Computer Science Journal of Moldova, vol.**1**, no.2(2), 1993

What does the simulation system SOL/PC do?

G.Magariu V.Madan L.Burtseva

Abstract

The simulation system SOL/PC is presented. It is intended for researching and projecting complex weakly formalyzed systems. SOL/PC provides models discription, checking and editing, executing experiments with models and analyzing the experiments results.

1 Introduction

The application sphere of simulation for investigation and projection of complex, hard-formalyzed system is recently being essentially expanded and now includes technical, economical, social, ecological and other objects.

Simulation presupposes the modelling of behavior and interaction of the investigated system parts with the due account of some influencing factors. In this case there are no restrictions on the level of model detalization, on the type of interdependence of model parameters. Besides, there is the possibility to investigate the dynamics of interaction among components of the model during time running. Thus simulation allows one to replace a natural experiment with a real investigated system by a computer experiment with corresponding model. As a result, the expenditures for the experiment are lower, the risk degree decreases when the experiment with a real system is dangerous, the computer experiment is possible when a real system does not exist yet, or it is not accessible. None the less simulation is a rather labour consuming procedure. The increase of the effectiveness of the application of the simulation method is provided by the combination,

in the frames of one project, of software tools which contain integrated simulation environment for supporting all stages of the problem solution by simulation: model building, simulation execution, analysis and presentation of results.

The tools for automatizing these stages in SOL/PC simulation system are implemented.

2 SOL/PC possibilities

The user is provided with the comfortable dialogue interface. He is offered the menu at each step of modelling.

When entering the system, the menu interface makes it possible:

- to create a file with a new model description;
- to edit the existing file with a model description;
- to check the model accordance to SOL definition;
- to create or edit a file with the values of model parameters for executing a series of experiments with the model;
- to execute a separate experiment or a group of experiments with the model;
- to obtain factors for measuring the model quality;
- to analyse the results of experiments.

In the last item the user is offered a submenu with the list of methods of analysis:

- graphic representation of certain statistics dependence on certain parameters;
- creating output tables with selective results of series of experiments;

- creating the Hunt diagrams for tracing object states during the simulation running;

- creating the Kiviat diagrams for evaluating various exploitation regimes of the modelled system.

3 Short characterization of model description language SOL

The source language for the simulation system is SOL — a general-purpose algorithmic Simulation-Oriented Language, designed in USA by D.E.Knuth and J.L.McNeley [1]. The language has been created for describing and simulating complex discrete systems and the best features of the other simulation languages (such as SIMSCRIPT and GPSS) have been taken into considaration. It has a rather small but powerful enough set of simple but flexible means. The language is easy to learn and the models described by it are easy to read and modify.

SOL includes basic modelling concepts: simulated time, transactions (events) passing in parallel, stores, facilities (devices), random variables, tools for automatic gathering of statistics about the objects involved. The user describes his model in terms of processes: model description consists of global objects declarations and processes descriptions. Processes are connected by their referring to global objects. As we will see below global objects are divided by trasactions of all processes.

Process is a kind of a pattern for transactions. There is only one transaction for every process at the initiation of simulation. It executes the statements given in the process description and is an analogue of a usual sequentially executable program. During the simulaton running, transaction of any process can create other transactions for the same process by executing **newtransactionto** statement. The initiated transactions will execute the statements of that process starting from the label indicated in the above-mentioned statement. Transactions are eliminated by executing the statement **cancel**.

Variables declared within the process are called the local ones. The names of the local variables of other processes are not used in the process description. So a transaction of one process does not refer to the local variables of other processes. Moreover, using the same names of the local process variables, transactions of the same process use different values of these variables. It means that the internal copy of the local variables set corresponds to each transaction of the same process.

Objects defined in the declaration part of the model are referred to as global objects. They are of three major types: global variables, facilities and stores.

Global variables can be referred to or changed by any transaction of any process in the model.

