International Institute for Applied Systems Analysis
IIASA www.iiasa.ac.at

User-Oriented Networks: A Series. Part III. User-System Communication

Orchard-Hays, W.

IIASA Working Paper

1975

USER-ORIENTED NETWORKS: A SERIES
PART III. USER-SYSTEM COMMUNICATION

Wm. Orchard-Hays

July 1975 WP-75-83

TABLE OF CONTENTS

	Page
Orientation with Respect to Central Computers	1
Some Additional Terminology	1
Protocols to Access a SYS from a TER	3
Connection Remote Unit Record Equipment	8
Logoffs and Crashes	12

ORIENTATION WITH RESPECT TO CENTRAL COMPUTERS

In Part II, a network was described in overall terms which easily handles inter-user communications. This largely ignored, of course, a major function of the network, namely, productive work on large central computing systems (denoted there and here by SYS). The sequence was deliberate. Networks are usually thought of as built around a SYS or set of them, in other words, the network is an adjunct to the central computer(s). The viewpoint in user-oriented networks is just the opposite: central computing systems are facilities available on the network but not indispensible units for all functions of the network. It seemed desirable to establish this viewpoint first.

The above observations, or even a working network as described in Part II, do not diminish the importance of central computers nor make the inherent difficulties of using a variety of them disappear. There will be much to say in subsequent parts of this series about the problems of incompatibility among systems and the confusing variety of conventions, formats and protocols. However, the network scheme of Part II is even more important in dealing with these problems than in handling inter-user communication. Properly used, it can deal fairly effectively with incompatibilities among systems so long as this is necessary, and can be employed to gradually force more standardization in the future.

SOME ADDITIONAL TERMINOLOGY

The symbology of Part II needs some extension. It is unnecessary to define geometric symbols for most new abbreviations since the latter will usually represent abstract concepts which are not readily stylized. First a succinct abbreviation is needed for the type of network suggested in Part II. Since it is a chain of hierarchical sub-networks, we will call it a CHINE. (A "chine" is a

backbone or spine and its surrounding parts.) The major nodes of the CHINE are the CONs, which are chained together with long-distance communication lines. The CONs are also the top level of the sub-networks as well as the connecting points for SYSs. It is the latter feature which will enable us to partially overcome incompatibilities between systems.

We should now be a little more specific about a SYS. What is meant here by a SYS is the hardware and software of a "host" computing system, usually thought of as large. (However, medium-sized computers such as a CDC 3300 might be used for the hardware of a SYS in some places.) A SYS must have some kind of comprehensive operating system (basic software) which always underlies any application programs or systems. While it might be of value to IIASA researchers in some circumstances to utilize small, stand-alone computers, this is not at all compatible with the concept of a network.

We now encounter the first dilemma in terminology with respect to computing system architecture. A good illustration is provided by the IBM 370s with virtual memories (a large one of which this writer desperately hopes will be available). The hardware/software host is called VM/370 but, even in IBM literature, it is defined ambiguously. The basic host system consists of the hardware and a control program called CP. The term VM/370 is also used to include, however, a conversational monitor system called CMS. If one is using CMS, then the combination CP/CMS is in fact the host software. However, batch operating systems may also be run under "VM/370" (i.e. under CP) in which case the host system does not appear to be a conversational monitor at all (except for direct CP commands which are more like system operator's commands).

Now when a user logs in to the host system, he is normally at CP level. (Installation conventions can cause automatic entry to CMS level but the user may still go back to CP and then initiate some other system in place of CMS.)

There is no difficulty in controlling all this from a terminal hooked exclusively to the system, by a user who understands (or at least knows by heart) the conventions in effect and what he wishes to do. But in attempting to standardize network terminology, it is a little difficult to define "host" precisely and to specify just how many protocols are required.

