Network Working Group
Request for Comments: 442
NIC: 13774

V. Cerf

24 January 1973

The Current Flow-Control Scheme for IMPSYS

BB&N quarterly report #13 outlines part of the current flow control scheme in the IMP operating system. A meeting held March 16, 1972, at BB&N was devoted to the description of this new scheme for the benefit of interested network participants.

This note represents my understanding of the flow control mechanism. The essential goal is to eliminate unnecessary retransmissions when the load is heavy, eliminate the retransmission time-out period when the load is light, increase bandwidth, prevent re-assembly lock-up, control traffic from HOSTS into the net more strictly than the earlier link blocking method, and secure the rights of life, liberty, and the pursuit of happiness for ourselves and our posterity,...oops.

Source IMP-to-Destination IMP Protocol

There are two different protocols depending on message length (i.e. single or multi-packet). We illustrate first the single packet case.

```
        Source Imp                        Destination Imp
        ----------                        ---------------

case 1) message (1) + implicit req (1)--->
                                    <--- RFNM (arrived ok)
        [discard copy of msg]

case 2) message (1) + implicit req (1)---> no room, don't respond
                                    <--- All (1)  (room available)
        message (1)                 --->
        [discard copy of msg]       <--- RFNM (arrived ok)
```

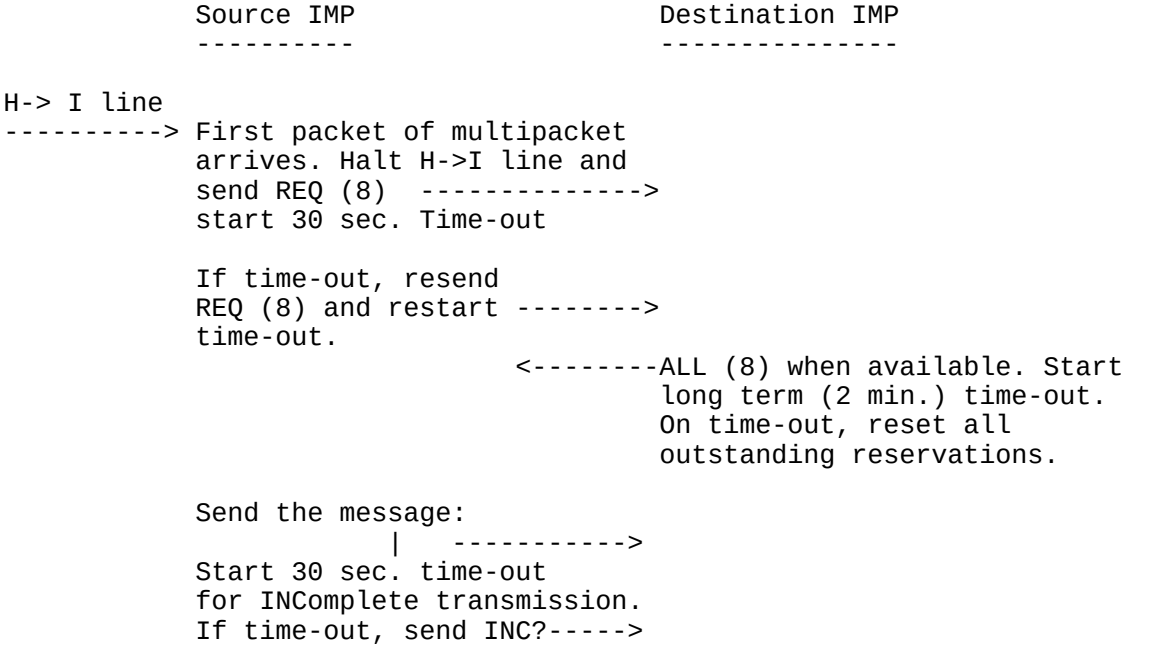
In the first case, a single packet message is sent to the destination IMP. This message acts as an implicit request for single packet buffer space. If there is room, as in case 1, the destination IMP responds with a RFNM. The source IMP, which has retained a copy of the message, deletes its copy and goes on.

The second case illustrates what happens when the source IMP sends a message to a destination IMP at which there is no room for the one-packet message. The arrival of the single packet message constitutes a request for single packet buffer space, and is recorded as such by the destination IMP in a first-come-first-served buffer reservation

request queue. When space is available, the destination IMP will transmit an ALL (1) to the requesting source IMP which can then send the single packet message again, this time knowing that space has been reserved at the destination.

