Putting the HP3000 to Work For Programmers

Thomas L. Fraser
Forest Computer Incorporated
East Lansing, MI

I. THE OPPORTUNITY

The demand for software is exploding as businesses and other organizations which use computers strive to be more productive, control costs, and improve the quality of management information. The acceleration of this demand is forecasted to continue throughout the early 1980s.

Software is produced for the most part by people, skilled people. These "programmers" are a limited resource. If the increasing demand is to be met, either the size of this resource must be increased or the productivity of the resource must be improved.

Looking at the issue from the viewpoint of an individual DP shop, increasing the size of the resource means hiring people. Skilled people are expensive, and costs are going up. Especially expensive are programmers, due to the already existing shortage. This shortage also makes it difficult to find quality people. So increasing the size of the resource is not always easy and is very costly.

Another trend that is evident is the decreasing cost of computer hardware. This contributes to the increasing demand for software, and thus is part of the problem. However, it can be made part of the solution by putting computers to work for the programmers.

This is the opportunity. Use the computer to increase the productivity of programmers. Provide software tools which allow the people to work efficiently and quickly. The expensive and scarce programmer should not have to wait for or adapt to the increasingly inexpensive computer. In a word, the computer needs to be made more friendly toward the programmer.

II. THE HYPOTHESIS

In the specific environment of the HP3000, programming is usually done online. A majority of the programs are written in COBOL with FORTRAN also popular. There are several types of tools which can be introduced to this environment. Report generators, high level file systems, COBOL generators, forms generators, and very high level languages such as RAPID/3000 can all help. However, in most shops programmers still spend a large amount of time at a terminal working with source code. This therefore is the first place to look when considering how to get the HP3000 working for the programmer.

By far the most prevalent software tool used by programmers is the HP editor. Compared to the primeval batch methods of source input and maintenance, EDIT/3000 is vastly superior. Because the editor is interactive, changes can be viewed as they are made within the context of the rest of the program. Also the editor provides many features such as searches and global changes previously unavailable. And best of all there is no problem keeping card decks in sequence.

However, the new features and capabilities come with a price, that of increased demand on the system resources. The programmers are competing with each other, as well as with production users, for precious disk accesses and CPU time. An obvious result of any delay in system response is lower productivity. This applies to all users of the system, including programmers.

EDIT/3000 is not without weaknesses. It is a line-by-line editor. This is a logical carryover from the days of cards. (Remember, the VDT was originally intended as a keypunch replacement.) All I/O is organized around the line as a standard unit. I/O from the terminal interrupts the hardware once for each character because of lack of block-mode handling. Moreover, the software must get involved each time "RETURN" is hit; this is a minimum of once per line with the exception of the "CHANGE" command, and with many commands can be several times per line. Disk I/O is blocked, but the binary search used to locate the card-image formatted records i very expensive in terms of disk accesses. This line orientation has obvious negative performance implications. Moreover, it means that the programmer must work with a line at a time. Despite the ability to display 20 lines on a single CRT screen, only one line at best can be entered or changed per transmission except for the noted exception.

Believing that the overall demand on resources might be reduced, and system performance improved, there still remain other areas to be investigated when seeking to improve upon the editor. For example, the "TEXT" and "KEEP" commands are very slow due to the fact they are actually file copying commands.

One of the nice features of EDIT/3000, that of being able to see the changes in context, is mitigated against by two major factors. The first is screen clutter. Unless one repeatedly does "LIST" commands, the screen becomes full of old source lines and already executed commands as well as current source lines. The second is the inability to access everything on the screen.

Performing some operations, even on a single source line, require several commands to be transmitted. This makes more effort by the programmer necessary, and slows the coding process. This, together with the other factors, are seriously impairing the speed of software development and system performance.

Thus the hypothesis, that a full screen block-mode editor, written for maximum features with minimum demand on machine resources, would dramatically improve programmer productivity. Improved response time for other users could also be anticipated.

III. THE METHOD

To test the hypothesis a full screen, block-mode editor was designed and written. The result of this effort, called "CHICKEN" by its architects, was a COBOL and SPL program which can be used to edit source code, documentation, stream-files, and other text. No operating system modifications are required, and the program runs in ordinary session-mode.