In spite of all the circular definitions and layered operating modes of virtual machines, it is still meaningful to divide all remote computing into two types: batch with remote job entry (RJE); and interactive with a conversational monitor system (CMS). Once in RJE mode, operations proceed in more or less traditional computing style.* However, application systems, which may run under CMS mode, can have elaborate characteristics of their own. This will not be discussed further in this paper.

PROTOCOLS TO ACCESS A SYS FROM A TER

Consider a part of a CHINE as shown in Figure 1. (Refer to Part II for symbology conventions.) Suppose an authorized user identified as BaPD wishes to use SYS A1 from TER Ba11 and to have printed output sent to PRT Ba01. We will assume SYS A1 operates in an interactive mode. What are the necessary protocols to start, continue and terminate the process?

First, BaPD (the person) must turn on Ba11 which may include a telephone dial-up, depending on the local physical arrangements. If GRP Ba (hereafter referred to as Ba00) is

---

*Somewhat of an exception exists with IBM's Time Sharing Option (TSO). This is a conversational mode with many of the features of a batch system and intended primarily to control batch-like operations remotely, plus providing interactive file editing.

not in operation, there will be no response and nothing further can be done. If BaOO is up, then transmission protocols must be carried out between Ball and BaOO. This is below our level of recognition and will be regarded as automatic. However, it may be necessary for BaPD to type one or two characters at Ball to identify the terminal, for example, ASCII at 30 cps. (Such conventions differ widely.) In any case, we can assume that BaOO now knows that Ball is connected and what kind of transmission mode is necessary for the messages between them.

BaOO must now indicate its readiness to accept messages from Ball. This may be a message to Ball something like

GROUP BaOO IN OPERATION

The expected response to this is a log-in, such as

LOGIN BaPD

BaOO will now look in its table of authorized users to see if "BaPD" is a listed userid and not already in use. (BaPD might have permitted someone else to use his account and it is already in use from a different terminal.) If BaPd is not listed or is in use, BaOO sends an appropriate message to

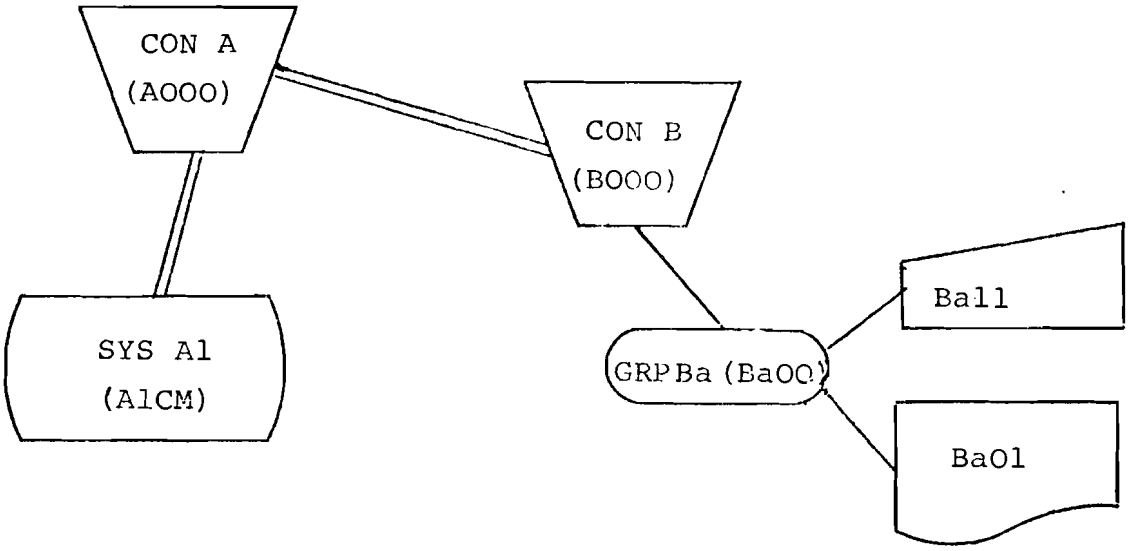

Figure 1. Part of a CHINE

Ball and then breaks the connection. Assuming BaPD is a legitimate userid, BaOO will next issue a demand for a password

PASSWORD:

BaPD will now type his password (and perhaps account number) which will be masked somehow so as not to appear on hard copy. If this is incorrect, BaOO should be programmed to re-request the password once. If incorrect a second time, the connection to Ball is dropped.

