Technical Report CS 79006-R

TO BE, OR NOT TO BE -- IS THAT THE QUESTION?

Richard E. Nance *
Department of Computer Science
Virginia Tech
Blacksburg, Virginia 24061
U.S.A.

August 1979

* Currently on sabbatical leave in the Combat Systems Department, at the Naval Surface Weapons Center, Dahlgren, Virginia 22448, U.S.A.

Presented at the Sorrento Workshop for International Standardization of Simulation Languages, Sorrento, Italy, September 20 - 21, 1979.

INTRODUCTION

To elaborate on my paraphrase of the famous line of the Danish prince, I suggest that we pause to consider (perhaps reconsider) the proposed exercise in simulation language standardization. My intent is not to be obstructive, and I beg the indulgence of those of you who are already muttering about "the same old tired arguments against standardization." My purpose is to identify three fundamental questions regarding language standardization, to share with you my answers to these questions, and to explore the rationale emanating from these answers. If the questions are irrelevant, inappropriate, or inconsequential, please accept my apologies. If the questions are relevant, appropriate, and of consequence and, more important, the answers seem meaningful, then perhaps we can perceive clearer directions toward the improvement of simulation methodology.

WHAT IS TO BE GAINED BY STANDARDIZATION?

The usual arguments favoring standardization are economic. Standard parts and standard sizes save money and time, both for the producer and the consumer. In some instances the savings are so small as to be classified as "conveniences," but often the hidden savings far outweigh the more obvious ones. Another economic benefit is the promotion of "fair trade" and the consequent protection of the small company's entry into and existence in the marketplace.

Simulation language standardization might have an impact on the economics of simulation model development and experimentation, but the influence is likely to be indirect and diffuse. That conclusion is based on the historical lessons taught by programming language standardization efforts and results in the United States. The claim that standards have led to better or less expensive program production, if advanced by anyone, finds few supporters. Programs, modules, procedures, and routines are not commonly exchanged or easily transported. Model portability in the sense of exchanging submodels to be run on different computers and operating systems is simply not going to achieve huge savings.

The real savings are to be realized in the communication of models, submodels, techniques, and concepts. These savings are more difficult to identify, but they are to be achieved in the evolution of concepts that can provide the sadly lacking foundation for discrete event simulation. Thus, model portability in the sense of an accepted representation form that assists in conveying model characteristics and describing experimental results is the motivating economic gain.

WHAT IS TO BE STANDARDIZED?

I standardization is to occur and if communication is the principal economic motivation, then is standardization at the programming language level the answer? Frankly, I do not think so. Risking the accusation of "speaking of rope in the house of one who has been hanged," I believe that "publication Algol" would have proved more successful as an algorithm representation form if an "execution" version had not been around. The distractions of implementation requirements ("compilability") are too pressing; the objectives of algorithm representation or model specification must be divorced from program execution.

To communicate model characteristics and experimental results, we need a representation form that

(1) facilitates model specification and model documentation through its semantic structure, independent of any syntactic structure,

(2) permits the model description to range from a very high to a very low (detailed) level,

(3) controls the level of detail (amount of description) by successive levels of refinement,

(4) exhibits integrity in its treatment of concepts,

(5) assists in the verification of any instance of use of the representation form, and

(6) enables convenient extension of the representation form.

In previous documents [NANCR77, NANCR79] I have called this form a Simulation Model Specification and Documentation Language (SMSDL). The SMSDL objectives clearly emphasize the model specification and documentation tasks in contrast to program execution.

My conclusion that we should not attempt to standardize at the programming language level is motivated also by pragmatic reasons. We seem to be at a threshold of major revisions in the way that programming languages provide the "conceptual crutch" between humans and computers on one hand and the "problem environments" on the other. The need for alternatives to the von Neumann languages is recognized, and the movements toward defining alternatives are gaining momentum. The nature of the alternatives, whether they are to resemble the functional programming of Backus [BACKJ78] or the higher level programming system described by Winograd [WINOT79], is yet to be perceived.

Not only do we find evidence of "revolutionary thinking" in the programming language community but also in the simulation community as well. The DRAFT projects of Mathewson [MATHS76, MATHS77, MATHS78] and the DELTA and BETA projects, the former at the Norwegian Computing Center [HOLBE77] and the latter jointly between NCC and the University of Aarhus [KRISB79], are two prime examples of efforts that stress model specification as opposed to program execution. From my vantage point I perceive a gradual "bootstrap" approach, beginning at the top (model specification) level as producing the most lasting benefit. Standardization at the conventional programming language level is likely to expend enormous effort in creating a dinosaur.

