# PROPOSAL FOR AN INTERNATIONAL END TO END PROTOCOL

V. Cerf, Stanford University (USA)
A. McKenzie, Bolt Beranek and Newman Inc. (USA)
R. Scantlebury, National Physical Laboratory (UK)
H. Zimmermann, IRIA (France)

## INTRODUCTION

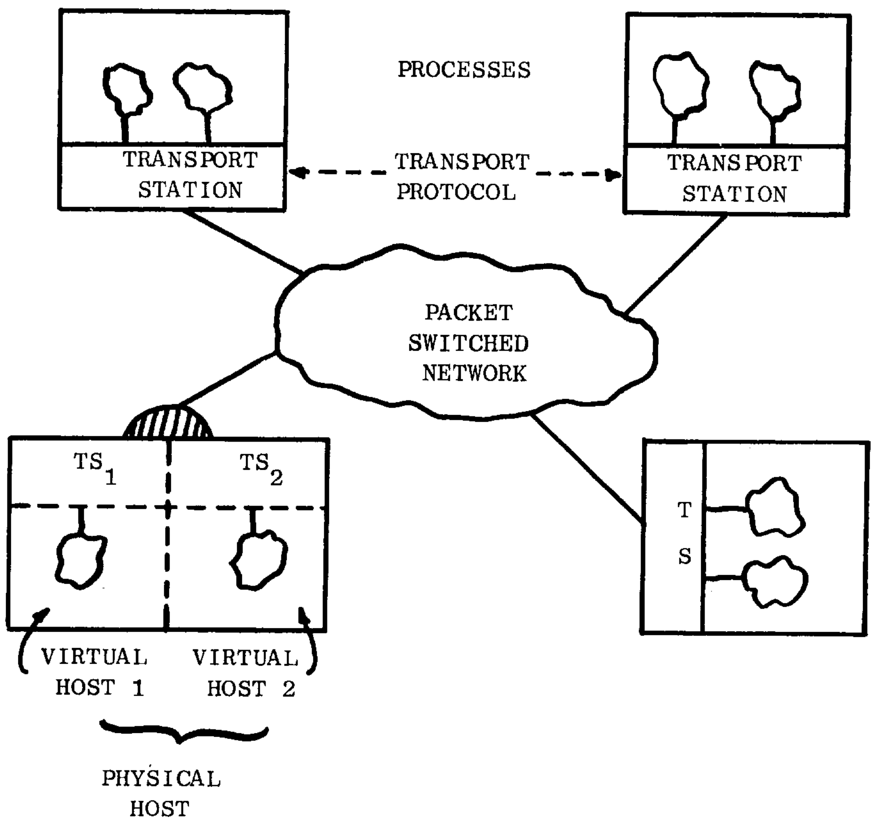
Computer networks are being developed or planned all over the world, using packet switching technology as a communication medium between computers and/or terminals. End-to-end protocols (often called "Host-Host" protocols) are installed on top of the packet switching service to provide users with an interprocess communication facility.

Since users of one network do not want to be tied to a single manufacturer, they need an end-to-end protocol that can be used on several types of computer. Since networks will sooner or later be interconnected, there is a strong need for a standard end-to-end protocol that could be used for interprocess communication.

Network interconnection issues have been especially addressed within Working Group 6.1 of the International Federation of Information Processing (IFIP), and several proposals have been made and discussed within that group [INWG61, INWG72, INWG74, CEKA74, INWP16, INWP20, INWP21]. A convergence of ideas has gradually emerged, leading to the present common proposal for a standard end-to-end protocol.

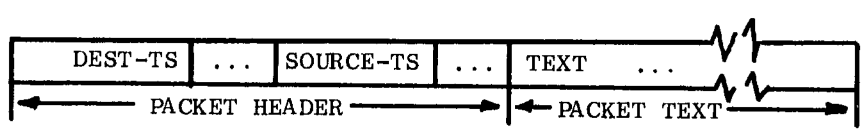
## NETWORK ARCHITECTURE AND TERMINOLOGY

In a network, since computers as well as terminals can be grouped and organized in various ways (e.g. one computer may contain several systems or one system can be made of several computers), we introduce the notion of a VIRTUAL HOST. A virtual host is a collection of resources appearing as a single entity to the subnetwork that transfers packets sent by a source virtual host to a destination virtual host. Usually, several entities such as jobs, processes, terminals, and so on, (processes for short) will be sharing the virtual host resources (i.e., most systems are multiprogrammed). Those processes have to share the interface with the subnetwork. Moreover, packet switching will be considered too basic for some of them. Therefore, each virtual host will contain a TRANSPORT STATION (TS) responsible for multiplexing the interface to the subnetwork and "adding value" to the packet switching service. Transport stations cooperate according to a TRANSPORT PROTOCOL (Fig. 1).




TRANSPORT ARCHITECTURE

Figure 1




SIMPLIFIED PACKET FORMAT

Figure 2

## PORTS and ASSOCIATIONS

The packet switching metwork carries packets from a source TS to a destination TS, both of which are identified within the packet header by their addresses (Fig. 2).

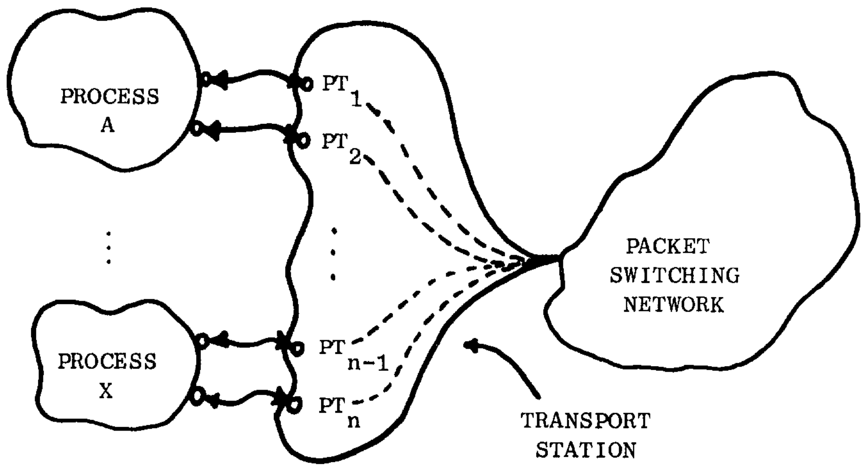
The final users of the information carried within packets are SUBSCRIBERS whose identification must also be associated with the pieces of information in transit (note that our use of the term "subscriber" applies to a customer of the transport station, rather than to a direct customer of the packet switching network). Since each system has its own way of naming (identifying) its subscribers' processes, resources, and I/O streams, we introduce an interprocess communication name space made of PORTS (PT's). As seen from the communications side, a transport station is just a collection of ports (Fig. 3).

