Section Editor:
Russ Abbott
Department of Computer Science
California State University,
Northridge
Northridge, California 91324

# DECset 8000 — A Review

DECset-8000 System User's Guide
Vol #1, #2
Dec #EK-TY800-OP-002, EK-TY808-PR-001

Vincent Manis and Peter van den Bosch

When it comes to designing software for people not used to dealing with computers, manufacturers of hardware are frequently the worst offenders in producing complex, inconsistent and frustrating systems. The problem is that they tend to think of all people as potential programmers. Now, a programmer, though he may grumble, will put up with almost any inconvenience thrown in his way, but a typist is not a programmer, nor is a bank clerk, nor is a journalist. Yet these people, and others in every aspect of business and industry, are expected to deal, in increasing numbers, with software designed for programmers and other members of the computer culture. They too will adjust themselves, since it is their livelihood, but they will not grumble, because they too often do not realize that matters could be better. When "the computer" makes a mistake it will afford them a moment of bitter pleasure during coffee-breaks, but in general they will feel frustrated, inadequate and oppressed, and their working lives will be made just a little more unpleasant because of it.

Why do computer system designers get away with it? If an architect designed unusable buildings he would very quickly run out of clients (unless he were first proclaimed a genius, in which case public inertia would keep him very busy indeed); more than likely he would be drummed out of his professional association. This is also true of surgeons whose patients were put back together in a biologically unusual way, engineers whose bridges only went half-way across the river, and lawyers whose contracts were full of loopholes.

An excellent example of software generally used by non-programmers and almost invariably inadequate in design is that intended for text formatting (and this includes typesetting). Here we have a class of users whose only interest in a computer is that it will speed up their processing of large amounts of text: they are not programmers, nor are they interested in becoming programmers -- any more than I am interested in becoming an electronic engineer when I buy a television set -- and yet this is precisely what is expected of them by computer system designers.

Text formatting is no simple task: typesetting is a skilled discipline which combines a sense of aesthetics with a deep knowledge of the available tools: even the typing of a business letter requires a considerable amount of experience before it is done well. A computer is no substitute for a sense of aesthetics. It is merely a tool to lighten the load of other, less difficult and therefore often duller tasks. However, when the use of a computer merely replaces one set of well-understood but dull tasks with another set, far less well-understood, the human function, the sense of aesthetics, is thwarted and the computer loses its effectiveness as a tool. A case in point is the DECset-8000 typesetting system.

It is worth pointing out, first of all, that a typesetting system like this one is generally a small operating system: there are facilities for managing story files, displaying formatted text, editing, sending material off to the photocomposition machine, etc. This is, in principle, an improvement over the sort of support generally provided by the photocompositor manufacturer, where the user produces a paper tape (usually at a Teletype) and is left only with a crude paper tape editor to make changes. But punching and editing paper tapes are two simple tasks which one can learn without undue effort; learning to use an operating system may well be another matter.

The DECset-8000 system is a turnkey PDP-8 configuration designed for newspaper typesetting. It comprises editing, filing, proofing, justification and hyphenation facilities, as well as the ability to generate tapes for a phototypesetter. It includes a Wire Service Subsystem for dealing with AP/UPI wire service input, a subsystem for handling classified ads, and the ability to generate reports on operator performance.

It appears, from the documentation, that the DECset-8000 performs many newspaper typesetting activities. However, there appears to have been little thought taken regarding human factors. For example, at the top level, there are at most 26 programs, one per letter of the alphabet. A single mistyped letter can put the operator into the disc patch routines, where angels fear to tread. One program is a general monitor, which allows the user to execute utility commands (which have a rather peculiar syntax). Thus, it appears that human engineering considerations, such as the provision of a system which always presents one face to the user, have not been met.

Regarding the programs themselves, there is relatively little to say. There is an editor, a "driver markup" program, a hyphenator, and so on. The fact that each of these programs has a different sort of command syntax from all others is regrettable. The documentation makes it appear that each program does all it is expected to.

Systems such as DECset-8000 are often defended on the grounds that they are turnkey, designed for those with no knowledge about computers. There is no general design principle which states that elegance and ease of use are contradictory -- ask an architect! There is no reason why such turnkey systems can't be implemented within the framework of a simple operating system such as Unix. Such a system provides named files, rather than numbered blocks, and a consistent environment for command execution. An OS approach also gives the sophisticated user or consultant the ability to add to the repertoire of programs, thus extending product life.

It may be argued that the PDP-8 in the DECset-8000 can't support such an ambitious software environment. This claim, even if true, is irrelevant. It was, after all, DEC who chose the processor -- an 11 would have added little to the overall system cost. The DECset-8000 points out a common problem with computer systems: it is not enough to imitate the crude methods in use before computers are introduced -- it is about time suppliers took care to provide something better.

We come now to that aspect of text formatting which all text formatters more or less have in common: the formatting language. This "language" is almost invariably a set of commands ranging in syntactic complexity from special character-codes (i.e., the Teletype School of Programming, strongly influenced by the presence of the CTRL-key), through codes with one or more parameters, to general-purpose languages, often entered by means of escape characters in the input stream. The DECset format language falls into the middle range of such languages: a command is followed by a list of parameters separated by commas; in most cases there are reasonable defaults for absent parameter values.

