KEDIT: A POWERFUL TEXT EDITOR FOR POST-PROCESSING SEARCHES

by Thomas E. Wolff
Amoco Corporation

The online information search generally begins with determining search requirements, continues with search strategy formulation and the online search, and concludes with review and presentation of search results [1-2]. Each step may be time-consuming, but the most repetitive ones and, therefore, most amenable to streamlining, are the post-search processing steps.

We have found that the text editor KEDIT is invaluable for editing captured search sessions and formulating search strategies for uploading to the online service. KEDIT is especially well suited for editing lengthy text database output, both because of the power of KEDIT and the regularity of the search session. Search output often consists of discrete citations that are separated from each other by blank lines, as if they were individual or groups of "paragraphs," and the internal structure of the citations often contains regular, ordered information fields. In addition, there may be regularly-occurring extraneous information, including logon and logoff text or undesired citations, fields or online database descriptors. While KEDIT is generally used by programmers, the reasons for their enthusiasm are at least equally valid for searchers. The characteristics, uses and advantages of KEDIT for post-search processing are presented in this article.

KEDIT (pronounced *KAY-edit*) is a full-screen text editor that runs on DOS- or OS/2-based personal computers and works with standard DOS ASCII text files. KEDIT is a rare program in that it is a remarkably improved implementation of an IBM mainframe program (CMS XEDIT editor). This lineage eases the migration of users to the personal computer from the mainframe. All new users can become familiar with KEDIT using the informative tutorial provided with the program. Thus, much of the power of KEDIT is readily accessible to the novice user.

We have found that the text editor KEDIT is invaluable for editing captured search sessions and formulating search strategies for uploading to the online service.

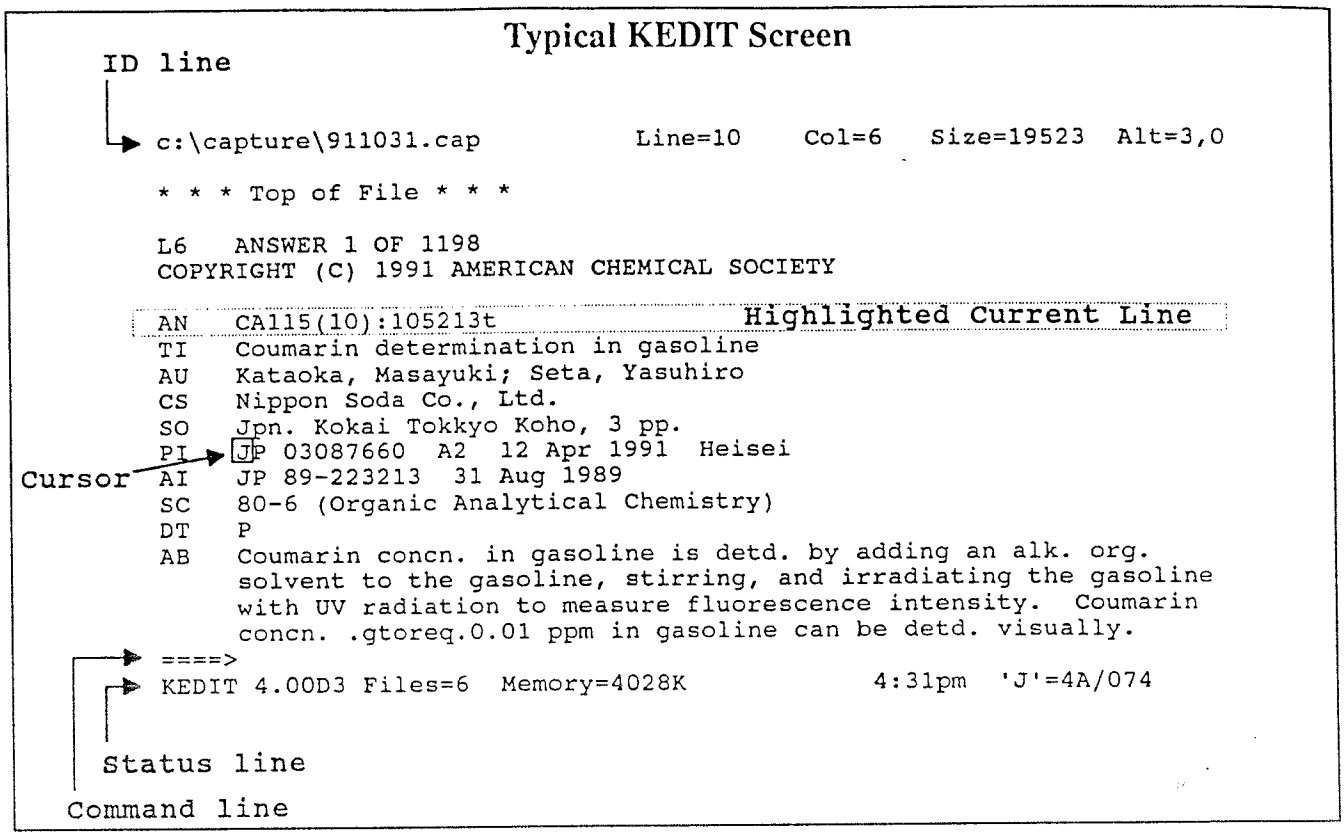
The *full* power of KEDIT comes from understanding its many advanced commands and features and using macros. The macro language of KEDIT is KEXX, a subset of Personal REXX language, although REXX macros are also fully supported by KEDIT. (REXX is an extremely flexible, general utility language, comparable in power and application to BASIC language, but with particular strength in interpreting and manipulating character strings.) Documentation is provided in an informative KEDIT User's Guide and a comprehensive Reference Manual. Technical support is available by telephone, on the Mansfield Software Group bulletin board, on CompuServe and on BIX. In addition, a news bulletin is published approximately twice a year for licensed users.

