since only those craft satisfying the first condition need be examined.

2. Graphic Display of Platforms

The NLS foundation upon which Query is constructed contains a comprehensive software package for generating and viewing line drawings consisting of mixed text and graphics. This NLS Graphics System will be interfaced to Query3 by means of an option, in both the FIND and SHOW commands, that will allow the user to obtain a graphical representation of the selected platforms in place of the normal textual display. This powerful Query3 facility will be accessible only from terminals equipped with the necessary graphic display devices.

Platforms will be mapped onto the display screen using mercator projection. The portion of the sphere displayed and, therefore, the magnification will be selected automatically by Query3, rather than being set by the user. Each platform will be represented as a small circle, square, diamond, or triangle (symbols already known to the Graphics System) and tagged with its name or (for unknown platforms) class. Rectangular grids marking degrees of latitude and longitude, concentric circles marking distances from a reference point, or other markings will be included in the displays as requested by NELC, subject to time and funding constraints.

Although the graphics facility to be added to Query will provide the user with greatly increased capabilitity, the facility is rudimentary. However, the NLS Graphics System itself is a flexible and powerful tool with which very sophisticated drawings can be created. Thus, increasingly sophisticated Query capabilities can be achieved as additional time and effort are invested, and NELC may wish to consider funding additional work in this area in follow-up efforts.

3. Transmittal of Graphic Output via Sendmail

In addition to the Graphics System described above, the NLS environment contains a comprehensive software package for transmitting messages and documents between users. This NLS

Sendmail System, which already supports messages containing mixed text and graphics, will be interfaced to Query3 by means of a command (for example, SEND) that will transmit the results (textual or graphical) of a previous FIND or SHOW command to one or more NLS users, with accompanying title and introductory text. Recipients of such messages will be required to view them using existing NLS file viewing techniques; no mail reading facilities will be included in Query3.

Although the SEND command will be accessible from any terminal, drawings can be viewed only from terminals equipped with the necessary graphic display devices. Users whose primary mailboxes are either offline (i.e. accessible via U.S. Mail) or on other ARPANET hosts (i.e. accessible via the File Transfer Protocol) will be unable to receive messages containing graphics, since the protocols necessary to transmit graphical information to such users do not exist.

Although the mail facility will provide the Query user with greatly increased capabilitity, it, like the graphics facility already described, is rudimentary. The NLS Sendmail System itself is a flexible and powerful tool with a wealth of options (for example, single or multiple authorship, distribution for action or information, recorded or unrecorded, access controls, keywords, subcollections). Therefore, the addition of dialog support capabilities to a system like Query is an open-ended proposition, and NELC may wish to consider additional funding for follow-up efforts.

D. Attachment of Query3 to the Datacomputer

As already mentioned, Query2 uses the NLS File System as the basis for its own database management system. Although this strategy yields satisfactory results with a database the size of the present one (approximately 300 platforms), NLS's file structure is probably ill-suited to databases of the size required in real command and control applications. Furthermore, it is the NLS Frontend System, allowing high-level description and, therefore, rapid alteration of the user interface, which is the primary focus of the Query experiment. In production systems, database management facilities will be provided by sophisticated systems devoted to that task.

For these reasons, Query will be expanded under this proposal to use as its database either a local NLS file (as is done currently) or a database maintained on the Datacomputer, accessible via the ARPANET by means of network connections and Datalanguage. The existing USE command will be modified to permit the user to switch between the two. In addition, to minimize the number of interhost exchanges required when the Datacomputer is used, Query3 will maintain the results of the last SHOW or FIND command locally in an NLS file; subsequent questions then could be answered without consulting the Datacomputer.

SRI assumes that the Datacomputer will permit floating-point arithmetic and spherical geometric computations to be used as criteria for platform selection. For example, we assume that the Datacomputer will permit direct selection of all platforms whose positions differ from that of a specified reference point by less than n miles. Such capabilities are crucial to the implementation of the "nearest" and "within" options of the SHOW and FIND commands.

SRI also assumes that each record of the Datacomputer database will contain at least the information already maintained in the Query2 database, as well as that required for implementation of the Query3 features described in this document. Although the physical representation of that information is necessarily different, there must be a logical correspondence between elements of the NLS database used by Query2 and elements of the Datacomputer database to be used by Query3.

As an interface to the Datacomputer, SRI expects to use the MACRO-10 subroutine package called DCSUBRS, which is supplied by Datacomputer personnel. This would require the construction of interface software for switching between the L10 and MACRO run-time environments.

As in past Query development efforts, NELC will assume responsibility for writing the documentation and supplying the database additions it deems necessary. We understand, for example, that NELC plans to supply database records for additional aircraft to serve as threats to other craft. This work will be carried out by NELC personnel, and no assistance from SRI will be required.

Two sources of online documentation are potentially available to Query users: the highly structured Help Database provided by NLS' Frontend System and accessible via the Frontend's HELP command and the more loosely structured Help Database provided by Query, which is maintained as a branch of its database and accessible via Query2's PRINT command.

The Frontend Help Database for Query2 has been written by NELC but is not yet available to the Query2 user. The Query2 Help Database has been written by NELC and is available to the user via Query2's PRINT command. Since Query1 has no PRINT command, only the Frontend Help Database is potentially available to Query1 users. No such database currently exists.

NELC will continue development of help databases for Query1, Query2, and Query3 as desired; no assistance will be required from SRI. However, as necessary SRI will expand Query's PRINT command to permit selection of one of several topics for which documentation is to be displayed, as an alternative to displaying the database in its entirety. PRINT will allow the user to choose from among only a small number of subdocuments known a priori to the software; in particular, it will not implement more complicated subdocument structures, such as tree structures, a function already provided by the HELP command.

IV QUALIFICATIONS.PBS;.H2=".Split;Qualifications";.Center;

A. Stanford Research Institute

SRI is an independent, nonprofit corporation performing a broad spectrum of research, development, and other professional services under contract to business, industry, and government. Most of SRI's work is directed toward problem solving rather than research in the abstract. SRI has developed a capability for working with a client organization, understanding its problems, and structuring a responsive program of professional services that provides realistic solutions to those specific problems. Typically, SRI has 800 to 1000 active projects at any one time that produce an annual business volume of over $95 million.

Research operations at SRI are organized into seven divisions representing major disciplines. Overall supervision of research is vested in the Office of Research Operations which reports directly to the Office of the President. Both formal and informal long standing arrangements exist to facilitate interdisciplinary research and development among the divisions and their subgroups. Staff members for this project will come from the Information Science and Engineering Division, which has experience in the design, implementation, and use of interactive systems.

B. Information Science and Engineering Division

The activities of the Information Science and Engineering Division are carried out in three laboratories and four research centers: the Augmentation Research Center, the Information Science Laboratory, the Engineering Sciences Laboratory, the Artificial Intelligence Center, the Electronics and Bioengineering Laboratory, the Telecommunications Sciences Center, and the Transportation Engineering and Control Center. Each laboratory comprises a number of groups with complementary interests and skills.

Staff members for this project will come from the Augmentation Research Center within the Division. This Center maintains a continuing development effort aimed at producing a broad-based computer support system that will effectively augment the human intellect in a highly communication-oriented society.

C. Augmentation Research Center .PBS;

The Augmentation Research Center (ARC) consists of a staff of about 42 people dedicated to the development and technology transfer of computer-based processes that augment people's abilities to handle data in many forms, including text and graphics. These computer aids are aimed at providing fast visual feedback of information, which is maintained in a hierarchically structured form so that it can be displayed to any level of detail and from many different points of view. This organization facilitates both rapid assimilation of concepts and rapid transmission of material in the appropriate level of detail to any desired audience. Communication time and effort are reduced because computers perform the necessary manipulation, reconstruction, and transmission.

ARC participates in all areas (operating system interface, terminal handler, command language interpreter, file system, tool building, networking, and applications) and all stages (design, implementation, maintenance, measurement, and documentation) of system development.

The computer aids developed by ARC are embodied in its oNLine System (NLS), which provides a coherent environment from which various cooperating subsystems can be accessed. To provide this environment, NLS: .IOvr=3;

- Supports interactive devices such as the teleprinter and the two-dimensional CRT display.
- Interprets and passes user commands to the appropriate subsystem.
- Maintains a hierarchically structured file system.
- Provides commands for editing, manipulating, cross-referencing, and cross-copying hierarchically arranged blocks of mixed text and graphics.
- Formats and outputs information obtained from files and other input sources to hardcopy or microfilm.
- Provides a range of tools to aid applications in data management, document production, message handling, software

engineering, and management. .IOvr=0;

V PROJECT ORGANIZATION AND MANAGEMENT.PBS;.H2=".Split;Project Organization and Management";.Center;

SRI believes that the work proposed will be best performed by the smallest number of people having the required technical competence. Therefore, the project team will consist of Mr. James E. White, who designed and implemented Query2, Mr. David S. Maynard, who designed and implemented Query1, and Mr. David C. Smith, who has experience with the design and implementation of other NLS-based data management systems. Other commitments may require alternative staffing.

Mr. White, who will work full-time on the project throughout its course, will also manage the project and be responsible for completion within time and budget constraints and with high technical standards. Mr. Maynard will work part-time on the project and assume responsibility for interfacing Query3 to the Datacomputer. Mr. Smith will help integrate Query3 with other ongoing NLS-based data management projects to eliminate redundancy and strengthen the NLS programming environment that makes systems like Query feasible.

The project will be under the overall supervision of Dr. Richard W. Watson, Assistant Director of ARC. Dr. Douglas C. Engelbart, Director of ARC, will be responsible for review and overall coordination.

Biographies of Messrs. White, Maynard, Smith, and Watson are included in Section VIII.

VI STATEMENT OF WORK.PBS;.H2=".Split;Statement of Work";.Center;

A. Background and Objective

To assist NELC in evaluating the potential of SRI's oNLine System (NLS) for application to Navy command and control problems, SRI has developed two versions--Query1 and Query2--of an NLS-based command and control demonstration system. NELC has requested SRI to make certain modifications and additions to that system to provide a more comprehensive and realistic demonstration system--Query3.

B. Scope

This effort will entail improvement of several existing Query2 capabilities, the addition of several new capabilities not currently available in Query2, and the attachment of Query3 to a remote database management system.

C. Tasks and Technical Requirements

SRI will provide engineering services to effect the following enhancements to existing Query2 capabilities: .IOvr=3;

- Allow the system to discriminate properly between opcon, task force name, and command.
- Allow the user to select from among current, maximum, and economical cruising speeds in the SHOW and HOW commands.
- Add "nearest" and "within" options to the FIND command.
- Provide variants of the SHOW and FIND commands for displaying the platforms in a particular fleet, force, group, unit, or element in accordance with the Naval task organization hierarchy.
- Provide shorthand terms for such command platform types as cruiser, submarine, destroyer, and frigate.
- Support new fields in the ADD, CHANGE, and FIND commands.
- Improve search efficiency and make other minor improvements to Query2. .IOvr=0;

SRI will add the following new capabilities to Query: .IOvr=3;

- A command for testing the feasibility of rendevous of a list of craft.
- Variants of the SHOW and FIND commands for identifying passive surveillance, active surveillance, and attack threats to a platform.
- Commands for adding, changing, and displaying static platform characteristics in a separate branch of the database.
- Internal retention of the results of the previous SHOW or FIND command for use as the search domain of a subsequent command.
- A command for graphically displaying the results of a SHOW or FIND command via the NLS Graphics System.
- A command for transmitting the results of a SHOW or FIND command to another user via the NLS Sendmail System. .IOvr=0;

SRI will interface Query3 to the Datacomputer and provide a command by which the user can switch between local and remote databases.