One cannot fault the format language for lack of command types. What is lacking is a sense of design, of principles underlying the entire language. It is admirable that the designers have tried to come up with a language which will drive a number of different photocomposition machines, but instead of trying to abstract some basic principles of text formatting and building on these, they appear to have simply extended the language whenever there was a feature on a given machine which the language couldn't yet exploit. The result is redundancy.

For example, there are "quadding" commands (a typesetting term for the process of filling a line out with white space) and "output mode" commands (which are concerned with centring lines, setting them with unjustified right or left margins, or justifying them). Now obviously, quadding is what the formatter does when it sets a line according to a current output mode, so quadding is the more primitive concept; but the manual

**Local Groups**

New York
Steve Mandell
245 East 19th Street
New York, N.Y. 10003
212/673-5111

San Francisco Bay Area
Tom Griffin
Hewlett Packard
1501 Page Mill Road
Bldg 6 - Upper
Palo Alto, CA 94304
415/493-1501 x2402

Düsseldorf
Robert Jones
MANDATA
Unternehmensberatung
GmbH
4 Düsseldorf-Heerdt
Werftsrasse 23
West Germany
(0211) 50 30 10

Toronto
Richard Nelson
Dept. of Computer Services
York University
4700 Keele Street
Downsview, Ontario, Canada
M3J 1P3
416/677-6308

Washington, D. C.
Christopher Higgins
Booz Allen & Hamilton
1025 Connecticut Ave., NW
Washington, D. C. 20036

presents the two in parallel (in fact, separated by a number of unrelated command groups) as independent concepts.

In a similar vein, the language gives the user limited ability to define commands in terms of existing commands. "Limited," because these user-defined commands are not permitted any parameters themselves, nor can they stand for any common sequence of two or more commands. In other words, in place of the sequence "X,a,b,c" it is possible to define the shorter string "Y", but

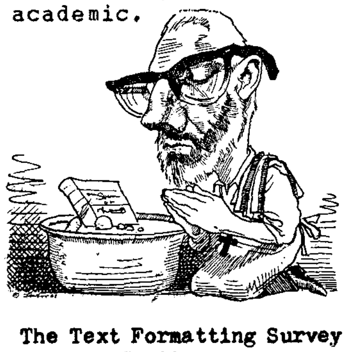
"K,a,b,c;L,d,e;M,f,g"

cannot be replaced by "Z". There is also a facility (called "hold group commands") for defining common text sequences as a shorter string. There are eight "registers" for this purpose. (Why eight? Why not more? Don't ask. It probably looked like a good number to deal with on an octal machine, and there may even be devious machine language tricks which make this an ideal number.) Thus the commonly used sequence "The party of the first part" can be defined as, say, "X,3". Again, there are no mechanisms for parameters, so that I cannot have a "The party of the |n-th| part" register which I could call up as, say, "X,3,fourth". These two facilities are enough alike to merit combining into a single concept -- one that has been around for a very long time -- the concept of the macro. But not eight macros, and certainly not numbered macros. Macros are at their best when unlimited, named, and given a parameter mechanism (it should, it almost goes without saying, be possible to nest macro-calls inside macros). Add to this the natural concept of a number as a string of digits, and macros can be made to do duty as counters. (There is only one counter in the DECset formatter: I don't know what one would do if one wanted a page-counter _and_ a footnote-counter.) It is simple considerations like these, applied at all levels, which make for a cleaner design, and that, in turn, makes the system easier to learn and use.

There are, here and there, some good ideas. For example, the facility for tabular material makes it possible not only to split the page into columns, but to split it into proportionately sized columns. The user also gets considerable control over the parameters which govern the formatting process, and there are means, though not always the best means, for dealing with a wide range of typesetting situations. In many ways the DECset formatting language compares favorably with such languages as CypherText and Harris Composition Language. However, there is the taste of assembly-level language about it: the DECset formatting language lacks even such high-level concepts as the "blocks" provided by HCL (not that HCL goes far enough in this respect).

One further topic of interest to readers of Sigdoc * is the documentation itself. This is in many ways inadequate, especially for users not used to computer manuals. Concepts are not clearly explained, nor is there a clear overview of the system: the many (too many) special purpose editors, the formatting language, the disk system etc., all flow together into an unmanageable plate of spaghetti. The manuals are confusing, and it is hard to figure out where you are, nor do important concepts always stand out. There is not even a glossary, although typesetting and computing terminology freely intermingle in the documentation, to the inevitable confusion of both kinds of readers likely to take up these volumes. No doubt this confusion reflects the overall confusion of the system's design.

However, we are skirting the real issue. As stated earlier in this review, text formatting is an activity requiring the strong intervention of human aesthetics. Since the results of this activity are judged on their merits largely visually, it is reasonable to conclude that the production of formatted text is best accomplished by visual means. Formatting tools, like this system, and all other systems of its kind, will fail to be adequate tools, no matter how well we refine them, until visual, interactive methods using high quality display and feedback equipment (already technologically feasible) are developed. This must and eventually will be the direction of such systems, and when that step is taken all such quibblings (over the design of better dinosaurs) as this review will be merely academic.



**The Text Formatting Survey Continues**

**The results of the Text Formatting Survey are being tabulated but more survey forms are still being received. If you wish to submit a form you are still encouraged to do so.**

**This review is an informal part of the survey: it is a review based solely on the documentation, and done by someone who has never used the formatter under scrutiny. The purpose is to get the perspective of the new user. Reviews of other formatters will be published in later issues.**

**Many thanks to the survey respondants. Your comments and manuals are being carefully examined.**