COMPARISON OF SIX INTERACTIVE TEXT EDITORS

by

RUSSELL WAYNE SOWELL JR

B.G.S., University of Nebraska at Omaha, 1969
M.B.A., Old Dominion University, 1974

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1983

Approved by:

[illegible]

Major Professor

TABLE OF CONTENTS

List of Tables 1
Acknowledgements 2
Chapter 1 Introduction 3
The Important Role of Interactive Text Editors 3
Literature Related to Interactive Text Editors 6
Chapter 2 Functions of Interactive Text Editors 9
Traveling 10
Editing 13
Viewing 16
Chapter 3 Survey of Six Interactive Text Editors 18
General 18
Brief Description of the Six Editors 20
Comparison of Traveling Features 29
Comparison of Editing Features 34
Comparison of Viewing Features 44
Comparison of Hardware Facilities Supported 47
Comparison of Features Related to Human Factors . . . 50
Advantages and Disadvantages of the Stand-Alone Editor 59
Chapter 4 Future Improvements of Interactive Editors . . 61
Chapter 5 Conclusion 67
Glossary 70
References 77
Bibliography 80

LIST OF TABLES

Table 1 Wang OS 30 Special Function Keys 26

Table 2 Comparison of Traveling Features 29

Table 3 Comparison of Traveling Features 32

Table 4 Comparison of Editing Features 35
(Insertion Functions)

Table 5 Comparison of Editing Features 37
(Deletion Functions)

Table 6 Comparison of Editing Features 40
(Modification Features)

Table 7 Comparison of Viewing Features 45

Table 8 Comparison of Hardware Facilities Supported . . 48

Table 9 Comparison of Features Related to Human Factors. 51

Table 10 Comparison of Features Related to Human Factors. 56

Acknowledgements

I would like to thank Dr. Elizabeth A. Unger for her support related to the writing of this paper and for her guidance and help during my graduate study at Kansas State University. I also want to thank my dear wife, Mary, for the many tiring hours she spent typing this Report.

CHAPTER 1

INTRODUCTION

The Important Role of Interactive Text Editors

In a class recently conducted at Kansas State University in automated office systems it was revealed that a recent survey indicated that between 1970 and 1980 the number of white collar workers in the United States increased to a point where white collar workers outnumber blue collar workers in the total work force. Additionally, the rate of increase in the number of white collar workers exceeds the rate of increase in blue collar workers. By 1985, it is estimated that two thirds of the United States work force will be composed of white collar or office workers.

Vincent Giuliano asserts in a recent Scientific American article that "In 1955 the odds were overwhelming that someone working at an alphabetic keyboard device was female and either a typist or a key punch operator. No longer. The keyboard workers are both female and male and the typewriterlike devices now accomplish an astonishing variety of tasks." (Giul82, p. 150) In this article he suggests that in the future office workers will operate from a "virtual office." Offices will not be defined as geographic locations in a building with a desk across which papers constantly flow. The office will be identified directly with the worker; the office can be where the worker is. The increasingly wide spread use of remote computing has contributed significantly to this condition. Consultants (including Giuliano) at Arthur D. Little, Inc. predict that 40 to 50 percent of all American workers will be making daily use of electronic terminal equipment by 1990. (Giul82).

To increase the productivity of office workers as well as increase the overall productivity of organizations through the use of better office tools, the application of computerized techniques is currently receiving considerable attention. Computerized text processing is one application where a lot of progress and improvements are being made.

Olson and Lucas (Olso82) suggest eighteen propositions for research and practice in the area of office automation. The first two of these propositions address the benefits of text processing in the automated office:

1. Automated office systems, especially text processing functions, can improve the quality of written documents produced.

2. Automated office systems, especially text processing functions, can permit increased specialization of skills to support administrative and clerical tasks.

According to Meyrowitz and van Dam, "The interactive editor has become an essential component of any computing environment. It uses the power of the computer for the creation, addition, deletion, and modification of text material such as program statements, manuscript text, and numeric data. The editor allows text to be modified and corrected many orders of magnitude faster and more easily than would manual correction.

Though editors have always been deemed important tools in computing systems, they have only recently become a fashionable topic of research, as they become key components in the office of the future. No longer are editors thought of as tools only for programmers, or for secretaries transcribing from marked-up hard copy generated by authors. It is now increasingly realized that the editor should be considered the primary interface to the computer for all types of knowledge workers, as they compose, organize, study, and manipulate computer-based information." (Meyr82, p. 321).

This paper addresses the subject of interactive text editors. The next section briefly discusses the most recent significant literature related to interactive text editors. Chapter 2 covers the functions of editors: traveling, editing and viewing. Chapter 3 is a survey of six editors opening with a brief description of each editor followed by a comparison of the traveling, editing and viewing features of each one. A comparison of hardware facilities and a comparison of features related to human factors is also included. Chapter 3 closes with a discussion of the advantages and disadvantages of stand alone editors. Chapter 4 suggests improvements for future editors and Chapter 5 provides a conclusion.

Literature Related to Interactive Text Editors

"Unlike the literature in areas such as programing languages or operating systems (with rich collections of written materials from basic definitions and tutorials to complex formalisms, analyses and implementation strategies), the literature in the field of editing consists primarily of functional descriptions of particular editors. Little work has been done to standardize terminology or to create a framework for comparing, contrasting, and analyzing editing systems." (Meyr82a, pp. 321-322).

There are, however, several writings which prasie the merits of certain editors (Coul76, Deut67, Haze80, TeiW79, vanD71). Still others offer suggestions for improving the human-oriented interface (Gain78, Jone78, Hans71, Mart73, Rouse73, Wass73). Other literature is listed in the references and bibliographic entries of more recent writings; Meyrowitz and van Dam (Meyr82b) list 169 writings. Embley and Nagy (Embl81) list 120, and Roberts (Robe83) lists 27.

The Embley and Nagy (Embl81) article reflects some basic work related to the behavioral aspects of text editors. They assert "In discussing editors firm opinions abound: everyone, from greenhorn to old hand, knows exactly what the best and worst features of given editors are and just how new editors ought to be designed. The only problem is the striking lack of consensus. Nor are there universally acceptable means of determining who is right. The distinction between conventional dogma and scientific fact is blurred." (Embl81, p. 35). They suggest four ways text editors can be studied: (1) introspection and intuition, (2) field studies and observations, (3) formal analysis, and (4) controlled experiments. Their paper discusses studies of all four types. Their conclusions include:

1. Error detection and correction, suboptimal choice of editing methods, and unpredictable mental activity account for 25 to 50 percent of the task time.

2. Flexibility and an increased number of options tend to improve the editing performance of expert users but slow the performance of novice users.

3. Editing efficiency can be improved through the increased use of English-like phrases as opposed to the notation commands found in most text editing interface languages.

4. There is not an overwhelming difference in the performance of editors. "Most users can perform equally well on any reasonable editor." (Embl81, p. 65).

5. The impact of working conditions for other keyboard operations has already been studied extensively and the results probably apply to editing as well. The ergonomic aspects of keyboard layout are well understood.

6. Menu selection in text editors appears to be less error prone but slower.

7. The mouse appears to be the best display selection mechanism.

One of the clearest and most comprehensive articles on interactive text editors is the one by Meyrowitz and van Dam

(Meyr82a and Meyr82b. This is a two part article, both parts appearing in the same issue of ACM Computing Surveys. Part I provides a general overview of text editors and text editing. It provides a brief historical background and a discussion of the functions of text editors. Part II discusses the types of text editors and presents technical details on specific editors. Over 30 editors are discussed to some extent.

One of the most recent evaluations of text editors appears in the April 1983 issue of Communications of the ACM. This article evaluates nine text editors: Wylbur, TECO, Wang, NLS, EMACS, BRAVO, BRAVOX, STAR and Gypsy. The article also recommends an evaluation methodology for future research. The methodology is for a "standardized" evaluation which "focuses on the common properties of text editors rather than on the idiosyncracies of particular editors" (Robe83, p. 266). The methodology focuses on the

Time for experts to perform tasks

Error costs for experts

Novice learning of basic tasks

Functionalality over a wide range of tasks

The results of the evaluation indicated that BRAVO, BRAVOX, STAR and Gypsy are better than the other five editors in most of the areas evaluated. One must be aware of possible bias in these findings since both authors work for Xerox and all four of the "best" editors are Xerox products.

CHAPTER 2

FUNCTIONS OF INTERACTIVE TEXT EDITORS

There are three basic functions that an editor must perform. These functions are referred to in the literature as traveling, editing and viewing. Meyrowitz and van Dam (Meyr82) discuss viewing in two parts: viewing and displaying. They do this in order to clearly define the parts of an editor. The viewing "module" of an editor addresses which portions of the document are to be displayed and in what format they are to be presented to the user. The display component receives the formatted view from the viewing component and maps it to a hardware device. In this paper viewing and displaying will be considered together as a single function of an editor and referred to as viewing. This is appropriate for this paper since it does not analyze the structure of the editors surveyed and the functions of viewing and displaying appear to users as a single function.

Traveling

The object or target of a text editor is a document such as a computer program, letter, report, essay, booklet or book. The user of the editor must understand the logical view of the document as it is addressed by the editor before performing the actions necessary to cause traveling through the document. Additionally, the user must understand what part of the document will be displayed as the traveling takes place.

Traveling takes place by moving one or more pointers logically through the document. The pointers may point to actual addresses in memory where the document is stored or to addresses in buffers where part of the document has been copied for editing and/or viewing. Normally the display device will display at least a line of text where the editing pointer is pointing (called the current line). Several lines, half pages or full pages of text may be displayed. If a CRT equipped terminal is used, a cursor may point to the current line.

Most systems provide a simple means for traveling to the top and bottom of documents. Traveling to the top quickly and easily is important since movement to other parts of the document is frequently made relative to the top of the document. Traveling to the bottom of the document is obviously necessary when text needs to be added to the end of the document.

Line oriented editors provide traveling in relation to visible or invisible line numbers assigned to the text lines. Movement may be to an absolute line number or to a line a given number of lines from the current line. In Control Data Corporation's XEDIT this relative movement is performed by a command like

```
N10
or  N-15
```

which causes the pointer to move to the line ten lines below the current line or 15 lines above the current line. (CDC79).