D. Reporting Requirements

Upon completion of the project, SRI will produce a Final Report describing accomplishments achieved during the contract period. In addition, SRI will prepare and submit to RADC, by means of the NLS Sendmail System, a brief Monthly Status Report describing for the reporting period, as appropriate: .IOvr=3;

- The approach taken and progress made.
- Changes in key personnel associated with the contract.
- Problems or areas of concern requiring government assistance or guidance.
- The number of man-hours and dollars expended both during the contract period and since the beginning of the contract.
- SRI's estimate of the percentage of technical completion as of the period being reported.
- A schedule of projected accomplishments for following months. .IOvr=0;

E. Security Classification

This work statement is unclassified.

VII SELECTED REFERENCES.PBS;.H2=".Split;Selected References";.Center;.IOvr=4;

A. Overview

1. Engelbart, D. C., Watson, R. W., and Norton, J. C., "The Augmented Knowledge Workshop," AFIPS Proceedings, National Computer Conference, Vol. 42, pp. 9-21, 1973.

B. NLS Programming Tools

2. Lehtman, H. G., and Victor, K. E., "Software Engineering," Final Report, Contract RADC-TR-75-304, SRI Project 1868, Stanford Research Institute, Menlo Park, California, December 1975.

3. NLS Programmers' Guide, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, January 8, 1976 (SRI-ARC Catalog Item 34210).

4. L10 Users' Guide, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, September 11, 1973 (SRI-ARC Catalog Item 18969).

5. Irby, C. H., Dornbush, C. F., Victor, K. E., and Wallace, D. C., "A Command Meta Language for NLS," Final Report, Contract RADC-TR-75-304, SRI Project 1868, Stanford Research Institute, Menlo Park, California, December 1975.

6. Dornbush, C. F., NDDT Symbolic Debugger User's Guide,

Augmentation Research Center, Stanford Research Institute, Menlo Park, California, January 7, 1973 (SRI-ARC Catalog Item 13716).

7. Meyer, N. D., "User Program System and Library," Final Report, Contract RADC-TR-75-304, SRI Project 1868, Stanford Research Institute, Menlo Park, California, December 1975.

C. NLS Frontend Environment

8. Watson, R. W., "Issues in the Design of the NLS User Interface," Final Report, Contract RADC-TR-75-304, SRI Project 1868, Stanford Research Institute, Menlo Park, California, December 1975.

9. Irby, C. H., CLI Implementation Memo, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, June 3, 1976 (SRI-ARC Catalog Item 28184).

D. NLS Backend Components

10. Lehtman, H. G., Kelley, K., van Nouhuys, D. H., and Beck, J. M., "Query/Help Software and Data Bases," Final Report, Contract RADC-TR-75-304, SRI Project 1868, Stanford Research Institute, Menlo Park, California, December 1975.

11. Irby, C. H., and Lehtman, H. G., "NLS File System," Final Report, Contract RADC-TR-75-304, SRI Project 1868, Stanford Research Institute, Menlo Park, California, December 1975.

12. Graphics in NLS 8.5, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, March 31, 1976 (SRI-ARC Catalog Item 34907).

13. White, J. E., "Recorded Dialog: the NLS Journal, Identification, and Number Systems," Final Report, Contract F30602-72-C-0313, SRI Project 1868, Stanford Research Intitute, Menlo Park, California, January 30, 1976 (SRI-ARC Catalog Item 22133).

E. Query2 Components

14. White, J. E., QUERY2 L10 Source Code, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, May 3, 1976 (SRI-ARC Catalog Item 28001).

15. White, J. E., QUERY2 Database, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, May 3, 1976 (SRI-ARC Catalog Item 28002).

16. White, J. E., QUERY2 Mini-Database, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, May 3, 1976 (SRI-ARC Catalog Item 28004).

VIII BIOGRAPHIES OF KEY PERSONNEL.PBS;.H2=".Split;Biographies of Key Personnel";.Center;.PES;

ORIGIN .H1="Proposal For Research No. ISU 76-185.GCR;Extensions to Query2 Subsystem"; .F="Part Two -- Contractual Provisions .Split; page .GPN;"; .SN=Off; .SNF=76; .SNFShow=[1,3]; .RM=72; .BLM=-3; .BRM=68; .YBS=1; .PN=0;

TABLE OF CONTENTS.PBS;.H2=".Split;Table of Contents";.GYBS=6;.Center;.GYES=1;

SCHEDULE D -- DOCUMENTATION COSTS .DotSplit; 22
SCHEDULE E -- COMPUTER SUPPORT COSTS .DotSplit; 23

I ESTIMATED TIME AND CHARGES.PBS;.H2=".Split;Estimated Time and Charges";.Center;

The estimated time required to complete this project and report the results is 6 months. SRI could begin work on receipt of a fully executed contract.

Persuant to the provisions of ASPR 16-206.2, attached are a cost estimate and support schedules in lieu of the DD Form 633-4. Also enclosed is a signed form complete except as to the "Detailed Description of Cost Elements."

II REPORTS.H2=".Split;Reports";.Center;

The following reports will be produced:

- Monthly Status Reports (6)
- Final Report

III GOVERNMENT-FURNISHED EQUIPMENT.H2=".Split;Government-Furnished Equipment";.Center;

The work proposed herein will require continued use of Government-Furnished Equipment (GFE).

For 6 months, SRI will need 5% of a PDP-10 computer located at USC-ISI, access to the ARPANET from a PDP-11 computer via IMPs now located at SRI, and terminal access to the ARPANET from TIPs.

IV CONTRACT FORM.H2=".Split;Contract Form";.Center;

Because of the nature of the work proposed, it is requested that any contract resulting from this proposal be awarded as a Supplemental Agreement to Contract F30602-76-C-0230. These services apply to Paragraph 4.1.2.4 of the statement of work for this contract.

V ACCEPTANCE PERIOD.H2=".Split;Acceptance Period";.Center;

This proposal will remain in effect until 30 September 1976. If consideration of the proposal requires a longer period, SRI will be glad to consider a request for an extension of time.

COST ESTIMATE.PBS;.H2=".Split;Cost Estimate";.Center;.SNF=0;

Personnel Costs		
Project Supervision	$	.LBS=0;
Professional		

Total Direct Labor	$ 16,173	
Payroll Burden @ 31%	5,014	

Total Salaries and Wages		21,187
Overhead @ 110% of Salaries and Wages		23,306

Total Personnel Costs		$ 44,493
.LBS=1;		
Direct Costs (See Schedules that follow)		
Travel	$ 332	.LBS=0;
Computer Terminals	765	
Materials and Supplies	200	
Communication	200	
Documentation and Reports	510	

Total Direct Costs		$ 2,007
Total Estimated Cost		$ 46,500
Fixed Fee		3,720

Total Estimated Cost Plus Fixed Fee		$ 50,220

.LBS=1;
SCHEDULE A -- DIRECT LABOR.PBS;.Center;
Direct labor charges are based on the actual salaries for the staff members contemplated for the project work plus a factor of 1.9% of base salary for merit increases during the contract period of performance. The precise factor applied is dependent on the estimated period of performance. Frequency of salary reviews and level of merit increases are in accordance with the Institute's Salary and Wage Payment Policy as published in Topic No. 505 of the SRI Administration Manual and as approved by the Defense Contract Administration Services Region.
SCHEDULE B -- OVERHEAD AND PAYROLL BURDEN.Center;
These rates have been found acceptable by the Department of Defense for billing and bidding purposes for the calendar year of 1976. We request that these rates not be specifically included in the contract, but rather that the contract provide for reimbursement at billing rates acceptable to the Contracting Officer, subject to retroactive adjustment to fixed rates negotiated on the basis of historical cost data. Included in payroll burden are such costs as vacation, holiday and sick leave pay, social security taxes, and contributions to employee benefit plans.
In response to Cost Accounting Standard 410, the Institute will be required on 2 January 1977 to convert to an indirect costing system composed of a separate G&A expense pool and a research operations rate. The specific expense pool that will result from this change is not predictable at this time; however, subsequent to that date indirect costs will be allocated to any contract resulting from this proposal in accordance with such revised accounting system. The standard provides for equitable adjustments should such be necessary because of this changeover.
SCHEDULE C -- MATERIALS AND SERVICES.PBS;.Center;
Travel

2 trips, San Diego, CA, @ $76	=	$ 152	.LBS=0;
4 days subsistence in San Diego, CA @ $30	=	120	
Auto Rental 4 days @ $15	=	60	

Total		$ 332	.LBS=1;

Air fares are based on prices established in the current Official Airline Guide.
Domestic subsistence rates and travel by private auto are established standards based on cost data submitted to and approved by RADC. .GCR;
Communication
This is an estimate of the toll charges for telephone calls during the period of performance. .GCR;
Materials and Supplies
This is an estimate of the cost of materials and supplies based on previous experience with similar projects.
SCHEDULE D -- DOCUMENTATION COSTS.PBS;.Center;
Report costs are estimated on the basis of the number of pages of text and illustrations and the number of copies of reports to be produced, in accordance with the following rates per page:

Editing	$ 2.55	.LBS=0;
Composition	2.50	
Coordination	.74	
Proofreading	.77	
Illustration	21.96	

```
    Press and Binding   .025 per impression .LBS=1;
  The following is a breakdown of the estimated cost of report
  production:
    Printing, 69 pages at $ 6.56 per page =  $  453 .LBS=0;
      (including editing, composition,
       report coordination, proofreading)
    Illustrations, 2 @ $21.96 =                  44
    Press and bindery at, 520 printed pages
    @ $ .025 per printed page =                  13
                                              -----
    Total Estimated Documentation Costs =    $  510 .LBS=1;
SCHEDULE E -- COMPUTER SUPPORT COSTS.PBS;.Center;
  Terminals: .LBS=0;
    NLS workstations (2)
      Display (1) for 6 months
        $ 85/mo x 1 x 6 =         $  510
      Display (1) for 3 months
        $ 85/mo x 1 x 3 =            255
                                --------
  Total                           $  765 .LBS=1;
TITLE PAGE .PBS; .H1Sw=Off; .H2Sw=Off; .FSw=Off; .LM=8; .RM=72; .BM=60;
.SNF=Off; .LBS=0; .YBL=0;
  Proposal for Research .Split; 12 July 1976.PBS;.GYBS=12;
  SRI No. ISU 76-185 .GYES=1;
  EXTENSIONS TO QUERY2 SUBSYSTEM .GYES=3;
  Part One -- Technical Proposal .GYES=1;
  Prepared for: .GYES=1;
  Rome Air Development Center
  Griffiss Air Force Base
  Rome, New York  13441 .GYES=1;
  Attention:  Mr. Duane L. Stone .GYES=4; .LM=40;
  Prepared by: .GYES=1;
  James E. White
  Research Engineer
  Augmentation Research Center .GYES=2; .LM=8;
  Approved: .GYES=4;
  -------------------------------------------
  Douglas C. Engelbart, Director
  Augmentation Research Center .GYES=4;
  -------------------------------------------
  Earle D. Jones, Executive Director
  Information Science and Engineering Division
TITLE PAGE .PBS; .H1Sw=Off; .H2Sw=Off; .FSw=Off; .LM=8; .RM=72; .BM=60;
.SNF=Off; .LBS=0;
  Proposal for Research .Split; 12 July 1976.PBS;.GYBS=12;
  SRI No. ISU 76-185 .GYES=1;
  EXTENSIONS TO QUERY2 SUBSYSTEM .GYES=3;
  Part Two -- Contractual Provisions .GYES=1;
  Prepared for: .GYES=1;
  Rome Air Development Center
  Griffiss Air Force Base
  Rome, New York  13441 .GYES=1;
  Attention:  Mr. Duane L. Stone
```

OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF
OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF
OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF

MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD
MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD
MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD MAYNARD

(DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (D
(DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (D
(DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (DSM)36044 (D

FRIDAY, JULY 16, 1976 13:32:21-PDT FRIDAY, JULY 16, 1976 13:32:21-PDT FRIDAY
FRIDAY, JULY 16, 1976 13:32:21-PDT FRIDAY, JULY 16, 1976 13:32:21-PDT FRIDAY
FRIDAY, JULY 16, 1976 13:32:21-PDT FRIDAY, JULY 16, 1976 13:32:21-PDT FRIDAY

NETWORK OPERATING SYSTEM
STUDY

ask Jim how to rewrite this
for NSF networking submission
How tightly is this tied to
the particular RFP from arpa

GOOD REFERENCES SECTION

ORIGIN .H1="Proposal For Research No. ISU 76-136.GCR;Network Operating System Study"; .F="Part One -- Technical Proposal .Split; page .GPN;"; .SN=Off; .SNF=Off; .SNFShow=[1,3]; .RM=72; .BLM=-3; .BRM=68; .YBS=1; .PN=0;

TABLE OF CONTENTS.PBS;.H2=".Split;Table of Contents";.GYBS=6;.Center;.GYES=1;

I INTRODUCTION.PBS;.H2=".Split;Introduction";.Center;.SNF=76;

Stanford Research Institute (SRI) is pleased to submit this proposal in response to RADC Request for Proposal No. F30602-76-R-0299 for a study to determine the technical problems, alternatives, and approaches for developing a network operating system (NOS) for a heterogeneous distributed computer network.

As requested in RADC's inquiry, the study here proposed by SRI will focus upon: .IOvr=3;

- Defining the functions to be performed by an NOS and identifying their interrelationship.
- Examining and setting priorities for these functions, with special regard to their criticality, benefit, and present implementation status.
- Developing several conceptual NOS design models.
- Estimating the time and other resources required to implement each design approach. .IOvr=0;

The availability at SRI of personnel experienced in operating system design, distributed systems, and network protocols; extensive data processing, library, and laboratory support; a powerful text editing, document production, and dialog support system; and access to the ARPANET and to several PDP-10 Tenex systems on it gives us confidence that the reports emerging from this project will contribute significantly to network development and hasten the day when heterogeneous computer networks will be effective vehicles for resource sharing.

II BACKGROUND.PBS;.H2=".Split;Background";.Center;

By virtue of their ability to bring the computer user into contact with otherwise inaccessible hardware and software resources, heterogeneous computer networks have great potential for increasing user effectiveness. Existing packet-switched networks have demonstrated the technical and economic feasibility of interconnecting large numbers of diverse computer systems. Users of the now international ARPA Network (the ARPANET), for example, routinely access remote time-sharing systems as if they were local users of those systems, transporting files from one machine to another as necessary.

As a result of this technical demonstration, various groups with extensive computer resources within the federal government (for example, AFSC, WWMCCS, and the Army and Navy) are turning to networks as a means of utilizing those resources more efficiently. Virtually all of these networks are distributed, meaning that network traffic is directed by a minicomputer located at each network site, rather than from a single, central location. Some of these networks are heterogeneous, meaning that they interconnect machines of varying size, manufacture, and software.

Although the "hands-on" use of remote resources made possible by existing technology is a natural and highly visible form of resource sharing, several limitations severely reduce its long-term utility:

- It burdens the user with the task of locating resources. Even if his problem is well defined, the user must often appeal to others for assistance in locating the hardware and software resources required for its solution. These preliminaries, which can be facilitated by a network information center, if one exists, may require searches of hardcopy documentation, phone calls to candidate installations, and so forth.
- It requires the user to attend constantly to network mechanics. To gain initial access to a resource once it has been located, the user must interact administratively with the remote installation to obtain a user name, password, and account number. To actually use the resource or transport a file to or from it, he must invoke a special network subsystem and supply the login parameters assigned to him. At the end of each month, the user receives a separate bill from each installation.
- It forces upon the user all the trappings of each remote system whose resources he desires to use, requiring him to leave the familiar working environment provided by his local system and enter an alien one with its own peculiar system structure (login, logout, and subsystem entry and exit procedures), file system (file naming conventions, access controls, and directory structure), and command language discipline (including command recognition and completion conventions and editing characters). Even the special subsystems he must employ to move himself and his files from one machine to another are a part of this changing milieu.
- It fails to provide a basis for bootstrapping new composite resources from existing ones. Because the network access discipline imposed by each resource is a human-engineered command language, rather than a machine-oriented communication protocol, it is virtually impossible for one resource to programmatically draw upon the services of others. Doing so would require that the program deal successfully with complicated echoing and feedback characteristics, unstructured and sometimes unsolicited system responses, and so forth.

Present technology thus prevents full realization of the resource sharing potential of heterogeneous computer networks. Of the casual user it demands an often prohibitive startup cost, requiring him to locate on his own the resources he needs and then to acquire detailed knowledge of the systems in which they happen to be embedded. It also fails to provide the more frequent user with the kind of organized and consistent workshop he requires to work effectively, and prohibits construction of the more powerful, composite resources that would allow him to make more effective use of his time.

Limitations like those above can be removed by interposing a software agent between the user and the network's resources. Early computing experience quickly demonstrated the inefficiency of requiring users to interact directly with the raw physical resources of even a single machine. This experience led to the introduction of operating systems, which perform many of the mundane and often complicated tasks associated with computer use, and so free the user to attend to higher-level tasks more directly related to his goals. Similarly, recent network experience even more clearly demonstrates the need for a network operating system (NOS) that will assist the user in using the composite machine to which the network provides him access. Unlike the field of packet-switching networks, in which considerable research has already been performed, the area of heterogeneous NOS design has received relatively little attention to date. ARPANET host-level protocol design efforts were early attempts to provide isolated elements of an NOS. The National Software Works, under joint construction by the Air Force and the Defense Advanced Research Projects Agency, is tackling the same kinds of problems in a more coherent fashion.

NOS development is crucial to the growth of resource sharing within existing and future computer networks; it also has potential for assisting in the solution of other important problems, such as program transportability. Because of its importance, NOS development warrants considerable government support. A previous study funded by RADC and entitled "Distributed Computation Study" examined some of the problems of NOS design, outlined some important characteristics of an NOS, and recommended a design approach. Because of the magnitude of the problem, further study is required. In particular, the specific services which an NOS will provide to applications programs, as well as the internal organization of the distributed system required to deliver those services, require further investigation and description in greater detail. Other important aspects of NOS design, such as security, debugging, and measurement facilities, which were not dealt with in the initial study, must also be examined.

III OBJECTIVES.PBS;.H2=".Split;Objectives";.Center;

In the proposed study, SRI will investigate and report on the technical problems in and alternative approaches for developing a network operating system for a heterogeneous, distributed computer network. The study will incorporate, build upon, refine, modify as necessary, and extend the work of a previous study entitled "Distributed Computation Study," performed under contract F30602-75-C-0222.

The proposed study will define NOS functions and identify their interrelationship, examine and set priorities for the functions in terms of their criticality, relative benefit, and present implementation status; develop one or more conceptual models for NOS design; give scenarios for the use of each; and estimate the amount of time and resources required by each approach. SRI will also provide a detailed comparison between the model(s) proposed in the study and that implemented by the National Software Works, giving detailed explanations of whatever similarities and differences are found to exist.

As requested in the inquiry, SRI will investigate at least the following aspects of an NOS:

- Interprocess communication .LBS=0;
- Interuser communication
- Data migration
- Job migration, control, and synchronization
- Network control
- System restart and recovery

- System reliability and availability
- Distributed data base access
- Automatic job assignment
- Dynamic job relocation .LBS=1;

SRI will also investigate other important areas of NOS design, including security, measurement, debugging, and the carrying out of command language interpretation local to the user. It has been SRI's experience that considering such areas from the outset is crucial to the success of the design process, and that existing systems are not easily retrofitted with such facilities.

IV METHOD OF APPROACH.PBS;.H2=".Split;Method of Approach";.Center;

To meet the objectives stated above, SRI has partitioned the proposed study into three major subtasks: specification of the functional interface between a network operating system and the applications programs running atop it, identification of ways in which the NOS might be organized internally, and determination of possible strategies for NOS implementation.

A. Specification of Functional Interface

1. Motivation

Like an operating system for a single machine, an NOS must, as its primary task, provide a run-time environment for the applications programs that will execute within its scope. The first and most important way of characterizing an NOS, therefore, is to describe the set of functions or primitives it will make available to the applications programmer. In recognition of the above and as a significant part of its work under this contract, SRI will describe the set of primitives that an NOS should provide.

The "Distributed Computation Study" defined major areas--job control, interprocess communication, terminal support, and so forth--in which primitives must be provided; in this study, SRI will much more fully detail the primitives required in those and other areas.

2. Investigation of Representative Operating Systems

A first-order understanding of the primitives that an NOS must provide is most readily obtained from an examination of the functions performed by conventional, single-machine operating systems. Some features of conventional operating systems will be meaningless in an NOS environment, and, conversely, some NOS features will find no counterpart in conventional operating systems. Nevertheless, we expect that a significant number of parallels will exist and that a thorough review of this more familiar software entity will yield many NOS features by logical extension. As a side benefit, this approach serves also to remind us that an NOS must, for pragmatic reasons, be fabricated from existing operating systems; therefore, it provides an initial feasibility check on our design. .PES;

SRI will examine, compare, and report on the primitive sets provided by several representative operating systems, including at least Tenex, Multics, and OS/VS2-MVS. This investigation will produce a comprehensive list of the broad categories of primitives that a conventional operating system and hence an NOS must provide, as well as an understanding of the particular primitives that must be provided in each category. As part of this investigation, SRI will also examine existing NOS work, such as RSEXEC and the NSW.

3. Development of a Model For Each Functional Area

From its analysis of the functions and capabilities of some representative operating systems, SRI will synthesize a functional description of the NOS. The task here is analogous to that of fashioning an operating system interface (OSI) for a program, written in a high-level language, that must run on two or more machines. The OSI in effect provides a virtual operating system upon which the program can rely, regardless of the machine on which it actually executes. This strategy, by confining operating system dependencies to the OSI, enables the program running above it to be transferred from one machine to another with a simple recompilation.

