# AN ARPANET FRONT-END FOR LARGE COMPUTERS

Ed Franceschini

Courant Institute of Mathematical Sciences
New York University

## ABSTRACT

The ELF system of David Retz was written to provide access to the ARPA network for communities of users which did not have a HOST computer. As such it incorporates support for many of the standard network protocols in a PDP-11 minicomputer. ELF thus provides a good foundation for an ARPANET front-end system. This paper describes certain extensions to the ELF system which provide time sharing and file transfer access to an arbitrary HOST computer. The emphasis is on the external specifications which define the interfacing requirements to be met by the prospective HOST system, rather than on internal design. For illustration, a current implementation using a CDC-6600 as HOST will be described.

This work was performed under contract AT (11-1) 3077, Energy Research and Development Administration.

## 1. SCOPE OF THE DISCUSSION

This paper describes the external specifications of the first of a two stage ARPANET front-end system currently under development at the Courant Institute of Mathematical Sciences. It is a preliminary document intended to provide material for discussion to groups which may be considering using this or a similar method for interfacing their HOSTS to the ARPA network.

No attempt will be made at this time to consider issues such as motivation for the front-end approach, internal program design, performance evaluation, or operational procedures. At a later date a more comprehensive document covering these topics will be prepared.

## 2. BACKGROUND

It is assumed that the reader is familiar with the ARPA network design, protocols, and access methods.

For background knowledge of the ELF system of David Retz, on which the NYU front-end system is based the reader is referred to various unpublished documents prepared by the Speech Communications Research Laboratory of Santa Barbara, California. For those not anxious to get deeply enmeshed, "An Overview of the ELF Operating System", by David Retz, is perhaps best.

However, I will mention a few things about the ELF system which will help with the matter at hand. The ELF system is written in assembly language and runs on the PDP-11. It supports various HOST TO IMP interfaces including the ANTS interface, the A Consultant interface for Very Distant Host Connection, and the DEC interface.

The system is hierarchical in design. Its lowest level, called the KERNEL is responsible for the management of hardware resources such as the processor, storage and peripherals. The second level includes the EXEC which provides terminal support, and the NCP which supports communications between processes within the ELF and processes in HOSTS at other network sites. Both the EXEC and NCP provide a set of system calls to the next level in the hierarchy which consists of the so-called user programs. The user program of interest here is TELNET, which provides conversational access to the network for ELF users, that is, it implements the User Telnet protocol.

## 3. THE NYU ELF EXTENSIONS

The purpose of the NYU effort is to provide access from the network to the time sharing system and file system of an arbitrary HOST computer system. (The notion "arbitrary " means that no HOST system dependencies should appear in the ELF). The ARPANET protocols needed to achieve these goals are Server Telnet and Server File Transfer. Thus, generally speaking, the NYU ELF extensions consist of programs to support these two protocols. Operationally, the programs run as detached user jobs which have access to the terminal multiplexor through the EXEC and to the network through the NCP.

### 3.1 HOST TO FRONT-END COMMUNICATIONS

Hardware communications between the two computers is achieved by interconnecting several ports of the respective asynchronous telecommunication device multiplexors. In effect each computer will see the other as several interactive terminals. As a future enhancement of this system we contemplate a direct channel to channel coupling, but that is beyond the scope of this discussion.

The ELF Server Telnet process and the HOST time sharing system use one physical connection for each network connection (that is, for each user), while the ELF Server File Transfer process and the HOST file system use one physical connection to multiplex all concurrent file transfer sessions.

For the sake of clarity the time sharing and file transfer functions will be considered separately.

## 3.2 TIME SHARING FUNCTION

Of the several ports allocated to the time sharing function, one is reserved for control messages between the ELF Server Telnet and the HOST time sharing system. Through this medium the following control functions are implemented:

1. ELF Control of output flow from HOST time sharing system.
2. ELF notification to HOST time sharing system of explicit CLOSE from user.
3. HOST time sharing system notification to ELF of normal user logout.
4. HOST time sharing system notification to ELF of auto-logout due to user inactivity.

