Discussion and Correspondence

An established U.K. software development

By E. L. Willey*

The PL/1 symposium at NPL

The excellent and stimulating symposium on PL/1 (*this Journal Vol* 10, *p.* 211) served to underline that most of the old problems and pitfalls remain with us, in the provision and use of any comprehensive high-level language.

It was, however, disappointing that the symposium was not longer, so that more time could be devoted to a discussion of the language as distinct from its implementation. Not unnaturally, in the time available users dwelt on the latter, since any realistic user with target dates to meet must be primarily concerned with the current state of the language and its compiler. As PL/1 will obviously aspire to be a candidate for international standardization one would have thought that the language itself should have been discussed in more depth.

History repeats itself

Delays and teething troubles have been the predominant feature of language implementation to date. It was therefore surprising to hear hardened users express astonishment that the PL/1 implementors should fail to meet targets, should find it expedient to remove facilities from the first implementation, and that object programs should require storage in excess of that anticipated. It is obvious that history is repeating itself, notwithstanding the vast resources behind PL/1.

What is more disappointing is that those in this country who are responsible for the evaluation, provision and selection of software should overlook a number of U.K. achievements which have either passed or avoided the difficulties apparently now being encountered by PL/1 and its users, and which now serve to provide very adequate working tools.

Seven-years development

I have particularly in mind two products of which I have first-hand experience as a user: Orion and Nebula. It is almost seven years since Messrs. Ferranti Ltd. detailed plans for a time-sharing computer, the Orion, and a "programming language for data processing"—Nebula. Indeed a description of the language together with a small program were published in this journal as long ago as 1961 (Braunholtz *et al.*, 1961). Whilst no one would claim that the ideas incorporated in the language remain unique, they combine to provide a software system which after the passage of seven years is still in the forefront of the field. Meanwhile the Ferranti Computer Department was taken over by I.C.T.

At the moment, the language has, unfortunately, only been implemented on one type of machine—Orion. Three factors, however, make it worthwhile for those concerned with languages to investigate it:

First, the language continues to be extended and improved, and close co-operation between the compiler team and users is maintained to this end.

Second, while Nebula is at present designed to meet commercial requirements, it can, I understand, be extended to meet the needs of non-commercial users.

Third, Nebula was written in an intermediate language (Fraser *et al.*, 1966) specifically with a view to facilitating implementation on other machines.

In view of the last point, and the fact that this approach was decided on some seven years ago, also that other software groups have been using similar techniques for some time (e.g. Computer Analysts and Programmers intermediate language PORDS), it seems strange that the Ministry of Technology should recently have given a grant in another quarter for the development of yet another intermediate language.

Indeed, one is most concerned that those responsible for the co-ordination and encouragement of the national software effort, such as the Ministry of Technology and the National Computing Centre, should not overlook the very substantial achievements that have been made in this country, albeit without much blowing of trumpets, as exampled by the Nebula language and time-sharing facilities on Orion.

It is to be hoped that these bodies are finding out what has been done to date, what potentialities exist for the further development of present achievements, and what is currently being developed. This will then ensure that valuable developments are not lost to the advance of the art. Who knows, we might ourselves have already produced a language that is as good as any other contender as the "standard language".

Facilities already available

The remainder of this note is directed to those who are interested in knowing something of the facilities provided by Orion and Nebula.

Time sharing has been working on Orion since late 1962. This is not just time sharing of peripheral activities but of main programs. The approach adopted and the facilities provided have not been outmoded by the passage of time. Thus, for example, the executive (Orion monitor) program does not crowd other programs off the machine, it does not appreciably delay the operation of the programs it is controlling, and it provides a sophisticated running record of failure of peripheral devices, program errors and the progress of jobs being run. The punched-tape record of the log can be used to produce costings of every job run, a valuable feature for a computer department which has to charge out work to other departments.

Nebula was and is most unusual in that it was designed in close co-operation with other experts in the field of languages and compilers (through the medium of B.C.S. study group No. 5) and with users. This is probably the main reason for the number of facilities it contains.

These features lie principally in the treatment of data structures and their handling, though there are some aspects of the procedure division which warrant attention, e.g. the

* *The Prudential Assurance Co. Ltd., Holborn Bars, London, E.C.*1.

verbs UPDATE and LOCATE, the latter being a scanning and search facility. Incidentally, the procedure division is truly "free form".