Some editing systems allow "paging" the text where simple commands cause the next page or the previous page of text to be displayed. "Scrolling" is a technique where the text displayed on the CRT equipped terminal is moved up or down as far as desired by the user.

Stream editors, like the Text Editor and Corrector (TECO), operate on text as one long continuous stream of characters. The pointer in a stream editor is also moved absolutely or relatively. Each character in the document is numbered, beginning with the first character, although the numbers are not displayed. An absolute movement causes the pointer to move to a specified numbered character. For example, the command 124J would cause the pointer to move to the 124th character in the document. Relative movement begins at the current position of the pointer. For example, the command 57C moves the pointer forward 57 characters and -32C moves the pointer backward 32 characters. The pointer can also be moved in accordance with a pattern search. (Meyr 82b).

At least two editors, NLS and FRESS, provide a facility for inserting "labels" into the text. These labels may then be used to travel through the text by issuing commands that cause the editing pointer to move directly to the text identified by the label. (Meyr82b).

The most advanced editors like those used in the Apple Lisa and the Xerox Star allow for the manipulation of multiple pointers in multiple files during the editing session. Selection of the text object to be manipulated is made using a cursor controlling device called a mouse. Operations related to the selected object are performed by using function keys on the mouse or on the keyboard of the terminal. Traveling in these systems is highly sophisticated and almost all of the pointer movements are transparent to the user. (Smith82).

Editing

The editing or modifying facility in an editor is, of course, the feature that makes editors so useful and such poweful tools. Editing text includes the functions of creating, inserting or adding, deleting and changing text in a document. This section discusses the general approach to editing performed by the different types of editors:

Line editors

Stream editors

Display editors

Syntax-directed editors

LINE EDITORS. Editing in line editors begins by moving the editing pointer to the line of text that requires deleting or changing. If text is to be added to the document, the pointer is moved to the line immediately above or below the position where the new text is to be inserted.

Addition of new text in line editors is normally entered following some kind of insertion command. New text is entered until another command is issued indicating the end of the addition process.

Deletion of text can be performed by deletion of a single character, groups of adjacent characters, a single line, groups of adjacent lines or occurrences of specified text strings scattered throughout the document. Care must be taken when deleting text to insure that only the appropriate text is deleted.

Changing data in line editors sometimes requires the complete replacement of the entire line of data that requires change. Other facilities allow for changing character strings within lines of text to new and different character strings. These character string changes may take place for a single line, several lines, or all lines in the text which contain the specified character string.

STREAM EDITORS. Insertion in a stream editor like TECO takes place between two characters after the pointer is moved to the correct position. Deletion occurs by issuing simple commands to delete a given number of characters in relation to the pointer. Text is modified by moving the pointer to a specified position or by searching for specified patterns and substituting the new text. (Meyr82b).

DISPLAY EDITORS. Display editors allow users to make additions, deletions and changes to the document by moving a cursor to the desired position on a CRT screen and issuing commands or simply typing in the appropriate text.

In IBM's XEDIT, text can be added in several different ways. One process allows lines of data to be added by entering an add command in the left hand margin of the screen. Single or multiple lines of text can be added. Insertion of text within a line can take place by execution of a CINSERT command. One advantage of display editors is that the user can observe changes to the text, including additions, while the changes are being made. (IBM80). Deletion of single characters or a few characters on the same line can be effected by moving the cursor to the characters to be deleted and simply depressing a delete function key or depressing the space bar. Deletion of a line or groups of lines can be performed by placing the cursor at the appropriate lines and depressing a delete function key or in IBM's XEDIT, marginal commands cause lines to be deleted. (IBM80). Changes to single characters and small groups of characters are easily made by moving the cursor to the target characters and simply typing in the new text. Pattern searching and replacement can also be performed in several different ways in display editors.

SYNTAX-DIRECTED EDITORS. Syntax-directed editors allow additions, deletions and changes to text in many of the same ways line oriented or display editors do. The major difference, however, is that, after each addition or change is made to the document (usually a computer program), the syntax is checked to determine if the structure is correct and valid in accordance with the rules of a given language. If the syntax is correct, the entry is accepted; if not, an error message is displayed to the user. This process is designed to assist programmers to produce more reliable code. The instant feedback relieves the programmer from having to wait hours or days to find out that his/her code is basically incorrect.

All types of interactive editors also provide a facility for copying and moving text around in a document. In some editors, another temporary file is used to hold the text during the move. In others, if a temporary file is used, it is usually transparent to the user.

Text can also be modified by inserting text from another file. Not all editors provide this feature.

Viewing

The viewing component in an editor is that component which determines what part of the entire document will be displayed to the user at any point in time. A buffer may be used to hold part of the document for viewing.

How much of a document is prepared for viewing is controlled somewhat by the display devices anticipated being used with the editor. In the case of many line editors, the editors are designed for use with typing devices which are not equipped with CRTs. In these editors generally only a single line is displayed to indicate the current line. Viewing of other parts of the document can be obtained by executing a print or list command. Single lines, multiple lines or the entire document can be displayed on command.

In display editors the user can view several lines, a half page or a full page of text continually during the editing process. Simple commands may allow paging where preceding or succeeding pages can be viewed quickly. Scrolling allows the user to view the document as it "rolls" up or down on the screen.

The display component of an editor is that component which prepares the text designated by the viewing component for display on a specific device. Much that has already been said regarding viewing is actually carried out for the user by the display component. This function is relatively simple for many line orieented editors and early display editors. In more advanced editors like those used in the Apple Lisa and Xerox Star the display function is complicated by the requirement to display multiple windows simultaneously, the display of icons and the display of overlapping windows. (Smit82).

CHAPTER 3

SURVEY OF SIX INTERACTIVE TEXT EDITORS

General.

Most of the information in this chapter was taken from the publications listed below. The comparisons of editor features in this chapter are reflected in tables and amplified in the accompanying narrative. It would be very irritating to the reader if every table entry and every narrative paragraph was repeatedly annotated with a reference. In order to make this chapter easier to read and give proper credit to the appropriate publication (and its author(s)), the editors and the publications used as references are listed only once, below:

EDITOR	REFERENCES
ed	Thom82
PEDIT	Perk79
XEDIT (CDC)	CDC79
XEDIT (IBM)	IBM80
Wang OS 30	Wang78
vi	Joy80

In those instances where publications other than those above are used in this chapter, they are referenced in the appropriate place.

The entries in Table 2 through 10 are designed to provide for the user something more than a simple "Yes" or "No" to indicate whether a particular feature exists in an editor. Short comments are used in many entries to indicate how a feature is implemented in the editor. "Yes" or "No" are entered when they appear to the author to be sufficient or when there is no short comment (suitable for a table) that would help the reader.

Brief Description of the Six Editors Surveyed

ed. ed (always printed uncapitalized: ed) is an interactive text editor designed to be used with UNIX operating systems. In some instances ed appears to be an integral part of UNIX. It is available in many different versions for many different computer systems and is provided by numerous suppliers. It is a variable length line editor, which eliminates line truncation problems found in other editors like PEDIT. It has a single line viewing buffer for the current line but provides for a variable length editing buffer which can logically include the entire document being manipulated. Although ed is used with CRT equipped terminals, text on the screen cannot be manipulated by moving the cursor around the screen and making changes.

ed has two operating modes: command mode and text editing mode. In the command mode operator entries are interpreted as commands. The insert, append and change commands put the editor in the text entry mode. Movement from the text entry mode to the command mode is effected by typing a period on a line by itself.

Each line in an ed document has an invisible line number. Movement to a line relative to the current line is performed by entering the prefix + (to move forward) or - (to move backwards) followed by an integer. The integer value determines the number of lines of movement.

The general command format for ed commands is:

```
address range     command letter     parameter...
```

The address range may be a single line or the first and last lines in a group of lines. Commands in ed are single letter commands. Parameters are optional and are not used with all commands.

PEDIT. PEDIT is an interactive text editor used in the Control Program/Conversational Monitor System (CP/CMS) environment. It is designed to manipulate several different types of files and requires when invoking the editor that the file type be specified. File types include BASIC, COBOL, FORTRAN and PL1. Specifying the file to be one of these types does not, however, indicate that PEDIT will perform syntax checking operations that are available with program editors. Identifying the file types simply indicates to the editor what default characteristics to assign to the file. File characteristics include fixed or variable length records, record length, upper case/lower case specifications, automatic line numbering option and tab settings.

PEDIT is normally used by operators at CRT equipped terminals but, like ed, the document or parts of it are displayed only as directed by the operator. Changes to the document cannot be made by moving the cursor around the screen and creating or modifying text. Creation and modification of text takes place by issuing commands and entering data.

PEDIT operates in two modes much like ed: edit mode and input mode. To move from the edit mode to the input mode enter INPUT. To reenter the edit mode simply type in a carriage return for a new line.

Responses to PEDIT commands are displayed in one of two ways. With VERIFY ON the current line is displayed following each command, thus displaying the results of the command. This is an important feature for new users. With VERIFY OFF the current line is not displayed after each command. This feature saves time and may be more acceptable to the veteran user.

Movement through the document is controlled by moving an invisible line pointer either to a specific line number or to a line specified by its displacement above or below the current line. Movement takes place automatically following many of the commands.

XEDIT (CDC). XEDIT is an interactive text editor developed by the University of Minnesota and available for time sharing in Control Data Corporation (CDC) CYBER 70 and 170 and CDC 6000 computers. It can be used with a terminal with or without CRT output facilities. If a CRT equipped terminal is used, XEDIT uses the CRT display very much the same way ed and PEDIT do. That is, text is created and modified through the use of commands and cannot be manipulated through the movement of the cursor and simply entering data like a display editor. Movement through the document is very similar to ed and PEDIT with facilities for movement to specifically numbered lines or to a line specified by its displacement above or below the current line. Movement also takes place as the result of the performance of most commands. XEDIT also operates in two modes: input and edit. The input mode is entered by typing INPUT. Movement from the input mode to the edit mode takes place by entering a carriage return for an input line.

