The following memo was a page of a document describing changes in
version 1.34 of the Tenex system. I believe that the author is Ray
Tomlinson or someone else in the BBN-RCC Tenex group. In any case Ray
has agreed that these comments should be circulated to the to the
network community, rather than to only the Tenex community.

Comments on RCTE from the TENEX Implementation Experience

The code to implement the RCTE option of the new TELNET protocol for
TENEX has been completed. The RCTE option permits a reduction in network
traffic by deferring the transmission of characters which will not cause
the receiving user program to be activated until a character which will
cause the user program to be activated. A further reduction is achieved
by minimizing the flow of echo characters back to the user TELNET
program. This is done by having the server instruct the user TELNET to
echo the group of characters up through the next wakeup character. By
sending this command as the user program is about to read the first
character of that group, the echo is guaranteed to follow any response
to the preceding group of characters.

Significant problems with the RCTE protocol were encountered. The
handling of spontaneous output was specified in a way that made the
implementation extremely difficult to do correctly (if, indeed, a
correct implementation is possible). The solution here was to completely
isolate the control of input transmission and echoing from the
characters flowing in the output stream. Synchronization of input and
output then occurs directly by virtue of the embedding of control
information in the output stream. This approach permits a simplified
coding of both the user TELNET and server TELNET.

A second problem was the handling of interrupt characters. The RCTE
protocol fails to provide an explicit mechanism for interrupt characters
thus necessitating the handling of interrupt characters as program
wakeup characters. Since the interrupt characters are not actually
handled as program wakeup characters and, in fact, bypass the terminal
input buffer, a special provision had to be made to get the command sent
back to the user TELNET to indicate that the character stream should be
echoed beyond the point where the interrupt character was typed. The
transmission must be synchronized with the processing of the terminal
input buffer so that it will be sent at the proper time. This was
achieved by putting a marker in the input buffer at the point where the
interrupt character was. This marker is never given to the user's
program and must not be counted as an input character. A new counter was
installed indicating the number of such markers in the input buffer and
the SIBE JSYS modified to indicate "empty" only if the difference of the
total characters in the buffer and the number of markers in the buffer
is greater than 0.

A third problem is handling the case where the input buffer is cleared. Since the buffer may contain various wakeup characters and special markers, when the buffer is cleared, the user TELNET and SERVER may get out of sync. It is infeasible to scan the buffer and send a RCTE command for each such wakeup character or special marker. instead, a command is sent to the user TELNET meaning "clear your input buffer and reset your counters". This command is implemented by sending "WILL RCTE". This is the reverse case from a normal RCTE (i.e. DO RCTE) and thus cannot be confused with the normal use of the RCTE option. This saves adding a new option.