To functionally describe an NOS, SRI will first develop a model for each area of user/system interaction. In the areas of file manipulation and terminal handling, for example, SRI will define virtual file structures and terminals, respectively. Although synthesized from conventional operating systems, these models will recognize the multimachine nature of the NOS environment, and methods of addressing files and terminals (for example) within the composite machine will be proposed.

4. Selection of a Set of Primitives

Once the underlying models have been defined, SRI will propose a set of primitives to implement them. This task will entail careful generalization of the operating system primitives already studied to eliminate obvious machine dependencies yet ensure the implementability of the proposed primitives on at least those machines examined. The task will produce logical calling sequences for each of the candidate NOS primitives.

B. Identification of Internal Organization(s) .PBS;

1. Motivation

The subtask described above treats the NOS as a black box. The principal output of the subtask will be a fairly detailed definition of the interface between the NOS and the user programs that will employ its services. This first subtask is an important one because it effectively defines what would be expected of an NOS implementer.

The second subtask proposed by SRI entails an examination of the contents of the black box. The principal output of this subtask will be one or more NOS implementation models, each describing a particular decomposition of the system into functional modules, organization of the necessary support data bases, and so forth. This subtask, too, is important because it will greatly assist the NOS implementer is carrying out his task.

From past experience in the development of distributed systems, we know that design problems often arise that require tradeoffs of various kinds for their solution. Although such problems can arise in any stage of a study like this one, this second subtask is especially prone to them. In its reports, SRI will identify any tradeoff situations it encounters and motivate the course selected.

2. Decomposition of System Into Modules

Although the NOS will probably present itself to the applications programmer as a single coherent entity, local to his own machine, in reality a much more complicated software configuration will be required. Because the NOS must be accessible (probably via a system call) to the application programmer of every machine, at least some code must reside there. Nevertheless, the NOS is necessarily a distributed system, and most of the code may actually reside on other hosts, on minicomputers attached to them, or both.

SRI will explore and report on several possible NOS configurations, identifying for each the functions of the required software modules and the manner in which the internal run-time data bases must be distributed among them. Resulting intermodule communication patterns will be described, and the necessary communication protocols identified and characterized.

3. Analysis of Reliability and Restart Characteristics

Important characteristics of any distributed system are its reliability, its overall sensitivity to failure of individual components, and its ability to recover from such failures with minimal loss of information. SRI will, therefore, give considerable weight to such factors in its configuration selections--by, for example, providing for distributed, rather than centralized control, wherever possible, and for redundant maintenance of crucial state information--and will analyze the proposed configurations in terms of these characteristics.

4. Presentation of Scenarios of System Use

In an effort to give a more complete description of the system configurations proposed, SRI will, as a final element of this second subtask, provide scenarios for the NOS execution of several operations that the applications programmer might typically ask it to perform. Separate scenarios will be provided for each proposed configuration.

C. Determination of Implementation Strategies

1. Motivation

The first two of the three subtasks that SRI proposes to undertake have as their aim the definition, in terms of both external behavior and internal organization, of a comprehensive and coherent, prototype NOS. The third and final subtask completes the study by suggesting strategies for realizing that objective.

2. Setting Priorities for Functions

Conventional operating systems are complex; packet switched subnetworks are complex; it is reasonable to expect that a full-blown NOS will be complex. Because a fully functional NOS will require many man-years for its detailed design and construction, it is important to provide for staged implementation. SRI will therefore, as requested in RADC's inquiry, suggest an implementation order for the NOS functions proposed, comparing the various functions in terms of their cost, benefit, and importance to continued evolution of the system.

3. Determination of Present Implementation Status .PBS;

Although not usually described as such, many existing ARPANET protocols clearly represent early attempts to provide isolated elements of an NOS. The Telnet and Graphics Protocols, for example, both built upon the concept of a network virtual terminal, perform the same kinds of functions as the terminal handler of an NOS will perform, for teletype-like and display terminals, respectively. Similarly, the File Transfer Protocol and Resource Sharing Executive represent attempts at construction of a network-wide file system. And the Remote Job Entry Protocol tackled some of the problems faced by an NOS in the area of job control.

To help RADC exploit the many hundreds of man years that have thus already been invested in NOS design and implementation, SRI will survey that work and relate it to the primitives and internal structures proposed in this study. Special emphasis will be placed on recording problem areas (and solutions, where they exist) that have already been identified by this early research, and to the possible temporary use of existing protocols as a means of speeding NOS implementation.

4. Comparison of Proposed Model(s) with National Software Works

The National Software Works (NSW) now being constructed by the federal government represents, in many ways, the most ambitious attempt to date at designing and constructing an NOS for a heterogeneous computer network. SRI will therefore provide a detailed comparison between the model(s) proposed in this study and that implemented by the NSW, pointing out NOS functions that may be absent in NSW, and giving detailed explanations of whatever similarities and differences may be found to exist.

At the completion of the study, SRI will produce a four-part Final Report. The first part of the report, an Operating System Survey, will describe and compare the primitive sets studied as part of Subtask A. The second part, a NOS Interface Specification, will provide a fairly detailed functional description of the NOS. It will describe the models developed for each area of user/system interaction and document the set of primitives proposed for implementing them. The third section of the report, a NOS Internal Organization Proposal, will document the results of Subtask B. It will discuss the alternative internal configurations developed, describing for each configuration the function of each module, the required internal data bases, intermodule communication patterns, and necessary communication. The fourth and final part of the report will be an NOS Implementation Strategy and Status Report, which will describe the results of Subtask C and report on the present implementation status of various NOS components and suggest priorities according to which they should be refined and others developed.

V QUALIFICATIONS.PBS;.H2=".Split;Qualifications";.Center;

A. Stanford Research Institute

Stanford Research Institute (SRI) is an independent, nonprofit corporation performing a broad spectrum of research, development, and other professional services under contract to business, industry, and government. Most of SRI's work is directed toward problem solving rather than research in the abstract. SRI has developed a capability for working with a client organization, understanding its problems, and structuring a responsive program of professional services that provides realistic solutions to those specific problems. Typically, SRI has 800 to 1000 active projects at any one time that produce an annual business volume of over $95 million.

Facilities at SRI's main offices in Menlo Park, California include extensive data processing, library, and laboratory support. SRI's comprehensive technical libraries are well supplied with literature in the fields of operating system design, distributed computation, and computer networks. The libraries have trained personnel to provide support for research activities through literature searches and the acquisition and distribution of technical documents.

SRI has many in-house computer systems. These include a CDC 6400, a B6700 dual processor system, and a PDP-10. Job processing can be accomplished in batch mode or online in a time-sharing mode. Besides its own facilities, SRI has ready access to numerous other nearby computer facilities, including various IBM, CDC, and Univac systems.

Research operations at SRI are organized into eight divisions representing major disciplines. Overall supervision of research is vested in the Office of Research Operations which reports directly to the Office of the President. Both formal and informal long standing arrangements exist to facilitate interdisciplinary research and development among the divisions and their subgroups.

Staff members for this project will come from the Information Science and Engineering Division, which has experience in the design, implementation, and use of operating systems, distributed systems, and computer networks.

B. Information Science and Engineering Division .PBS;

The activities of the Information Science and Engineering Division are carried out in three laboratories and four research centers: the Augmentation Research Center, the Information Science Laboratory, the Engineering Sciences Laboratory, the Artificial Intelligence Center, the Electronics and Bioengineering Laboratory, the Telecommunications Sciences Center, and the Transportation Engineering and Control Center. Each of the laboratories is composed of a number of groups with complementary interests and skills.

Staff members for this project will come primarily from the Augmentation Research Center, with support from the Computer Science Group within the Information Science Laboratory and from the Telecommunications Sciences Center. The Augmentation Research Center maintains a continuing development effort aimed at producing a broad-based computer support system that will effectively augment the human intellect in a highly communication-oriented society. The Information Science Laboratory is predominantly oriented toward research, development, and implementation of techniques of broad applicability, focusing on the design and development of computers, computer programs, and computer systems. The Telecommunications Sciences Center uses a system engineering approach to solve problems created both by technological advances in communication theory and practice and by economic and social constraints placed on communication technology.

C. Augmentation Research Center

General Research Goals

The Augmentation Research Center (ARC) consists of a staff of about 42 people dedicated to developing computer-based processes that augment people's abilities to handle textual and pictorial information. These computer aids are aimed at providing fast visual feedback of information, which is maintained in a hierarchically structured form so that it can be displayed to any level of detail and from many different points of view. This organization facilitates both rapid assimilation of concepts and rapid transmission of material to any desired audience in the appropriate level of detail. Communication time and effort are reduced because computers perform the necessary manipulation, reconstruction, and transmission.

ARC participates in all areas (operating system interface, terminal handler, command language interpreter, and file system) and all stages (design, implementation, maintenance, measurement, and documentation) of system development.

The computer aids developed by ARC are embodied in its oNLine System (NLS), which provides a coherent environment from which various cooperating subsystems can be accessed. To provide this environment, NLS: .IOvr=3;

- Supports interactive devices such as the teleprinter and the two-dimensional CRT display.
- Interprets and passes user commands to the appropriate subsystem.
- Maintains a hierarchically structured file system.
- Provides commands for editing, manipulating, cross-referencing, and cross-copying hierarchically arranged blocks of mixed text and graphics.
- Formats and outputs information obtained from files and other input sources to hardcopy or microfilm. .IOvr=0;

Specific Qualifications for This Project

In the context of developing and delivering services like those described above, ARC has gradually expanded its user base from a small group of in-house users to what is now a large, nationwide community. In the process, ARC has gained over 6 years of experience in the design, implementation, maintenance, and daily use of network-based and distributed systems. This experience includes work done in fulfillment of the following contracts:

Network Information Center
and Augmented Knowledge Workshop Development
Contract No. F30602-70-C-0219 and No. F30602-72-C-0313
Defense Advanced Research Projects Agency
February 1970 through June 1974
Dr. Douglas C. Engelbart, Principal Investigator

Under this contract, ARC helped design the following protocols for the ARPA Computer Network (the ARPANET): The Host-Host Protocol, which provides the basis for a general interprocess communication facility; the Telnet and Graphics Protocols, which attach teletype-like and display terminals, respectively, to remote time-sharing systems; the File Transfer Protocol (FTP), by which files are transported from one computer to another; the Remote Job Entry Protocol (RJE), which provides access to remote batch processing facilities; and the Mail Protocol (MAIL), which enables distant users to correspond via the Network.

ARC also interfaced NLS' Journal subsystem to FTP and MAIL, converted the Journal to a dual-site system, and designed the network protocol necessary to implement a general, multisite Journal.

National Software Works
Contract No. F30602-75-C-0156 and No. F30602-75-C-0320
Defense Advanced Research Projects Agency
July 1974 through April 1976

Dr. Richard W. Watson, Principal Investigator

Under this contract, ARC is implementing the Front-End system by which remote tools will be accessed within the National Software Works (NSW), installing NLS as the first NSW tool, and designing and implementing a distributed-system debugger for use in the NSW environment. ARC also designed and implemented a Distributed Programming System (DPS), which provides a high-level programming environment for the construction of distributed systems.

Specification of TCP and THP Protocols for AUTODIN II
Contract No. DCA100-76-C-0034
Defense Communication Agency
February 1976 through June 1976
Dr. Jonathan B. Postel, Principal Investigator

Under this contract ARC is specifying the Transmission Control and Terminal-to-Host Protocols that will underlie AUTODIN II.