For multi-packet messages, the procedure is somewhat different. When a message enters an IMP from a HOST, and the "last bit" flag is not set when the number of bits in a maximum length single packet have arrived, the IMP halts the HOST->IMP transmission line while it determines whether space has been reserved at the dest. IMP. If space (8 packets worth) has been reserved, the HOST->IMP line is re-opened, and the message is sent out normally. If space has not been reserved, the HOST->IMP line is kept closed while the source IMP makes a request for multi-packet buffer storage at the destination IMP. When 8 buffers are available, the destination IMP responds with an ALL (8). The source IMP then transmits the message, and waits for a combination RFNM and ALL (8) from the destination IMP. The destination IMP will delay its RFNM, if necessary, until it has another 8 buffers available for the next multipacket message.

This sequence is illustrated below:



```
                                <------On recept of message, send
                                       RFNM + implicit ALL (8). On
                                       receipt of INC? send RFNM +
                                       ALL(8) if MSG(8) received,
                                       or send INC! if MSG(8) not
                                       received. Start 2 min. time-out
                                       on ALL(8).

          Queue ALL(8); start 125 ms.
          time-out when it reaches
          head of queue. If time-out
          on ALL(8), send GVB(8)----->
                                <----- Ack.
          else send next message ----->
```

A key point in this protocol is that a source IMP, after receipt of a RFNM and implicit ALL(8) from the destination IMP, has 125 msec. in which to initiate the transfer of at least the first packet of a multi-packet message to the destination IMP. The source IMP may have several allocate responses queued up in which case these time-outs occur one after the other (one has to time-out before the next 125 msec time-out starts).

Time-outs exist in the source IMP which cause it to send INC? messages to the destination IMP if it has received no response from some earlier message.

Buffer Allocation

A total of 40 buffers are available for store/forward and re-assembly purposes. At most 32 can be allocated for re-assembly, and at most 24-25 can be allocated for store and forward use. This prevents either kind of traffic from completely shutting out the other kind.

Message Ordering (Source IMP-to-Destination IMP).

As an aid to congestion control, an IMP can have at most 4 messages outstanding (un-RFNMed) for each other IMP. Link numbers in the message leader are ignored by the IMPs. Instead, IMPs mark messages leaving for other destinations with an 8-bit message number. In addition, a 2-bit priority number is also used in case a HOST has marked a message as a priority message. The key notion here is that the IMPs treat all HOSTs on a given IMP as if they were a single HOST. A single sequence of message and priority numbers is used in each direction between each pair of sites.

The receiving IMP remembers the message number of the last message delivered, as well as the priority number of the last priority message delivered. It uses this information to correctly sequence messages out the IMP-HOST line (s). Since there is only one sequence of numbers for each pair of sites, messages for one HOST at a site may get in the way of messages for another HOST at the same site. In fact, if some message, m, is the next in line to go to some HOST, and that HOST delays receipt for 30 seconds, any messages for another HOST may be delayed that long also. However, only the first message is lost, since the second one could not even start into its destination HOST until the first one had been delivered. There is a tighter coupling between HOSTs sharing an IMP than before, but not much tighter.

An example of the use of message and priority numbers is given below.

Order sent by Source IMP	Order received by Dest. IMP	Order received by HOST
11,12P(1),13P(2),14 -->	13P(2),12P(1),14,11 -->	12P(1),13P(2),11,14
11,12P(1),13P(2),14 -->	13P(2),11,14,12P(1) -->	11,12P(1),13P(2),14

where 13P(2) is interpreted to mean message #13, priority number(2).

Note that there are only 2 classes of messages, priority and non-priority, and that the priority numbers simply allow ordering at the destination of multiple outstanding priority transmissions from the same site.