KEDIT BASICS

The default screen layout of KEDIT (see the figure) illustrates some KEDIT features. The file area contains the contents of the subject file and lies between the ID line at the top of the screen and the command line near the bottom. The file contents are marked with top-of-file and bottom-of-file indicator lines. The arrow and <Tab> keys move the cursor as expected; the

→

Typical KEDIT Screen



<Home> and <End> keys move the cursor to the command line and the end of the cursor line, respectively. Text entry mode is toggled between typeover and insert with the <Insert> key. Characters may be deleted with the <Delete> or Backspace keys. One shifts the file contents through the file area "window" with the <Page-up> and <Page-down> keys. The <Control> key also enhances the functions of most of the keys described above.

A critical feature of KEDIT is the command line from which most KEDIT commands may be invoked. The <Enter> key causes the typed command to be executed when the cursor is at the command line; otherwise the <Enter> key moves the cursor in the file area down to the beginning of the next line. These commands may be basic KEDIT functions or extensive user-written macros. Standard KEDIT commands include cursor movement, text editing (change; find and locate; copy, move, add, delete or sort lines; block manipulation), editor configuration, page formatting, printing, and many DOS commands. Another feature is that commands from the command line act upon the current line, usually highlighted near the middle of the file area. The current line also provides a means of improved scrolling through a file, as explained below.

KEDIT commands and macros are frequently best invoked from function keys (e.g., <F1> - <F12>) and combination keystrokes (e.g., <Ctrl><A>, <Alt><1>, <Shift><F6>). These keystroke commands usually apply to the cursor-containing line, but may act on the current line as the cursor location and the command or macro definition require. Some keystroke commands are informative or global: <F1> is help; <F3> quits the current file; <F6> redisplays the previous command stored in a 15 command recall buffer; <F9> reissues the previous command. An additional means of entering commands is by the prefix area, a limited-value artifact of the conversion from the mainframe XEDIT program.

Most information about the current KEDIT session can be learned by the Status command or from lines on the KEDIT screen. The ID line shows the full path of the edited file, the cursor position by file line and column number, the total number of lines contained in the file and the number of alterations since the file was saved last. Included in the status line are the number of files being edited in this one KEDIT session, available RAM (random access memory), time of day and the value of the cursor character in ASCII and hex. The Status command provides current information about 64 KEDIT settings, but only a few are of interest normally.