Ports are identified by a 16 bit port identifier. In a TS, port identifiers are locally associated with internal names for communication purposes. That mapping will have to be done dynamically in some cases, but it is recommended that it be done statically when possible, to avoid the problems of synchronizing the dynamic allocation of names in a distributed environment.

A transport station subscriber is identified by the high order bits of the port identifier. The low order bits of the port identifier represent the PORT NUMBER within the subscriber. It is recommended that in any implementation, the port number be at least 4 bits, but we do not rule out a larger size (see Fig. 4). Furthermore, we make no implication that the mapping from subscriber identifi to process or job be uniform throughout all TS implementations. Nor do we insist that use be made, by higher level protocols, of the proposed adjacency of port numbers within a subscriber. However, this adjacency can be a convenience for some higher level protocols.

A subset of the port identifiers (0-255, for example) should be set aside for use as "well-known ports" (mainly devoted to standard services such as loggers, file transfer service, host status reporting, remote job entry, etc). Aside from these recommendations, the association of port identifiers with subscriber I/O channels is left to the discretion of the protocol implementer.

For the sake of simplicity, each packet contains only information going from a single source port to a single destination port. According to the DATAGRAM format proposed by IFIP WG 6.1 to CCITT (INWG83), port numbers are placed in the low order bits of the source and destination address fields (Fig. 5).




A TS IS A COLLECTION OF PT'S

Figure 3

| 12 | 4 |
|---|---|
| SUBSCRIBER | PORT NUMBER |

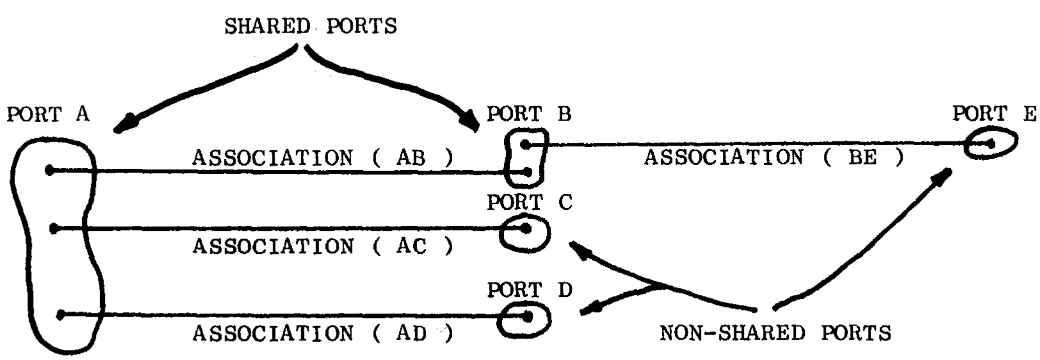
PORT IDENTIFIER
Figure 4

| OCTET | 0 1 2 3 | 4 5 |
|---|---|---|
| | SOURCE OR DESTINATION TS IDENTIFIER | PORT IDENTIFIER |

COMPLETE PORT ADDRESS FORMAT
Figure 5

The transport station can now be considered as an extension of the packet switching network up to the port level, and a set of port-to-port mechanisms providing the value added services. Ports which wish to communicate with each other must somehow become "associated." The ASSOCIATION of a pair of ports is controlled by the port-to-port protocol. An association is full-duplex, and is identified by the combination of a source and destination address. Since port addresses (including a TS identifier) constitute a minimum network-wide name space, they will advantageously be used as a network-wide naming scheme for functions, services or resources (e.g. a time sharing system can be identified with a well known port, or a terminal can be identified with a port). 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 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 philosophy can be applied to ports. If a port is the access means to a sharable resource (e.g. a time sharing system), that port should be sharable, i.e. able to handle several associations at the same time. The naming conventions for an association make it possible for any port to be shared by several associations, as shown in Fig. 6.