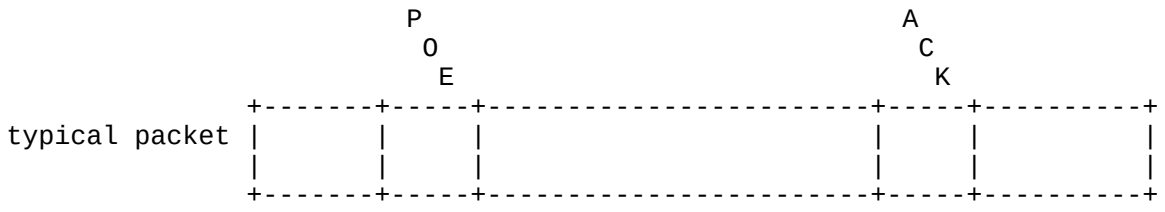
If HOSTs use link numbers to de-multiplex messages to processes, then it would be a mistake to arbitrarily assign short messages priority. If a file transmission were carried out such that the last short message had priority, the file might not enter the receiving HOST in the same order it was sent!

ACK Mechanism

IMPs treat their physical channels (phone lines) as if they were pairs of simplex communications paths. Each IMPSYS has a sender and receiver module for each full duplex channel. Each module has an "ODD/EVEN" bit which is used to keep track of the state of the last packet on the line. The object is for the sender module to "block" a channel until the corresponding receiver has received a packet indicating that the send packet was received on the other end (i.e. an acknowledgment).

In the present system, acknowledgments are separate IMP-IMP packets. In the new system, they are a single bit in a packet flowing in the opposite direction on the reverse path of a full duplex channel.

Every packet sent between IMPs has an ACK bit and an OE bit, as shown below.



We need some terminology: Let POE be the packet OE bit, and SOE, ROE be the send module OE bit and Receive module OE bit respectively. For two IMPs, A and B, we distinguish SOE/A and SOE/B as the two send module OE bits at IMPs A and B respectively.

The rules of operation are as follow:

Sender
------
if ACK != SOE then do nothing
--
else SOE <- !SOE (i.e. flip SOE bit) and free channel.

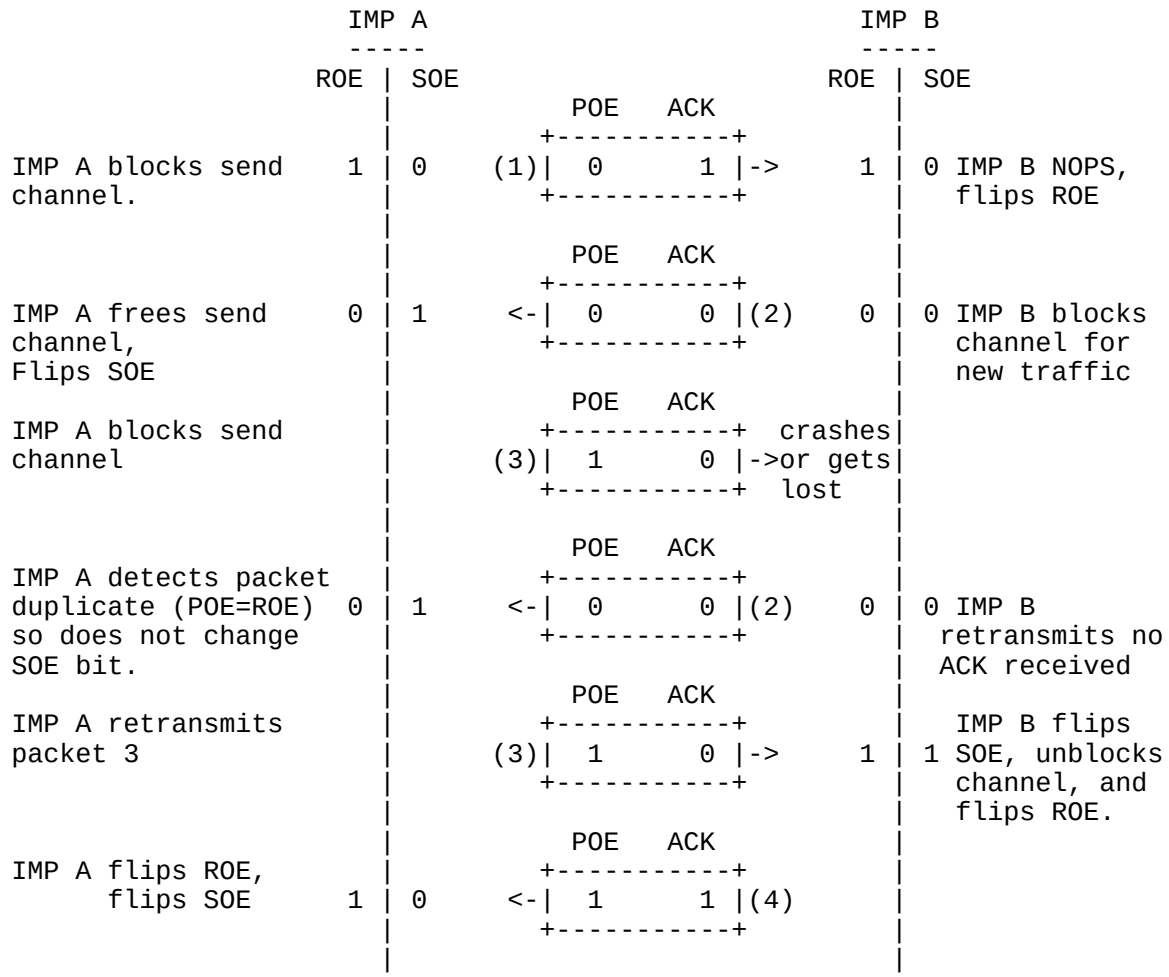
Receiver
----
if POE = ROE then packet is a duplicate so throw it away.
--
else ROE <- !ROE