Once the userid and password are verified correct, BaOO looks to see if BOOO (i.e. CON B) is in operation. If not, BaOO sends a message such as the following to Ball

> CONCENTRATOR B NOT IN OPERATION. ACCESS TO GROUP Ba FACILITIES ONLY.

BaPD may now choose to log off if his only interest is in accessing SYS A1.

Assuming BOOO is up, then BaOO sends a prompting flag to Ball which may type a characer (such as >), unlock the keyboard, or somehow indicate readiness to accept a message from Ball. User BaPD (the person) must know the identification of the system he wants to use. We will assume this to be A1CM (CMS on SYS 1 attached to CON A) so far as the network is concerned. The actual identification which the SYS accepts need be known only to AOOO.

We now need a network command which will establish the availability of A1CM to Ball. In order to avoid conflict with commonly-used command mnemonics (such as ACCESS, USE, FIND), we invent the command HOOKUP, explained by example below. BaPD types the following command at Ball

HOOKUP SYS A1CM

Strictly speaking, the "SYS" is redundant since "A1CM" can only refer to a SYS. However, redundancy of this sort is useful. Suppose BaPD typed AiCM instead of A1CM. BaOO can

immediately check for this inconsistency and send an error message back to Ball, without putting any further load on the network.

The command HOOKUP (assuming correct syntax) starts a whole chain of events which are carried out automatically. The scenario goes as follows.

1. BaOO creates a message which can be denoted as follows:

   (to from request sys ter user)

   AOOO/BaOO. HOOKUP AlCM/Ball/BaPD/password

   This message is sent to BOOO but it is kept at BaOO in a temporary pending file.

2. BOOO receives the message for AOOO. If AOOO is in operation and communicating with BOOO, the message is forwarded to AOOO and step 3 executed. If not, BOOO sends a message back to BaOO. (The original message is discarded at BOOO.)

   BaOO/BOOO. UNAVAILABLE AOOO

   BaOO then looks through its pending file for any action requiring AOOO, sends appropriate messages to its terminals and cancels the entries.

3. AOOO receives a message addressed to itself, which marks it as an internal message requiring special action. The request HOOKUP sends control to an appropriate routine in AOOO. (There is an additional layer of logic throughout to take care of garbled internal messages, i.e. network errors. We will ignore this here.) This routine must do several things.

   a) It must first be determined whether communication has been established (today, that is) between AOOO and AlCM. Whether or not this can be done on demand or must be within agreed-upon schedules is a matter which must be negotiated and built into the AOOO-AlCM communication logic. If

communication has been or now is established, proceed to step 3 b). Else, AOOO sends a message back to BaOO

BaOO/AOOO. UNAVAILABLE AlCM

On receiving this message, BaOO takes actions on its pending file as in step 2, but only for requests specifically to AlCM.

b) There must be available ports on both ends of the AOOO-AlCM line and also room within AOOO to handle whatever temporary files may be required. (Note that these considerations are in addition to those in step 3 a).) If any of these requirements fail, an action as in step 3 a) must be taken but with possibly a different word, such as OVERLOAD, instead of UNAVAILABLE. The distinction is only important to the human user BaPD. If all is in order, proceed.

c) AOOO now simulates a log-in to AlCM as though from BaPD, using the userid "BaPD" and its password. (Conceivably, these might be translated to predetermined forms by AOOO but this would appear to only add confusion without any particular benefit.) A table of "hooked-up" relationships must be maintained within AOOO and an entry made for the following:

   i) Line from AlCM to AOOO against BaPD. (AlCM will receive/return messages from/for BaPD as though AOOO were the terminal.)

   ii) Ball against BaPD. (AOOO must be able to route messages from AlCM directly to the terminal where BaPD is working in case he is not at his "home base," which does not apply in the present example.)

