MACHINE INDEPENDENT SIMULATION MODELING

George E. Juras
Frederick K. Goodman

ABSTRACT

Computer simulation and modeling is a form of scientific experimentation. As such, simulation experiments must satisfy the reproducibility criterion imposed by the scientific method, i.e., they must be machine independent. A model is machine independent if it runs on all machines of comparable scale and produces results that are equivalent. To achieve machine independence, models are implemented in the user language of a host system, which in turn is written in portable FORTRAN. If the host system is made machine independent, then the models hosted by it are necessarily machine independent.

The host system approach is discussed in the context of NUCLEUS, a programming methodology under development at Battelle's Columbus Laboratories for the past ten years. The validation of the approach is discussed in the context of the FORSYS model, a multinational, macroeconomic forecasting model currently under development by the four Battelle research centers -- Battelle-Geneva, Battelle-Frankfurt, Battelle-Northwest, and Battelle-Columbus.

INTRODUCTION

Almost invariably, all simulation models are machine dependent. A model is machine dependent if it runs on (computing) machine A but not on machine B. Or, if it runs, it does not produce on machine B the exact same results as on machine A. In this definition, machines A and B are assumed to be of comparable scale, i.e., manipulate information in words of size greater than some minimum, say, 32 bits per word.

Simulation models may be measured in terms of their machine independence. The machine independence of a simulation model denotes the likelihood that the model will run and produce the same results on any randomly selected machine (of comparable scale).

In this paper, machine independence is discussed in the context of simulation models written in the FORTRAN language. In the FORTRAN context, machine independence of a model is usually measured as that proportion of the model which is written in ANS (American National Standard) FORTRAN. This definition of machine independence is supported by the erroneous assumption that if a model is written in ANS FORTRAN, it will be usable on any machine having a FORTRAN compiler. After all, every major FORTRAN compiler must include ANS FORTRAN as part of its language interface.

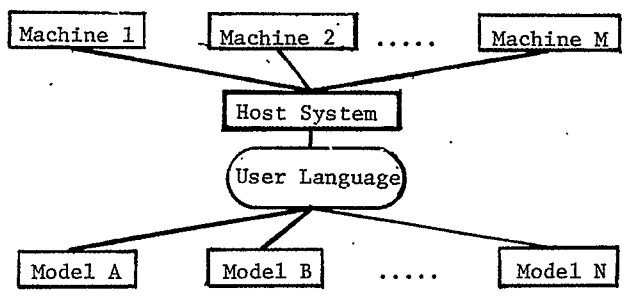
However, machine independence is both a syntactic and a semantic problem. A model written in ANS FORTRAN is supposed to be compiled successfully by any FORTRAN compiler; however, this does not imply that it will also run in the same manner on every machine. ANS FORTRAN is not semantically a machine-independent language.

The results produced by a FORTRAN model are not just dependent on the syntax of that model. They are also dependent upon the internal characteristics of the machine compiling and executing the model. It is the semantics of a FORTRAN model that finally determine its machine independence, given that it has syntactic independence. Being written in ANS FORTRAN is a necessary but not sufficient condition for a model to be machine independent.

It is the purpose of this paper to discuss a methodology for writing simulation models in FORTRAN that have high degrees of machine independence. This methodology is shown schematically in Illustration 1. This configuration consists of a host system and a user language.

ILLUSTRATION 1

A Machine Independent Configuration for Modeling



The host system is an ANS FORTRAN program that interprets and executes instructions written in its own user language. The user language of the host system consists of a finite set of fixed words, an infinite set of user-supplied words, a finite set of special characters, and a finite set of formation rules by which the user may construct the instructions defining a simulation model.

Here the basic principle is this: if the host system is machine independent then any model written in the user language of the host system is necessarily machine independent. Thus, by making a single FORTRAN program machine independent, one guarantees the machine independence of all models written in the user language of this program.

This methodology has been developed and tested over the past ten years and is currently in practice at Battelle's Columbus Laboratories in a software system called NUCLEUS (NUmerical CLassification and EvalUation System). NUCLEUS is a software system featuring: a structured programming user language, an interpreter for continuous simulation modeling (using the same difference equation techniques as those used by DYNAMO), a report generator, data management, conversational modeling, and batch or on-line access. NUCLEUS has provided the base for a series of machine independent host systems supporting a series of simulation models in the areas of regional demographic/economic forecasting, electricity demand forecasting, input-output econometric modeling, aviation activity forecasting, marketing research analysis, coal preparation process simulation, and others.

