Network Working Group
Request for Comments #163
NIC #6775
Categories: D.7
Updates: None
Obsoletes: None

V. Cerf
19 May 71
UCLA - NMC
Computer Science

DATA TRANSFER PROTOCOLS

This is an informal statement of material discussed at the SJCC. There are two peoblems.

1. Movement of data from one site to another.
2. Interpretation of the data at receiving site.

The first task (1) requires a simple protocol which accomplishes the following

1) Standard connection procedure for connecting transmitting and receiving processes

2) Standard packaging which allows network to collect the transmitted data stream in the right order and know when the end of the file has been reached.

Standard Connection Procedure

Suppose every HOST has a process charged with the responsibility of sending and receiving files between -HOSTS-(processes?)[The Data Manager]. If the Data Manager offers to listen on a given socket for file xmt requests, then ICP is sufficient to establish a connection between a serving Data Manager and a using process.

We have completely avoided the discussion of data interpretation, and also the problem of control. For instance, we have not said how a process can ask the Data Manager to send a file of a par- ticular name, nor how to end the transmission of a file. This is deferred for later.

Another desirable ability is to have processes transmit files to each other independent of the HOST Data Manager. ICP should suffice, for the creation of a full duplex connection. File naming, and format interpretation are left to the individual process to solve.

It is of interest to note that files need not have names. If two processes are connected, then the file name is in a sense implicit in the sending and receiving socket pair. One imagines, however, that connections with Data Managers for the purpose of file transmission are too transient to serve as permanent file names, so information about file name will be needed by the Data Manager. This information could be supplied either embedded in the file transmission data stream, or supplied over a separate control connection established at ICP time.

It seems reasonable that a Data Manager have a network-wide, fixed socket number on which it is listening to service data transmission requests.* In this sense, it acts much like the Network Logger. For inter-process file transmission, less rigidity seems called for, and we can leave such decisions to the individual peocesses communicating with each other. Public processes at serving HOSTS could have known (nia NIC?) sockets over which file transmission is acceptable.

Standard Packaging

We naively imagine that very little in the way of formatting is needed to move data across the connection. A few bits (8?) at the beginning of transmission could specify the formatting protocol (e.g. arbitrary bit string until connection closed, count field + data, break chars, etc.) Depending on the selected format mode, the appropriate control bits will or will not appear interspersed betweeen the data bits. Message boundaries are totally transparent.

A way of ending the file, possibly without closing the connection, is useful, although closing the connection after the RFNM from final "record" sent is received by the sending process might be adequate (sufficient, but not palatable?)

----
*ICP causes sockets to be dynamically assigned for the ensuing conversation (which might be all 1-way).

Control

A great many problems come up if the Data Manager serves as a part of the HOST filling system. For example, the Data Manager must know whether the process it is serving wants to send a file or receive one. In either case, some sort of file name + qualifiers (user ID, security codes, access requested, etc.) will be needed to resolve the usual access legality, and potential file name ambiguities. This information can be supplied either within a single full duplex data stream (1 per ICP request) established by a modified ICP for data transmission. The former seems simplier, sufficient, and immediately implementable.

Data transmission between arbitrary processes probably does not need as much formal control protocol as process-to/from-DM (data Manager) connection. Ad hoc procedures can be established by trading information on previously established connections; regularity is nice, so perhaps a standard set of control protocols can be devised which work, regardless of the identity of the processes transmitting data. Control data must be formatted and probably identifiable by prefix codes so that unnecessary control information can be left out if desired. (I am thinking specifically of file names.)

It remains to establish a set of format protocols which permit packaging of data and identification of control information. This should be the task of the renamed Data Transmission Committee.