Appendix A shows the details of these transactions.

The implementation of these functions may be easier in some HOSTS if some characters are reserved for control of flow. For example, TELENET controls flow from its HOSTS by the X-ON and X-OFF characters in the data stream. The other controls are analogous to disconnect events associated with real teletypes and can be implemented by manipulation of the modem control functions of Data Set Ready and Data Terminal Ready. We chose the control port method in order to maintain full transparency with respect to the data streams and to avoid modifying the ELF KERNEL modem drivers.

## 3.3 FILE TRANSFER FUNCTION

All communications between the ELF file transfer process ( FTPSVR) and the HOST file system process take place over a single full-duplex teletype line (although not necessarily at teletype speed). The protocol used has been adapted from Michael A. Padlipsky's "Host-Front End Protocol" and is outlined in Appendix B.

The teletype line is driven from ELF by a new kernel driver (KDFE). This new module is a version of the standard ELF keyboard

driver (KDKB), modified to eliminate the special handling of certain control characters.

The H-FE protocol has just five message types. All message types have interpretations in both directions, although some fields have no meaning or relevance in one direction or other. In the diagrams of messages given in the appendix, each field has its constituent number of bytes marked above it.

The changes and restrictions applied to the H-FE protocol for our use are as follows:-

| Field | Use |
|---|---|
| SOCKET | is always 3, indicating that only an FTP connection is valid |
| CONNECTION TYPE | is always 1 for duplex |
| PAD | is one byte long and used to indicate the type of message (see below) |

Three message types are used (in the PAD field). The three possible values (in octal) are:-

10 - FTP command or reply
11 - FTP ASCII data
12 - FTP image (binary) data

The FTPSVR in ELF decodes each incoming FTP command and decides whether to process it itself, or to pass it on to the HOST. This choice is defined in the table given in Appendix C. (FTP commands not shown in this table are not implemented in the first stage of the project).

In order to facilitate communication and synchronization between ELF and the HOST, we have introduced two new "pseudo-FTP" commands called OPEN and CLOS.

Upon establishing an FTP connection, FTPSVR sends the HOST an OPEN. The HOST, if alive, replies with a 300-type greetings message. If the HOST does not reply in a certain time, FTPSVR closes the connection.

Whenever a connection is closed, FTPSVR sends the HOST a CLOS command. The CLOS never needs a reply; it terminates ELF to HOST communication associated with the particular FTP connection.

All FTP replies sent out by the HOST are passed on to the network by FTPSVR. However, they are all decoded on the way, so that ELF can keep track of the state of the connection. In particular, this enables FTPSVR to know when a data transfer is in progress. Then, if data is received when FTPSVR thinks that no transfer is in progress, it is discarded. While a transfer is in progress, all data is passed blindly through ELF.

## 4. AN IMPLEMENTATION USING A CDC-6600 AS HOST

The CDC-6600 at NYU is currently running the CDC KRONOS Operating system and a change to NOS is planned for the near future. Unfortunately with respect to transportability, CDC makes available three operating systems and many users have either modified these or built their own. Our operating system is close to standard except in the one area impacted by the ARPANET connection. Figure 1 shows that the ELF PDP-11 is connected to the CDC-6600 through a telecommunications front-end consisting of a Honeywell H516 minicomputer. Figure 2 is a slightly idealized illustration of the interconnection of processes in the three computers. TELEX is the name of the KRONOS time sharing subsystem and ITD is its handler for H516 communications. EXPORT is the name of the NYU remote batch subsystem and RIO is its handler for H516 communications.

The Server Telnet commands to the HOST are handled completely in the H516 so that no change has been made to TELEX or ITD.

The HOST-FRONT END protocol used for the file transfer function is implemented in the H516 but all file transfer protocol commands and data are passed on to EXPORT, which provides the interface to the file system.

