Correspondence

To the Editor
The Computer Journal

Sir

SIMPL/I—a new simulation language from IBM

SIMPL/I is a new simulation language which is sold as a program product by IBM. I have previously, for my employer, made a thorough investigation of existing simulation languages. Based on that investigation, I recommended SIMULA 67 as the language best suited to our purposes. That investigation was made before the presentation of SIMPL/I. This note is an assessment of SIMPL/I using the same criteria which I used in my previous assessment.

SIMPL/I is based on PL/I. A pre-processor translates SIMPL/I programs to PL/I programs. These are then compiled using the ordinary PL/I compilers. The IBM manuals say that the pre-processor usually expands each SIMPL/I statement to 15-20 PL/I statements. If this is true, then SIMPL/I will be inefficient with computer time and memory space compared to other simulation languages which compile directly to object code.

SIMPL/I is similar in many ways to SIMULA, which makes a direct comparison with SIMULA of interest. The SIMPL/I programmer can include ordinary PL/I statements in his program, just as SIMULA programs can contain statements in ALGOL 60 extended with list and text processing.

SIMPL/I is based on the process concept of SIMULA. The time sequence of events is in both languages described as a number of coordinated parallel processes. Such a process is in both languages described by ordinary programming language statements, extended with special statements to suspend execution of one process for some simulated time so that other processes can act in the meantime. This process concept gives clear, natural models and does not restrict the freedom of the programmer as much as for example the network flow of GPSS.

In both SIMULA and SIMPL/I, an inactive process can be activated again either at a certain time, or when some other process activates it. SIMPL/I has a third way of activating a process. Certain program variables can be defined as notifiers. Whenever a notifier changes its value, a Boolean condition on the notifier is checked to decide if and how some processes are to be activated. In SIMULA, the programmer can get the same effect by (*a*) inserting Boolean tests into the notifier dependent process and (*b*) replacing assignments to the notifier by calls to some simple procedure. When notifiers are useful, this will make the SIMULA program somewhat longer, but on the other hand more explicit. In SIMPL/I a notifier can change the behaviour of a process from the outside with no indication in the process program. This may be confusing.

Both SIMPL/I and SIMULA use list structures for the description of the structure of the system which is simulated (that a certain crane serves a certain ship, a certain truck is on the way to a certain warehouse, etc.), Here, the SIMULA system is vastly superior. SIMPL/I uses the ordinary list structure mechanisms of PL/I, which are not much better than machine code. Explicit FREES and DESTROYS are used instead of garbage collections; this technique is at least ten years outdated. The technique gives longer, less readable and less secure programs. Checking of the correctness of pointer references must be done at execution time in SIMPL/I, which will either be very inefficient or else give a large risk of difficult-to-find programming errors.

SIMPL/I is only available on IBM 360/370 computers. SIMULA is available on a broad range of computers from different manufacturers (CDC, Univac, IBM, soon on DEC). SIMULA programs can easily be transferred from one computer type to another.

My conclusion is that SIMPL/I has a process concept which is as good as that in SIMULA, but that the list processing methods are not as secure and easy to use, and that efficiency probably will be much less good with SIMPL/I.

Yours faithfully,
J. Palme

Datalogy Section
Research Institute of Swedish National Defense
S-10450 Stockholm 80
Sweden
27 November 1972

References

SIMPL/I (Simulation Language Based on PL/I) General Information Manual. IBM GH19-5035-0, 1972.

Kornhauser, Fred M. G. (1972). *Simulation Using SIMPL/I.*

Palme, J. (1970). SIMULA 67—an advanced programming and simulation language, Norwegian Computing Center, Forskningsveien 1b, Oslo, Norway.

Palme, J. A SIMULA System for the DEC System 10 Computer, Research Institute of Swedish National Defense, S-10450 Stockholm 80, Sweden.

To the Editor
The Computer Journal

Sir

Letters to The Computer Journal

It is valuable to learn from the comment to the letter from G. Moon (this *Journal*, Vol. 15, p. 124) that letters to *The Computer Journal* are refereed. One infers from this that the British Computer Society wishes to publish in its Journal letters of a standard comparable with the quality of the papers published therein.

I wish to suggest that letters published in *The Computer Journal* are therefore accorded the same respectful presentation as papers. In particular the splitting up of a letter by several pages of unconnected material, whilst using up excess space on pages containing the end of papers, is most annoying and gives the impression that the importance of letters to *The Computer Journal* can be equated with that of articles in popular digests which suffer the same fate.

Furthermore, it would direct attention to letters of interest to readers if a title was included in the Correspondence section of the contents in addition to the current practice of providing a surname only. Many writers of letters do indeed already provide titles.

A short item in the Notes for intending authors giving the policy concerning publication of letters, requesting the provision of a title and recommending a typical length, together with an improvement in the presentation of letters as indicated, would contribute to the standing of *The Computer Journal* as a means of communication between computer professionals and between the layman and the computer professional.

Yours faithfully,
G. H. Kirby

Centre for Computer Studies
University of Hull
Hull HU6 7RX
24 October 1972

Editor's comment:
The reason why letters are split, as Mr. Kirby acknowledges, is to make the fullest use of space. The points which he makes are noted, however, and will be taken into account in any future planning of the Journal's layout.