FORTRAN is an almost universal language in which to write host systems but it is not a good host system itself. For this reason, others also have used FORTRAN to implement various host systems (see, for example, (1), (2), (3), and (4)).

THE HOST SYSTEM APPROACH

The host system approach is defined in the context of the NUCLEUS methodology. In this methodology, a host system consists of a user language, an interpreter which interprets and performs the operations specified in the user language, and an interface to FORTRAN. A software system with these components is not a host system, however, unless it can "host" on all machines all simulation models written in its user language. Hereafter, unless otherwise specified, the word "system" will mean a host system and the word "model" will mean a simulation model "hosted" by the system, i.e., written in the user language of and interpreted by the interpreter of the system. A "hosted" model will run equivalently on all machines that have a FORTRAN compiler.

The user language of the system is a structured programming language according to whose rules and conventions the user writes and organizes his instructions and model algorithms for entry into the system. As a structured programming language it has the following two characteristics: (i) it has no statement labels and thus no GO TO statements, and (ii) major system functions--such as report generation, data base access, model compilation, model execution, storage classification and allocation, etc.--are accessed or specified via either single or well localized groups of statements. The user language is especially useful for writing models which construct, manipulate, and display arrays of numeric data.

The interpreter of the system is a set of ANS FORTRAN programs which interpret the user language statements of a model and generate interconnected tables of internal instructions which later direct the execution of these statements. Only the direct access routines which read and write on disk violate the ANS character of the system. These routines, however, are well localized (less than 20 statements each) and have non-ANS FORTRAN equivalents on all major computers.

Locally, the machine-independence of the host system may be sacrificed for the benefit of improving the run-time efficiency of or adding special functions to the system, by taking advantage of the system's interface to FORTRAN or to assembly language. This interface allows the user to "call" his own additional FORTRAN subroutines or to replace system subroutines with equivalent but more efficient assembly language subroutines.

Model implementation using the host system approach contains the following three steps:

1. Design a computer language, the user language, in which to write the model. As much as possible, this language must be related closely to the notation of the application area being modeled.

2. Using FORTRAN, construct a machine-independent host system which is able to interpret and execute the statements of the user language.

3. Write the desired model in the user language and run the model for testing, validation, and simulation using the host system.

VALIDATION OF THE METHODOLOGY

The host system methodology has been applied in the implementation of numerous machine-independent models. These models and their host systems have been developed on the Battelle-Columbus computer -- a CDC machine -- but have been validated on many other machines. These include: IBM System/360 and System/370, Sperry Univac 1100 Series, DEC-10, Burroughs B6700/B7700, and others.

The most rigorous validation test of the methodology was performed recently in connection with the FORSYS model. FORSYS (FORecasting SYStem) is a multinational, macroeconomic forecasting model for the principal OECD countries (the main Common Market countries in Europe, the United States, Canada and Japan). The core of FORSYS is a set of dynamic input-output models providing forecasts on an annual basis. These models, which use national data, will be fully compatible and provide detailed analyses and forecasts both at the national and international levels.

FORSYS is being jointly developed by the four main Battelle research centers in Europe and the United States: Battelle-Geneva, Battelle-Frankfurt, Battelle-Northwest and Battelle-Columbus Laboratories. The data management and modeling requirements of this application are considerable. The individual country models are being developed by separate teams of economists, econometricians, industry specialists, and software scientists at each of the four Battelle centers. Three different and unlinked hardware configurations are involved in the implementation: a CDC 6500 and a CYBER 73 at Battelle-Columbus (used also by Battelle-Northwest) a CII SIRIS 7/SIRIS 8 computer of French manufacture at Battelle-Geneva, and a Burroughs B6700 at Battelle-Frankfurt. The need for a machine-independent, common representation of the developing model that produces equivalent results on all Battelle computers is obvious.