SHARED/NON-SHARED PORTS

Figure 6

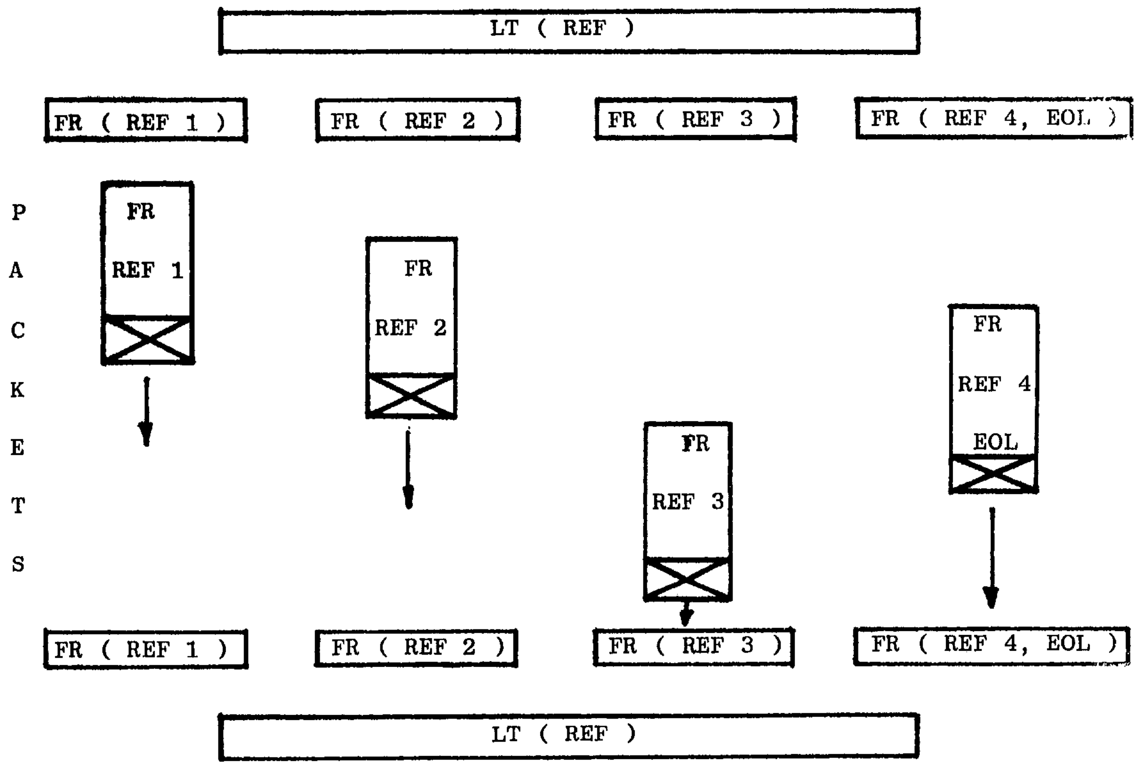
## LETTERS

The transport protocol provides for transfer of letters (LT) from one port to another within the context of an association. A letter is a variable length piece of information with a maximum size of over 25 thousand octets of data. The idea is that almost any physical record can be placed in a letter, thus avoiding the fragmentation of meaningful data at the subscriber level. Thus, the letter is given as a whole by the sending process, fragmented if necessary into several packets by the transport station, reassembled upon arrival, and then delivered as a whole to the receiving process. Thus, in the transport station, buffer management is handled at the letter level. Since error control and flow control are tied to buffer management, they will also be introduced at the letter level.

When necessary, the letter text is divided into fixed length, 216 octet, FRAGMENTS (FR), except for the last one, each fragment being sent in one internetwork packet with proper control information (*). Each letter gets a 16 bit reference number (MY-REF) which is unique within the association, thus avoiding confusion between fragments of different letters sent on the same association. Fragments are numbered (FR-NB) within the letter and an END-OF-LETTER (EOL) flag indicates the last fragment of the letter. Both FR-NB (7 bits) and the EOL (1 bit) go in one octet. Upon arrival, fragments are reassembled into a copy of the letter (see Fig. 7). The pre-processing function could be performed in a front-end processor, thus presenting a letter switching service to the main processor.

---------

(*) Chosen so that the internetwork header and text of an internetwork packet can be embedded in the text of a local network's packet (assumed to be able to carry 255 octets of text). This requirement may be relaxed if all constituent networks adopt a standard datagram header. Furthermore, 216 octets = 1728 bits, which is a multiple of almost every useful machine word length (8, 12, 16, 18, 24, 32, 36, 64).




FRAGMENTATION AND REASSEMBLY

Figure 7

## OPERATION OF ASSOCIATIONS

An association can be operated simultaneously in LIAISON and LETTERGRAM mode. In lettergram mode, letters are sent independently of each other with the possibility for the sender to request an acknowledgement. The users are responsible for having their ports activated prior to exchanging letters. Lettergram mode provides a letter switching service that can be compared to the postal system.

In liaison mode, initialization commands are exchanged prior to transmitting any letter. That initialization is intended

(1) to make sure that both ends of the liaison are active,

(2) to agree on the set of services to be put into operation

(3) to initialize parameters coherently.

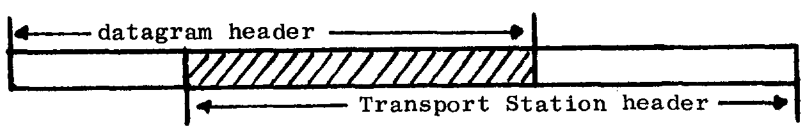
Various options are available, including, in particular, error and flow control. A session in liaison mode ends with the exchange of termination commands.

## TRANSPORT COMMANDS

A DATAGRAM is the unit of exchange between a TS and a packet switching network (PSN). We have assumed a standard D1 datagram format as proposed in IFIP WG 6.1's earlier contribution to CCITT [INWG83, INWG84], but this is merely for concreteness. Datagrams are used to carry transport commands between transport stations. The datagram header contains the complete source and destination port addresses of an association; a PSN typically uses only the high order part (TS address) for routing to the destination TS.

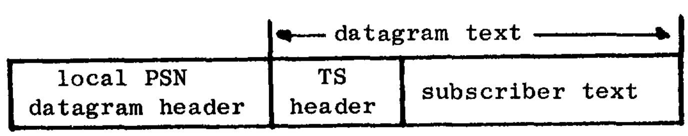
In our earlier proposal, we suggested that the TS level header could overlap (share fields) with the datagram header. In Fig. 8, we illustrate the overlap of the D1 datagram header with the TS end-to-end protocol header. Note that the proposed TS header is almost identical with the E1 format proposed in [INWG83] but has slightly longer field widths. Figure 10 illustrates in detail the datagram and TS header formats.

It may happen that a standard datagram format cannot be established before internetworking is effected between PSN's. If this is the case, then a slightly different picture emerges, in which there is no overlap. Figure 9 illustrates this case.