XEDIT (IBM). IBM's XEDIT is an interactive, display text editor which can be used at teletypewriter terminals but is designed primarily for full screen CRT equipped terminals with cursor control capability. The screen is separated into several sections each of which is used for different purposes. The first line displayed is a "file identification" line which constantly reflects the file name, file type and file mode of the file or document being processed. This line also indicates whether lines are fixed or variable length, the record length, the current number of lines in the file, the line number of the current line and the column being addressed on the current line. The second line is the "message line". Whenever the cursor is placed on this line the editor will accept XEDIT commands. The cursor is moved to this line when the ENTER key is pressed. The remainder of the screen is reserved for display of text from the document being processed. Beside each line displayed are five equal signs (= = = = =). This area on each line is used to enter special "prefix commands".

The current line in XEDIT is identified by the placement of a scaled line immediately below the current line. Movement of the cursor does not affect the position of the current line. The current line is also highlighted. The current line can be changed by placing a / in the prefix command area of the appropriate line. Text from the document is displayed by scrolling up or down in the document by pressing special program function keys (PF7 and PF8) if the terminal is equipped with these keys. Otherwise typed commands are used to perform these functions.

XEDIT provides a so called Power command which allows "power typing". This feature is used when large amounts of text are to be entered simply and rapidly. In this mode the operator does not need to be concerned with word length, line length or word splitting. All of this is taken care of automatically.

Unlike the three editors discussed above, XEDIT allows the operator to change text by moving the cursor to the appropriate place on the screen and typing over the incorrect text or by using special keys to insert or delete characters.

In addition to scrolling and using the / prefix command to identify the current line, text is located using objects referred to in XEDIT as "targets". Targets can be:

an absolute line number
relative displacement from the current line
line tag assigned by SET POINT
a string pattern

These targets are used in commands to affect current line selection while processing the command.

XEDIT allows multiple files to be edited in virtual memory at the same time. These files are placed in a "ring". That is, when successive files are called with an XEDIT command they are placed as the last file in the ring. Each file points to the next file in the ring and the last file points to the first file. When moving from file to file an XEDIT command without a file name will cause the "next" file to become the current file. If a file other than the "next" file is desired, the file name must be specified.

WANG OFFICE SYSTEM 30 (OS 30). The Wang OS 30 interactive text editor is the only editor discussed in this paper which is a "stand alone" editor. The editing software is part of a commercial word processing system which is marketed as a package: a microcomputer central processor, a disk unit, one or more printers, and one or more work stations. This entire system is used exclusively to support a word processing operation. The work stations are specially designed terminals to work with the Wang system. The terminals are equipped with a full sized CRT for display and a standard typewriter keyboard with four cursor positioning keys and 21 special function keys. See Table 1 for a brief description of these keys. A brief review of this table should cause the reader to begin to understand the editing power available through the use of function keys.

Because the Wang system uses a display editor, correction to text can be made by moving the cursor to the incorrect text and typing in the correct text.

Documents are usually "filed" on "archive diskettes". This means that, when an operator finishes processing a document, the document is copied from the System Disk to a diskette for filing. Later the text on the diskette can be reloaded on the System Disk for further processing.

vi. vi (always printed uncapitalized: vi), like the Wang system discussed above, is an interactive, display text editor. Unlike the Wang system, however, vi was designed to be used in a timesharing mode on a general purpose computer with general purpose terminals. The terminal screen acts as a window to the file. Changes made to the screen are also made to the stored version of the document. vi has a very large number of commands (most are single characters) which requires most keys on the keyboard to be related to at least two commands; one command when the key is pressed in the lower case mode and a different command when the keyboard is shifted to upper case.

TABLE 1

SPECIAL FUNCTION KEYS

The special function keys on the Wang OS 30 terminal perform the following functions:

KEY	FUNCTIONS
INDENT	Indents a line and all subsequent lines until RETURN is pressed.
PAGE	Defines the end of a page of text.
CENTER	Automatically centers a line of text.
DECTAB	Automatically aligns columns of numeric data by decimal points.
FORMAT	Allows the definition of vertical spacing, tab stops and line length.
MERGE	Allows document/text merging.
STOP	Allows operator to indicate where print is to stop in a document.
SEARCH	Allows a character-defined sequence to be located within a document.
NOTE	Allows operator to write notes within a document.
COPY	Allows portions of text to be copied from one portion of a document to another. This operation does not affect the original text.
MOVE	Allows portions of text to be moved within a document. The moved text is deleted in the original position.
COMMAND	Allows various special operations to be performed in conjunction with other keys.

TABLE 1

(Continued)

KEY	FUNCTION
SUBSCRIPT/SUPERSCRIPT	Allows formulas, equations, etc. containing subscripts and/or superscripts to be created.
GO TO PAGE	Instructs system to proceed directly to a specified page within a document.
CANCEL	Forces termination of operator defined function.
INSERT	Allows text to be inserted within a document.
DELETE	Allows text to be deleted from a document.
PREVIOUS SCREEN	Allows the previous screenload of text to be displayed.
NEXT SCREEN	Allows the next screenload of text to be displayed.
EXECUTE	Causes system to accept operator entry.

Vi allows for the use of intelligent terminals which can modify text on the screen and in the buffer of the terminal without immediately affecting the document in memory. Vi is designed to be used with the UNIX operating system.

Comparison of Traveling Features

Tables 2 and 3 compare the features in each of the six editors related to traveling through the document being processed.

As indicated in Table 2 five of the six editors reference lines in the document by a line number. Even in a display editor like IBM's XEDIT absolute line numbers are used as "targets" for movement through the file. The Wang display editor does not use line numbers during the editing process. Movement through the document in the Wang editor takes place by scrolling, paging, and moving the cursor. Even though line numbers are used by five of the editors, the line numbers are normally invisible to the operator. CDC's XEDIT and IBM's XEDIT will display the line numbers for special purposes. The automatic creation of line numbers by the CDC editor is useful when writing programs in the FORTRAN language.

The current line position indicator is an important feature in any editor. (See Table 2). The operator needs to know where the editing pointer is pointing before entering many commands. In some editors the position of the editing pointer (the current line) is rather vague; there is no indication where the pointer is pointing. With time the operator learns how the editor works and most of the time knows which line is current. If he/she does not know, some command in the editor can be used to identify the current line. ed is especially bad about keeping secrets regarding its current line. However, the line number can be displayed

TABLE 2

COMPARISON OF TRAVELING FEATURES

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Use of line numbers	Yes	Yes	Yes	Yes	No	Yes
visible	No	No	Optional	Optional	N/A	No
Identification of current line	Enter .= or .P	Use LIN-ENO or TYPE commands	Usually last line affected. Use PRINT or WHERE commands to deter-mine cur-rent line	Places a scale under current line. Use prefix command to set current line	Not line oriented. Cursor marks current position	Cursor position indicates current line
Movement using labels or tags	No	No	No	Use SET POINT command to assign label	Use NOTE feature	Yes
Interfile Access	Yes, use READ and WRITE commands	Yes	Yes, use COPY and READ com-mands	Yes, uses a "Ring of files"	Use func-tion keys: COPY, CUR-SOR CONTROL, EXECUTE	Yes
Recursive file access	No	Yes, to 9 levels	No	No	No	No

by typing .= and the text on the current line can be displayed by typing .P. In CDC's XEDIT the current line will be displayed by executing the PRINT command and the line number can be obtained by typing WHERE or W. In PEDIT the command LINENO will return the line number of the current line and the TYPE command will display the current line. IBM's XEDIT identifies the current line by placing a scaled line under it. The Wang editor and vi indicate the current line by the cursor on the screen. The Wang editor actually does not identify a current line but a current position on a specified line.

As indicated by Table 2, three of the six editors allow movement through the document with the use of tags, labels or notes. This feature is important if return to a given position in a document must be repeated often. These tags, labels and notes are displayed and used only during the editing process. When the document is printed, they are automatically deleted.

Table 2 also indicates that all editors provide some facility for accessing different files while editing a given file. ed, CDC's XEDIT, IBM's XEDIT and Wang provide relative simple procedures for merging text in two different files (documents) into one. PEDIT is the only editor that provides a recursive feature for moving from file to file. It appears to be a little tricky and its value and power are not clear.

Editors vary somewhat in the procedures they use for moving the editing pointer through the document (See Table 3). All six of the editors surveyed provide for pointer movement to specific lines and in some manner to specific characters. The display editors, IBM's XEDIT, Wang, and vi provide for retrieving the next

TABLE 3

COMPARISON OF TRAVELING FEATURES

FEATURES	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Pointer Movement						
by character	Yes	Yes	Yes	Yes	Yes	Yes
by word	No	No	No	No	Yes	Yes
by line	Yes	Yes	Yes	Yes	Yes	Yes
by paragraph	No	No	No	No	Yes	Yes
by page	No	No	No	No	Use GO TO PAGE function key	Yes
by screenful	No	No	No	Yes, Use PF7 and PF8 keys	Use Special Function keys	Half screen
Pattern searching						
global searches	Yes, use SEARCH feature	Use LOCATE and AGAIN commands	Yes, use L*	Global change possible	Use SEARCH and EXECUTE function keys	With / and REPEAT commands
ellipsis type searches	No	No	Yes	No	No	No
"not" searches	Use V command	Yes	Yes	Yes	No	No

or previous screenful of text using function keys. Wang and vi allow displaying of specified pages. vi offers the most comprehensive pointer movement of all the editors by allowing movement of the pointer by character, word, sentence, paragraph, section or page.

Finding text through context or pattern searching is a powerful feature which all editors provide. As Table 3 indicates, all editors provide some facility for global searches. In some editors the procedure is cleaner and simpler than others. In ed and CDC's XEDIT the procedure is simple and straightforward. vi and Wang require an initial pattern search and then execute a command that repeats the search. IBM's XEDIT provides for making global changes but not for simple global searches. PEDIT, ed, CDC's XEDIT and IBM's XEDIT allow for "not" searches. That is, the editor will identify lines which do not contain the pattern specified. "Ellipsis type searches" are pattern searches that allow the operator to identify the beginning and ending characters in a pattern without typing in the entire pattern. In CDC's XEDIT if the operator wishes to find a pattern beginning with "Now" and ending with "country", with anything in between, he/she would type

```
L/Now...country/
```

If the pattern is found the entire text from "Now" to "country" would be displayed on the terminal.