ARC has been heavily involved in the development of the ARPANET since 1969, when one of the its initial four nodes was installed at SRI.

As indicated above, ARC has played a principal role in the design of network-wide protocols which are principal elements of a network operating system. ARC staff members have also implemented versions of each protocol and, in particular, were among the first implementers of Network Control Programs (NCPs).

ARC staff members, who spend a large portion of their working day online, are heavy users of the ARPANET; since March 1975, they have relied exclusively on remote systems for their daily computer support. Of necessity, therefore, ARC has gained experience in measuring and improving the performance of distributed systems.

For the last 13 years, ARC has been working methodically toward the development of an Augmented Knowledge Workshop (of which NLS is a prototype), a network-based, distributed system which will make a large collection of geographically separated software and hardware tools available, within a coherent framework, to knowledge workers of all kinds. ARC is, therefore, committed to the development of sophisticated distributed systems, and an effective and widely implemented network operating system is crucial to the realization of its goals.

D. Computer Science Group

General Research Goals

The general research goals of the Computer Science Group are (1) more systematic methods of designing, verifying, and evaluating systems and programs; and (2) new computer architecture and computation techniques. The group performs research ranging from theoretical studies (in areas like the theory of programming and the theory of computation) to the logical specification of prototype hardware and software (for example, languages, compilers, operating systems, and computers). Although most of its work is directly sponsored, the group occasionally contributes to other SRI projects where knowledge of future computer developments is needed.

The objectives of the group's current research in programming are: .IOvr=3;

- Systematic means for creating correct programs.
- Techniques for designing secure and reliable computer systems.
- Techniques for designing knowledge-based decision-support systems based on large data files. .IOvr=0;

These objectives are closely related to each other and to the research goals in computer architecture. They form a basis for an evolving set of formal programming techniques, including software specification languages, programming languages, system structures, analytic methods, and heuristics for extremely complex systems.

Specific Qualifications for This Project

Over the past 4 years, the Computer Science Group has engaged in various projects related to the research proposed here. Most of these projects, some of which are outlined below, have resulted in important publications:

Secure Operating System Design Study
Contract No. DAAB03-73-C-1454 and No. DAAB03-75-C-0399
U.S. Government
April 1973 through June 1976
Dr. Peter G. Neumann, Principal Investigator

Under this contract, the Computer Science Group is designing and proving the security kernel of an operating system. Hierarchical design is being used to simplify the proof.

Verification of Hierarchically Structured Programs
Grant No. DCR74-18661
National Science Foundation
March 1975 through March 1977
Drs. Karl N. Levitt and Lawrence Robinson,
Principal Investigators

Under this contract, the Group is developing a hierarchical methodology for structuring design, implementation, and proof, that is will suited to the semiautomatic proof of large systems.

Study of a Methodology for Reliable Software
Contract No. 5-35932
June 1975 through August 1976
Drs. Karl N. Levitt and Peter G. Neumann,
Principal Investigators

Under this contract, the Group is applying the above methodology to general software production, using as an example a secure data management system.

A Methodology for Modular Operating Systems
Contract No. N00123-76-C-0195
January 1976 through December 1977
Dr. John H. Wensley, Principal Investigator

Under this contract, the Group is developing methodology to support families of systems, and to stress and test the methodology with a detailed example. Message processing systems are being used as an example.

E. Telecommunications Sciences Center

General Research Goals

The Telecommunications Sciences Center consists of teams of experts possessing the diverse talents required to solve complex telecommunications problems of both commercial and government organizations. The Center's spectrum of professional expertise includes laboratory, field, and equipment specialists; design engineers; technology specialists, communication and information theorists; media and antenna experts; traffic, network, and switching engineers; and senior analysts in communication requirements, operations, and systems.

The Center uses a system engineering approach to solve problems created both by technolgical advances in communication theory and practice and by economic and social constraints placed on communication technology.

The professional competence and project experience of the staff include the following areas: .IOvr=3;

- Communication systems design and evaluation.
- Communication network analysis and planning.
- Communication requirements analysis.
- Mathematical modeling of communication channels, equipment, systems, and networks.
- Packet switching. .IOvr=0;

Specific Qualifications for This Project

The Center is heavily involved with and Technical Coordinator of ARPA's Packet Radio Project, which is developing packet-switching technology to support inter-machine communication via satellite. One of the Center's staff members also has extensive experience in operating system development and network protocols, having designing and implemented the ELF operating system for the PDP-11 computer, a network-based system providing terminal access to remote resources.

VI PROJECT ORGANIZATION AND MANAGEMENT.PBS;.H2=".Split;Project Organization and Management";.Center;

SRI believes that the proposed study is best performed by the smallest number of people that include all the technical competence required. SRI has, therefore, assembled the following basic team for the study: Mr. James E. White, Dr. Richard J. Feiertag, Dr. Peter G. Neumann, and Dr. David L. Retz. Dr. Jonathan B. Postel will also be available for informal consultation. Other commitments may require alternative staffing.

Mr. White, who will work nearly full-time on the project throughout its course and perform most of the work, will also manage the study and be responsible for coordination and completion on time, within budget, and with high technical standards. Drs. Neumann and Feiertag will provide assistance in developing the overall NOS design strategy and in dealing with NOS security issues. Dr. Retz will be primarily concerned with specification of the NOS file system. Dr. Postel will provide assistance in the areas of NOS internal organization and interprocess communication.

The study will be under the overall supervision of Dr. Richard W. Watson, Assistant Director of ARC. Dr. Douglas C. Engelbart, Director of ARC, will be responsible for review and overall coordination.

VII STATEMENT OF WORK.PBS;.H2=".Split;Statement of Work";.Center;

A. Background

Present technology is insufficient to fully realize the resource sharing potential of heterogeneous computer networks. Rather than require the user to interact directly with each computer system, as at present, a software agent or network operating system (NOS) can be interposed between the user and the network's resources to assist him in using the composite machine to which the network provides access. The development of such an agent would assist various groups within the federal government in making more efficient use of their computer resources.

B. Objective

The objective of this effort is to investigate and report on the technical problems in and alternative approaches for developing a network operating system (NOS) for a heterogeneous, distributed computer network.

C. Scope

This effort includes: .IOvr=3;

- Defining NOS functions and identifying their interrelationship.
- Examining and setting priorities for the development of those functions.
- Investigating the present implementation status of those functions.
- Developing model(s) of an NOS's internal organization.
- Preparing scenarios of NOS use for each model.
- Estimating the resources required for implementation of each model.
- Comparing and contrasting the proposed model(s) with the National Software Works. .IOvr=0;

D. Tasks and Technical Requirements .PBS;

The contractor shall provide engineering services to accomplish the following.

The contractor shall incorporate, build upon, refine, modify (if supported by sufficient rationale), and extend the work completed under contract F30602-75-C-0222 entitled "Distributed Computation Study".

The contractor shall determine critical system functions, system components, and their interrelationship necessary to develop a distributed network operating system to support distributed computation in a heterogeneous computer network. There may exist sets of NOS functions, each set being appropriate to the environment in which the NOS is to operate.

The contractor shall develop conceptual models for NOS design. Each model shall describe the manner in which the user interacts with the NOS and the particular application area addressed. Models may vary depending on mission objective or the application area the NOS is intended to address.

The contractor shall examine the defined functions and determine which are most critical to the NOS concept; which, if implemented, will produce the greatest benefit; and which have been or are already being implemented.

The contractor shall provide several approaches to NOS design and indicate advantages and disadvantages of each approach. The contractor shall also provide estimates of the time and resources necessary to implement each design approach.

The contractor shall investigate at least the following areas relevant to NOS design:

- Interprocess communication .LBS=0;
- Interuser communication
- Data migration
- Job migration, control, and synchronization

- Network control
- System restart and recovery
- System reliability and availability
- Distributed data base access
- Automatic job assignment
- Dynamic job relocation .LBS=1;

The contractor shall compare the NOS model(s) developed under this effort with the model of the existing National Software Works. This comparison shall at least include similarities and differences, and contain a detailed explanation of why those similarities and differences exist.

E. Reporting Requirements

Upon completion of the study, SRI will produce a Final Report with the following four parts:

- Operating System Survey
- NOS Interface Specification
- NOS Internal Organization Proposal
- NOS Implementation Strategy and Status Report

In addition, SRI will prepare and submit to RADC, by means of the NLS Sendmail system, a brief Monthly Status Report describing for the reporting period, as appropriate: .IOvr=3;

- The approach taken and progress made.
- Changes in key personnel associated with the contract.
- Noteworthy trips, meetings, or special conferences held in connection with the contract.
- Problems or areas of concern on which government assistance or guidance is required.
- The number of man-hours and dollars expended both during the contract period and since the beginning of the contract.
- The contractor's estimate of the percentage of technical completion as of the period being reported.
- A schedule of projected accomplishments for following months. .IOvr=0;

F. Security Classification

This work statement is unclassified.

VIII REFERENCES.PBS;.H2=".Split;References";.Center;.IOvr=4;

A. Host-Host Protocol and its Implementation

1. Postel, J. B., "Survey of Network Control Programs in the ARPA Computer Network," Technical Report MTR-6722 revision 1, The MITRE Corporation, October 1974.

2. White, J. E., An NCP for the ARPA Network, Computer Research Laboratory, University of California, Santa Barbara, California, December 1970 (SRI-ARC Catalog Item 5480).

3. White, J. E., Dynamic Extension of OS 360/ for a Network Environment, Computer Research Laboratory, University of California, Santa Barbara, California.

B. Function-Oriented Protocols

4. Anderson, R. H., Cerf, V. G., Harslem, E. F., Heafner, J. F., Madden, J. M., Metcalfe, R. M., Shoshani, A., White, J. E., Wood, D. C., "The Data Reconfiguration Service -- An Experiment in Adaptable, Process/Process Communication," Presented at the Second Symposium on Problems in the Optimization of Data Communication Systems, sponsored by the SIGCOMM of ACM and the IEEE Computer Society, Palo Alto, California, October 1971.

5. Cerf, V. G., Harslem, E. F., Heafner, J. F., Metcalfe, R. M., White, J. E., "An Experimental Service for Adaptable Data

Reconfiguration," IEEE Transactions on Communication Technology, June 1972.

6. Crocker, S. D., Heafner, J. F., Metcalfe, R. M., Postel, J. B., "Function-oriented Protocols for the ARPA Computer Network," AFIPS Proceedings, Spring Joint Computer Conference, Vol. 40, pp. 271-279, 1972.

7. Postel, J. B., A Possible Protocol Plateau, ARPA Network Working Group Request for Comments 48, Stanford Research Institute, Menlo Park, California, April 1970 (SRI-ARC Catalog Item 4750).

8. Postel, J. B., Official Initial Connection Protocol, Network Measurement Center, University of California, Los Angeles, California, June 11, 1971 (SRI-ARC Catalog Item 7101).

9. Postel, J. B., Standard File Formats, ARPA Network Working Group Request for Comments 678, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, December 19, 1974 (SRI-ARC Catalog Item 31524).

10. Postel, J. B., Telnet Protocol, ARPA Network Working Group Request for Comments 318, Stanford Research Institute, Menlo Park, California, April 1972 (SRI-ARC Catalog Item 9348).