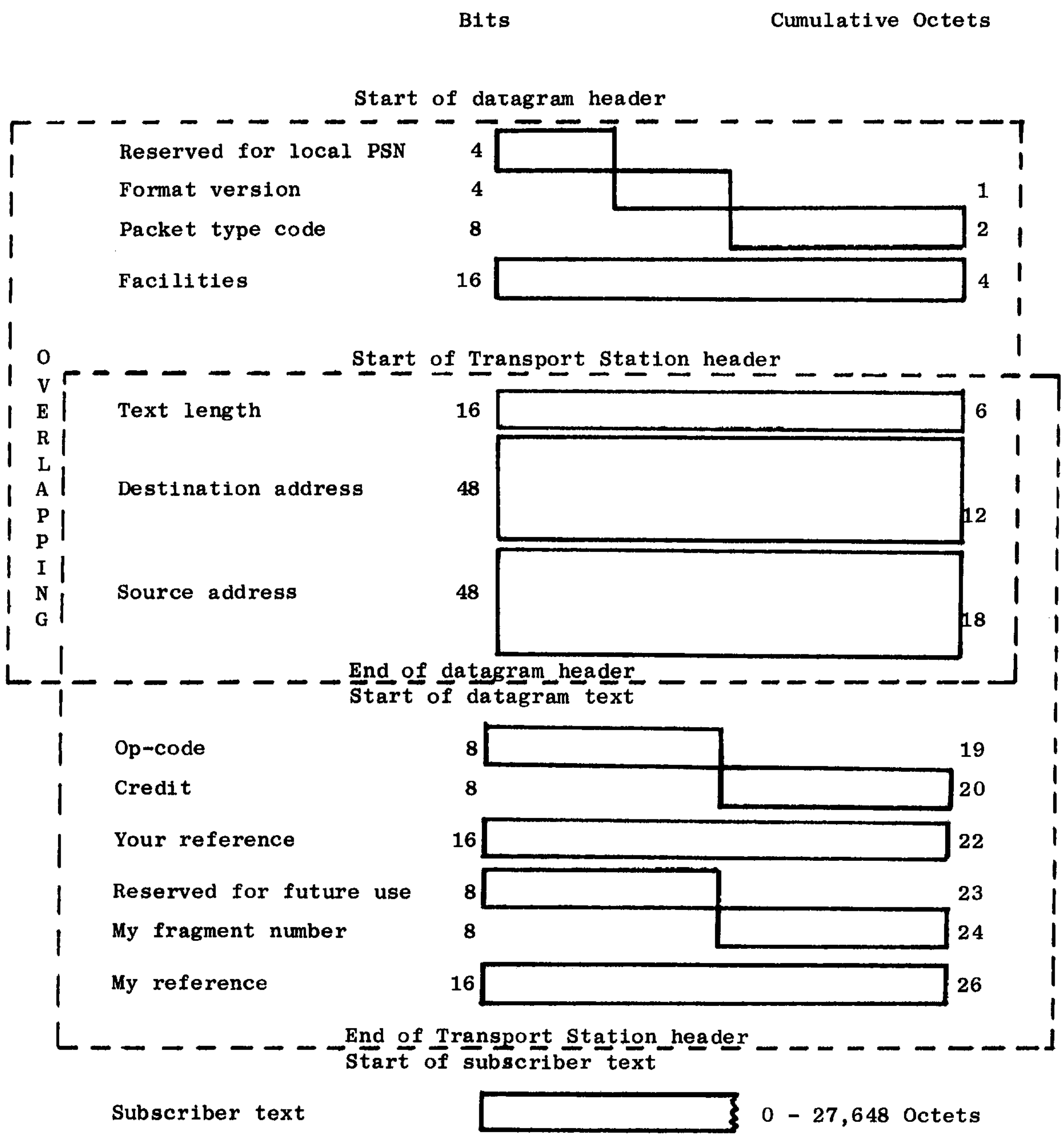

Overlapping datagram and TS header

Figure 8




Embedding of TS header and subscriber text within a local PSN datagram

Figure 9




Format of the datagram with
Transport Station header

Figure 10

Referring to Fig. 10, the meaning of the various header fields is given below.

Local PSN - 4 bits

This field allows PSN's to operate easily on their own local formats, which they may already have, and also on internetwork formats.

Format version - 4 bits

This field allows a variety of other formats to be recognized by the PSN, e.g. for future developments, or services other than datagrams.

Packet type code - 8 bits

This field allows recognition of a variety of specific information packets carried within a datagram, e.g. calls, diagnostics, or network services.

Facilities - 16 bits

This field is to be divided into many small subfields, both for the local PSN and standard internetwork services, such as: measurements, diagnostic aids, special routing, accounting, etc.

Text length - 16 bits

This field indicates the length of text carried in the packet. When used for datagrams (CCITT standard), only 8 bits are significant as the maximum text length is 255 octets. However, some PSN's accept packets of larger size, and it is predictable that future technologies will allow a substantial increase in packet size. In the end-to-end protocol described here, only 14 bits could be used as subscriber text length, as subscriber text is limited to one latter that has a maximum size of 27,648 octets (i.e. 128 fragments of 216 octets each).

Address - 96 bits

The source and destination addresses form two separate groups of 48 bits each. The high order 32 bits of each address are reserved for standard internetwork TS address

(including PSN identifier). The low order 16 bits of each address are reserved for a port identifier.

Op-code - 8 bits

This field identifies actions to be performed by the receiver of the command (see Fig. 11). Bit 0, called the R bit, is used to request (R = 1) the receiver of the command to send back an acknowledgement. In lettergram mode, and in liaison mode wthout error control, the acknowledgement is requested for the letter being sent. In liaison mode with error control, the acknowledgement will indicate the last letter of the complete seqqence of letters correctly received. In liaison mode, the acknowledgement can be piggy-backed in an LI-LT command.

Bits 1 and 2 are reserved for future use

Bit 3 indicates the association mode (1: lettergram, 0: liaison)

Bits 4-7 identify the command.

Credit - 8 bits

This field is used to control the traffic flow of letters between source and destination ports, when operating in liaison mode. CREDITs refer to permission to transmit letters beyond the one last acknowledged, and are granted by the receiving TS to the sending TS on a liaison-by-liaison basis. The receiving TS is free to take back credit by sending a smaller credit number, since the sum of the last acknowledged letter reference number and the latest credit determines the maximum reference number which can be emitted by the sending TS.

