

To: Distribution
From: Richard J.C. Kissel
Date: April 6, 1977
Subject: Current editor problems

The standard Multics editor must support a wide variety of users and uses, ranging from novice users just learning the editor and doing very basic interactive editing, to sophisticated users doing complex interactive editing and writing editor macros to solve general problems. Furthermore, as more Multics are sold, the percentage of novice and unsophisticated users is going to increase.

We have identified the following general problem areas common to the current Multics editor:

- 1) It is difficult to learn (and use for the novice).
 - a) The command level interface is confusing (in general, the object that a command operates on should be specified in the command invocation, i.e. the segment to be edited should be specified in the editor command call).
 - b) The syntax is cryptic and offers little protection for the unwary user (typing "cwd" when one thinks he has quit the editor but is really in edit mode, replaces the current line with one consisting of "wd" and stays in input mode; many other commands can have similar unexpected effects in the same situation).
 - c) There is very little redundant information or feedback for the novice user (such as, which editor mode he is currently in, or prompting for input or requests).
 - d) Simple editing tasks, such as moving text around in the current buffer, are hard to do and require learning new and complex features (other buffers, buffer expansion, etc.).
 - e) There is no protection for the user from quitting from the editor without writing, or from losing a lot of work if the system crashes.
- 2) Addition of new features is difficult.
 - a) Again, the cryptic syntax makes adding new features an exercise in obfuscation (see Ted capital letters, special symbols, etc.), since new requests can only be single

Multics Project internal working documentation. Not to be reproduced or distributed outside the Multics Project.

characters with no delimiters.

3) It lacks features.

- a) Corrections for the above problems.
- b) Current CRT type terminals (e.g. TTY 40) provide the user with a controllable cursor and often even local editing capability. The editor should provide support for these terminal features.
- c) A subroutine interface for use by other commands, and, an interface from the editor to the rest of the Multics commands and active functions.
- d) Better features for writing macros (conditional requests, macro parameters, etc.).
- e) It should be recursive to provide environment independence (currently care must be taken that Multics commands executed from within the editor do not invoke the editor).
- f) Many other features which, if the syntax were extensible, would be straightforward to add.

One very important property which a new editor must retain is minimization of typing, which yields ease of use for the experienced user.

Many of the features we want in a new editor are already incorporated into Ted. However, because Ted has chosen to maintain compatibility with qedx syntax, it has the same problems (1 and 2 above) as qedx. Furthermore, the standard Multics editor must be implemented as efficiently as possible since it is used in system benchmarks, and the current Ted implementation is too inefficient.

An MTB with a proposal for a new editor is planned to come out shortly.

Please address any comments on this MTB to:

Rick Kissel
Honeywell Information Systems
575 Tech Sq.
Cambridge, Ma. 02139

or send mail to Kissel.Multics or call 617-492-9319 or HVN 261-9319.