To guarantee the machine independence of the FORSYS model, a single host system has been implemented and the model for France has been written in the user language of the host system. The remaining country models will be implemented in similar fashion. The user language has been designed specifically for large-scale econometric applications of this type. This language includes specific capabilities for continuous dynamic simulation modeling; time series data storage, retrieval and manipulation; report generation; and allows for both batch or interactive use. Its matrix notation for equations is specifically suited for dealing with multi-sector, input-output economic data. The direct disk access and dynamic storage allocation capabilities of the system are well suited for the management of vast amounts of multi-country, multi-sector time series data.

The commonality of the FORSYS host system on all Battelle computers and the single code representation of the FORSYS model in the user language of the host system provide a sound technical basis upon which to integrate in a coherent fashion the geographically dispersed and multidisciplinary resources contributing to the development of this model. Moreover, the separation of purely software from purely modeling issues, provided by the separation of the host system from the actual model representation, facilitates project management, allows for a better distribution of labor along clearly defined discipline lines, and eliminates duplication of work.

DOCUMENTATION

The syntax of the user language, the rationale for the approach and complete documentation for the methodology are given elsewhere (5). A more complete presentation of the methodology is in preparation for later publication (6).

BIBLIOGRAPHY

1. Lucas, J., and J.V. Wait, "DARE P, A Portable Continuous-System Simulation System," Simulation, Jan. 1975.

2. Korn, G.A., and J.V. Wait, Digital Continuous-System Simulation, Prentice Hall, Inc., Englewood Cliffs, N.J., 1978.

3. Hay, J.L., "Interactive Simulation on Minicomputers, Part 1-ISIS, a CSSL Language", Simulation, Vol. 31, No. 1, July 1978.

4. Pearce, J.G., "Interactive Simulation on Minicomputers, Part 2-Implementation of the ISIS Language", Simulation, Vol. 31, No. 2, August 1978.

5. Goodman, F.K., Juras, G.E., and J.H. McCreery, User Handbook for the NUCLEUS-FHS System (the FORSYS Host System), Battelle-Columbus Laboratories, June 1978.

6. Goodman, F.K., and G.E. Juras, "Machine Independent Simulation Modeling", to be published.

CHARLES M. SHUB

Charles M. Shub has been on the faculty of the University of Wyoming since 1974. His previous experience includes being a graduate teaching and research assistant in electrical engineering. He has also been a part-time instructor in computer science. He has taught at a summer training institute for the use of computers in the humanities. He has also taught courses in computer science to inmates at the U.S. penitentiary in Leavenworth, Kansas. He was awarded an honors dissertation fellowship in 1973. More recently, he has directed a National Science Foundation project for improving high school level teachers of computer science within Wyoming. His current research interests are mostly within the simulation area and include studies, through simulation, on the effect of scheduling overhead in computer systems; computer modeling and analysis of adaptive cognitive processes in neurologically handicapped children; problems in validation of simulation models; and the effect of simulation language methodology on the design and implementation of models.

Dr. Shub has a doctorate in computer science from the University of Kansas. His bachelor's (with high honors) and master's are in electrical engineering from the University of Maryland. He is a member of Tau Beta Pi, Eta Kappa Nu, and the Association for Computing Machinery (SIGOPS, SIGMETRICS, SIGSIM). He is a member of the Albany County Association for Retarded Children, a state advisor for the Developmental Disabilities Protection and Advocacy System in Wyoming, and a member of the National Ski Patrol.

After a brief sojourn at Cornell University, Dr. Shub spent two years, ten months and twenty-nine days in the U.S. Army where he was awarded the Air Force Outstanding Unit Award. He then received a BSEE with high honors from the University of Maryland in 1967 and an MS also from Maryland in 1968. He then served as a three-fourths time instructor in computer science at the University of Kansas where he finally, at long last, earned his Ph.D. in that discipline in 1974. Since then, he has been an Assistant Professor of Computer Science and (in name only) Electrical Engineering at the nations highest institute of higher learning (7,200 feet).

He has published a number of papers on the simulation of computer systems and is an infrequent contributor to other journals. His current research interests include using a XEROX Sigma 7 (which might be considered a virtual mainframe with virtual support) to study, through simulation, the effect of scheduling overhead in computing systems; computer modeling of adaptive cognitive processes in neurologically handicapped children; validation of computer simulation models; and the effect of simulation language methodology on the design and implementation of computer models. He has been a consultant for the Utah State University program for training generic workers to teach handicapped children.