Fragment number (FR-NB) - 8 bits

In an LI-LT command, the seven low order bits of the FR-NB field give the number associated with the first (*) fragment of letter text appearing in the subscriber text field of a packet. The high order bit of FR-NB is called the End-of-Letter (EOL) bit. It is used to indicate that the last (*) fragment of the letter text in the packet is the last fragment of the letter. In other commands, the FR-NB field is used to augment the OP-CODE field (see Fig. 12).

--------

(*) The first fragment in a packet is the only fragment, unless a liaison mode association is operating with the multi-fragment packet option.

| Bit / Mnemonic | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Meaning |
|---|---|---|---|---|---|---|---|---|---|
| LG-LT | R | 0 | 0 | 1 | 0 | 0 | 0 | 0 | Used to send fragments of letter in lettergram mode |
| LG-ACK | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | Used to send acknowledgement of a letter in lettergram mode |
| LG-NACK | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | Used to send negative acknowledgement in lettergram mode |
| LI-LT | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Used to send fragments of a letter and reverse acknowledgements in liaison mode |
| LI-ACK | R | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Used to send acknowledgement in liaison mode |
| LI-INIT | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | Used to initialise liaison mode |
| LI-TERM | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | Used to terminate liaison mode |
| LI-PURG | R | 0 | 0 | 0 | 0 | 1 | 0 | 1 | Used to pass an interrupt and purge possibly remaining letters in liaison mode |
| LI-ERR | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | Used to signal an error in liaison mode |
| LI-NACK | R | 0 | 0 | 0 | 0 | 0 | 1 | 0 | Used to send negative acknowledgements in liaison mode |

Bits 4–7: Identification of the command

Bit 3: Lettergram/Liaison mode

Bits 1–2: Reserved for future use

Bit 0: Request acknowledgement

Transport Command Op-Codes

Figure 11

| Command | Use of the Fragment Number Field |
| --- | --- |
| LG-LT | EOL, FR-NB |
| LI-LT | EOL, FR-NB |
| LI-INIT | Set of Services required |
| all other commands | UNUSED |

Use of Fragment Number Field

Figure 12

| Command | Use of Your Reference Field |
| --- | --- |
| LG-LT | NOT USED |
| LI-INIT | Maximum size of letter to be received |
| LI-ERR | Reference of the letter in error |
| LG-NACK | Not yet fully defined |
| LI-NACK | Not yet fully defined |

Use of Your Reference Field

Figure 13

Your Reference (YR-REF) - 16 bits

This field indicates the reference of the letter being acknowledged, except as shown in Fig. 13.

My Reference (MY-REF) - 16 bits

This field indicates the reference of the last letter sent or the letter being sent (when subscriber text is present, e.g. in a LI-LT or LG-LT command). This field is not used

in the LG-ACK and LI-ACK commands.

The format of the various transport commands is summarized in Fig. 14.

| | CRD-NB | YF-REF | | FR-NB | MY-REF |
|---|---|---|---|---|---|
| OP-COD | CREDIT NUMBER | YOUR REFERENCE | RESERVED FOR FUTURE USE | | MY REFERENCE |
| LG-LT | | | | FR-NB | MY-REF |
| LG-ACK | | YR-REF | | | |
| LG-NACK | | YR-REF | | | |
| LI-LT | CRD-NB | YR-REF | | FR-NB | MY-REF |
| LI-ACK | CRD-NB | YR-REF | | | |
| LI-INIT | CRD-NB | MAX-LT-LG (Octets) | | SERVICES | MY-REF |
| LI-TERM | TERM CODE | YR-REF | | | MY-REF |
| LI-PURG | CRD-NB | YR-REF | | | MY-REF |
| LI-ERR | ERR-CODE | YR-REF | | | Last value sent as MY-REF |
| LI-NACK | CRD-NB | YR-REF | | | |

FORMAT OF THE TRANSPORT COMMANDS

Figure 14

## ASSOCIATION OPERATION IN LETTERGRAM MODE

In lettergram mode, the set of commands is restricted to LG-LT and LG-ACK (and possibly LG-NACK), allowing the exchange of letters with possible acknowledgement.

The sender is responsible for giving to each letter a MY-REF unique within the association (and independent of references used by the same association operating in liaison mode). Since (1) the reference field is 16 bits long, (2) it is recommended by IFIP WG 6.1 that the maximum lifetime of a datagram be limited to a short time (e.g. 30 seconds [INWG82]), and (3) lettergram mode is not intended for high bandwidth, there should be no problem to generate these unique MY-REF's. One counter for the whole transport station might even suffice.

If the R bit of the LG-LT command is set to one, the receiver is requested to send back an LG-ACK with the YR-REF equal to the MY-REF of the received letter, as soon as the whole letter has been received. A timeout allows the sender to detect possible loss of the letter (or of the acknowledgement) and to report this to the subscriber. If the destination port does not exist, or is not active, or if no buffer is available, no positive acknowledgement will be sent back and the result at the sender's end will be equivalent to the loss of the letter.

Possible recovery is left to the subscriber who can send the lettergram again. We do not rule out the possibility of an LG-ERR command which could make more explicit an error detected at the receiving TS, but this is left for further study. Since packets may be lost, reassembly at the receiver is protected by a timeout associated with each letter. This timeout is set when receiving the first (in time) fragment, it is reset on receipt of each fragment, and finally turned off when all fragments of the letter have been received. If the timeout occurs, reassembly is aborted and the letter considered erroneous (if error control is used, the error will hopefully be recovered).

## ASSOCIATION OPERATION IN LIAISON MODE

In liaison mode, initialization must take place before any letter is sent, and termination ends the session in liaison mode.

Initialization and termination are done using a simple symmetric rendez-vous scheme (see Fig. 15). Initialization succeeds if the LI-INIT commands exchanged are compatible, i.e. request the same set of options with compatible parameters (if any). Transient states are

protected by time-outs. Unexpected commands are ignored without action.