11. Postel, J. B., The Batch Job Package, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, November 22, 1974 (SRI-ARC Catalog Item 24583).

12. Postel, J. B., The File Package, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, November 22, 1974 (SRI-ARC Catalog Item 24582).

13. White, J. E., A Proposed Mail Protocol, ARPA Network Working Group Request for Comments 524, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, May 31, 1973 (SRI-ARC Catalog Item 17140).

14. White, J. E., Network Specifications for Remote Job Entry and Remote Job Output Retrieval at UCSB, ARPA Network Working Group Request for Comments 105, Computer Research Laboratory, University of California, Santa Barbara, Caifornia, March 22, 1971 (SRI-ARC Catalog Item 5775).

C. A Network-Wide Virtual Programming Environment

15. Maynard, D. S., The Middle End: A Protocol Interface, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, October 17, 1975 (SRI-ARC Catalog Item 26694).

16. Postel, J. B., White, J. E., Notes on a Distributed Programming System, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, March 21, 1975 (SRI-ARC Catalog Item 25613).

17. Watson, R. W., Some Thoughts on System Design to Facilitate Resource Sharing, ARPA Network Working Group Request for Comments 592, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, November 20, 1973 (SRI-ARC Catalog Item 20391).

18. White, J. E., "A High-Level Framework For Network-Based Resource Sharing," Submitted for publication in the AFIPS Conference Proceedings of the 1976 National Computer Conference.

19. White, J. E., "A Network-Wide Virtual Programming Environment," Presented at the Berkeley Workshop on Distributed Data Management and Computer Networks, sponsored by Lawrence Berkeley Laboratory, Berkeley, California, May 1976.

20. White, J. E., "Elements of a Distributed Programming System,"

Submitted for publication in the Journal of Computer Languages, 1976.
21. White, J. E., DPS-10 Version 2.5 Implementer's Guide, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, August 15, 1975 (SRI-ARC Catalog Item 26282).
22. White, J. E., DPS-10 Version 2.5 Programmer's Guide, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, August 13, 1975 (SRI-ARC Catalog Item 26271).
23. White, J. E., DPS-10 Version 2.5 Source Code, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, August 13, 1975 (SRI-ARC Catalog Item 26267).

D. Debugging Network-Based Distributed Systems

24. Victor, K. E., Programmers' Guide to the Debugger (First Draft), Augmentation Research Center, Stanford Research Institute, Menlo Park, California, September 23, 1975 (SRI-ARC Catalog Item 26529).
25. Victor, K. E., The Design and Implementation of a Multi-Process, Multi-Machine, Multi-Language Debugger, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, January 23, 1976 (SRI-ARC Catalog Item 27400).
26. Victor, K. E., Users' Guide to the Debugger (Second Draft), Augmentation Research Center, Stanford Research Institute, Menlo Park, California, January 23, 1976 (SRI-ARC Catalog Item 27299).

E. Operating System Structure and Proof .PBS;

27. Feiertag, R. J., Preliminary Modularization of Multics [for the proposed Security Kernel], Computer Science Group, Information Science Laboratory, Stanford Research Institute, Menlo Park, California, March 10, 1976.
28. Levitt, K. N., Neumann, P. G., Robinson, L., An Interactive Environment for the Specification, Implementation, and Certification of the Multics Security Kernels, Computer Science Group, Information Science Laboratory, Stanford Research Institute, Menlo Park, California, February 1976.
29. Neumann, P. G., Fabry, R. S., Levitt, K. N., Robinson, L., Wensley, J. H., "On the Design of a Provably Secure Operating System," Proceedings of the Workshop on Protection in Operating Systems, IRIA, Rocquencourt, France, pp. 161-175, August 1974.
30. Retz, David L., "Operating System Design Considerations for the Packet-Switched Environment," AFIPS Proceedings, National Computer Conference, Vol. 44, 1975.
31. Robinson, L., Levitt, K. N., Neumann, P. G., Saxena, A. K., "A Formal Methodology for the Design of Operating System Software," R. T. Yeh (ed), Current Trends in Programming Methodology, Vol. 1 (Prentice Hall, New York 1976).

F. Miscellaneous

32. Engelbart, D. C., Watson, R. W., Norton, J. C., "The Augmented Knowledge Workshop," AFIPS Proceedings, National Computer Conference, Vol. 42, pp. 9-21, 1973.
33. Irby, C. H., Dornbush, C. F., Victor, K. E., Wallace, D. C., "A Command Meta Language for NLS," Final Report, Contract RADC-TR-75-304, SRI Project 1868, Stanford Research Institute, Menlo Park, California, December 1975.
34. White, J. E., "Recorded Dialog: the NLS Journal, Identification, and Number Systems," Final Report, Contract F30602-72-C-0313, SRI Project 1868, Stanford Research Intitute, Menlo Park, California, January 30, 1976 (SRI-ARC Catalog Item

```
      22133).
      35. White, J. E., Description of a Multi-Host Journal System,
      Augmentation Research Center, Stanford Research Institute, Menlo
      Park, California (SRI-ARC Catalog Item 23144).
IX  PERSONNEL.PBS;.H2=".Split;Personnel";.Center;.PES;
ORIGIN .H1="Proposal For Research No. ISU 76-136.GCR;Network Operating
System Study"; .F="Part Two -- Contractual Provisions .Split; page
.GPN;"; .SN=Off; .SNF=76; .SNFShow=[1,3]; .RM=72; .BLM=-3; .BRM=68;
.YBS=1; .PN=0;
TABLE OF CONTENTS.PBS;.H2=".Split;Table of
Contents";.GYBS=6;.Center;.GYES=1;
     I  ESTIMATED TIME AND CHARGES .DotSplit; 14
    II  REPORTS .DotSplit; 15
   III  GOVERNMENT-FURNISHED EQUIPMENT .DotSplit; 16
    IV  CONTRACT FORM .DotSplit; 17
     V  ACCEPTANCE PERIOD .DotSplit; 18
    VI  MONTHLY FUNDING .DotSplit; 19
        COST ESTIMATE .DotSplit; 20
           SCHEDULE A -- DIRECT LABOR .DotSplit; 21
           SCHEDULE B -- OVERHEAD AND PAYROLL BURDEN .DotSplit; 22
           SCHEDULE C -- MATERIALS AND SERVICES .DotSplit; 23
           SCHEDULE D -- DOCUMENTATION COSTS .DotSplit; 24
           SCHEDULE E -- COMPUTER SUPPORT COSTS .DotSplit; 25
I  ESTIMATED TIME AND CHARGES.PBS;.H2=".Split;Estimated Time and
Charges";.Center;
   The estimated time required to complete this project and report its
   results is 12 months.  SRI could begin work on receipt of a fully
   executed contract.
   Persuant to the provisions of ASPR 16-206.2, attached are a cost
   estimate and support schedules in lieu of the DD Form 633-4.  Also
   enclosed is a signed form complete except as to the "Detailed
   Description of Cost Elements."
II  REPORTS.H2=".Split;Reports";.Center;
   The following reports will be produced:
      Monthly Status Reports (12)
      Final Report
III  GOVERNMENT-FURNISHED EQUIPMENT.H2=".Split;Government-Furnished
Equipment";.Center;
   The work proposed herein will require continued use of
   Government-Furnished Equipment (GFE).
   SRI will need access to the ARPANET from a PDP-11 computer via IMPs
   now located at SRI.  SRI will also need terminal access to the
   ARPANET from TIPs.
IV  CONTRACT FORM.H2=".Split;Contract Form";.Center;
   It is requested that any contract resulting from this proposal be
   written on a firm fixed price level of effort basis.
V  ACCEPTANCE PERIOD.H2=".Split;Acceptance Period";.Center;
   This proposal will remain in effect until 31 August 1976.  If
   consideration of the proposal requires a longer period, SRI will be
   glad to consider a request for an extension of time.
VI  MONTHLY FUNDING.PBS;.H2=".Split;Monthly Funding";.Center;
   The following are the projected monthly expenditures for the 12-month
   contract period:
      Month    Expenditure  Cumulative Expenditure
        1        $  8,476                 $  8,476 .LBS=0;
        2        $  8,476                 $ 16,952
```

3	$ 8,476	$ 25,428
4	$ 8,476	$ 33,904
5	$ 8,476	$ 42,380
6	$ 8,476	$ 50,856
7	$ 8,476	$ 59,332
8	$ 8,476	$ 67,808
9	$ 8,476	$ 76,284
10	$ 8,476	$ 84,760
11	$ 8,476	$ 93,236
12	$ 10,918	$ 104,154 .LBS=1;

The final month's expenditure includes the cost of producing the Final Report.

COST BREAKDOWN.PBS;.H2=".Split;Cost Breakdown";.Center;.SNF=0;

Personnel Costs

Senior Professional	$	.LBS=0;
Professional		

Total Direct Labor	$ 29,495	
Payroll Burden @ 31%	9,143	

Total Salaries and Wages		38,638
Overhead @ 110% of Salaries and Wages		42,502

Total Personnel Costs		$ 81,140

.LBS=1;

Direct Costs (See Schedules that follow)

Travel	$ 534	.LBS=0;
Computer Facility & Terminals	10,020	
Materials and supplies	350	
Communication	200	
Documentation and Reports	2,441	

Total Direct Costs		$ 13,545
Total Cost		$ 94,685
Profit		9,469

Total Firm Fixed Price		$ 104,154

.LBS=1;

SCHEDULE A -- DIRECT LABOR.PBS;.Center;

Direct labor charges are based on the actual salaries for the staff members contemplated for the project work plus a factor of 6.3% of base salary for merit increases during the contract period of performance. The precise factor applied is dependent on the estimated period of performance. Frequency of salary reviews and level of merit increases are in accordance with the Institute's Salary and Wage Payment Policy as published in Topic No. 505 of the SRI Administration Manual and as approved by the Defense Contract Administration Services Region.

SCHEDULE B -- OVERHEAD AND PAYROLL BURDEN.Center;

These rates have been found acceptable by the Department of Defense for billing and bidding purposes for the calendar year of 1976. Included in payroll burden are such costs as vacation, holiday and sick leave pay, social security taxes, and contributions to employee benefit plans.