Comparison of Editing Features

Tables 4, 5, and 6 compare the features in each of the six editors related to editing (creating and modifying) the document being processed.

Three of the six editors surveyed are strictly line oriented editors: ed, PEDIT and CDC's XEDIT. They use line numbers for traveling through the document and line numbers are frequently used in document modifying commands. The whole editing orientation is toward editing lines of text. The Wang editor and vi are display editors which are not oriented toward document lines; rather the orientation changes throughout the editing session from a focus on characters, words, paragraphs, groups of lines, sentences, sections or screensful. IBM's XEDIT falls somewhere between the other two groups. It is a display editor but has a strong focus on manipulating lines of text. It is very important from the outset that an operator understand the scope of editing (the view of the editor) addressed by the editor.

Creation of the original text is a very simple matter in all six editors. Little more is required than calling the editor, giving a simple command and begin typing. Most text editors do not provide much support in text creation. Their real power lies in the facilities provided to insert, delete, change and rearrange text.

Three of the editors (vi, IBM;s XEDIT and Wang) have specific character insert functions using commands and/or function keys.

TABLE 4

COMPARISON OF EDITING FEATURES

(INSERTIONS FUNCTIONS)

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Characters in lines	Use SÛBSTI-TUTE com-mand	Use CHANGE command	Use CHANGE command	Use INSERT MODE key	Use function keys: IN-SERT, CURSOR CONTROL, EXECUTE	Use I command
Lines before text	Use INSERT command	No	Use INSERTB command	Use ADD prefix command	Data inserted on string text rather than as lines	Use O command
Lines after text	Use APPEND command	Use INPUT command	Simple operation	Use ADD prefix command		Use O or o command
Lines between other lines	Use APPEND or INSERT commands	Use INPUT command	Simple operation	Use ADD prefix command		Use O or o command
From other text files	Use READ command	Use PUT and GET commands	Use READ command	Use PUT and GET subcom-mands	Use function keys: COPY, CUR-SOR CONTROL, EXECUTE	No

(See Table 4). The other three editors (ed, PEDIT and CDC's XEDIT) use a change or substitute command that allows insertion of characters in a word, line or sentence by "changing" the original text to new text, which amounts to simply adding characters.

All editors will allow new text to be inserted before text already in the document. Some editors, however, provide a simple direct facility for this operation and in others the process is somewhat complicated. In PEDIT there is not a direct "insert before text" command and in the Wang editor insertions are made at any place in the text without regard to lines. "Insert after text" facilities are available, simple and direct for all six editors. There are also simple procedures for inserting lines of text between lines of text already in the document in all six editors.

Insertion of text from other documents stored in the system is obviously a powerful and frequently useful feature. It is the electronic counterpart in text editors of "cutting and pasting" in manual editing. All editors except vi provide for this operation in a relatively simple manner.

Deletion of incorrect or unnecessary text is sometimes as important to document preparations as creation or insertion. Three editors (ed, PEDIT and IBM's XEDIT) do not have specific character deletion commands. (See Table 5). Instead (like the insertion function for these editors) the change or substitute commands are used where the change or substitution is in fact a deletion of text. The Wang editor has a function key used with the cursor control keys for character delection. vi uses

TABLE 5

COMPARISON OF EDITING FEATURES

(DELETION FUNCTIONS)

FEATURES	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Characters in lines	Use SUBSTITUTE command	Use CHANGE command	Use CHANGE command	Use DELETE command	Use function keys: DELETE, CURSOR CONTROL, EXECUTE	Use X key
Single lines	Use DELETE command	Use DELETE command	Use DELETE command	Use DELETE prefix command	Not directly	Type dd
Groups of lines	Use DELETE command	Use DELETE command	Use DELETE command	Use DELETE prefix command	Not directly	Type #dd
Single occur-rences of a pattern	Use SUBSTITUTE command	Use CHANGE command	Use CHANGE command	Not directly	Use SEARCH, DELETE, EXE-CUTE function keys	Not directly
Between two patterns	No	No	Use CHANGE command	Awkward using DELETE	No	Not directly
Multiple occurrences of a pattern	Use SUBSTITUTE command	Use CHANGE command	Use CHANGE command	No	Not directly	Not directly
Recovering de-leted text	Use UNDO command (limited use)	No	No	Use RECOVER command	No	Use UNDO command

the cursor control keys and the "X" key to delete characters.

All six of the editors provide for simple deletion of single lines or groups of lines. The Wang editor, however, does not provide for deletion by line directly since it is not line oriented.

As indicated previously, another powerful feature of all modern editors is the ability to find text in a document through a pattern matching process where the editor is requested to find text in the document that matches a given string. All six editors provide a simple facility for this function. Additional power is provided in some editors with features which allow (1) direct match and deletion of a single occurrence, (2) direct deletion of text between two patterns and/or (3) direct deletion of multiple occurrences of matched text in a document.

Direct match and deletion of a single occurrence is performed in ed, PEDIT and CDC's XEDIT through the use of change (or substitute) commands with string match parameters. The function is not performed directly by the other three editors (IBM's XEDIT, Wang and vi).

Direct matching and deleting of text between two patterns is performed directly only by CDC's XEDIT. This transaction is performed by a single command that performs an ellipsis type search and deletion of the matched text.

Direct deletion of multiple occurrences of matched text is also provided for in ed, PEDIT and CDC's XEDIT through the use of the change (or substitute) commands with string match parameters and parameters indicating the number of occurrences to delete. Direct global string matches and deletions are also relatively simple in these editors.

Subsequent recovery of deleted text is sometimes a helpful feature. This feature is provided by IBM's XEDIT, ed and vi. In IBM's XEDIT all lines deleted in an editing session can be recovered before leaving the session. In ed the UNDO command will restore the editing buffer to the state it was in prior to the last substitute command. In vi the last nine deleted blocks of text are saved in a set of numbered registers. These registers can be used to recover the deleted text.

Table 6 compares the six editors' text modifying features. Five of the editors (all except Wang) provide commands for changing characters in the text. In these five editors the change process includes a pattern matching and substitution where one string is substituted for another. In the Wang editor, changes to text displayed on the screen are made by moving the cursor to the appropriate position and typing in the new text.

Complete lines of text are replaced in ed, PEDIT, IBM's XEDIT, CDC's XEDIT, and vi by using change or replace commands. In Wang old lines can be overtyped on the screen, thus deleting the old text and entering new. IBM's editor also allows complete line substitution using the prefix commands Delete and Add.

It is frequently useful to perform a pattern match and text modification as a single, direct action. This function is performed relatively simply by all six editors. Wang uses a Replace function key to perform this function.

TABLE 6

COMPARISON OF EDITING FEATURES

(MODIFICATION FUNCTIONS)

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Characters in lines	Use SUBSTITUTE command	Use CHANGE command	Use CHANGE command	**Use CHANGE command**	Simply type over charac- ters on screen	Use REPLACE command
A line of text	Use CHANGE command	Use REPLACE command	Use REPLACE command	**Use A and D prefix commands**	Simply type new line over old line	Use CHANGE command
By single ocur- rences of a pattern match	Use SUBSTITUTE command	Use CHANGE command	Use CHANGE command	Use CHANGE command	Use REPLACE and EXECUTE function keys	**Use S command**
By multiple occurrences of a pattern match	Use SUBSTITUTE command	Use CHANGE command	Selected or global. Use CHANGE command	Global changes using CHANGE command	Use REPLACE and EXECUTE function keys	Not directly

TABLE 6

(Continued)

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Repeat command	No	For four commands only. Use AGAIN command	Use a period (.)	Use PF9 key	Repeatedly press EXECUTE function key	Use a period (.)
Moving	Use MOVE command	Use MOVE command	Use COPYD and READ commands	Use MOVE and F or P prefix commands	Use function keys: MOVE, CURSOR CONTROL, EXECUTE	Use Y and P keys
Copying	Use TRANSFER command	Use COPY command	Use COPY and READ commands	Use COPY and P or F prefix commands	Use function keys: MOVE, CURSOR CONTROL, EXECUTE	Use Y and P keys
Macros	No	Yes	No	Yes	No	Yes

Pattern matching and changing text in one step is additionally enhanced when the operator can cause a single, simple entry to match and change multiple occurrences of a given string. This operation is performed fairly simply by ed, PEDIT, CDC's XEDIT and the Wang editor. CDC's XEDIT will allow changes to the next n lines or to the entire document (global change). IBM's XEDIT will allow pattern matching and changing to single lines (one at a time) or to the entire document with a single command. vi does not provide a single command which will perform this function.

Another useful feature in the editing process is the ability to simply repeat the previous command. Sometimes a command will require 20 or 30 keystrokes (normally when long strings are matched It is helpful if the depression of a single key or two allows the command to be repeated without reentering the command. CDC's XEDIT and vi allow commands to be repeated by simply entering a period each time the operator wants to repeat a previous command. PEDIT has a REPEAT command but it can only be used with four other commands: BLANK, BLCOL, OVERLAY or OVCOL: hence it is not very useful. IBM's XEDIT uses a special function key (PF9) to cause the repetition of commands. In the Wang editor the operator repeats commands by repeatedly pressing the EXECUTE key.

Table 6 reflects that all six text editors have a facility for moving and copying sections of text from one place in the document to another place in the document. Moving text implies that the text in the original position of the document is deleted when the move is complete. Copying, on the other hand, causes the text to be repeated in another place in the document but does not delete the text in the original place. PEDIT, ed, and the Wang editors perform move and copy operations with single commands. These single commands identify the range of text to be moved or copied and the position in the document where the moved or copied text is to be placed. In vi, CDC's XEDIT and IBM's XEDIT, two commands are used to identify the range of the text to be moved or copied and the position where the text is to be placed. CDC's XEDIT writes the text to be moved or copied to a special buffer with a COPY command. A READ command is then used to place the text in the buffer into the document. IBM's XEDIT performs the move and copy functions through the use of the prefix commands which are entered along the lefthand margin of the screen. The Wang editor moves and copies text by using function and cursor control keys.