Various options can be put into operation in liaison mode. The options requested are indicated in the service field of the LI-INIT command, by setting the appropriate bits (0: off, 1: on), as is illustrated in Fig. 16.

In liaison mode, letters are sequentially numbered, starting initially with the MY-REF following the one sent in LI-INIT. Since the reference field is large (16 bits), references are cyclicly re-used, without any problem of uniqueness.

The LI-PURGE command is used to pass an "interrupt" signal from one subscriber to another over a liaison. The exact interpretation of an "interrupt" signal will be dependent upon the receiving subscriber, but should take the general form os suspending or terminating current processing and beginning to scan incoming letters sent after the interrupt. The LI-PURGE contains the field MY-REF, assigned as with a LI-LT, although the FR-NB field is not used, since there is no text. The sending TS must transmit "interrupt" information even if the flow control option (if in use) would prohibit transmission of a LI-LT with this reference number. Similarly, the receiving TS must accept (and acknowledge) an LI-PURGE immediately, even if there are previous unreceived letters, and this has the potential (if error control is in use) of losing previous letters since they would be implicitly acknowledged by the acknowledgement of the LI-PURGE. Thus, the LI-PURGE command has the following effects:

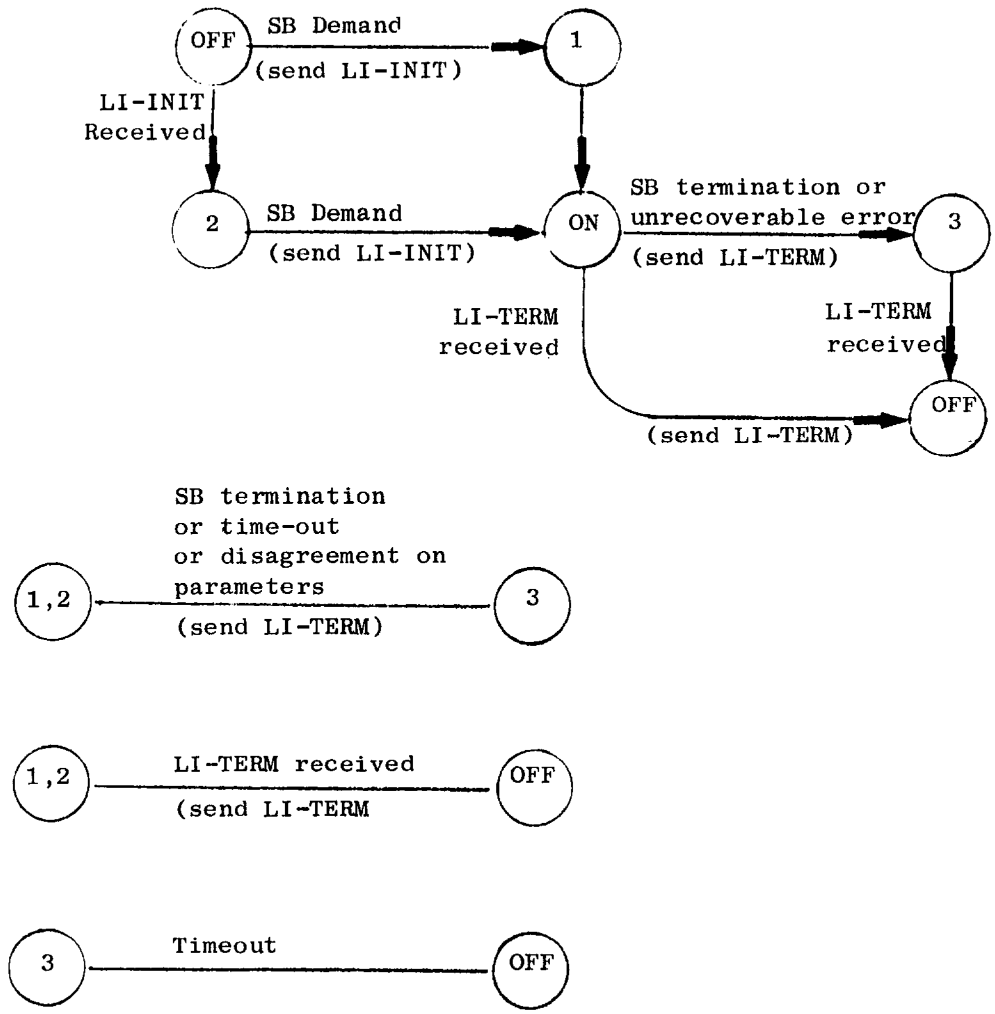
(1) Pass an "interrupt" signal to the receiving process

(2) Purge unprocessed letters up to the point of the LI-PURGE command.

| Bit Number | Optional Service |
|---|---|
| 0 | Error Control on letters |
| 1 | Flow Control on letters |
| 2 | Multifragment packets |
| 3 | Checksums on letters |
| 4 - 7 | Reserved for future use |