If AlCM refuses the log-in for any reason, its messages must be sent back to Ball. In any event, a message must be sent from AOOO to BaOO confirming or denying the requested HOOKUP. If all is in order proceed.

4. BaPD now appears to be in direct communication with AlCM and may proceed in accordance with his intent and AlCM's language. In reality, however, every message from BaPD is routed Ball-BaOO-BOOO-AOOO-AlCM and all replies go back along the same route in reverse. Note that BaOO must know that all normal messages from Ball are to be addressed AlCM/BaPD and that AOOO must know that such messages go through a particular line to AlCM. Coming back, AOOO must address messages from that line as Ball/BaPD/AlCM (to/for/from).

Considerations relating to log-offs and crashes will be taken up in the last section.

CONNECTING REMOTE UNIT RECORD EQUIPMENT

It was assumed in the above example that printed output for BaPD's run on AlCM was to be routed to PRT BaOl. This leads to a new set of considerations which have not previously arisen in our examples and which do not occur in conventional networks. We proceed to analyse this situation.

Note first that AOOO is the only connection to the CHINE for AlCM. This is a great advantage for hooking up users to the SYS since any special translation of protocols, symbology, etc. need be provided only at AOOO. However, for returning voluminous information, such as printed output, it creates additional timing and forwarding problems which do not occur in conventional networks.

It must be recognized, first of all, that a SYS does not transmit output files (except those destined for an actual TER) as they are generated. The handling of output files

(also large input files, as from a RDR or TAP) is actually quite an involved process, and goes somewhat as follows.

1. As output lines (i.e. records) are generated by executing programs, they are first stored in internal buffers. When a buffer fills, it is output to a temporary file assigned to the job.
2. When the job is completed, the temporary file is moved or added to a somewhat more permanent file. This destination can often be controlled by commands in the conversational or job control language. Such commands can also cause whatever output exists at the time to be transferred, the temporary file then being started over. The normal destination--and the one of interest here--is what is usually called an output spool. Output files belonging to a certain job and residing in the output spool are the ones which, in our example, are to be transmitted to BaOl.
3. The mechanisms for manipulating files in the output spool are a part of the host system. Files are labelled for a physical destination, such at the printer (of a certain designation) in a particular location. Files are held in the output spool for some maximum length of time (such as 48 hours or sometimes up to a week) and output on demand. Scheduling of such demands may be automated for equipment at the SYS's own installation but demands must be made by telecommunication protocols for remote equipment. When the appropriate line is connected and the proper identifications given, the file is then transmitted at whatever speed the equipment is currently capable of. When finished, the file in the output spool is destroyed. (If transmission is interrupted, the entire file is usually re-transmitted on the next attempt.)

Since A000 is the only connection to A1CM, all output files from A1CM for unit record equipment attached to nodes of the CHINE must be sent by A1CM to A000. There is no difficulty in this, per se; it is as though the entire network had only one printer for A1CM. But now A000 must take on three additional chores:

1. A000 must keep track of all desired routings on the CHINE for output from A1CM;
2. A000 must recognize demands from CHINE nodes to transmit and then initiate and monitor the transmissions;
3. A000 must distribute output from A1CM through a single port to the various output devices attached to the CHINE.

If A000 can actually do all this, then there is no need for any significant amount of extra storage. A000 simply passes through lines (records) of output to go along an appropriate route just as for any other messages. The difficulty is with chore number 2 which may require some modification in the spooling software of A1CM. The reason is as follows:

> Although files in the output spool are identified by job, the spooling mechanism is more concerned with destinations. Thus when a printer is connected, its identification is by location, not by user. The spooling mechanism then proceeds to transmit all files it can find with this destination and usually in unpredicable order. For example, an aborted transmission may be put at the end of the queue, and short files may be transmitted before long ones. All this makes perfectly good sense in a conventional situation where a printer at a certain place is serving all users at that place.