Whenever a packet is sent by the sent module, its two bits, POE and ACK are set up by:

POE <- SOE
ACK <- ROE

The mechanism is designed to use real traffic to accomplish the acknowledgment protocol by piggy-backing the ACK bits in the header of real packets. If there is no real packet waiting for transmission in the opposite direction, a fake packet is assembled which carries the ACK, but which is not acknowledged by the receiving side.

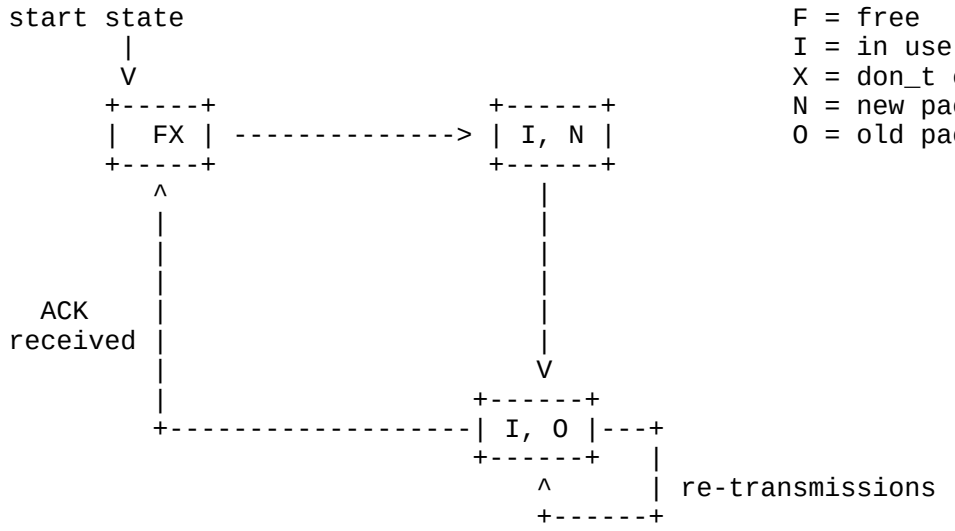
We give an example of the operation of this mechanism between two IMPs.



In fact each send/receive module has 8 OE bits, so up to 8 packets can be outstanding in either direction.

How things really work

Actually, a single send module is responsible for trying to transmit packets out on the 8 pseudo-channels. Each channel has a two-bit state (in addition to an OE bit). Each channel is either FREE or IN USE and if IN USE, it may be sending OLD or NEW packet.



Between IMPs, packets are sent repeatedly, until they are acknowledged. However, the choice of what to send is ordered by priority as follows:

1. Priority Packets (as marked by HOST)

2. Non-Priority Packet

3. Unacknowledged packets (on I,O state channels)

4. Others

It was pointed out that a heavy load of type (1) and (2) traffic might prevent retransmissions from occurring at all, and W. Crowther responded that the bug would be fixed by a 125 ms time-out which forces retransmission of old packets in class (3).

Note that each packet must carry a "pseudo-channel" number to identify the POE-to-channel association, and 8 ACK bits (which are positionally associated with the pseudo-channels). Thus a single packet can ACK up to 8 packets at once.

[This RFC was put into machine readable form for entry]
[into the online RFC archives by Helene Morin, Via Genie, 12/99]