KEDIT STRENGTHS

KEDIT can handle very large files and multiple files in each session, and it is fast. All files are loaded directly in system RAM. Up to twenty files may be edited in one session. File management is facilitated by creation of a directory file using the DIR command; any listed file may be loaded in memory by placing the cursor on that line and entering <Alt><X>. The total size of all edited files is limited to 8MB in DOS using EMS LIM 4.0 RAM or 16MB in OS/2. The session in the figure has a file of 19523 lines and a total of six files in the session. Files totalling about 100,000 lines can be edited with 5MB RAM and a memory management program. Since each file is read entirely into RAM memory, simple movement throughout files is essentially instantaneous. Simple editing commands are also very fast. For example, an entire 100,000 line file can be searched for a five-letter word in about 17 seconds using

a Compaq 386/25 MHz computer and KEDIT as a DOS session in Windows 3.0.

The document review and editing process is enhanced by the powerful KEDIT locate and change commands. The commands have in common their reference to some file location that is associated with a target. Targets may be absolute or relative numbers of lines or columns, such as the eleventh column of the file or back thirty lines or from the current line through the end of the file. The most important targets are the character string targets, as for locating the line containing "US 3678124" or the column in which begins the word "Journal." A powerful locating command is the **ALL** command. It searches for all lines containing the specified target and then it uses KEDIT's selective line editing facilities to display only those lines. This is especially useful in bringing together lists of important data, such as database accession numbers, patent numbers or classification codes, or registry numbers. These lists can then be sorted for display or used to prepare search strategies for uploading. Some Boolean logic can also be applied to targets; the available operators are **AND, OR, NOT** or combinations of these. For example, lines can be located that contain any of several targets, or some targets but not others. Searching backward through the file is accomplished by preceding the locate command with a hyphen. The change command can be specified to replace one string target with another in the current line or any specified groups of lines. The selective change queries the user about changing the located string target or searching for additional targets.

Although KEDIT is not a word processor, we find it satisfactory for editing search reports because it contains most standard word processing functions except those controlling font changes or headers, footers and pagination. KEDIT defines three types of text blocks: line blocks consist of one or more consecutive lines; box blocks are rectangular areas of text; and string blocks are streams of consecutive characters spanning one or more lines. The blocks may be copied, moved, deleted or otherwise acted upon by commands such as lowercase or uppercase. Another group target is the paragraph, which is defined as text between two blank lines. Although this definition may be modified, it is especially useful for search output in which citations are generally separated by blank lines. Paragraph text may be "filled" to fit neatly within the designated left and right margins using the **Flow** command. One may choose to "right justify" the text, although KEDIT just adds extra spaces to fill each line since the font is nonproportional. Lines of text may be centered or adjusted to right or left margins. For printing reports from KEDIT files, one may use print facilities from the common DOS utilities programs, such as Norton Utilities or Central Point Software's PCTOOLS, to add headers and pagination. For the few instances in which a full-featured word processor document is required, the edited KEDIT file can be imported as an ASCII file into most word processing programs. Word processor control characters can be added to the KEDIT file to enhance the importation process.

Although KEDIT is not a word processor, we find it satisfactory for editing search reports because it contains most standard word processing functions except those controlling font changes or headers, footers and pagination.

Almost any function or setting may be customized in KEDIT. A profile may be established, which is invoked every time KEDIT is run, or settings may be changed for the current session only. With few exceptions, each keystroke combination can be substituted with simple commands or custom macros. The screen layout may be modified extensively and each area's color can be changed. For example, the command line may be moved to the top of the screen, the current line raised to maximize the amount of citation viewable when the first line of the citation is made the current line, or the colors of the file area and command lines changed to make them more palatable. In addition, the screen may be divided into two through eight different sections for viewing multiple files. It may be divided vertically, horizontally or in both directions. Over 80 Set commands are available, although the default settings are generally satisfactory for effective operation.