Macros are files of commands and/or statements which can be created, filed and executed as a single command. Macros expand the basic editor's language and reduce repetitive tasks. PEDIT, IBM's XEDIT and vi allow for the use of macros.

Comparison of Viewing Features

Table 7 compares the features in each of the six editors related to the viewing of the document during the editing process.

In all six editors, except Wang, reference to a current line throughout the editing session is very important. Some editors do better than others at keeping the operator informed as to the location of the current line. In IBM's XEDIT the current line is always clearly delineated by the placement of a scaled line under it. When another line becomes current, it always appears immediately above the scaled line. In vi the current line is always identified by the position of the cursor on the screen. The Wang editor does not really have a current line as such. The cursor in the Wang editor identifies the current position or character rather than the current line. In ed, PEDIT and CDC's XEDIT the current line is almost always "the last line affected". That is a pretty fuzzy concept and so is its implementation in these editors. With repeated use the operator using these editors learns to know what line is current without it being explicitly stated. These three editors also provide simple ways to determine the current line if there is any doubt.

The Wang editor and vi have the capability of addressing parts of the document as pages and can display a page of text on the screen at one time if the page is small enough to fit on the screen without overflow. Both of these editors allow for simple actions to display the previous page and the next page in

TABLE 7

COMPARISON OF VIEWING FEATURES

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Current line	Use period (.)	Use TYPE command	Use PRINT command	Displayed above a scaled line	Not line oriented. current position identified by the cursor	Identified by cursor
Page	No	No	No	Only if page coincides with screen size	Yes	Yes
Next page	No	No	No		Yes	Yes
Previous page	No	No	No		Yes	Yes
Next screenful	No	No	No	Use PF8 key	Yes, special function key	Not directly
Previous screenful	No	No	No	Use PF7 key	Yes, special function key	Not directly
Groups of lines	Use PRINT command	Use TYPE command	Move pointer, Use PRINT command	A screenful only	Not directly on lines	Use scrolling feature

the document. The Wang editor uses a GO-TO-PAGE function key to move directly to a specific page.

IBM's XEDIT and the Wang editor allow for requesting the display of the previous or next screenload of text. Both of these editors use function keys to perform this function.

Sometimes an operator is interested in reviewing a specific group of lines in a document. All editors allow for displaying groups of lines but some are simpler and/or more specific than others. To display a specific group of lines in CDC's XEDIT, it is necessary to move the pointer to the first line to be displayed and give a command to print a given number of lines. In some instances this may be a single command. PEDIT and ed have similar procedures for performing this function. Groups of lines can be reviewed by scrolling the document on the screen in vi, IBM's XEDIT and the Wang editor.

COMPARISON OF HARDWARE FACILITIES SUPPORTED

Table 8 reflects a general comparison of the terminals and computers used with the six editors surveyed. Two of the editors (ed and vi) are designed to be used on timesharing computer systems using the UNIX operating system. There are many versions of these editors adapted to several versions of UNIX and developed to support a very large number of terminals. Both ed and vi are used with dumb and intelligent terminals (with or without CRTs). vi works best with intelligent, CRT equipped terminals.

PEDIT and IBM's XEDIT are used on IBM and "IBM-like" computers using the Conversational Monitor System (CMS) operating system in a timesharing environment. PEDIT supports dumb terminals only (with or without CRTs). IBM's XEDIT is designed to work best with IBM 3270 or 3270-compatible terminals which allows the user to take advantage of special function keys and cursor control keys. It can also be used with typewriter type terminals without using the full set of features.

CDC's XEDIT works only on CDC CYBER 70 and 170 and CDC 6000 computers using the Network Operating System (NOS) in a timesharing environment. This editor only supports dumb terminals (with or without CRTs).

Of the six editors surveyed, only one is a stand alone or dedicated text processing system: the Wang editor. The hardware used in this system consists of:

--Work stations (up to 14) with a keyboard (including 21 function keys and 4 cursor control keys) and a CRT screen. The

TABLE 8

COMPARISON OF HARDWARE FACILITIES SUPPORTED

HARDWARE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Terminals	Many different terminals, printer types, dumb terminals, intelligent terminals	Printer type or dumb CRT **and intelligent terminals**	Printer type or dumb CRT	Dumb CRT and teletypewriter terminals. Best used with intelligent IBM 3270 or 3270-compatible terminals	Special purpose Wang terminal with CRT and 21 special function keys and cursor control keys. Intelligent	Many different terminals: printer types, dumb terminals, intelligent terminals
Computer	General purpose, time sharing. Many versions for many systems using the UNIX operating system	General purpose, time sharing. Operates with the CMS operating system	Available on CDC 6000 and CYBER 70 and 170 computers. Uses CDC NOS time sharing operating system	General purpose, timesharing. Operates with the CMS operating system	Dedicated hardware (microcomputer) and software	General purpose, timesharing. Many versions for many systems using UNIX operating system

screen displays 80 characters per line and 24 lines per screenful. Each work station has an internal microprocessor.

-- Microcomputer control-processor which is connected to each work station, the disk drives and the printers.

-- Archive disk drive which uses floppy diskettes that store 120 pages of text.

-- Hard disk which stores 4,000 pages of text.

-- Printers. Normally one dot matrix printer for draft work and one or more letter quality printers for finished work are employed.

Comparison of Features Related to Human Factors

Tables 9 and 10 compare the features in each editor related to human factors. This is obviously not an exhaustive list but was selected because at least one of the editors provides some facility for each of the features. There is no benefit in comparing features which none of the surveyed editors possess even if these features are recognized as being important in text editors.

Meyrowitz and van Dam (Meyr82, p. 327) state "Typing-oriented systems require familiarity with the system language, as well as some expertise in typing. Function-key-oriented systems often have either too few keys, thus requiring keyboard-overloading by binding single keys to several interpretations and necessitating multiple key-stroke commands, or have too many unique keys, resulting in an unweildy keyboard. In either case, even more agility is demanded of the user than by a standard keyboard. The menu-oriented user interface is an attempt to address these problems."

Only one of the editors, Wang, uses a menu selection process at all and it makes only limited use of it. (See Table 9). Since it is a display editor, much of the work on the document can be performed directly by modifying text on the screen. The use of a large number of function keys also obviates the need for continous menu selection.

Wang and IBM's XEDIT are the only two editors which use special function keys for editing. In order for IBM's XEDIT to

TABLE 9

COMPARISON OF FEATURES RELATED TO HUMAN FACTORS

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Use of menus	No	No	No	No	Yes	No
Use of function keys	No	No	No	Yes	Yes	No
Cursor positioning keys	Terminal dependent but not relevant	Terminal dependent but not relevant	Terminal dependent but not relevant	Terminal dependent	Yes	Terminal dependent, uses standard keys otherwise
Different prompt characters or messages	No prompt character is used usually	No	Different prompt characters for edit and input modes	No prompt characters used. Clear text messages indicate mode or give instructions.	No prompt characters used. Clear text messages indicate mode or give instructions.	No

TABLE 9

(Continued)

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Confirmation of location	Not indicated usually	Usually last line displayed	Usually last line displayed	Always displayed	Always displayed	Always displayed
Profiling	No	Yes	No	Yes, with Profile macro	No	Yes
Multiple commands on a single line	No	No	Yes	No	Most commands use function keys and menu selection	No

take advantage of the function keys, the operator must be using an IBM 3270 or 3270-compatible terminal. The function keys on the Wang system are custom designed into the terminal for use with the editor. The author has not found in the literature any research reflecting an evaluation of function keys in text editing.

Even though Meyrowitz and van Dam contend, as quoted above, that "more agility is demanded of the user than by a standard keyboard", function keys appear to be powerful tools. In a study by Roberts (Robe79) using four expert editor users and the editors TECO, Wylbur, NLS and Wang, she performed an experiment where she measured the mean time to perform a group of tasks. The Wang editor proved to be significantly superior to the other three. The Wang editor also ranked high in a subsequent evaluation by Roberts (Robe83). Obviously, a high performance rating of an editor cannot be attributed to a single factor like function keys, but the author suggests that they contributed significantly. Perhaps further research will confirm or repudiate this assertion.

There is considerable difference between the editors in the way the cursor is used in CRT equipped terminals. In the case of ed, PEDIT and CDC's XEDIT, which are line oriented editors, cursor movement is of very little importance since all commands used to edit are typed in on the keyboard. IBM's XEDIT, like the other editors, supports several terminal types. The availability of cursor positioning keys depends on the terminal used. Cursor movement in the IBM editor is more important than in ed, PEDIT and CDC's XEDIT but not as important as in vi and the Wang editor. In vi and Wang the cursor is used extensively and easy cursor positioning is essential to efficient use. Since vi supports numerous terminals, the availability of cursor positioning keys is once again terminal dependent. All Wang terminals are equipped with cursor positioning keys.

Another useful feature in most editors is the ability to prompt the user for input and provide prompting characters or messages to help the user remember "where he/she is". ed is especially weak in this regard. Norman (Norm81) accuses ed of being shy and silent. New users are not quite sure what the silence means. What it usually means is, if something is wrong, it will tell you; otherwise, all is well. CDC's XEDIT displays a different prompt character when it is in the command mode than when it is in the input mode. IBM's XEDIT and the Wang editor do not use prompt characters but provide clear text messages to aid the operator.

Somewhat related to the use of prompting characters and messages is the feature in an editor which keeps the user informed regarding the location in the document where he/she is working. That is, where is the editing pointer pointing or what is the current line or position. As already stated, ed does a poor job in this regard. In PEDIT and CDC's XEDIT the current line is usually the last line displayed and users fairly rapidly become comfortable with this procedure. In IBM's XEDIT, Wang and vi the current location is always displayed and easily identified.

Meyrowitz and van Dam (Meyr82) suggest another important factor that should be considered in editor design: profiling. Three of the editors surveyed, (PEDIT, IBM's XEDIT and vi) provide such a feature. PEDIT allows the user to create and use a special profile file which may contain run commands to define a working environment for a specific user. These run commands may establish different settings than the normal settings for some of the editor's command set. In IBM's XEDIT the user is allowed to develop a profile MACRO which is stored and executed automatically each time XEDIT is invoked. The macros may contain EXEC2 statements, CMS and CP commands, and any XEDIT subcommands or macros. Vi also provides a profiling feature where statements are placed in the EXINIT file of the UNIX environment. Options such as autoindent, autowrite, and terse can be place in the variable EXINIT. When the editor is executed and EXINIT is invoked, the selected options become effective.