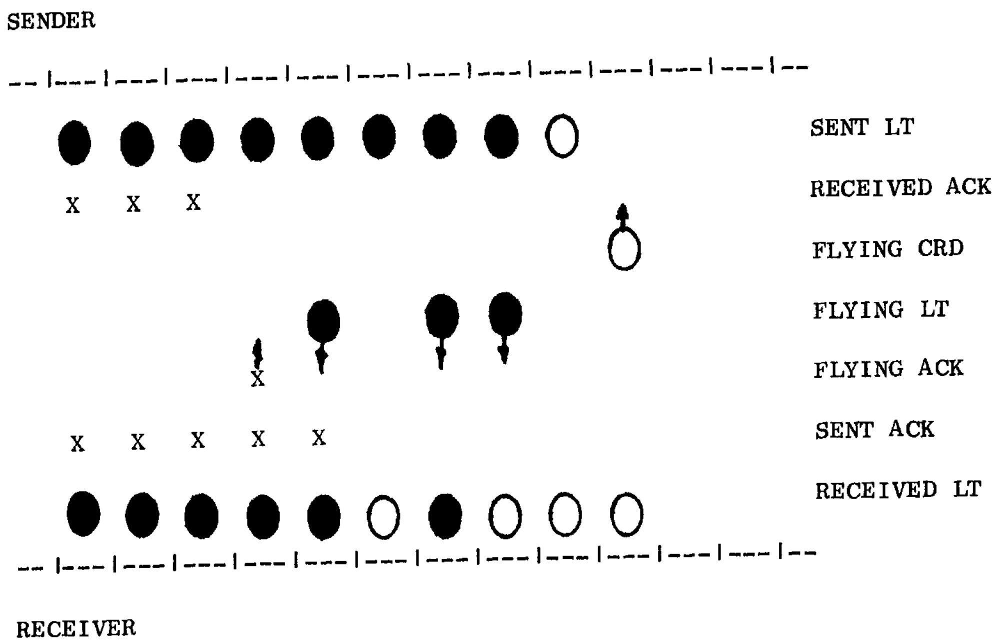
Service Field of the LI-INIT command

Figure 16




Initialisation and Termination
of Liaison Mode

Figure 15




Error Control on letters

Figure 17

## Basic Service

If no optional service is on, letters are exchanged in basic service mode. The basic service is similar to the lettergram mode, with the following differences:

(1) Initialization makes sure that both ends are (were!) active and willing to communicate.

(2) A purge and interrupt function is available.

## Error Control

If error control is on, all letters must finally be acknowledged, i.e. (1) if a letter is missing, the YR-REF will not progress, and (2) one single LI-ACK can acknowledge several letters. The sender expects to receive acknowledgements within a maximum delay after the last fragment of a letter has been sent. If the acknowledgement is not received, all unacknowledged letters will be assumed lost and sent again, starting with the first unacknowledged letter. Acknowledgement will again be expected. If acknowledgement is not received then, the process will be repeated. If a letter has been sent "N" times without success, the sending transport station will report this condition to the subscriber. This strategy is simple, but may not be optimal in some cases. Alternative retransmission strategies, possibly involving negative acknowledgements could be taken into account during further studies.

## Flow Control

This option allows flow control (for buffer management) at letter level, on top of Fragmentation-Reassembly. When in use on a liaison, it is associated with error control and performed in both directions as follows:

- At initialization of flow control, both ends of the liaison agree on the maximum size of letters to be sent in each direction (the size may be different for each direction), as indicated in the MAX-LT-LG field of the LI-INIT command.

- The receiving end allocates credits (CRD) to the sending end. One credit represents the permission to send one letter. Each allocation is associated with an acknowledgement (YR-REF) within a LI-LT or LI-ACK command. The 8 bit credit number parameter (CRD-NB) means: "you may send letters with references going from (YR-REF) + 1 up to (YR-REF) + (CRD-NB)" unless (CRD-NB) = 0 which means: "don't send any letter."

- CRD-NB = 255 means: "you may send as many letters as you want."

- The sending transport station restricts itself to this upper limit that progresses under receiver control. This limit is updated as LI-ACK or LI-LT packets are received.

Multi-fragment Packet

This option is intended to make communication possible between networks with different packet sizes. The nimimum size of the datagram text (as proposed by CCITT) is 255 octets. The fragment size (216 octets) has been chosen here to go with the transport header in the CCITT datagram. However, as new technologies are introduced, networks may carry packets much larger than datagrams. The multi-fragment packet option, when it is on, allows each end to put several consecutive fragments of the same letter in the subscriber text field of the LI-LT command, the only limit being packet size. The FR-NB field gives the number of the first fragment in the subscriber text and the EOL bit indicates whether the last fragment in the subscriber text is the last one of the letter.

As the packet passes through a gateway between two networks which use different packet sizes, it may be necessary for the gateway to divide it into several smaller packets with properly reconstructed transport headers.

Checksum on Letters

A checksum is associated with each letter. It is placed in the last octets of the letter and used to detect possible errors in the contents of the letter. This option will be precisely defined at a later time.

## USER INTERFACE

A variety of user interfaces can be provided. We just indicate a partial set of primitives that could be offered as a Network Access Method.

OPEN-PT:
Activate a port.

RECV-LG:
Receive a letter from any distant port in lettergram mode.

SEND-LG:
Send a letter in lettergram mode.

CLSE-PT:
Deactivate a port.

OPEN-LI:
Initialize a liaison

RECV-LI:
Receive a letter in liaison mode.

SEND-LI:
Send a letter in liaison mode.

CLSE-LI:
Terminate the liaison.

## BIBLIOGRAPHY

CEKA74 - V. Cerf, R. Kahn, "A Protocol for Packet Network Interconnection," IEEE Transactions on Communications, May 1974.

INWG60 - L. Pouzin, "A Proposal for Interconnecting Packet Switching Networks," March 1974, IFIP W.G. 6.1 General Note 60.

INWG61 - H. Zimmermann, M. Elie, "Transport Protocol. Standard Host-Host Protocol for Heterogeneous Computer Networks," IFIP W.G. 6.1 General Note 61.

INWG72 - V. Cerf, Y. Dalal, C. Sunshine, "Specification of Internetwork Transmission Control Program (revised)," IFIP W.G. 6.1 General Note 72, December 1974.

INWG73 - V. Cerf, "Minutes of the Stockholm meeting of IFIP W.G. 6.1," IFIP W.G. 6.1 General Note 73, August 1974.

INWG76 - A. McKenzie, "Comments on the document 'Contribution on packet formats in public data networks' contained in INWG 55," IFIP W.G. 6.1 General Note 76, February 1975.

INWG79 - L. Pouzin, "Standards in Data Communication and Computer Networks," IFIP W.G. 6.1 General Note 79, March 1975.

INWG82 - IFIP W.G. 6.1, "Timing Parameters of Packet Switched Data Networks - User Implications," IFIP W.G. 6.1 General Note 82, May 1975 (Submitted to CCITT as Delayed Contribution COM VII D68).