HOW SHOULD STANDARDIZATION BE ACCOMPLISHED?

The first step is to achieve some agreement on a vocabulary. How can we hope to communicate when we have at least three different uses of the term "event," and the use that relates state and time have at least three variations. While the originators of the terms "activity scanning," "process interaction," and "event scheduling" intended to catagorize simulation programming languages. A more recent paper uses the first two as characteriations of the purpose of a model [HIGHH79].

With the existance of a sound, accepted vocabulary, the second step toward standardization should be to define what is meant by "model specification." The succeeding steps must address the issues of model documentation, experimentation, validation/verification, and implementation (including maintenance) among the several requirements cited by Lee [LEEJO77]. We need to examine the ongoing efforts such as DRAFT (see Mathewson [MATHS78]) and the DELTA and BETA projects to assess their capabilities vis a vis a SMSDL.

Previous references, in particular [NANCR79], expand on the description of SMSDL characteristics listed above. I also have some personal requirements. At the least a SMSDL should:

(1) exist in an interactive environment for model development,

(2) provide a structured format for model definition,

(3) permit a domain of discourse tailored to a particular application area,

(4) monitor the model creation so as to assist the modeler in discovering inconsistencies or incomplete specifications, and

(5) offer direction through a more axiomatic approach to model validation and verification.

Note that I have avoided the organizational/jurisdictional questions related to responsibility for standards development, sources of support, procedural matters, etc. At this point I feel compelled to consider only the technical issues; after all, there is sufficient quicksand within them to trap us all.

My perspective on simulation language standardization is quite biased toward discrete event simulation (des). I am well aware that important and unresolved issues in des are neither unresolved nor important in continuous system simulation. Additionally, continuous simulation has benefitted immensely by an accepted descriptive formalism -- the differential and integral calculus -- unlike des. Therefore, my remarks might not be appropriate for continuous simulation, and we might ask yet one more question: Should the standardization question be considered separately for continuous and discrete event simulation?

I believe that those who tend to react negatively to the suggestion implied in the above question do so because of their belief that, by joining the two, simulation modeling approaches the methodology can achieve a more unified and impressive foundation. A joint treatment might promote that laudable goal. However, at this juncture the potential for achieving some benefit from standardization seems considerably greater for continuous simulation than for des.

REFERENCES

BACKJ78 Backus, John. "Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs," Communications of the ACM, 21(8): August 1978, 613 - 641.

HIGHH79 Highland, H.J. "A Taxonomy Approach to Simulation Model Documentation," SIMULETTER, 10(3): Spring 1979, 19-23.

HOLBE77 Holbaek-Hanssen, E., P. Handlykken and K. Nygaard. "System Description and the DELTA Language," DELTA Project Report No. 4, Publication No. 523, Norwegian Computing Center, Forskningsvn 1 B, Blindern, Oslo 3, Norway, February 1977 (second printing).

KRISB79 Kristensen, B. B., O. L. Madsen, B. Moller-Pedersen, and K. Nygaard. "BETA Language Proposal (as of April 1979)," Working Note No. 5, Publication No. 635, Norwegian Computing Center, Forskningsvn 1 B, Blindern, Oslo 3, Norway, June 1979.

LEEJO77 Lee, J. A. N. "Considerations for Future Programming Language Standards Activities," Communications of the ACM, 20(11): November 1977, 788 - 794.

MATHS76 Mathewson, S. C. and J. E. Beasley. "DRAFT/SIMULA," Proceedings of the Fourth SIMULA Users' Conference, Norwegian Computing Center, 1976.

MATHS77 Mathewson, S. C. and J. A. Allen. DRAFT/GASP: A Program Generator for GASP, Proceedings of the Tenth Annual Simulation Symposium, Tampa, 1977, 221 - 227.

MATHS78 Mathewson, S. C. "A DRAFT/SIMON Manual," Department of Management Science, Imperial College, 1978.

MATHS78 Mathewson, S. C. "Computer Aided Simulation Modelling and Experimentation," Proceedings of the Australian SIGSIM Conference, April 1978, 9 - 13.

NANCR77 Nance, R. E. "The Feasibility Of and Methodology For Developing Federal Documentation Standards for Simulation Models, Final Report to the National Bureau of Standards, Department of Computer Science, Virginia Tech, June 26, 1977.

NANCR79 Nance, R. E. "Model Representation in Discrete Event Simulation: Prospects for Developing Documentation Standards," in Current Issues in Computer Simulation, N. Adam and A. Dogramaci (eds.), Academic Press, 1979.

WINOT79 Winograd, T. "Beyond Programming Languages," Communications of the ACM, 22(7): July 1979, 391 - 401.