Editor users can frequently perform their work faster if several edit commands can be placed on the same line and executed one after the other while pressing the RETURN, ENTER or EXECUTE key only once. Although this seems to be an attractive feature and potentially simple to implement, only one editor surveyed provided this capability, CDC's XEDIT. In the Wang editor multiple commands are entered through function keys.

Two of the editors surveyed allow for changeable response (or feedback) modes, PEDIT and CDC's XEDIT. This means that the editor allows the user to determine the level of feedback that he/she wants. Normally novice users want and need the maximum feedback and response the editor can give during the first sessions of use. When a user becomes more experienced, however, repeated and verbose feedback sometimes becomes annoying. It is a nice option to reduce the amount of feedback received. The options in PEDIT and CDC's XEDIT are not very sophisticated. In

TABLE 10

COMPARISON OF FEATURES RELATED TO HUMAN FACTORS

FEATURE	ed	PEDIT	XEDIT (CDC)	XEDIT (IBM)	Wang OS 30	vi
Display editing	No	No	No	Yes	Yes	Yes
Changeable response modes	No	Yes	Yes	No	No	No
Use of graphics	No	No	No	Yes, (limited use)	Limited number of graphic symbols used	No
Autosave	No	Yes	No	Yes	No	No
Display of unprinted characters	No	No	No	No	Yes	No

PEDIT when the VERIFY option is set ON, lines that are located or changed are displayed, as well as lines to which the pointer moves (the current line) after GET, PUT and DELETE commands. With VERIFY set OFF the current line is not displayed except after TYPE or DISPLAY commands. In CDC's XEDIT the feature is limited to the DELETE command. In VERIFY mode the editor displays each deleted line so the user can verify the editor's action. In the BRIEF mode (opposite of VERIFY) the editor does not display the deleted text.

The six editors surveyed are not known for their extensive use of graphics. As a matter of fact, only two (IBM's XEDIT and Wang) use any graphics display worthy of mention and these editors' use of graphics is very limited. In IBM's XEDIT a graphic scaled line is used to denote the current line. Five equal signs (= = = = =) are used to help the operator easily use the prefix commands. An arrow (= = =>) is used to identify the command line. Multiple asterisks are used in the lines representing top and bottom of the file. The Wang editor provides thirteen graphic symbols used to denote the functions: indent page, center, tab, note, carriage return, decimal tab, merge, do not merge, stop, format, superscript, and subscript.

Two editors (PEDIT and IBM's XEDIT) provide a very helpful feature designed to prevent the operator from editing text for a long period and then losing a lot of work due to a system failure or communication line drop. This feature is called by both editors "autosave". In IBM's XEDIT the operator may use a command

```
SET AUTOSAVE n
```

Where n equals the number of lines of new or changed text entered before the editor automatically saves the document with the changes That is, n is the number of lines to be updated between automatic saves. The original disk resident version of the file is not affected since the automatically saved text is placed in a new disk file to be retrieved in the event the system fails. The PEDIT autosave feature operates almost identical to the IBM editor feature.

The Wang editor has another noteworthy feature not found on the other editors. That is, this editor causes some symbol to be displayed on the screen for each character entered during the editing process. There are no "invisible" characters in the document to cause unexpected results when certain commands are executed or the file is edited on another system. Many of us, including the author, have wasted considerable time with data processing and text processing systems attempting to determine the cause of some unusual behavior of the system to find out that some "unprintable" character was causing the anomaly. Unfortunately, the explanation for the situation is usually buried in some obscure footnote of a manual that is not normally accessible to the user.

Advantages and Disadvantages of the Stand-Alone Editor

It is theoretically possible to eliminate all differences between so called stand-alone or dedicated text editors and editors which use the resources of a large computer along with other non-text editing jobs. At least this is possible insofar as the user is concerned. At present, however, if the Wang editor is representative of stand-alone editors and the other five editors surveyed are representative of timesharing editors, then there appears to be some advantages to the stand-alone system. All resources in a stand-alone system are reserved for the text editing process. Therefore, hardware selection, operating system features and memory type and size can all be customized to provide the most effective and efficient base for text editing. This may not happen but the potential exists. There are potentially more system resources available for formatting, prompting and graphics. The Wang editor is at least as good as the others in all these areas and exceeds most of them. Special hardware in the Wang system allows 21 special function keys which are not possible with editors designed for general computer systems with standard terminal devices. (vi is a good example of keyboard overloading where certain keys on a standard keyboard are used in conjunction with the control key to operate like function keys.) The Wang system provides the user with numerous labeling and filing conventions not available in the other editors. A repagination feature, automatic wrap-around without word splitting, use of superscripts and half spacing for mathematical equations are still other attractive features found in the Wang editor. Finally, the Wang editor allows for merging two files while printing which is useful, for example, if the text to a letter is in one file and addresses for multiple copies of the letter are in another file.

Stand-alone systems obviously have several disadvantages also. Since they are designed for special hardware and operating systems, they are not very portable. Editors like ed and vi, for instance, operate on many different computers and with many different terminals. This is obviously not possible with a stand-alone editor. If a user wants to use a stand-alone system, he/she must buy or lease the whole package even if several large systems are already available. With ed and vi, if you use the UNIX operating system, you can use these editors with little additional fuss and expense. Additionally, the Wang system does not provide the user any data processing support. If a time-sharing user of a text editor wants to switch from text editing to data processing, it is usually not a major effort to do so. With the Wang editor, it is impossible.

CHAPTER 4

FUTURE IMPROVEMENTS OF INTERACTIVE TEXT EDITORS

As was stated earlier everyone who produces or uses text editors has their own ideas regarding what defines the ideal editor. (Embl81). As with programing languages, the desirable characteristics of a text editor are largely determined by individual, subjective preferences. However, Meyrowitz and van Dam (Meyr82a) have presented suggestions for future editors worthy of note. These suggestions are based on over ten years of study in this area. Future editors should:

1. Be developed based on a well defined, consistent conceptual model that will allow the user to be familiar and comfortable with the philosophy behind the system.

2. Provide easily understood on-line and off-line help facilities which explain the conceptual model and details of the user interface and functions of the editor.

3. Provide an easy to use interface for editing text, pictures and voice. The interface should be so attractive that authors can and will use the system without the help of technicians or secretaries.

4. Provide for an infinite undo and redo capability which will allow users to experiment without fear of serious loss of text.

5. Respond immediately without delay except for

rare exceptions.

6. Provide complete editing facilities so that an author can perform all the functions on an "electronic document" that he/she can perform with pen, ruler, scissors and tape.

7. Allow for global substitution of strings of text, replication of standard headings, phrases or paragraphs and automatic renumbering of sections or references after (or while) the document is being edited.

8. Provide controlled access to other files and computer resources.

9. Provide a simple ability to mix targets such as text, graphics, programs and forms.

10. Provide the ability to simply edit a document in its near final form.

None of the editors discussed in this paper come close to this ideal. Most of the editors do provide good response times. When they do not, it is normally a function of the computer or communication system and not due to editor caused delays. Related to all the other suggestions the surveyed editors are inadequate.

Embley and Nagy (Embl81) suggest several improvements in future editors related to human factors:

1. Users strongly prefer screen-oriented editors to hard-copy-only versions.

2. Editors should provide more user support in the area of error detection and correction.

3. Scanners may be used to load the original document in cursive handwriting, hand printing, typed copy, line drawings and continous pictures.

4. Editors should take advantage of new technology in color graphics and audio input and feedback.

The most advanced text editors commercially available today are the Xerox Star and the Apple Lisa.

The Xerox Star, announced in April 1981, is a personal computer designed for office professionals. It is advanced relative to other editors in its use of graphics. The graphical displays and the use of a mouse for "command selections" dramatically reduce the number of operations using the keyboard. Star is the first commercially available text editing system to make extensive use of icons. Icons are pictures of objects that can be manipulated by Star through the use of the mouse and the keyboard. An excellent description of the Xerox Star is contained in reference Smit82.

The only system that compares with the Xerox Star is Apple's Local Integrated Software Architecture (Lisa). Lisa was developed at a reported cost of $50 million and over 200 person-years of work. It sells for $10,000 which includes a computer with a one megabyte memory, two floppy disk drives, one 5¼ inch Winchester hard disk and six application programs. According to a recent article in Byte (Morg83), the Lisa editor contains four major design elements:

1. A graphics-mouse orientation.
2. A "desktop" metaphor conceptualization
3. A "data-as-concrete-object" metaphor
4. An integrated design of the hardware and software

The graphics-mouse orientation of the editor solves a number of the problems associated with the use of a keyboard and the memorization of a large number of commands. Like Star, Lisa displays icons or pictured symbols which are accessed by using the mouse to move the cursor on the screen. When the icon has been "located" by the mouse, it can be "selected" by pressing a button on the mouse. This selection causes the execution of application programs which perform a requested function. Apple rejected light pens and touch-sensitive video panels in favor of the mouse. The Apple mouse has only one button instead of two or three used in other systems. Pointing to pictures is simple (and perhaps more fun) than keying in some abstract command code. With Lisa the user can do what he/she wants to largely by pointing and pushing the mouse button. Lisa also provides cursor positioning (arrow) keys on the keyboard so user's fingers do not have to be removed from the keyboard to the mouse to position the cursor. Instead of providing a mouse or cursor positioning keys, Lisa provides both. Lisa provides other graphics support in the Lisa Draw and Lisa Graph application software packages.

The "desktop" metaphors conceptualization pertains to the idea that the video display terminal acts as an "electronic desktop". That is, almost everything that can be placed on a desktop can be displayed on the screen in a similar form. Lisa even allows the screen to display overlapping documents in much the same way as documents overlap on an actual desktop. The objects on the screen are manipulated in much the same way as papers, reports, files and mail are manipulated at a desk. Additionally, each peripheral component (printer, disk drives, etc.) in the system is represented by an icon. Selection of an icon is often followed by the display of all or part of the object that the icon represents. For example, if a letter icon is selected, the text of the letter will be displayed. The text can then be manipulated through the use of the mouse and the keyboard. Lisa also makes very good use of temporary menus.

