Exploiting the GCOS File System

Prior to tackling the organization of the graphics package, a version of Thoth's Inclusion Builder was implemented for the GCOS file system. This tool handled Zed and B (another BCPL derivative) source and has been used successfully for the maintenance of moderately large text manipulation software, such as a spelling checker and a family of screen editors. However, to accommodate the different nature of the GCOS file system, the Inclusion Builder required adaptation to GCOS.

The fact that files must be leaf nodes could have been finessed by designating a special name, and interpreting any file with that name to be the information associated with its parent catalog. Taken to an extreme, this leads to a file system in which all *files* have the same name and there is never more than one *file* below any catalog. If a sufficient set of tools were provided for working with such a structure, the fact that the file system was being used in this fashion would be obscured from the user. But if one is prepared to go to such lengths, then a virtual environment, such as that developed at Lawrence Berkeley Labs[7], is called for.

The preferred alternative here is to continue to use the file system in a conventional manner, letting a naming convention indicate the cases where one file is to be considered the child of another, even though they are actually siblings in the file system. This is accomplished by designating a special character, "-" say, as a delimiter and then treating the file F-V as though it were a son of F named V. Similarly, F-V-W is treated as a son of F-V named W.

This cannot be used as a universal mechanism, because the lengths of names are quite severely bounded. Moreover, there is rarely a justifiable call for more than one level. Once versions of versions become widespread, it is time to rationalize and simplify the software's structure to use more appropriate abstractions. What this convention does mean is that a version of a file can be introduced naturally and rapidly as the software is modified. It avoids the difficulty of creating catalogs and moving files every time a version is introduced.

Using this convention, the majority of the standard tools which are available for operations on the file system can be used meaningfully on the structured source text without modification. Listing, copying, editing, looking at the structure of the source files and so forth can all be be performed immediately with the software tools at hand.

Structuring the Graphics Package

The structure of the package is presented top-down, showing more and more detail about less and less of the package. The root of the structure is the file system node GRAPHICS/SOURCE, that is the catalog SOURCE at the first level below the root of the structure owned by the timesharing account GRAPHICS. Immediately below this are the root catalogs for each of the five levels as shown in Fig. 2.

Below each of these catalogs is a number of files and catalogs. A fixed organization and naming convention is used to structure these, as shown in Fig. 3.

The files named by the reserved names CONST, TYPE and VAR, are files which contain the constant, type and variable declarations which are global to the procedures of level 1. The catalogs named by the reserved words EXPORT and PRIVATE are the roots of the subtrees containing the source for the procedures accessible by the next higher level in the package and those used exclusively by this level, respectively. (The distinction between exported and private procedures is informal; neither Pascal nor any compilers used on the package support this directly). Alternate versions of the CONST, TYPE and VAR files can be introduced, either temporarily or permanently, by files named F-V where F is a reserved file name and V is the version name.

The organizations of the structures below the EXPORT and PRIVATE catalogs are similar. The sons of each of these nodes are named by the Pascal procedures stored there. Where a

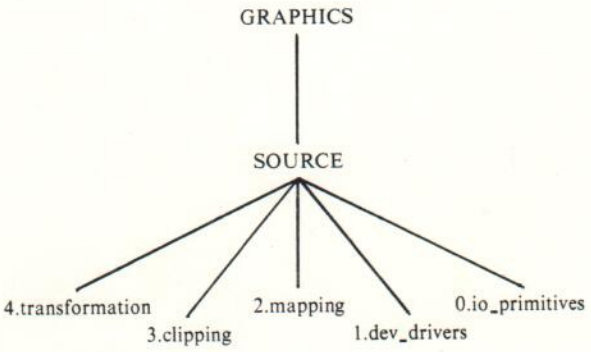

Fig. 2 The "levels" of the package

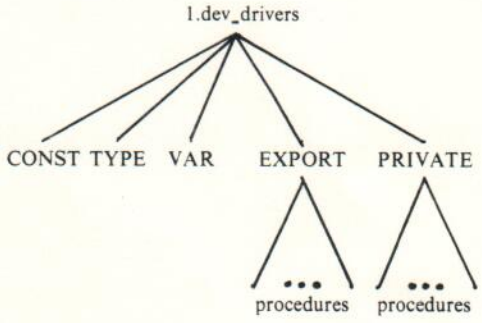

Fig. 3 Below root of level 1