SCHEDULE C -- MATERIALS AND SERVICES.PBS;.Center;

Travel

```
      1 trip, Rome, NY, @ $444                  =  $  444 .LBS=0;
      2 days subsistence in Rome, NY @ $30 =        60
      Auto Rental 2 days @ $15                  =     30
                                                    -----
      Total                                        $  534 .LBS=1;
   Air fares are based on prices established in the current Official
   Airline Guide.
   Domestic subsistence rates and travel by private auto are established
   standards based on cost data submitted to and approved by DCAA.
   Communication
   This is an estimate of the toll charges for telephone calls during
   the period of performance.
   Materials and Supplies
   This is an estimate of the cost of materials and supplies based on
   previous experience with similar projects.
SCHEDULE D -- DOCUMENTATION COSTS.PBS;.Center;
   Report costs are estimated on the basis of the number of pages of
   text and illustrations and the number of copies of reports to be
   produced, in accordance with the following rates per page:
      Editing              $2.55 .LBS=0;
      Composition           2.50
      Coordination           .74
      Proofreading           .77
      Illustration         21.96
      Press and Binding     .025 per impression .LBS=1;
   The following is a breakdown of the estimated cost of report
   production:
      Printing, 281 pages at $ 6.56 per page =  $ 1,843 .LBS=0;
         (including editing, composition,
          report coordination, proofreading)
      Illustrations, 15 @ $21.96 =                   329
      Press and bindery at,  10,750 printed pages
      @ $ .025 per printed page =                    269
                                                   -----
      Total Estimated Documentation Costs =     $ 2,441 .LBS=1;
SCHEDULE E -- COMPUTER SUPPORT COSTS.PBS;.Center;
   1) PDP-10 TENEX Computer Time .LBS=0;
      August 1, 1976 to July 31, 1977
         1.5 units * $500/unit-mo * 12 mo =        $  9,000
   2) Terminals,
      a) NLS workstations (1)
         1) Display (1)
         $ 85/mo x 1 x 12 =                           1,020
                                                   --------
        Total (Items 1-2)                          $ 10,020 .LBS=1;
   * PDP-10 COMPUTER SUPPORT SUBCONTRACT COSTS: .GYBS=2; .GYES=1;
   Charges based on estimates from Bolt Beranek and Newman (BBN).
TITLE PAGE .PBS; .H1Sw=Off; .H2Sw=Off; .FSw=Off; .LM=8; .RM=72; .BM=60;
.SNF=Off; .LBS=0; .YBL=0;
   Proposal for Research .Split; 21 May 1976.PBS;.GYBS=12;
   SRI No. ISU 76-136 .GYES=1;
   NETWORK OPERATING SYSTEM STUDY .GYES=3;
   Part One -- Technical Proposal .GYES=1;
   Prepared for: .GYES=1;
   Rome Air Development Center
   Griffiss Air Force Base
```

Rome, New York 13441 .GYES=1;
Attention: Mr. Thomas F. Lawrence .GYES=1;
Request for Proposal No. F30602-76-R-0299 .GYES=2; .LM=40;
Prepared by: .GYES=1;
James E. White
Research Engineer
Augmentation Research Center .GYES=2; .LM=8;
Approved: .GYES=4;

Douglas C. Engelbart, Director
Augmentation Research Center .GYES=4;

Earle D. Jones, Executive Director
Information Science and Engineering Division
TITLE PAGE .PBS; .H1Sw=Off; .H2Sw=Off; .FSw=Off; .LM=8; .RM=72; .BM=60;
.SNF=Off; .LBS=0;
Proposal for Research .Split; 21 May 1976.PBS;.GYBS=12;
SRI No. ISU 76-136 .GYES=1;
NETWORK OPERATING SYSTEM STUDY .GYES=3;
Part Two -- Contractual Provisions .GYES=1;
Prepared for: .GYES=1;
Rome Air Development Center
Griffiss Air Force Base
Rome, New York 13441 .GYES=1;
Attention: Mr. Thomas F. Lawrence .GYES=1;
Request for Proposal No. F30602-76-R-0299 .GYES=2; .LM=40;

 Stoney's msg

< KJOURNAL, 36319.NLS;1, >, 4-AUG-76 14:49 XXX ;;;; .HJOURNAL="DLS 4-AUG-76 14:18 36319"; Title: .H1="Group Idents Associated With RADC...Some Changes"; Author(s): Duane L. Stone/DLS; Distribution: /FEED([ACTION]) JAKE([ACTION]) EJK([INFO-ONLY]) RJC([INFO-ONLY]) TFL([INFO-ONLY]) ; Sub-Collections: RADC; Clerk: DLS; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES;

.PEL; .PN=PN-1; .GCR;Is there a document that explains the meaning of the fields in the Ident Record...Expanded, Groups, Comment...how are they used in preparing the NIC Directory, by SENDMAIL, by MESSAGE? There continues to be confusion on my part (and others) as to the meaning of the group ident RADC, when used as the argument in SENDMAIL and MESSAGE for the addressee field. The RADC ident is (was) used here to broadcast information to NLS users about the availablility of the TIP, printers, to announce presentations of general interest, etc. I suspect that the RADC MULTICS machine becomming a host and the recent NIC update have contributed to the confusion. RADC apparently includes RADC-TIP, which in turn contains a number of idents that weren't there a short time ago and some which shouldn't be there now...primarily RADC-MULTICS----with secondary groups of MIT-MULTICS & RADC-TIP. The net (no pun intended) result is that MULTICS people at RADC receive mail intended for RADC NLS users. They ask a lot of questions, generating unnecessary traffic, etc.

Here are some examples

07/28/76 1259-edt white.558110002 at RADC-MULTICS: Mail from RADC-MULTICS
Distribution: STONE
Received at: 28-JUL-76 10:03:42-PDT
What is the Message Group, and why am I being honored with mail from Kennedy and yourself?

07/28/76 1427-edt Karger.SDruid at RADC-MULTICS: I don't know about that mailman - he seems to be sending me stange garbage like this
Distribution: STONE AT OFFICE-1
Received at: 28-JUL-76 11:31:29-PDT
<< Mail from Network host OFFICE-1 (NETML.NetAdmin) 07/28/76 1213.7 edt Wed >>
Date: 28-JUL-76 0641-PDT
Sender: STONE At OFFICE-1
Subject: Mailman of the future -- a computer?
From: DLS
Message-id: [OFFICE-1] JOURNAL 36227 : (KJOURNAL, 36227, 1:w)
Message-class: [INFO-ONLY]
Comments: Received this from the Message Group...It shows what the 'real' world thinks about electronic mail.

26-JUL-76 0858-PDT KENNEDY: Why did you get that??
Distribution: KARGER.SDRUID AT RADC-MULTICS, WALKER AT RADC-MULTICS., RWALKER, KENNEDY, STONE
Received at: 26-JUL-76 08:58:17-PDT
I have no idea why you got that. The information on the printer was sent to the gereric address RADC. This address is supposed to be for the RADC OFFICE-1 users. It is not impossible that someone, reinventing the wheel, has improved(?) things by adding the RADC Multics group to the RADC group ident.
I'll try to find out and see if that is the reason and if it is what can be done to correct it. Sorry about that and thanks for

the feedback.
Let me suggest the following remedy:
That the RADC-TIP group include only those individuals who use (or can use) the network via the RADC-TIP. This includes groups and/or individuals that access the RADC hosts via the net.
That the RADC ident include only those individuals who have a mailing address at Griffiss AFB regardless of their host mailbox. That the RADC group NOT include the RADC-TIP group...rather the other way around.
That three new group IDENTS be created to reflect the hosts/interests at RADC. RADC-NLS, RADC-NSW, RADC-GCOS
A map of the RADC ident space would then look like the following, where each branch should be expanded to include the plex down from the group ident.

RADC-TIP
- RADC
  - RADC-NLS
  - RADC-NSW
  - individuals using MULTICS & GCOS
- RADC-MULTICS
- RADC-GCOS
- DMAP
- ROCHESTER
- individuals outside RADC

Ident: RADC-TIP
Name: Rome Air Development Center TIP Users
Expand
Membership: RADC RADC-MULTICS RADC-GCOS DMAP ROCHESTER ;
Coordinator: TFL
Organization Type: Tip
Groups: TIPG NICLR
Mail Addresses:
Hardcopy Address:
Sub-Collection: RADC
Delivery: Online

Ident: RADC
Name: Rome Air Development Center
Expand
Membership: RADC-NLS RADC-NSW JER SAB2 FCB DAW2 JMI JVL ECM2 RED2 WDJ RKW WFS ;
Coordinator: TFL
Mail Addresses:
Hardcopy Address: Griffiss Air Force Base, New York 13441

Delivery: Online

Ident: RADC-NLS
Name: RADC NLS Users
Expand
Membership: JAN SLR JLK RCS JVG BJD2 IGT YS ML2 DFR JJV3 AMR OAR JLP ARK LRP RAR3 RAM4 SLR RC3 FJH RKW FPS JFF RWW2 RDK RN2 JBM2 MLK2 ELF ACD ARB LML RJC FJT EJK RAL DVA DFB FSL WER RFI TJB2 RBP JLM JPC DLS TFL WFS ;
Coordinator: DLS
Mail Addresses:
Hardcopy Address: Griffiss Air Force Base, New York 13441

Delivery: Online
Ident: RADC-NSW
Name: RADC NSW Users
Expand
Membership: FJH RAM4 DLS RAR3 TFL JLK LRP ;
Coordinator: DLS
Mail Addresses:
Hardcopy Address: Griffiss Air Force Base, New York 13441

Delivery: Online
Ident: RADC-MULTICS
Name: RADC MULTICS Users
Expand
Membership: to be determined by RKW ;
Coordinator: RKW
Organization Type: Server
Groups: SERVERG
Mail Addresses:
Hardcopy Address: Griffiss Air Force Base, New York 13441

GROUPS
RADC
RADC-NLS
RADC-NSW
RADC-GCOS

Delivery: Online
WALKER@RADC-MULTICS
Ident: RADC-GCOS
Name: RADC GCOS Users
Unexpanded
Membership: to be determined by WALKER@RADC=MULTICS ;
Coordinator: RKW
Mail Addresses:
Hardcopy Address: Griffiss Air Force Base, New York 13441

Delivery: Online
WALKER@RADC-MULTICS
assoc DMAP to be determined by DMAP group coordinator — already exists doesn't
ROCHESTER to be determined by ROCHESTER group coordinator
There are other individuals and organizations that are sponsored by RADC in their use of the net and NLS. They should be grouped with appropriate names. If the following don't conflict with existing names, then use them as indicated to set up the groups.
assoc AFSC
Ident: AFSC
Name: Air Force Systems Command
Unexpanded
Membership: RWJ JAJ WWP2 + others to be determined by RWJ ;
Coordinator: RWJ
Mail Addresses:
Hardcopy Address: Hq., Andrews Air Force Base, Maryland 13441
Delivery: Online
BJOHNSON@OFFICE-1
assoc TRW
Ident: TRW
Name: TRW
Unexpanded
Membership: NWP + others to be determined by NWP ;
Coordinator: NWP
Mail Addresses:
Hardcopy Address: TRW Systems Group, 1 Space Park, Redondo

Orgs
RADC-TIP
RADC-MULTICS
RADC-XPER
ROCHESTER
assoc DMAP
AFSC
TRW

Beach, California 90278
Deliverv: Online
PETERSON@OFFICE-1

GSGI
The ident GV should be added to the membership list.

36760

Workshop Utility Service News

September 1976

Issue Number 2

Augmentation Research Center
Stanford Research Institute

TABLE OF CONTENTS

Paragraph

EDITOR'S STATEMENT [Robert Lieberman] 3

You have to believe now--a SECOND edition of our internationally famous Newsletter. Well, we are really pleased to gather all these bits of information together for another issue and hope you find it informative. Please keep those many letters and journal items coming. Comments, suggestions, etc., are encouraged and appreciated. Thanks.

This issue can also be found in <arc-log,Sept-news,>.

Previous issue <35119,> April 1976; also found in <arc-log,April-news,>

OUR STAFF [Robert Lieberman] 4

In this section we will just list the changes in our staff since the last issue.