KEDIT is good for the typical user without programming experience, and is especially powerful when KEXX or REXX macros are invoked. (Examples of macros useful for editing search reports will be given later.) Macros may be assigned to keystrokes or called from the command line. Some simple macros may just augment default functionality, such as having the <End> key toggle the cursor between the beginning and end of the text line. Other macros may involve extensive programming operations. The latter are usually called from the command line, although there may be advantage to assigning these macros to keys via the KEDIT Profile.

The KEXX macro language has many features:

- expressions (character strings, variables or numbers);
- operators (arithmetic, comparison or concatenation);
- instructions (conditional, looping);
- functions (accessing arguments passed to it; parsing character strings; determining the case or length of character strings; changing case of character strings);
- query and extract operands, which can obtain information about almost any relevant editor or computer setting or contents of any screen area (e.g., the current or command line).

Macros can also call other macros, in effect allowing subroutines. Especially important features are the full support for **IF . . . THEN . . . ELSE** conditional operators, simple and augmenting-variable **DO** loops, and string parsing. The latter allows, for example, determination of the value of any word or character in the current line, which itself may have been selected as a result of a locate command. Subsequent action could then be determined on the basis of the text so evaluated. The reader is referred to the excellent manuals accompanying both KEDIT and Personal REXX for further details.

→

KEYSTROKE MACROS

All KEDIT macros are created by programming in KEXX (or REXX) language, and no recording or "learn" function, commonly found in word processing programs, is available. Thus, while creation of keystroke macros in KEDIT may require more *initial* investment in learning the language, the resulting macros are very easy to maintain. The distinction between keystroke macros and command line macros in KEDIT is mostly a matter of convenience. Those functions called frequently may be conveniently assigned to keystrokes. Our usual practice has been that macros requiring user input of parameters are called from the command line where parameters can be added immediately after the macro name. This could also be accomplished using the KEXX Readyv command, imbedded in a keystroke macro, for accepting user input during execution of the macro. Keystroke macros are often bundled in files with extension KML (KEDIT Macro Library), such as the PROFILE.KML file invoked when KEDIT is first called.

The following are keystroke macros used by us in post-search processing:

Scrolling by paragraph. The <Ctrl><D> and <Ctrl><S> are defined in the PROFILE.KML for scrolling down to the next and up to the previous paragraph, respectively. The first line of the paragraph becomes the current line. This facilitates browsing through output, looking for "false hits" or especially useful citations.

Half screen scrolling. A complement to scrolling by paragraph, keystrokes such as <Ctrl><CursorUp> and <Ctrl><CursorDown> or <Alt><PageUp> and <Alt><PageDown> are assigned to shift the file area contents one-half screen up or down. This is useful when the bibliographic part of the citation fills a half-screen and one wants to read the abstract or main text after the bibliographic information.

Paragraph blocks. Since <Alt><G> deletes a block by default, <Ctrl><G> has been defined the related function of marking as a block the entire paragraph containing the cursor. If a block had already been marked, perhaps previously selected paragraphs or citations, the newly selected paragraph and all intervening text are added to the marked block. The whole block can then be acted upon by commands that can target line blocks: delete; copy or move within the file or to other files; shift right or left; change to upper or lower case. This approach has been used with minor modification to mark entire citations that contain internal blank lines but are otherwise clearly delineated. For example, Predicasts PROMT citations, obtained on Dialog Information Services, have many internal blank lines, but each starts with a record number indicator.

Classification. Paragraph blocks as described above may be classified by moving them selectively into twelve files using the <Alt><F1> through <Alt><F12> keys. The target files are generically called CLASS1.TXT through CLASS12.TXT. We have several periodic searches that are routinely classified manually during post-processing for the convenience of the requestor.

