Correspondence

Dear Sir,

Implementation of BEDSOCS: An interactive simulation language

I read John Stephenson's letter[1] with interest, as I did the earlier article by Eidelson and Robinson.[2] Grant Weddel, a senior student at University of British Columbia, did a very nice job of implementing BEDSOCS in BCPL in the mid-seventies. (This work was begun under a grant from the British Columbia Y.E.P. Project.) As a result, I was able to do some benchmark testing on the Amdahl at the University of Alberta, using IBM's then current version of CSMP for comparison. Had I suspected that benchmark testing of BEDSOCS would have been of interest in 1980, I would have submitted a paper for publication at the time. However, the general conclusions were very much in line with John Stephenson's comments. The significant advantage that I had was that it was possible (using CPU time) to compare BEDSOCS and a compiled language on the same machine.

The results were surprising. In general, the execution speed of BEDSOCS (which as John Stephenson mentions, includes 'compilation time') was faster than executing the *object* code of CSMP. CSMP is, of course, a much older design, but the process of translating the source code into FORTRAN, and compiling and linking the resultant code soon seemed very tedious. For simple problems (ten linear and non-linear differential equations) the problem could be fully solved in BEDSOCS before the errors in syntax had been eliminated from CSMP. In CPU terms, the total cost of the BEDSOCS solution was generally less than the first successful CSMP compilation.

This led me to question the continued validity of benchmark testing as it has been traditionally applied. Where the modelling is incorporated in a feedback or feed forward loop, the time of execution is obviously critical. Where the problem is under investigation by an engineer, physical, life or social scientist the ability to set up the problem *quickly* and *easily* and to *manipulate* it is of overriding importance. The execution must be reasonably fast for the user to *see* what is happening, but the general conclusion that John Stephenson seems to imply is that even for very large engineering problems, BEDSOCS is fast enough. It would not be difficult to implement BEDSOCS in compiled mode on a mainframe computer, yet he implies there has been no real motivation for doing this. One of the reasons, I suspect, is that not only can' the problem be set up and manipulated with remarkable ease using an interactive system, but the user can make an intelligent analysis of what is happening, can quickly reach an intuitive understanding of the nature of the problem and if need be can stop the run as soon as something unexpected occurs and analyse it in more detail.

My only surprise is that BEDSOCS—or systems derived from it—are not in more widespread use. I know that the University of Calgary has implemented it and I understand it is well liked by the users there. I would have hoped that by now I could buy a BEDSOCS card for my APPLE and would have thought that a programmable calculator with differential equation solving ability (based on a BEDSOCS PROM) would have found a ready market. Perhaps this new found interest will generate some activity of this sort.

Yours faithfully,

ROGER HART
Knowledge Network,
Box 3200,
Victoria,
British Columbia V8W 3H4,
Canada

August 1981

References

1. J. Stephenson, Implementation of BEDSOCS: an interactive simulation language (letter), *The Computer Journal* **24** (No. 2), 191 (1981).
2. A. F. Eidelson and I. J. Robinson, Implementation of BEDSOCS: an interactive simulation language, *The Computer Journal* **23** (No. 3), 233–236 (1980).

Dear Sir,

MU5 Descriptors

We are grateful to R. N. Ibbett for putting a date on the change undergone by the MU5 descriptor format.[1] That this change occurred prior to 1972 was certainly not evident from the otherwise clear discussion in the text by Ibbett and his colleague.[2] As we point out in our paper,[3] the original design had definite merits and proved very useful in highlighting the drawbacks of the ICL2900 format. That these drawbacks are not mere 'theories' is borne out by Morris and Ibbett themselves and by another recent study.[4]

Far from omitting to 'read the literature for the past decade' we have been diligently seeking out information on the MU5 and other structured architectures as the references in our paper show. It is to be regretted that the point that Ibbett regards as so important was covered in a conference, the proceedings of which were not widely circulated.

Yours faithfully,

J. M. BISHOP
Computer Science Division,
University of the Witwatersrand,
1, Jan Smuts Avenue,
Johannesburg 2001,
South Africa

D. W. BARRON
Mathematics Department,
The University,
Southampton SO9 5NH,
UK

February 1982

References

1. R. N. Ibbett, MU5 descriptors (letter), *The Computer Journal* **24** (No. 4), 377 (1981).
2. D. Morris and R. N. Ibbett, *The MU5 Computer System*, pp. 20–21. MacMillan, London (1979).
3. J. M. Bishop and D. W. Barron, Principles of descriptors, *The Computer Journal* **24** (No. 3), 210–221 (1981).
4. W. T. Izatt and E. A. Schmitz, Data structures and descriptors in the ICL 2900 series and beyond, *The Computer Journal* **24** (No. 4), 301–308 (1981).