The "data-as-contrete-object" metaphor is said to aid in demystifying the editing process. The user works with objects that look like reports, folders and conventional files. Allegedly he/she feels more secure and comfortable with these objects. Through the use of the icon symbols, data files and data bases appear less abstract to the user.

The hardware and software are obviously integrated to produce a powerful tool. A discussion of how this integration came about can be found in the February 1983 issue of Byte (Morg83).

The six application programs in Lisa raise interactive text editing to a new level. The applications are:

Lisa Draw--uses pop-up menus to allow the user to create all sorts of graphics images: lines, boxes, circles, ellipses, arcs, polygons. Text can be added in 352 distinct kinds of type. Drawings can cover up to 25 pages.

Lisa Write--the system's word processor. The user uses the mouse and keyboard to add, change, delete and move text.

Lisa Project--a managerial tool designed to use PERT, Gantt charts, and task charts to keep track of scheduled projects and personnel.

Lisa List--a data base that allows records with up to 100 fields and 1,000 bytes. Manipulation of the data base takes place right on the screen. Eight data types are permitted and data is type checked when entered.

Lisa Calc--performs spread sheet calculations that can be used as stand alone documents or "cut and pasted" into other documents prepared by Lisa Write.

Lisa Graph--uses data from other applications or data entered by the user to produce bar charts, line charts, scatter diagrams or pie charts almost instantly. It has very simple features for labeling the charts and provides 36 varieties of shading for sections of the charts.

Lisa is just now hitting the market. Time will tell what the user response will be. Many revisions are already promised. At $10,000 a shot, it better be good!

(The information for the above paragraphs related to Lisa were taken from references Will83 and Morg83.)

CHAPTER 5
CONCLUSION

The offices of United States businesses and other organizations are now experiencing dramatic change related to the expanding use of automated office equipment. Some offices have used automated office systems for years; others are only beginning. A major component of most of these systems is an interactive text editor which may be embodied in a stand-alone word processing system or may be included in the timesharing facilities of a medium to large scale computer system. In many instances the text editor is the principal interface with the computers.

Interactive text editors allow users to create, modify, rearrange, and delete text documents consisting of letters, reports, books, computer programs and other text forms. Users are furnished with editor commands and devices (including keys, joysticks, and mice) to move through the document for reviewing or specifying a place where the document is to be modified. Numerous modifications are allowed to the document once it is created. Additional characters, lines, paragraphs, or sections may be inserted at the beginning, at the end or somewhere within the original document. Text, in small or large portions may be moved from one place to another in the document or from one document to another. Characters, lines, paragraphs and sections may also be deleted from the document through the use of simple commands. Many other features are also provided, such as automatic line centering, page numbering, indentation, right justification and autosave.

There are many different text editors; one article references more than 30 and this includes probably less than half of the editors in use today. This paper compares a number of features in six editors: ed, PEDIT, CDC's XEDIT, IBM's XEDIT, Wang OS 30, and vi. Three of these (ed, PEDIT and CDC's XEDIT) are relatively unsophisticated line-oriented editors. IBM's XEDIT, the Wang editor and vi are display editors which display a full screenful of text throughout the editing session. The Wang OS 30 word processing system is the only stand-alone system discussed.

As this paper indicates, all six of the editors are different in several aspects. This paper does not attempt to evaluate the editors but to indicate the difference in their features. Evaluation is performed subjectively by everyone who uses text editors but very few scientific evaluations have been attempted. (See Robe83 and Embl81 for reports on two evaluations that have been performed.)

Everyone who has used text editors can suggest ways they can be improved. Several authors have documented their suggestions including Meyrowitz and van Dam (Meyr82b). Improvements related to the ergonomic aspects of editors have been suggested by Embley and Nagy (Embl81). Both sets of these suggestions are discussed in Chapter 4 of this paper. The two most advanced editors are components of the Xerox Star and the Apple Lisa. It remains to be seen how many of the features in these two systems will become commonplace in the editors of tomorrow.

Given the literature related to interactive text editors, there is a great deal yet to be researched and implemented. There is also a lot to be done in providing the public and academia with tools for understanding the features and differences in text editors that already exists. Several articles referenced in this paper have attempted to do this. This Master's Report is also an attempt to make information related to six text editors readily available to the faculty and students at Kansas State University.

GLOSSARY

Autosave	A feature used by some editors (e.g. PEDIT and IBM's XEDIT) which causes an automatic write to storage of the modified document after a given number of lines have been added or changed.
Browsing	A technique employed by some editors (e.g. FRESS) which allows the user to move around in the document in a random fashion. (see hypertext). (Meyr82a, p. 339).
Cancel	The cancel command allows a user to cancel the command that is currently being executed (Meyr82a, p. 348).
Case insensitive	Pattern searching is case insensitive if the target string is matched by text strings without regard to the case (upper or lower) of the letters in the strings.
Clipboard	Temporary storage in the Lisa system used to hold text during the editing session. (Will83, p. 33).
Command line	A specific line in some display editors (e.g. IBM's XEDIT) where editing commands can be entered.
Components of editors	Traveling component, editing component, viewing component, and display component. (Meyr82a, p. 330).
Copy	A command used in editors to replicate text in one position of a document in another position in the document. The original text is not affected.
Current line	The referenced line where addition or changes will take place (or begin) in line oriented editors when a command is used.
Cursor positioning keys	Special keys on some terminals (normally labeled with arrows) which are used to move the cursor on a CRT screen.

Data tablets	A flat, rectangular, electromagnetically sensitive panel used with a stylus or a puck. As the stylus or puck is moved across the data tablet, the coordinates are returned to a system program which uses them to cause a display on the CRT screen. (Meyr82a, p. 324).
Desktop metaphor	A concept used by the Xerox Star and Lisa systems to simulate a desktop on the screen of a CRT equipped terminal.
Display editor	An editor which uses a CRT equipped terminal. These editors usually display a full screen of text and the user is allowed to modify text by moving the cursor to the point of addition or change and entering the new or corrected text directly.
Editing buffer	A portion of memory that holds all or part of a document during the editing process.
Ellipsis type search	Pattern searches that allow the operator to identify the beginning and ending characters in a pattern without typing in the entire pattern.
Filtering	A process in the viewing component of an editor which determines what part of a document will be displayed at any point in time.
Function keys	Special keys on some terminals used to control the editing process. These keys obviate the use of typed in commands using the standard keys. (e.g. the Wang editing system has 21 function keys.)
Glass teletypes	A name given to CRT equipped terminals which are used like a teletypewriter with the output displayed on the screen instead of being printed. The use of terminals in this manner does not take advantage of their full capability since text cannot be modified directly by moving the cursor and entering additional or changed text.
Global changes	A feature in some editors that allows for a single change to a character string to be repeated on every matching string in the document.

Hypertext	The hypertext feature, found in some editors, allows for the establishment of semi-permanent or permanent paths through the document or between documents. Through the use of "tags" and "jumps" the user can browse through the document on'line. (Meyr82b, p. 339).
Icon	A pictorial representation of a Star or Lisa object that can exist on a desktop. Data icons represent objects on which actions are to be performed, e.g. documents, folders, or record files. Function icons represent objects that perform actions. (Smit82, p. 519).
Intrafile movement	Refers to movement of text from one place in a document to another place in the same document. (See Copy and Move.)
Interfile movement	Refers to movement of text from a document in one file to a document in another file.
Label	Labels are character strings inserted in the text which allow travel through a document by using the labels as an address in a command. They are called tags or notes in some editors. (Meyr82a, p. 338).
Light pen	A light sensitive device used to point to text or other objects on special terminal screens.
Light gun	A light pen mounted in a pistol-grip mount, with the somewhat awkward switch on the original light pen replaced by a trigger. (Embl81, p. 61).
Line-oriented editor	Interactive editors which allow users to create or modify documents by referencing lines within the document. Lines are usually given invisible line numbers which are used for direct movement through the document or as addresses in commands.
Macro	Macros are files of commands and/or statements which can be created, filed and executed as a single command. Macros expand the basic editors language and reduces repetitive tasks. (IBM80, p. 7-1).

Menu	A menu in an editing system is (like a menu in a restaurant) a list of options available to the user. The use of menus tends to reduce the number of commands in an editor interface language and reduce the number of function keys required for a terminal. (Meyr82a, p. 330).
Modeless	The modeless environment is an ideal proposed in reference Meyr82b. The contention is that editors should have as few modes as possible to reduce the number of restrictions on the operations of the editor normally imposed by a large number of modes. It is recognized that no system can be entirely modeless. (Meyr82b, p. 403).
Mouse	A pointing device used to move the cursor on the screen of a terminal. It is about the size of a package of cigarettes with a ball on the bottom that turns as the mouse slides over a flat surface. Electronics sense the ball rotation and guide the cursor in corresponding motions. Mice come with one, two or three buttons which are used as functional keys. (Smit82, p. 518
Move	An editor command which causes text in one part of a document to be moved to another part of the same document. The text in the original part is deleted.
Pattern searching	An operation in the editing process where a target string is specified by the user; the editor then searches through the document and attempts to match the target string with text. If a match is found, an insertion, deletion or change command may be executed. Selective pattern searches locate a specified number of occurrences of a target string. Global pattern searches locate every occurrence of a target string in an entire document.
Pointers	Constructs implemented in the editor software used to identify the area of the document selected for editing and/or viewing. Pointers are transparent to the user but are moved through the document by user entered commands or by movement of the cursor on display editors.