NAME	IDENT	CHANGE or PRIMARY FUNCTION
HARRENSTIEN, Ken	KLH	Nic programming
LOUVIGNY, Josette	JRL3	ARC-APP secretary
METZGER, Laura	LJM	Moved to southern California
MEYER, Dean	NDM	Graduated MBA; now at Xerox, Rochester, NY
MIRANDA, Sandy	SLM2	Gone to the lush Santa Cruz Mt.
MOUTON, Kate	KATE	Programming
PARTEL, Jean	JLP2	Left for another job
WATSON, Dick	RWW	Now at Lawrence Livermore Labs
ZOLOTOW, Nina	NINA	Documentation

UTILITY CLIENTS [James Norton] 5

The following is a list of present Subscriber Organizations with their acronyms.

AFDSDC	AF Data Systems Design Center
ARC	Augmentation Research Center
ARPA	Advanced Research Projects Agency
ATT	ATT Long Lines
BELL	Bell Canada
BRL	Ballistic Research Laboratory
DARCOM	Army Materiel Development & Readiness Command
DMA	Defense Mapping Agency
IBM	IBM Federal Systems Division
NIC	Network Information Center
NSA	National Security Agency
NSF	National Science Foundation
NSRDC	Naval Ship Res and Dev Center
RADC	Rome Air Development Center
SRI	Stanford Research Institute
TRW	TRW Systems Group

DATES ON THE CALENDAR

6

SEPTEMBER 1976

Sep 8-10: COMPCON 76; Washington, DC
Sep 8-10: International Symposium on Technology for Selective Dissemination of Information, Palazzo dei Congressi Repubblica di S. Marino, Italy
Sep 16-17: Society for Management Information Systems (SMIS) Annual Conference, Philadelphia, PA
Sep 20-24: KWAC meeting in Montreal, Canada

OCTOBER 1976

Oct 4-9: ASIS conference, San Francisco, CA; Presentations by friends and staff: R. Belleville, J. Feinler, and R. Uhlig, with J. Bair as chairman of a session.
Oct 13-15: Second International Conference on Software Engineering, San Francisco, CA
Oct 18-19: Graphics Seminar, Houston, Texas
Oct 20-22: ACM 76 Annual Conference, Houston, Texas
Oct 24-27: INFO/EXPO 76, Las Vegas, Nevada
Oct 25-29: AKW Seminar for ALFA-LAVAL

NOVEMBER 1976

Nov 1-5: AKW Seminar
Nov 17: Conference on Computer Networks: Trends and Applications, Gaithersburg, MD

DECEMBER 1976

JANUARY 1977

Dec 24-26: Conference on Decision Support Systems, San Jose, CA

COMPUTER OPERATIONS [Robert Lieberman] 7

As most of you now realize, OFFICE-1 has doubled its core size to 512K. Although we have not fully analyzed the effects, we do have a general impression from talking to many users that the response is improved at a given load average and the load averages are lower. There has been no other changes in the hardware configuration and none are expected in the short term.

A new software fix in the scheduler is planned for the next month or so that will (hopefully) further improve interactive response at the sacrifice of other jobs (e.g., Output processor, compilers, CPU bound jobs). Also, we are actively making plans for computer support at Office-1 for next year.

TRAINING NEWS [Susan Roetter] 8

User Services News

Since the last Architect's meeting two new clients have contracted for Office-1 services and have received initial training: NSF and TRW. Client assignments are unchanged except for the addition of these two new ones. TRW has been assigned to Susan Roetter, and NSF has been assigned to Jeanne Beck. In addition to training for new clients, the following groups have also received training: AFDSDC (Gunter), NUSC, AT&T, SRI, and DMA.

The User Services staff worked closely with the Application Development group to revise the Introductory Course and began work on a course to train people at subscribing sites to be on-site trainers.

The Office-1 usage statistics have been compiled twice a month instead of monthly to provide more timely information to users. The Feedback operation has switched from a one-person job to a rotating job among the trainers.

NOTICES 9

At some time in the near future, five core procedures (ccopgrp, cmovgrp, crepgrp, ctragrp, ctrasta) will be changed in way that might effect programmers (NOT NLS Users) that call these procs. In order to give precise information of these changes (rather minor ones effecting the RETURN parameters only) and the exact date when this will happen, we are setting up a group ident for all those interested in L10/CML programming news. This ident will be used in the future for any and all announcements about L10/CML programming. If you desire to be put into this group, please let FEEDBACK know. The ident is NPIG for NLS Programming Interest Group.

BUG REPORTS AND FIXES (Dave Hopper and Karolyn Martin) 10

FIXES

In DNLS, most characters (a, b, c, etc.) are transmitted as one character. Others (OK, bugmarks, command delete, etc.) require several characters to transmit all the information for cursor positions and so forth. These are called big character sequences. Occasionally a character from a big character sequence gets lost in transit. This can cause problems in deciphering that (big) character. We have added a check for a lost 3rd character from a sequence, which formerly caused Illegal Instructions. The system will now display the message "Transmission Error - Third character lost in big character", discard all typed-ahead input, and generally resynchronize input. The user may then just continue by typing the character again.

In addition, a lost 2nd character now causes the message "Transmission Error - Second character lost in big character" to be displayed, discards typed-ahead input, and resynchronizes input. The user may then just continue by typing the character again. The former message was "Illegal Input - Bad BCCNT". We hope the new message is clearer.

The Load File command now accepts remote host name as part of link. For example, <ISIC,xprograms,xtable,>.

Other Universal Subsystems are possible:

It is now possible to replace the Supervisor (commands that always appear in each subsystem, e.g. Execute, Goto, Jump, etc.) with another subsystem as the universal subsystem. For example, user-programmers can edit the CML to make subsets of the Supervisor.

A different universal subsystem can be specified by changing the User Profile.

CTRL O has been modified to be almost instantaneous (if there are no Network delays) instead of taking effect after the information that has already been delivered to the output buffer is printed, displayed, or executed. (The code now clears the output buffer as well as the input buffer.)

In the past, if viewspec v was on (no automatic display refreshing), it didn't always display correctly in the viewspec window. It has been fixed so that it displays as v (not u) except for a very short time during which an explicit refresh (viewspecs f or F) is taking place.

Formerly, if a process commands structure included a Simulate Lineprocessor command, the rest of the process commands structure was lost. It has been fixed to correctly keep track of the source of command input in this case.

The Renumber Sids command has been fixed to handle graphics statements correctly. The problem was that SIDS sometimes got used twice.

The Control-S mechanism has been fixed to accommodate very long command syntax strings.

SUGGESTIONS RECEIVED [Robert Lieberman] 11

A new command to allow immediate formatting for the proof subsystem, Output (to) Proof, was suggested.

A new OP directive to specify a minimum number of lines for a statement that must be printed at the bottom of the page has been mentioned by several people.

Many users have asked that the TENEX level program SPELL be fixed to run under NLS.

For the GRAPHICS subsystem, a new feature to allow shrinking and inflating of a figure or group by a single command was suggested. Additional suggestions include a Replace Figure command and an Align Arrowhead command.

PROGRAMS AND NEW FEATURES [Karolyn Martin and Robert Lieberman] 12

The Insert Record command has been added to the Sendmail subsystem. It inserts the ident record for the specified ident after the specified statement.

The name of the sendmail file has been changed to contain the user's ident in the first name of the file (not in the extension as before). This allows several users to operate in one directory without conflict in the use of sendmail files.

Two new TENEX commands have been added: TNLS and DNLS. Each one does the necessary setting of terminal types and so forth, then calls NLS.

A DNLS user who is detached from his/her job can now attach to that old job and continue in DNLS. Please make sure you respecify terminal type before continuing; and note that it may be necessary to use the system reset button on the LP after returning to DNLS.

TRICKS and TREATS [Robert Lieberman] 13

The NLS privacy feature allows restricted access to individual files on a person or group basis. This is quite handy when there is a real need to maintain some privacy; however, it may cause some unnecessary lack of communication to others who have the need and the permission to read a protected file. This is very likely to occur if a new staff member becomes involved with a particular project and needs to 'catch up' on the happenings that appear in the Journal.

We have one suggestion that may be of some help in this situation. When specifying the distribution include "&groupident" as an ident. (Hopefully there is an appropriate group ident.) This will only send a citation to the coordinator of the group and NOT any of the members. Nevertheless, if any member of the group wishes to read this journal item, no problems will ensue since she does belong to the group. Also remember to set the mail item to be private with the SENDMAIL command <>PRivate.

For individual files, you must NOT include the '&' as part of the group ident in the 'AccessList' itself in statement zero; the '&' should be used only in a sendmail distribution list.

USER APPLICATIONS [Robert Lieberman] 14

In this section we list applications at new subscriber sites.

National Science Foundation (NSF)

The Access Improvement Program is composed of three professionals and one secretary. This small office has plans to be committed to using NLS as much as possible. Chronological files, indexes, event databases, memos, forms, and reports will all be done with NLS.

NIC NEWS [Dave Maynard] 15

The Network Information Center is currently in the process of producing a new version of the ARPANET DIRECTORY using the Output Processor and COM. It is now in the printing stage, and an announcement will be made when issues are available.

Many of you will be interested in a new user subsystem that the NIC has developed called MEMLIST which extracts selected portions of the IDENTIFICATION DATA BASE. This subsystem lets you produce 'mini' Arpanet Directories for any set of individual, group, and organization idents you choose. It also lets you make a brief listing of name and online network mailboxes, and lets you produce distribution lists suitable for use with the TENEX sndmsg program. This subsystem will shortly be available for your use in the <XPROGRAMS> directory. We invite you to try it, and send comments, gripes, and suggestions to FEEDBACK or FEINLER@OFFICE-1.

DEVELOPMENT NEWS [Jon Postel] 16

NLS 9 Development

As part of the National Software Works (NSW) project, NLS 9 is designed as two distinct modules called the Frontend (FE) and the Backend. The Frontend is that part of the system that acts as the human interface; it talks to the user in his language, and then communicates instructions to the Backend, which carries them out. The FE handles everything the user does at a terminal (e.g., command parsing, command recognition, echoing), while the Backend puts changes in the actual files.

Except in minor areas where required by the changes in system architecture, the NLS 9 user interface will be the same as that of NLS 8.5. NLS 9 is being written in XL10, an improved version of L10. Most of the major NLS subsystems required for the NSW have been converted to NLS 9. ARC has tested an experimental TENEX version (as opposed to an NSW version) of NLS 9. A good deal of further work is needed in the NSW protocols area, in efficiency and reliability features, as well as general bug fixing.

Frontend Work Continues

ARC is building a generalized Frontend for use with many tools. This is made possible by a modularized design that enables the FE to talk to other processes in many ways, switch between applications, and easily add new applications. The Frontend will work on both a PDP-10 and PDP-11, making it possible for a local user to the PDP-11 (a minicomputer) to communicate with a remote PDP-10. This will allow fast response for all FE functions. As part of the FE implementation, improvements have been made in the Command Meta Language (CML), Command Language Interpreter (CLI), and display interface. The FE has been tested in a variety of situations on the PDP-10 and testing is just beginning on the PDP-11.

The Do All Debugger

The first stage of the Do All Debugger (DAD) has been completed and work continues on the implementation of this multi-process, multi-host, multi-high level language, interactive debugger for the NSW environment. A version has been in use for several months by ARC programmers working on NLS. Interested individuals should contact Ken Victor (KEV).