As regards data handling, perhaps the most interesting feature is the distinction between the internal logical structure of a record and its physical arrangement for input or output, i.e. the data structure used for internal storage and processing is independent of the representation of the data on the input and output media. This divorce between the internal logical structure and physical arrangement of data contributes greatly to the power of the language.

All input and output is described on a tabular form in terms of the physical layout. Thus cards are described in terms of columns and rows, and printing is described in terms of line and character position. The facilities associated with printing, for example, are very powerful: a variable-length item may be positioned on another variable-length item, and printing can be made conditional. Furthermore codes can be automatically converted to plain-language words and phrases.

Nebula has very good facilities for handling *variable-length records*, and data may be described as:

(*a*) Variable-length alphanumeric data items.
(*b*) Repeated group of items (a fixed or variable number of occurrences).
(*c*) Optional item (requiring only one bit of storage when absent).

These data types can be used in any combinations and may be nested, e.g. one may have a repeated variable-length item within an optional repeated group within an optional group.

Everything, of course, has a cost and these facilities have to be used with intelligence, e.g. a repeated variable-length group is particularly expensive in terms of addressing routines.

Where a file contains variable-length data it expands or contracts with the presence or absence of that data. This can be a major factor affecting tape usage. For example, in an insurance application one may have to allow for the possibility of as many as 500 items per policy record as against an average of only 100 actually present.

There are two facilities of particular interest to those handling suites of programs using complex files. First, while the programmer may leave the detailed file organization entirely to the compiler, he can if he so desires take this function over and "hand pack" his files. thus ensuring the most economic use of every bit. Second, he is able to describe part or all of one file structure as being identical to another. This is a most valuable means of achieving compatible interfaces between files.

Two aspects of vital importance to the user, diagnostic facilities and compiler listings, have been shown in actual practice to be good.

The extent of our own use of Nebula may be judged from the fact that we have already implemented in it a suite of 39 programs, producing a total of over a quarter of a million three-address instructions. Our use of Nebula is continuing and we now have a total of some fifty programmers engaged on two even larger suites of programs.

We have, of course, suffered from our share of the usual teething troubles of a new compiler. Furthermore the speed of compilation is still not as fast as we would like, although this is largely offset by the excellent checking and diagnostic facilities already referred to, and by the power of the language.

It is therefore felt that in Nebula we have something which merits the closest attention, before we start going through the wilderness again with a new language such as PL/1.

References

BRAUNHOLTZ, T. G. H., FRASER, A. G., and HUNT, P. M. (1961). NEBULA: A Programming Language for Data Processing, *The Computer Journal*, Vol. 4, p. 197.

I.C.T. *NEBULA Reference Manual.*

FRASER, A. G., and SMART, J. D. (1966). The COMPL language and operating system, *The Computer Journal*, Vol. 9, p. 144.

To the Editor
The Computer Journal

Sir,

At the PL/1 Symposium on 18th May Mr. D. F. Hendry of the Institute of Computer Science emphasized that the facilities provided by a high level language for the description of data were as important as the procedural facilities of the language. Suitable data description facilities can result in a drastic reduction in the programming effort required because the majority of red-tape operations do not need to be explicitly stated. This also clarifies the logic of the program and reduces its size. PL/1 does not provide facilities of this nature and it was noticeable that many of the speakers for the PL/1 users criticized the language in this respect.

A language which does satisfy this criterion is the Nebula language which not only provides powerful file structure facilities but also separates the description of the physical appearance of the data from the description of the logical structure of the file. Thus a programmer is not constrained in the logical design of the file by the physical lay-out of the data. Moreover, this separation allows the programmer to describe the physical appearance of the data in a simple and direct manner, and the compiler can relieve the programmer of all the work of organizing the input and output of the data.

The Nebula language was originally described by Braunholtz *et al.* (1961) but has been considerably developed since in close collaboration with the users. It is now a most advanced and flexible high level programming language for commercial data processing.

Yours faithfully,
K. H. M. NOBLE

c/o I.C.T., 61 Broadway,
Bracknell, Berks.
10 July 1967

Corrigendum

When reporting the PL/1 Symposium, this journal Vol. 10, p. 211, it is regretted that a misleading impression was given of experience at B.O.A.C. The third sentence of the first full paragraph in col. 2 of p. 211 should be replaced by:

"It was easy to write and the reluctance of experienced programmers to use the new language gradually wore off; COBOL and FORTRAN programmers easily made the change. The new language proved unsuitable for some Operational Research projects and the OR programmers had to revert to the use of FORTRAN."