                        TIP Message Buffers

     We have recently heard some groaning about the size of the TIP's
message buffers. While we realize these aren't as big as some Hosts
might desire, they aren't as small as the intensity of the groans
suggest either.

     Let's first consider messages going from a TIP to another Host.
The buffers have the following sizes:

device numbers	buffer size (in 8 bit characters)
1-2	64
3-16	32
17-41	16
42-62	8
63	6

The TIP user has the option of having his messages sent 1) every
character, 2) on line feeds and/or com's, 3) every nth character, or
4) the OR of 2) and 3). Selecting to have messages sent every large
number of characters, say 100, will result in the TIP sending the
longest messages it can for a given device. Hosts which don't like to
receive very short messages might advise users accessing them from a
TIP to set the TIP's parameters to use the maximum length buffer.

     Now let's consider messages going from another Host to a TIP.
The buffers have the following sizes:

device numbers	buffer size (in 8 bit characters)
1	96
2	64
3	48
4-17	24
18-35	16
36-63	8

The TIP double buffers its terminal output. Thus, when a TIP terminal
makes a connection to a Host, the TIP sends off an allocation of
between 8 and 96 characters, depending on the terminals device number,
and when a message comes using up the allocation, the TIP immediately
sends another allocation for the same number of characters while it
prints the first buffer.

     For traffic both to and from the TIP, lower numbered devices have
bigger buffers. Therefore, users of line oriented systems, as well as
users of higher speed devices, should try to come in through the lower
numbered ports on the TIP's multi-line controller, if possible.

     The sizes of the TIP's message buffers and the number of each
size are not permanently fixed and can be changed if a better
distribution is suggested. We didn't know what size buffers to
provide so we have provided a variety, What is fairly fixed is the
total amount of buffer space: two output buffers and one input buffer
for each of 63 devices must come out of this total buffer space.

     The answer to your question "Why not dynamically allocate buffers
at run time?" is: It is a complicated job to do that. It requires
memory compactions, a mechanism to reclaim space from current users
when a new user comes on, etc. Our guess is that the code to
dynamically allocate buffers at run time would reduce the total space
available for buffers by about one-fifth.

         [This RFC was put into machine readable form for entry]
         [into the online RFC archives by BBN Corp. under the   ]
         [direction of Alex McKenzie.   12/96   ]