Pagination. "Widow" and "orphan" are standard word processing terms for the first line of a paragraph printing as the bottom of a page or the last line of a paragraph printing at the top of a page. We extend these concepts to avoid splitting any citations awkwardly. In fact, many clients would prefer to have only one citation per page if this did not create such thick reports. As a compromise, we strive to have citations never extend to a second page, which is possible except when citations are longer than the page length. The pagination process can be done manually or automatically. Manual placement of the form feed character for most printers ♀ (ASCII 012 or hexadecimal 0C) is done with <Ctrl><T>. Movement throughout the file is facilitated by locating these form feed characters using the <Alt><T> (next) and <Alt><Y> (previous) keys, respectively. Automatic assignment of the next form feed character at 56 or fewer lines from the current line is invoked by <Ctrl><E>. The entire file is paginated automatically by <Ctrl><6>. Pages just slightly longer that one page, i.e., 57-60 lines, can be identified by invoking <Ctrl><2>, which stops at each long page to allow paring it to only 56 lines. Form feed may also be checked by moving down and up 56 lines using <Alt><7> and <Alt><8>, respectively.

COMMAND LINE MACROS

Copyright. Source database file or copyright owner is not always indicated on each citation but may be given in the file header. Several macros have been written that add this information as a new line at the beginning or end of each citation or append it to an existing field. For example, CASCOPY appends the copyright information to the accession number field in Chemical Abstracts Service citations. PTS PROMT (File 16 on DIALOG) always indicates copyright owner for full text or excerpted information but not for abstracts. A macro was written to add abstract copyright statements after the text part of the citation that does not already have a copyright statement.

Removing extraneous characters. Macro HEXCLEAN replaces undesired characters from downloaded files, such as the ◄ (^Q, or ASCII 017) from DIALOG output. It also exchanges graphic highlighting characters ♫ (ASCII 014) and ☼ (ASCII 015) in STN Express transcript files for triple asterisks ("***"), the text highlighting indicator used in STN files downloaded in text mode.

Source database file or copyright owner is not always indicated on each citation but may be given in the file header. Several macros have been written that add this information...

Selective removal of information fields. Most online systems and database files allow custom display formats. Nevertheless, it is often desirable to streamline the search report by removing information fields from search output. Macro **WEED** has been written to weed out information fields from STN, ORBIT or tagged DIALOG output, whether the fields are single or multiple lines. The macro also removes the answer number and copyright lines from STN output and can condense files by weeding out extra blank lines, i.e., reducing successive blank lines to

→

Information On Packages

KEDIT, version 4.00
Mansfield Software Group, Inc.
P.O. Box 532
Storrs, CT 06268
203/429-8402
$150 (DOS only), $175 (DOS and OS/2), free demo version.

Technical support on MSG bulletin board at 203/429-3784; CompuServe (GO PCVEN); BIX (JOIN MANSFIELD).

KEDIT macros, including many referred to in this article, may be found in the PCVEN library and on the MSG bulletin board.

Personal REXX, version 3.0
Quercus Systems
P.O. Box 2157
Saratoga, CA 95070-0157
408/867-REXX
$150

Technical support on CompuServe (GO PCVEN).

single blank lines. Up to fifteen arguments are accepted by the **WEED** macro. A derivative of this macro is **WEEDACB**, which acts on STN output to remove answer number, copyright and extra blank lines and add copyright statements to accession number fields.

Selective field display. The **ALL** command has been described as bringing together all single lines containing the desired target. For tagged search output, the **SHOW** macro was written to display whole information fields, single or multiple lines, for browsing or selective line editing. The **SHOW** macro accepts up to fifteen field codes for output from the three online systems listed above for the **WEED** macro. For example, all title and index term fields can be displayed and **SHOW** separates those fields from different citations by blank lines. Using the paragraph block command <Ctrl> <G>, the entire citation could then be manipulated, though only these selected fields are displayed. (This is accomplished by setting the "scope" to all lines, rather than the default of displayed lines only.) To make the effect of the **SHOW** macro permanent, the **KEEP** macro retains only the displayed fields and erases all hidden lines. The **SHOW** command may be reversed using the **ALL** command without arguments or the **UNSHOW** macro, which retains some of the selective line editing parameter assignments for further use of the **SHOW** command.