Prefix commands	Commands used in the IBM XEDIT editor. These commands are executed by placing single or double character commands in the left margin of the screen and pressing the ENTER key.
Profiling	A feature in an editor which allows the user to "tune" the editor environment when the editor is invoked. This allows preferred environment settings to be handled automatically and removes the need for all users to accept the normal defaults. (Meyr82a, p. 350).
Property sheet	A screen displayed "sheet" which allows the user to review and/or modify the properties assigned to objects in the Star editor. Objects include text characters, paragraphs, graphic lines and icons. Properties include style, size, and face of text characters; indentation, leading, and alignment of paragraphs; and name, size creator, and creation date of icons. (Smit82, p. 523).
Puck	A 3 inch square by 1 inch high device that fits in the palm of the hand and is moved over the surface of a data tablet. (See Data tablet.) (Meyr82a, p. 324).
Recursive editing	A process in at least one editor (PEDIT) which allows the user to invoke the editor to address a second file without losing the first file.. When the user "quits" the second file, editing resumes on the first file. Files can be edited in this fashion up to 9 levels deep. (Perk79, p. 24).
Redo	A facility provided by some editors to re-perform editing operations that have been "undone" by an UNDO command. (See UNDO.) (Meyr82a, p. 348).
Rings of files	In IBM's XEDIT multiple files are kept in a "ring". Each time a new file is used in a command, the file is added to the ring with the previous file pointing to it. The latest file added points to the first file addressed, thus creating a ring. This procedure allows for commands to be used without operands when the files are used in the sequence of the ring. (IBM80, p. 5-1).

Stand-alone editing systems	These editing systems are dedicated to text processing or word processing and are not used for data processing or anything else. (e.g. the Wang OS 30.) These systems are usually comprised of customized computers, customized terminals and customized software. They are contrasted with editors which are used in a timesharing mode on a large system.
Stream editors	These editors address target documents as a single stream of characters as opposed to individual lines of text. Each character in the document has an invisible number which is often referenced in editing commands. Although lines are not referenced in the editing process, the editor provides for line construction to accomodate display terminals and printing requirements. TECO is a stream editor. (Meyr82a, p. 334).
Structure editors	Editors which conceptualize the target document as specific structures (hierarchical trees, computer programs, etc) as opposed to simply linear strings of characters or lines. Operations on the documents are based on the structures as well as the content of the document. NLS was the first structure editor developed. (Meyr82a, p.335)
Target document	The object manipulated by an editor. Target documents include letters, reports, computer programs, tables, diagrams and other art work. (Meyr82a, p. 322).
Timesharing editors	Editors which operate on larger computers and generally use the same resources used for data processing operations and at the same time data processing operations are taking place. Usually they are used with general purpose terminals.
Touch screen panels	Terminal display devices which allow the user to enter instructions and data by pressing pictorial representations on the screen.
Traveling	A major component in a text editor which provides for movement through a document. Traveling may be addressed to specific characters, lines, paragraphs, pages, selections, labels or screenful. (Meyr82a, p. 337).

Undo	A critically important, time-saving feature in some editors. The most basic version allows the user to undo the last command issued. More useful systems have an n-level UNDO stack that allows the user to undo commands n levels back (sometimes to the beginning of the session or even back to previous sessions.) The UNDO feature frees the user from the burden of making sure that each command does exactly what is wanted by guaranteeing that any result can be undone. (Meyr82a, p. 348).
Window	Windows are features offered in the Star and Lisa editors. They are rectangular areas that display the contents of icons on the screen. On the Star system only six windows can be displayed simultaneously and they do not overlap. Lisa does allow for overlapping windows. (Smit82, p. 523).
Wordwrap	An editor feature which eliminates the need for typing a carriage return at the end of each line. When the editor senses that the word currently being typed has exceeded the right margin, it breaks the line at the first blank space before the overflowing word and pushes it to the next line. (Meyr82a, p. 345).

REFERENCES

CDC79 Control Data Corporation, XEDIT Version & Reference Manual, St. Paul, Minn., 1979.

Coul76 Coulouris, G. G., "The Design and Implementation of an Interactive Document Editor", Software Practice and Experience, April-June 1976.

Deut67 Deutsch, P. L. and Lampson, B. W., "An Online Editor", Communications of the ACM, Dec 1967.

Embl81 Embley, David W. and Nagy, George, "Behavioral Aspects of Text Editors", ACM Computing Surveys, Mar 1981.

Gain78 Gaines, B., "Programing Interactive Dialog", Pragmatic Programing and Sensible Software, Online Conferences Ltd, 1978.

Giul82 Giulian, Vincent E., "The Mechanization of Office Work", Scientific American, Sep 1982.

Hans71a Hansen, W. J., "User Engineering Principles for Interactive Systems", Proceedings of the Fall Joint Computer Conference, 1971.

Haze80 Hazel, P., "Development of the ZED Text Editor", Software Practice and Experience, Jan 1980.

IBM80 International Business Machines, Systems Product Editor User's Guide, Endicott, New York, Jul 1980.

Jone78 Jones, P. F., "Four Principles of Man-Computer Dialog", IEEE Transactions, Dec 1978.

Joy80 Joy, William, "An Introduction to Display Editing with vi", University of California, Berkeley, Sep 1978.

Kern78 Kernighan, Brian W., "A Tutorial Introduction to the UNIX Text Editor", Bell Laboratories, Murray Hill, New Jersey, Sep 1978.

Mart73 Martin, J., Design of Man-Machine Dialogue, Prentice-Hall, Englewood Cliffs, N. J., 1973.

Meyr82a	Meyrowitz, Norman and van Dam, Andries, "Interactive Editing Systems, Part I", ACM Computing Surveys, Sep 1982.
Meyr82b	Meyrowitz, Norman and van Dam, Andries, "Interactive Editing Systems, Part II", ACM Computing Surveys, Sep 1982
Morg83	Morgan, Chris; William, Gregg; and Lemmom, Phil, "An Interview with Wayne Rosing, Bruce Daniels and Lang Tesler", Byte, Feb 1983.
Olso82	Olson, Margrethe H. and Lucas, Henry C., "The Input of Office Automation on the Organization: Some Implication for Research and Practice", Communications of the ACM, Nov 1982.
Perk79	Perkin-Elmer Corporation, PEDIT VM/370 Editor Manual (Updated at KSU Computing Center, Dec 1979).
Robe79	Roberts, Teresa L., "Evaluation of Computer Text Editors", Ph.D. dissertation, Dept. of Computer Science, Stanford Univ., Stanford, Calif, Nov 1979: also Xerox Research Report SSL79-9, Palo Alto, Calif 1979.
Robe83	Roberts, Teresa L. and Thomas P. Moran, "The Evaluation of Text Editors: Methodology and Empirical Results", Communications of the ACM, Apr 1983.
Rous75	Rouse, W. B., "Design of Man-computer Interface for On-line Interactive Systems", Proceedings of the IEEE, Jun 1975.
Smit82	Smith, David C.; Irby, Charles; and Kimball, Ralph, "The Star User Interface: An Overview", in Proceedings of the National Computer Conference 1982.
Teit79	Teitelman, W., "A Display Oriented Programmer's Assistant", International Journal of Man-machine Studies, Mar 1979.
Thom82	Thomas, Rebecca and Yates, Jean, A User's Guide to the UNIX System, Osborne/McGraw Hill, Berkeley, Calif 1982.
VanD71	Van Dam, A. and Rice, D. E., "On-line Text Editing: A Survey", Computing Surveys, Sep 1971.
Wang78	Wang Laboratories Inc., Wang Word Processor Operator's Guide, Lowell, Mass. 1978.

Wass73 Wasserman, A. I., "The Design of Idiot-proof Interactive Systems", Proceedings of the National Computer Conference, 1973.

Will83 Williams, Gregg, "The Lisa Computer System", Byte, Feb 1983.

BIBLIOGRAPHY

Chamberlin, D. D., Bertrand, C. F., Goodfellow, M. J., King, J. C., Slutz, D. R., Todd, S. J. P., and Wade, B. W. "Janus: An Interactive Document Formatter Based on Declarative Tags", IBM Systems Journal, International Business Machines Vol 2, No 3, 1982.

Goodman, Jana T., "The Design, Implementation, and Use of LEDIT: A Real Time Editor for Lisp", KSU Master's Report, 1982.

Heidorn, G. E., Jensen, K., Miller, L. A. Byrd, R. J. and Chodorow, M. S., "The Epistle Text-Critiquing System", IBM Systems Journal, International Business Machines, Vol 2, No 3, 1982.

Maclean, M. A. and Peck, J. E. L., "CHEF: A Versatile Portable Text Editor", Software Practice and Experience, May 1981.

Noot, Han, "Structure Text Formatting", Software Practice and Experience, Jan 1983.

Peck, J. E. L., and Maclean, M. A., "The Construction of a Portable Editor", Software Practice and Experience, May 1981.

Stallman, Richard M., "EMACS The Extensible, Customizable Self-Documenting Display Editor", AI MEMO 519a, Massachusetts Institute of Technology Artificial Intelligence Laboratory, Mar 1981.

Thimberly, Harold, "A Text Editing Interface: Definition and Use", Computer Languages Vol 7, No 1, 1982.

COMPARISON OF SIX INTERACTIVE TEXT EDITORS

by

RUSSELL WAYNE SOWELL JR

B.G.S., University of Nebraska at Omaha, 1969
M.B.A., Old Dominion University, 1974

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1983

COMPARISON OF SIX INTERACTIVE TEXT EDITORS

Russell Wayne Sowell Jr
Kansas State University
Manhattan, Kansas 66506

Abstract

The use of interactive editing systems has become an important feature in many business and other organization offices throughout the United States. Although the literature related to text editors is sparse, several recent articles provide valuable discussions related to the subject. A brief summary of these articles are included in this Report.

Text editors perform three major functions referred to as traveling, editing, and viewing. These three functions are implemented in a variety of ways in the six editors surveyed by this paper. A comparison of the traveling, editing and viewing features in each editor is provided. A comparison of the hardware supported by the editors is also provided. Finally, twelve features related to human factors are compared.

Everyone who has used text editors can suggest ways they can be improved. Several authors have documented their suggestions including Meyrowitz and van Dam (meyr82b). Improvements related to the ergonomic aspects of editors have been suggested by Embley and Nagy (Embl81). Both sets of these suggestions are discussed in Chapter 4 of this paper. Future editors will quite

likely be influenced by the advanced features in the Xerox Star and Apple Lisa systems. These two systems are also discussed in Chapter 4.