Design Notebook

Appendix H

FROM: J. Saltzer

SUBJ: On the Format of Files Containing Programs

Some conventions on the format of files containing symbolic programs are necessary so that common editing routines can be provided for all symbolic program files, regardless of language. It is desirable that the conventions permit as flexible a format as possible for the individual translator, ~~however.~~ It should be possible for the same general editor programs to be used on memorandum files of English text.

In all these applications, a fundamental subdivision of a file is a record corresponding usually to a line of ~~typical~~ typed input and to one program statement. Three ways have been suggested to separate records:

1. A carriage return character between the last character of one record and the first character of the next record.
2. An otherwise illegal character bit pattern between the last character of one record and the first character of the next record. In this case, a single record may contain several carriage returns.
3. The record-mark. Here, the first word of each record would contain a count of the number of words in the record. The first word may also contain some otherwise illegal bit combination.

Upon comparing these three schemes, no outstanding advantages can be found for methods 2 or 3 to compensate for the simplicity of method 1. The record mark of method 3 has the minor disadvantage that if the word count of a record should be wrong for some reason, a special program procedure must be provided to scan the file for the next record, or else the whole file after the error must be considered lost. On the other hand, a missing record mark in method 1 or 2 will merely cause a long record to be encountered, and no special procedure is needed to salvage the rest of the file. (The string-pointer format of MEMØ - MØDIFY and DITTØ files have this same flaw - a fouled up pointer word means the whole file is lost; such failures have proven to be fairly common and are in fact one of the reasons this command has remained unused.)

Also, methods 2 and 3 both require more storage space than does method 1. For all these reasons it seems quite reasonable to adopt the simple format of a carriage return to separate records.

Within a record, one format convention appears useful. Although the commands ED and TYPSET have established that line numbers are not essential to the editing process, it is equally clear that some users prefer to work with files having their records sequence numbered. The subdivisions of a record we will call fields. Following the logic expressed above with respect to record delimiters, the delimiter which separates fields should be the tabulate character. Thus the internal representation of a record is identical with the representation in the mind of the typist who created the record.

We may now add the sequence number as an extra field, complete with tab character; placing this field either at the beginning or the end of every record. Either convention could be used, but it seems preferable to put the field at the end, so that a program which wishes to ignore the field can do so trivially. The sequence number field would be introduced by the editor program, which would simply tack it on the end of whatever was typed at the console. (Before the carriage return, of course.)

These conventions are sufficient to be able to provide a general editor program. They leave to the designer of a translator considerable freedom in the format of the records his translator will accept.

He may wish to restrict the format further, if he desires, by limiting the character set and record size to that which can be mapped onto an 80 column Hollerith card, for example. (Such a restriction is probably necessary, say, for a FØRTRAN IV translator.) On the other hand, a new, interactive, translator may take full advantage of all possible flexibility.

One other similar problem should be considered; that of obtaining fan-fold program listings in bulk, off-line. Here, the most desirable solution would be, of course, a printer with a full 8-bit character set. The program driving the printer would also have to make some standard interpretations of the meaning of the tab characters. The present convention, that tabs are placed every 15 columns, seems perfectly satisfactory.

Any attempt to regulate tab settings more precisely will require that tab settings be recorded in the file somehow. If such an approach is used, this information would also be of use to editor and console print programs. The conventions used to record tab settings is again one which must be observed by all programs which read BCD files, but unless they will interpret tab setting information as BCD data for the program.

One possible convention which would maintain at least an illusion that the BCD file contains an exact representation of the character typed on the typewriter would be the following. Invent two new 8-bit characters which represent "TAB set" and "TAB clear" respectively. Each of these 8-bit characters is immediately followed by an 8-bit binary number, designating the column in which a tab is to be set or removed.

With this convention, one could change the tab setting by inserting in a file a record containing tab set and tab clear characters. There is, in fact, no reason why a future typewriter could not accept and interpret properly these characters, and also transmit them when the tabs are set by the user.