Facility is a global object and can be controlled by only one transaction at a time. A "control strength" is associated with each request for the facility, so that it can be intercepted by another requesting transaction with the strength higher than that of the transaction possessing it; otherwise, the requesting transaction stops and waits until the controlling transaction releases its control. Once interrupted, transaction may continue its execution as soon as it regains control over the facility.

Store is a space-shared global object, to which a specific storage capacity is assigned. The request for space is satisfied if there is sufficient storage to accomodate the requesting transaction; otherwise, the latter waits until the requested space becomes available.

Simulated time passes in discrete units by executing a special statement **wait**.

The simulation process ends with the statement **stop**. A number of standard statistics about the model objects are yielded.

4 Extention of SOL in SOL/PC system

During the language implementation and the simulation system utilization, a number of supplementary tools were designed and introduced into the language [3] [4]. They are very useful for model debugging and

revealing its adequacy to real system and for more detailed investigation of the critical situations in the model behavior.

4.1 Statistics

In the source version of the language at the finishing of simulation experiment the standard statistical characteristics about the model objects are output.
For facility, they are:
KFAC - utilization coefficient,
TFAC - average time of facility occupation,
NFAC - number of addresses to facility.
For store, these characteristics are:
AVSTORE - average number of occupied units,
USTORE - average utilization,
MSTORE - maximal number of occupied units.
For table, the following characteristics are provided:
INTABLE - the number of hits of value x in table interval,
MTABLE - the mean of x,
DTABLE - dispersion,
MAXT - maximal value of x hitted in table,
MINT - minimal value of x hitted in table,
OUTTABLE - the number of nonhits in table interval (a,b).

In SOL/PC system these characteristics are accesible in every simulation moment: they can be used in the model in the capacity of initial arithmetic expressions. It allows one, for example, to control the changing of the most important for user characteristics according to the changing of model time, and to stop the simulation experiment when this changing becomes small enough.

4.2 Information about queues

With the help of a special statement it is possible to provide in the model the output of the current queue to the facility. The name of the corresponding process, the number of the transaction, the force of

seizure, priority, the model time moment when the transaction arrives in the queue are output about each transaction waiting control over the facility. Such possibility allows one to control the interaction of transactions receiving the control over the facility in complex situations when transactions have different priorities and nested interruptions occur.

4.3 Model state

The statement of output of the model current state is involved in the language. Its execution consists in outputing the imformation about all transactions existing in the model at a given moment: the name of the process, the transaction number, the model time value, the names of facilities which are controlled by the transaction, the cause of the transaction interruption and so on. Such statement makes it easy to analyse critical situations in simulation experiment running.

4.4 Random numbers control

The using of random generator allows one repeatedly to execute the same experiment with the same excerption of random numbers. The statement of control of random flow allows one to change this excerption and to reveal whether or not the characteristics he is interested in depend on the excerption of random numbers.

4.5 Experiments making tools

The **experiment** statement allows one to execute the necessary number of experiments with the model [4]. Parameters values for each experiment are input from the file which is pointed in the statement. The results of all experiments are put in one common file which then serves as the source file for subroutines of the analisys of the results of the experiments with the model.

5 Analysis of simulation experiments results

A number of tools for the automatic application of the known methods of the complex systems analysis to the results of the simulation experiments with the models were also designed and implemented [2] [3] :

- graphic representation of certain statistic characteristics dependence on a parameter;
- graphic representation of the facility or store states in the course of the experiment;
- the histogram graphic representation;
- creating the Hunt diagrams [5] for the comparison of various facilities (stores) states during one experiment, or for one facility in the course of executing different experiments;
- creating the Kiviat diagrams [5] for the comparative evaluation of various modes of exploitation of the simulated system by two statistical characteristics (criteria) or multicriteria model evaluation on the basis of one experiment results.

For user's convenience and to prevent him from posible errors and from the necessity of inputting the names of parameters, facilities, stores, tables, in the system, the output of them on the screen and the following user's choice with the help of standard keyboard are provided in the system. When an object is chosen, the system shows the list of possible for this object statistical characteristics for the user to choose the one he needs. Let us examine briefly the most interesting means of the analysis of the experiment results.