Reformatting citations from untagged output and preparation for importing in word processors. Extraneous fields are readily removed from field-delimited (tagged) output but more complicated programming is required for untagged output as is standard from DIALOG. Major reformatting of PTS PROMT output is carried out by us using a series of single-application keystroke macros. Functions included are:

- removal of ISSN, word count, full text available indicators;
- removal of bylines, source addresses not needed by the requestor, and null dates or page numbers (e.g., November 00, 1991 or p. 0);
- addition of copyright of abstract;
- enclosure of the document's language in parentheses and moving it to end of abstract;
- conversion of the title to uppercase characters;
- relocation of source information from end of citation to just below the title;
- conversion to blank characters from indentation of the first line of each paragraph to tab characters;
- elimination of all extra blank spaces resulting from text output in right-justified format;
- insertion of CR/LF (new paragraph) indicators and font control characters to facilitate importing of the modified report into a word processor document for final printing.

The font control characters are based on Rich Text Format used by Word for Windows. These are generally intuitive strings: e.g., \tab, \b{...} or \i{...} where the last two cause the intervening characters to be bold-faced or italicized. Many word processors can import Rich Text Format documents.

Text conversions for importing into personal bibliographic databases. A simple conversion is relocating the document type (**DT**) field to the top of each CA (STN) record, as required by several database programs that accommodate multiple record (or document) types. Additional examples are described in my article in the April *DATABASE* [3].

Conversion of word processor ASCII output to clean flat file. Occasionally word processor documents need to be converted for transfer as flat files via electronic mail. A KEDIT macro is used to translate tab characters into blank spaces, weed out extra blank lines and format text within flat file margin settings.

CONCLUSIONS

KEDIT is our standard editing tool because of its strength in handling large, structured text files and its ease of use, versatility, powerful text commands and macro language. While KEDIT has over 100 commands and many customizable settings, one can make good use of the program by understanding and using only a small fraction of these. KEDIT does lack menus and other user aids, which some users might find indispensable. For them, other DOS editors or word processing programs might be a better choice.

Mansfield Software Group is a small organization that has been responsive to their loyal customers' concerns. Many KEDIT features have come about from listening to user feedback. The Fall 1991 issue of their newsletter promises additional exciting features, possibly including:

- mouse support for many common actions;
- addition of more REXX language functions and facilities for the KEXX language subset;
- the availability of the KEDIT Reference Manual online;
- expanded undo facility; and
- implementation of Windows and Presentation Manager versions of KEDIT.

While KEDIT seems in no sense inadequate as a text editor, these improvements will be as welcome as all previous upgrades.

ACKNOWLEDGEMENT

The author would like to acknowledge the contributions of Mark L. Winzenburg of Amoco Corporation,

Information Research and Analysis Group, who pioneered development of KEXX macros in our group.

REFERENCES

[1] Quint, Barbara. "Inside A Searcher's Mind: The Seven Stages of an Online Search—Part 1." *ONLINE* 15, No. 3 (May 1991): pp. 13-18.

[2] Quint, Barbara. "Inside A Searcher's Mind: The Seven Stages of an Online Search—Part 2." *ONLINE* 15, No. 4 (July 1991): pp. 28-35.

[3] Wolff, Thomas E. "Personal Bibliographic Databases: An Industrial Scientist's Perspective." *DATABASE* 15, No. 2 (April 1992): pp 34-40.

THE AUTHOR

THOMAS E. WOLFF is a Staff Research Scientist in the Amoco Corporation Information Research and Analysis Group. He received his S.B. (Chemistry) from the Massachusetts Institute of Technology in 1974 and Ph.D. (Inorganic Chemistry) from Stanford University in 1980. After graduation, he was a Research Chemist in Amoco Chemical Company for ten years in the areas of polyolefin catalysis and oxidation of alkylaromatics to polycarboxylic acids. Two years ago he made the auspicious move to the Amoco Corporation information area where he provides literature, patent and business information services, principally to former colleagues in Amoco Chemical.

Communications to the author should be addressed to Thomas E. Wolff, Amoco Corporation, Mail Station F-1, P.O. Box 3011, Naperville, IL 60566-7011; 708/420-4662; Internet—mhs!amoco!thomas_e_wolff@attmail.com; STNmail—1749C; DIALMAIL—27246.