Block-mode transmissions dramatically reduce the overhead of terminal I/O. This is especially true when the line is driven from the Asynchronous Data Communications Controller (ADCC). The number of transmissions is also reduced making life easier for the programmer. The terminal has a microprocessor; block-mode enables taking advantage of this to reduce load on the HP3000. If one has paid for a "smart terminal," it behooves one to use it. By the way, CHICKEN can automatically switch the terminal between block-mode and character as needed. Implicit is that the VDT being used is an HP compatible terminal with block-mode capability.

Full screen access is another way of putting the terminal to work. With the new editor, all twenty-four lines of the screen are used. One line is for entering commands, one line for error messages, and the other twenty-two are used to display source lines. The programmer can change, delete, or insert lines of code any place on the screen by using just the terminal capabilities. Only after completing an entire screen, is the source transmitted to the HP3000. At that time CHICKEN will determine which lines should be deleted, changed, or added to the file. There is no need to use commands to tell it what is a change, delete, etc.

The disk organization of source files also effects significant advantages. Standard MPE files are used, but CHICKEN has its own access techniques. The old card-image format is replaced by a compressed format which is designed to maximize performance while using less disk space. Because of the file organization and access methods, CHICKEN can retrieve any single line of source code in one disk access, and any twenty-two consecutive lines in an average of 1.4 seeks with a maximum of two required. This single technique has great performance implications.

Ease of use is always an important design consideration and CHICKEN is easy to use. The command set uses language similar to EDIT/3000 to make it easy to quickly get acquainted. Any command can be issued at any time. Moreover, it is seldom necessary to issue multiple commands to accomplish a single task. Recall also, that the software frequently will figure out what you want done without having to be specifically told. Probably the biggest factor in ease of use, though, is the full screen access. A simple list of commands is below.

CHICKEN has other features which contribute to improved productivity:

- Screens are automatically formatted for COBOL, FORTRAN or SPL source if desired.
- The programmer has access to most MPE commands from the editor.
- Compiles can be submitted without leaving the editor.
- Special passwords are put on source files.
- An optional log of changes provides a means of recovery and a means of "backing out" modifications. This also can be used to provide an audit trail.

Several commands are listed below to show general syntax and to compare their operation with the similar commands available in EDIT/3000. In general, the commands follow a standard format as shown here:

CMD <starting-line <ending-line>> required-params <optional-params>

CMD <starting-line <ending-line>> required-params <optional-params>

Most command key-words are the same as found in EDIT/3000, and all can be invoked by entering only the first letter. For instance, "LIST 120.5" can be entered as "L 120.5".

CHICKEN attempts to give the user as much flexibility in entering a command as possible, so as to accommodate differing user styles acquired through exposure to various other editors. Thus the following commands would all have the same effect if entered:

DELETE 20/30
D (20.00:30.00)
DEL 20 30
DELETE 20,30

The goal here is to make the editor easy to learn by not requiring strict adherence to particular syntax rules, and easy to remember by keeping command formats simple and regular.

Following are some representative commands:

TEXT edit-file < NEW < mpe-source-file > >

This command opens and grants access to an edit-file.

If another edit-file is currently open and being worked on, it is automatically closed. If the NEW option is entered, a new edit-file is created. The "mpe-source-file" refers to an EDIT/3000 source file which can be copied to the CHICKEN edit-file. This command executes very quickly because there is no copy operation from a source file to a work file as in EDIT/3000, except when an MPE source file is copied in, which happens only rarely.

KEEP < A < B > > mpe-source-file < PURGE >

This command makes a copy of the currently accessed edit-file to an EDIT/3000 formatted source file. Normally all lines will be copied. If line A is specified, all lines from line A through the end of the edit-file will be copied. If line B is specified, the copy will only include the lines from line A through line B. If the PURGE option is entered, the edit-file is closed and purged from the system after a successful copy operation.