To restate the difficulty in a few words: one cannot request output by job or userid, but only by destination. Since A1CM has only one destination, namely A000, for the entire CHINE, this can create intolerable storage problems at A000 (or any CON with an attached SYS).

The only modification that is actually needed is the ability for AOOO to specify different locations (such as BaOl) to AlCM but to have their output line selectable immediately prior to transmission. It is possible that the necessary protocols for this already exist but it is an extremely important matter to ascertain. If the capability does not exist, then negotiations with the organization operating the SYS must be entered into to have it provided.

Assuming the proper arrangements have been made, then we can describe the protocols necessary to cause transmission from AlCM to BaOl. It must be recognized that BaOl does not belong to either BaPD or Ball, but really to BaOO. Some instruction must be given to BaOO to start transmission from a particular SYS, not by job or userid, but for any and all output files in AlCM's output spool destined for BaOl. This is accomplished by assigning BaOl a distinctive userid, such as BaOE (mixed numeric, alphabetic). BaOO would be programmed to recognize this userid and execute a special sequence, i.e. an installed subroutine specifically for BaOl (just as AOOO has one for AlCM). Some human user must log into an actual TER and play the role of BaOE. Assuming the necessary units are in operation, the scenario would go something like this, say from TER BalO:

(BaOl turned on and readied as necessary)	
(BalO turned on by someone)	
(from BaOO)	GROUP BaOO IN OPERATION:
(at BalO)	LOGIN BaOE
(from BaOO)	PASSWORD:
(at BalO)	password typed
(at BaOO)	Connection to BaOl checked, probably a signal sent to BaOl to skip to a new page and type some start-up message. Then a prompt sent to BalO.
(at BalO)	OUTPUT AlCM.

At this point, BaOO would create a message to send to AOOO something as follows:

AOOO/BaOO. OUTPUT AlCM/BaOl/BaOE

The following actions would be almost like a HOOKUP command except that AOOO would create any necessary password and protocols for AlCM's output spool. AOOO would then proceed to forward lines (records) from AlCM to BaOO until an end-of-file signal was encountered. BaOO would transmit these to BaOl in an appropriate mode.

The userid BaOE serves another purpose. When BaPD is setting up his job on AlCM, he must designate BaOE as his output destination. This is the destination label which would be attached to his files in the output spool. Other users might also be using the same or different designations at AlCM at the same time. Furthermore, PRT BaOl could be used at a different time to print output from some other system.

The same arrangement can be used for a RDR or TAP. Another command, say INPUT, is needed for transmission into an input spool at a SYS.

In order not to tie up BalO once transmission is started, a DISCONNECT command is needed, in the same sense as used by existing interactive systems. BaOO would drop the connection to BalO but continue the transmission to BaOl as long as necessary.

LOGOFFS AND CRASHES

When BaPD is through with AlCM, he will log off the SYS using the appropriate command of its language (usually LOGOFF or LOGOUT). However, this does not log him off the network. Furthermore, many SYSs have a feature which permits a user to log off one account and log in to another without breaking the connection. (This is typically done by issuing a "LOGIN new-userid" instead of LOGOFF, or "LOGOFF HOLD" followed by LOGIN protocols.) Both situations create new problems within the CHINE.

Consider first the effect of a normal SYS LOGOFF. If user BaPD, hooked up to AlCM from Ball, types LOGOFF, AlCM will go through its usual session close-out procedure and break the connection to AOOO. It is essential that AOOO be signalled when this connection is broken. For recall that AOOO has a table of "hooked-up" relationships which includes an entry for the "BaPD/Ball/AlCM-AOOO line" relations. This entry must be deleted, and a message must be sent to BaOO. For BaOO has a record of the hookup which causes messages from Ball to be routed to AlCM. This entry must also be deleted.