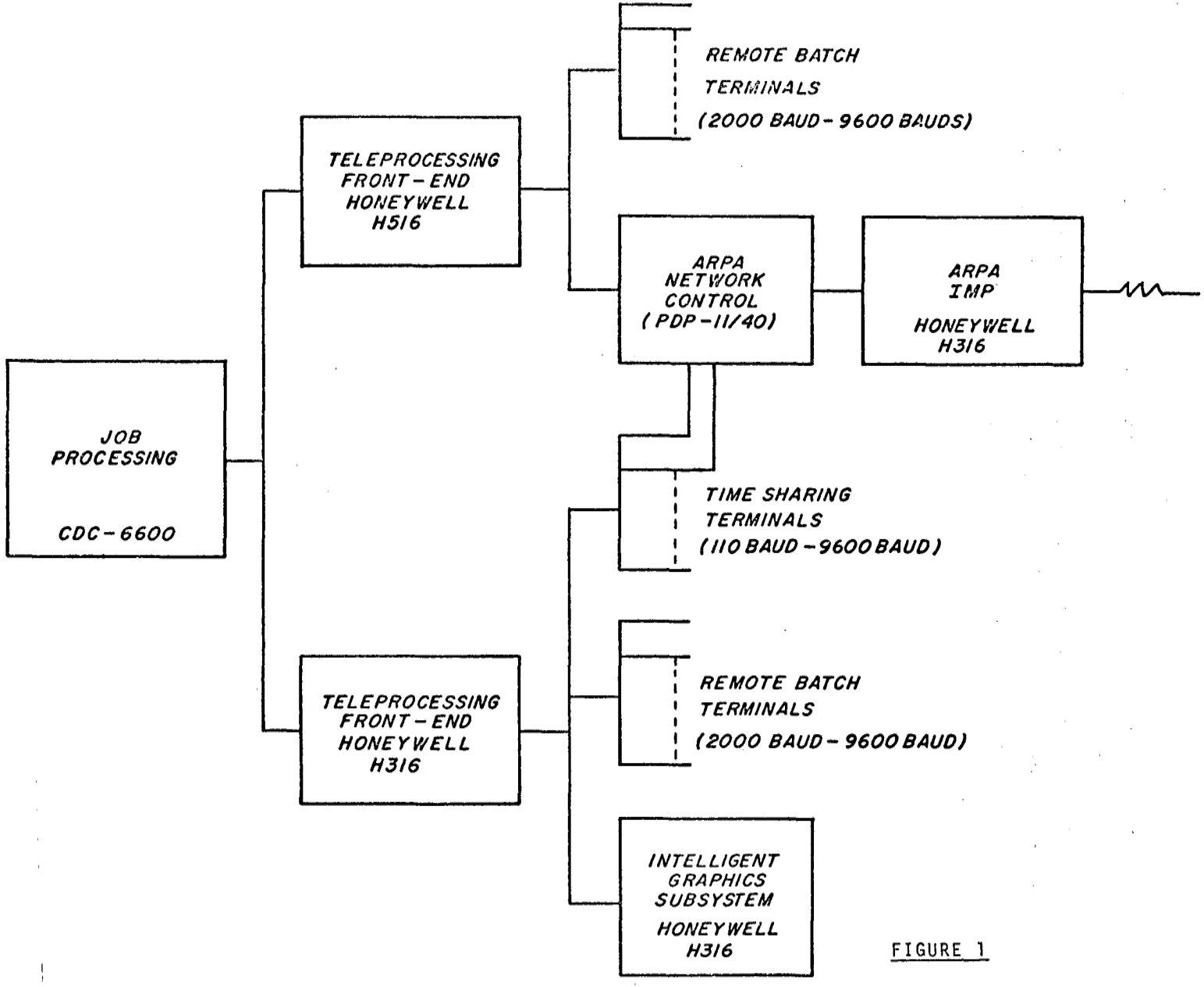

REMOTE BATCH
TERMINALS
(2000 BAUD - 9600 BAUDS)
TELEPROCESSING
FRONT - END
HONEYWELL
H516
ARPA
NETWORK
CONTROL
(PDP-11/40)
ARPA
IMP
HONEYWELL
H316
JOB
PROCESSING
CDC - 6600
TIME SHARING
TERMINALS
(110 BAUD - 9600 BAUD)
TELEPROCESSING
FRONT - END
HONEYWELL
H316
REMOTE BATCH
TERMINALS
(2000 BAUD - 9600 BAUD)
INTELLIGENT
GRAPHICS
SUBSYSTEM
HONEYWELL
H316