This command is used infrequently, usually for backup purposes. Since compiles can be implemented directly from within the editor on the existing edit-files, there just isn't much need to KEEP files. If one edit-file is TEXTed in and modified, a second TEXT automatically closes the first edit-file with changes intact. The improvement in response time to access edit-files can be dramatic even on only a moderately loaded system.

LIST < { A / LAST } >

A simple LIST command without parameters will display the first 22 lines of text in the edit-file. Subsequent transmission will display the next 22 lines, in effect paging through the text. If line A is specified, then line A and the next 21 lines of text following line A will be displayed. Again, paging applies after entering the command once. If "LAST" is specified, then the last line of text and 21 blank lines are displayed.

This is where some of the power and flexibility of a full screen block-mode editor can be seen. The user can now be free to move the cursor anywhere on the screen, modifying, inserting, and deleting lines. Changes can be reviewed in context of the surrounding text. Even line numbers can be changed simply by typing over the old ones displayed. All of this goes on without bothering the host computer. Of course, this frees up the HP3000 for other tasks at hand.

FIND < A < B > > *textl* < ALL >

This command performs a search for the next occurrence of textl and displays the line containing the textl along with the following 21 lines of text. If line A is specified, the search will begin at line A and continue until a match is found or the end of the file is reached. If line B is specified, the search will only encompass lines A through B. The asterisks surrounding textl represent delimiters, which can be any non-alphanumeric characters including a space.

Examples:

FIND MEN	<spaces as delimiters>
F/ALL MEN/	<slashes as delimiters . . . space is part of the search string>

The ALL option will cause the editor to attempt to find all occurrences of textl and display all corresponding lines. If 22 occurrences are found before the search line limit, the lines containing occurrences of textl are displayed along with a message stating that the search is not finished. The user can modify any of the lines on the screen. To resume the search, the user only needs to enter "F", and the editor picks up the search where it left off. The user can even begin a search and then use other commands such as LIST or CHANGE, add and delete lines, etc., and will still be able to resume a search.

RENUMBER < A < B > > < BY N >

Renumbers the edit-file. If no parameters are entered, all text lines are renumbered. If line A is specified, the numbering will begin at line A and continue through the end of the file. If line B is specified, the renumbering will only be done on lines A through B. The BY N option allows the user to override the default line number increments used by CHICKEN in a renumber operation.

The renumbering is done to the file in place, rather than through a copy procedure. This significantly speeds up the operation in comparison to, say, EDIT/3000's GATHER command.

Other commands found in EDIT/3000 as well as many other line editors, are either unnecessary or have their utility reduced with a full screen editor. The ADD command in EDIT/3000 is a good example. With CHICKEN lines are inserted right on the screen between other lines of text, and transmitted back to the editor. The line number does not even have to be included. The editor identifies the surrounding text and calculates a line number for the new line. To add text at the end of a file, the user enters L LAST. CHICKEN displays the last line of text followed by 21 numbered blank text lines. Along the same vein, line deletes can be handled on the screen simply by placing the letter "D" before a displayed line of text. The DELETE command itself is only needed for global deletes, as in D 100/200.

The above are just a sampling of the full screen editor's command list. As was mentioned previously, command keywords have, for the most part, been kept the same to facilitate learning to use CHICKEN. Thus the user will find such familiar key-words as GATHER, JOIN, HOLD, CHANGE, EXIT, etc., along with a few new commands, such as ZIP which initiates a compile for an edit-file without having to either exit the editor or KEEP the source code.

THE RESULT
(A Personal Digression)

The most notable difference upon the installation of

CHICKEN was not programmer productivity, it was programmer euphoria. After using it for even a short time, one gets hooked. In our shop we have a mixture of block-mode and character-mode terminals used for program development. To say that the character-mode terminals are collecting dust would be an exaggeration, but we have noticed people arriving quite early in the morning to stake claim to a "CHICKEN" terminal.

The productivity, response time, and performance improvements are also accomplished. As of this writing (December 1981), quantitative data are not available. Anyone desiring more information of this nature, or having any further interest in learning more about CHICKEN can write to:

Tom Fraser
Forest Computer
P.O. Box 1010
East Lansing, Michigan 48823

or call (517) 332-7777.