AOOO must also be notified if AlCM crashes, for the same reasons. It would be nice to know the difference so BaPD could be notified in a more meaningful manner, but this is not essential. When BaOO deletes its hookup entry, it should send a message to Ball reporting this to the user BaPD. If he did not expect it, he will know the system has crashed.

The possibility that a user can switch accounts within a SYS is equivalent to saying that one userid on the network can have multiple userid's in a SYS. This cannot be permitted with the scheme discussed in earlier sections. If one user actually has two or more accounts (which might be desirable in some situations), then he should identify himself to the network (and not just to a SYS) with the pseudonym he is currently using. There are still two disadvantages to this approach:

1. It increases the number of userid's at a GRP unnecessarily. There seems no reason why one user needs two names for purely network functions. This is not to say that one person might not, on occasion, use another person's name.
2. If a SYS permits a user to switch userid's without notifying its connecting CON, and a user does this, there is no direct way to detect it. However, message addressing will fail within the CHINE.

A possible way out of this difficulty is to require a second userid in a HOOKUP command, i.e. the person's network userid and his SYS userid may be different. If he then wants to change to a different SYS userid, he must issue a network command to effect this. If he persists in trying to switch within the SYS, he will not be able to continue and perhaps one or two experiences will discipline him sufficiently. However, this can leave incorrect entries in network tables. The common expedient for this trouble is a "dead-time" limit. For example, if a route has no action for ten or fifteen minutes, the connections are broken peremptorily. Many SYSs have this feature built in. In a CHINE, only the CONs would need to have this logic with respect to SYSs if the latter did not provide it.

It was noted in Part II that messages to other users must be flagged in some way to the GRP. This is not strictly necessary if a user has not issued a HOOKUP command, but, if he has, all type-ins must be considered as part of the interactive conversation with the SYS. It is dangerous to use a special character for such a flag since, among any small set of systems, virtually all possible special characters have some meaning. Since terminals are always equipped with some kind of attention or break key (ATTN button), this is the only safe signal to use. The SYSs themselves make use of the ATTN button but this creates no conflict since the GRP can translate a flagged type-in as an ATTN for a SYS and forward the appropriate signal.

Consequently, to log off the network, the ATTN button can be used to get the GRPs attention. Then the command LOGOFF will be sufficient to cause the user to be logged off the network. The GRP must make a number of rather obvious checks before terminating all action with respect to the user. For example, the user may have been hooked to SYS and neglected to log off the SYS before logging off the network. The GRP has a record of the hookup and must itself issue

the LOGOFF command to the SYS. There may be similar actions to be taken with respect to other network services which have not yet been discussed. A great deal of accounting information will, of course, have to be recorded in appropriate files. This is an extensive subject which will not be taken up in this paper.

It is clear that a great many pieces of equipment may fail, causing crashes of various kinds. If a SYS crashes, it is mainly an inconvenience to the user, just as on existing networks. However, if a CON crashes or the telecommunication lines between two CONs, or between a CON and GRP, fail, the situation is much more complicated.

Suppose BaPD is using AlCM as before and the line between CON A and CON B goes down (either the line, a modem, or whatever). One of two situations occurs. One possibility is that BaPD types in at Ball and the message goes to BaOO and then to BOOO. BOOO either tries to send it to AOOO and cannot or already knows the line is down. BOOO may send a message back announcing AOOO unavailable, but this goes to Ball (the return address), not BaOO. BaPD is notified but is helpless to do anything about it. The other possibility is that AlCM sent a message to BaPD. AOOO receives it but cannot forward it to BOOO. Should AOOO log off BaPD? Or should it just hold, in hopes the line will be restored? But then, dead-time limit may run out. In any event, it cannot notify BaOO that communication is lost.

There is a whole class of such situations which will have to be thought through and provided for. In extreme cases, resort to personal telephone calls may be necessary to straighten out the network.