FIGURE 1

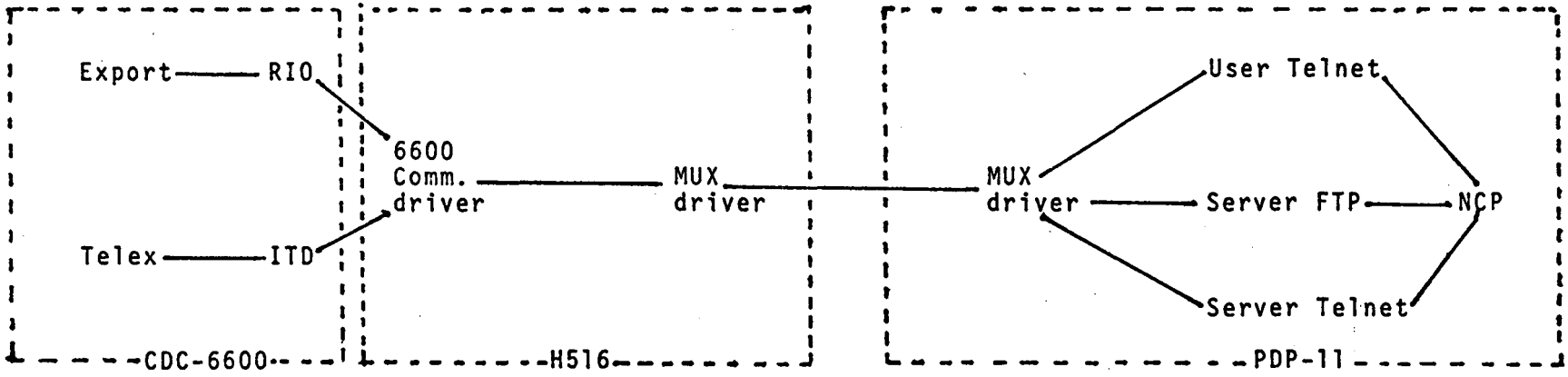
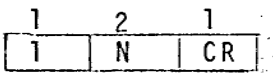



FIGURE 2

## APPENDIX A

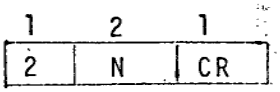
ELF Server Telnet to HOST Time Sharing System Control Messages.

SUSPEND OUTPUT



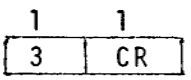

N = port number on which output is to be suspended

RESUME OUTPUT

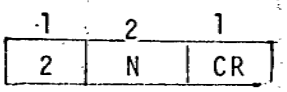



N = port number on which output is to be resumed

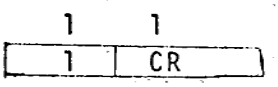
REQUEST FREE PORT



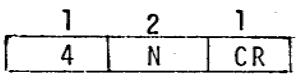

The HOST is being asked whether one of its (preallocated) ARPA ports is free. (It should be noted that if a LOGOUT has occured but no CLOSE, a port is available but ELF is not aware of it.) The HOST responds either with the number, N of an available port:




or, if no ports are free:




CONNECTION CLOSED




Informs the HOST that the ARPA connection associated with port N has been closed. (The HOST should treat this as if a teletype had disconnected)

In all the above cases where specific responses have not been indicated the HOST replies with the general acknowledgement:

| 1 | 1 |
|---|---|
| 1 | CR |

All messages are in ASCII. Port number is a two digit octal number represented in ASCII.