INWG83 - IFIP W.G. 6.1, "Basic Message Format for Inter-Network Communication," IFIP W.G. 6.1 General Note 83, May 1975 (Submitted to CCITT as Delayed Contribution COM VII D69).

INWG84 - IFIP W.G. 6.1, "Data Communications Standards," IFIP W.G. 6.1 General Note 84, May 1975 (Submitted to CCITT as Delayed Contribution COM VII D70).

INWP10 - A. Belloni, M. Bozetti, G. LeMoli, "Routing and Internetworking," IFIP W.G. 6.1 Protocol Note 10, August 1974.

INWP11 - A. Belloni, M. Bozetti, G. LeMoli, "Remarks on the D-Format," IFIP W.G. 6.1 Protocol Note 11, November 1974.

INWP16 - A. McKenzie, "Proposed Revision of D-Format," IFIP W.G. 6.1 Protocol Note 16, May 1975.

INWP19 - Rapporteur on Packet Switching, "Proposals for Recommendations," CCITT Rapporteur Group on Question 1/VII - Point C, March 1975.

INWP21 - G. LeMoli, "A Proposal for Fragmenting Packets in Internetworking," IFIP W.G. 6.1 Protocol Note 21, April 1975.

INWX3 - D. Lloyd, M. Galland, P. Kirstein, "Aim and Objectives of Internetwork Experiments," IFIP W.G. 6.1 Experiment Note 3, January 1975.

# APPENDIX A
## Topics for Further Study

Several elements of this end-to-end protocol appear to require further study before being completely defined. This appendix briefly mentions some of the issues to be resolved during the course of such definition.

1. An "interrupt" facility which does not result in the potential loss of data.

The end-to-end protocol defines a command called LIAISON-PURGE which is intended to convey an "interrupt" condition over a liaison in a reliable (retransmitted until acknowledged, and duplicates detected) manner. However, the LIAISON-PURGE is carried within the sequence number space of the liaison (for reliability) but it is not constrained by the flow-control mechanism (i.e. the credit mechanism) since it may be the lack of credits which makes the transmission of an "interrupt" important. Acknowledgement of the interrupt implicitly acknowledges an arbitrary number of intervening letters which the receiver may not be able to process after being "interrupted," so we have defined this command to purge the liaison data stream of all unprocessed data which precedes it.

It has been suggested, however, that an "interrpt" signal is required which is both reliably transmitted and which does not cause the data stream to be purged. In an earlier proposal, Zimmermann and Elie suggested a TELEGRAM to carry "interrupts," but the telegrams were not protected by either a duplicate detection or a retransmission mechanism. Further, there was no solution to the potential problem of TELEGRAMS arriving after the point in the data stream to which they pertained had been processedby the receiver.

During the preparation of this document, we have had extensive discussion of possible changes to the TELEGRAM concept, but were unable to find a solution which was both relible and straightforward, and did not require the possible purging of data. We also discussed several other possibilities and were unable to find a general solution, although we did note that a subscriber process could create a second liaison for reliable transport of "interrupt" information. This, of course, would require some method of synchronization between the "interrupt" signals transmitted over one liaison and the data transmitted over the other

liaison. A standard method for accomplishing this synchronization is to insert some unique data element (a "data mark") in the data stream; this can be done by the subscriber, but it is difficult for the transport station to do this without loss of data transparency. Thus, we have been unable to specify a satisfactory mechanism which is:

- Straightforward
- Reliable
- Does not result in potential data loss
- Allows synchronization of "interrupts" with the datastream

and we therefore suggest that this topic should be studied further.

2. Use of LIAISON-NACK

Recognizing the desire of many designers for a greater level of efficiency than may be provided by an ACK + timeout/retransmission strategy, we have defined a LIAISON-NACK code within the command set, although we have not completely defined the parameter, or the use of this command. We believe that the LIAISON-NACK should be defined in such a way as to permit a receiver to negatively-acknowledge selected fragments, and for a sender to retransmit selected fragments. Since the LIAISON-NACK is intended to increase efficiency of retransmission, we would never expect a receiver to be obliged to send a NACK, nor would a sender ever be obliged to process a NACK. Since WG 6.1 has not discussed negative acknowledgement strategies previously, we believe a specific proposal in this document would be premature.

3. Fragment acknowledgement/total fragment count

Some end-to-end protocol designers have argued that a fragment-based, (rather than an exclusively letter-based) acknowledgement strategy would result in efficiencies within a sending transport station, since storage buffers would be released more quickly, and would also result in more efficient use of the communications services proveded by the network. In addition, some end-to-end protocol designers have argued that each letter should contain a could of the total number of fragments contained within it (possibly replicating this information in each fragment); it is believed that this would result in efficiencies within a receiving transport station, since buffers could be assigned more flexibly and new credits could be given out more rapidly.

we have not chosen to incorporate either of these concepts in the specification, but we agree that either or bith concepts might prove desirable. We recommend further study and we note that some experiments might make use of the currently unassigned octet in the transport header to test these ideas.

## 4. Checksum option for liaisons

We note that we have not specified the checksum algorithm to be used when the checksum option is chosen. On the one hand, a polynomial checksum provides for the detection of a wide variety of error conditions, but is likely to be costly to compute if done by software (some table look-up methods may reduce this cost). On the other hand, an additive type of checksum, which may be easy to compute, provides a smaller measure of protection. In INWG General Note 74, McKenzie proposed use of a CCITT-recommended 16 bit ploynomial checksum for this option, and in INWG Protocol Note 20, Zimmermann requested the specification of some simpler option. Since none of the authors of this specification is an expert in the analysis of the strengths and weaknesses of various checksum algorithms, we have decided to defer the specification of the exact algorithm for further study.

## 5. Use of lettergrams

The specifications for lettergrams are intentionally made rather simple, and are intended to be a similar as possible to datagrams, although much greater in length. In particular, we have only completely specified the LG-LT (lettergram-letter) and the LG-ACK (although a code is reserved for an as yet undefined LG-NACK). We expect that further studies may lead to additional specifications regarding the use of lettergrams.