5.1 Graphical representation of statistical characteristics changing according to the changing of one model parameter

For each experiment the values of a parameter and chosen characteristics are taken out from experiments series. According to these values

graphics, are built (from 1 to 4 in one picture).

5.2 Graphical representation of change of the facility or store state according to the change of the model time

Interesting for this objects must be declared in the model by "monitor". For facilities such diagram marks the model time intervals when the facility is busy, for stores it marks also the number of occupied units. Figure 1 showns the changing of store Q0 state in the model of service station with two request flows: in each time moment the quantity of nonpriority requests in queue on service is marked. This graphics corresponds to the time interval when new requests do not come in and the service of already existing ones is being completed.

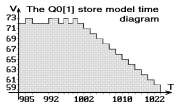

Figure 1

5.3 Building of the Hunt diagram

It represents the state of one facility in different experiments (or states of different facilities in one experiment). On the absciss axis the values of the model time are marked. The interrupted line in the picture corresponds to one facility in one experiment. The segments of this line represent the model time intervals when the facility is busy. The number of such lines corresponds to the number of experiments (or facilities in the second case). Figure 2 showns the changing of the state of TV-sets retuning point in the model of TV-sets tune control.

The diagram is built according to the results of three experiments with different probability of returning of a TV-set on recontrol.

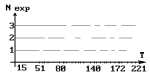

Figure 2

5.4 Creating of the Kiviaty diagram

There are two possibilities. The first one is the comparative evaluation of a series of experiments (which can be interpreted as various regimes of exploitation of simulated system) by two statistical characteristics. The second one is the evaluation of the only experiment results by a set of statistical characteristics. Figure 3 illustrates the changing of the total number of amoebae in the population and the percentage of those died in a natural way according to changing of amoeba division time. Thus Figure 3 allows one to evaluate the experiment results by these two criteria: each beam of the star represents one experiment results.

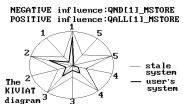

Figure 3

6 Measuring of model quality

The user can obtain a set of factors which characterize the model quality: the length of the model, the volume of occupied memory, the dimention of data bases, the quantity of work etc.[6] [7]. All necessary information for the reception of characteristics is accessible for the SOL compiler. It can fulfil the function of calculation of the model complexity vector. Since debugging implies multiple utilization of the compiler, we can obtain a set of measurement vectors. This information is saved in the system informatical base and is necessary for making the analysis of the model comlexity. Further, we propose to include in the system the multicriterial analysis based on the quality measures set. As a result of the analysis, the system can give the indices of the best model version.

Aknowledgments

The SOL/PC simulation system was elaborated by a group of researchers and engineers from the Institute of Mathematics. In SOL/PC the results were used obtained by the group in the last 10-15 years, such as creation of methods of translation systems construction for the general-purpose programming languages, realization of basic concepts of simulation languages on computers with various architecture, elaboration of software for the analysis of the results of experiments with models and measurement of program quality. We would like to thank our collaborators and co-creators of the system Grecu G., Marichuc M., Ungureanu M., Tofan T., Costyuc S., Soroncheanu E., Kukotina N.

References

[1] Knuth D.E. McNelley J.L. The formal definition of SOL//IEEE Transaction. 1964 Vol EC-13 N 4. P.409-416 (Engl).

[2] Magariu G.A. The possibilities of outputting additional information about model in the SOL language. High-performance computers

software. Mathematical research. V 79. Kishinev. Shtiintsa.1984. P.84-97 (Russian)

[3] Magariu G.A. The SOL language extension by the means of models analysis. Mathematical research. V.81. Computers and systems software. Kishinev: Shtiintsa. 1985. P.87-90.(Russian)

[4] Kolesnick J.V. Magariu G.A. Marichuc M.N. Series of experiments with model execution and statistical information about model controlled outputting in the system SOL/EC. Mathematical research.V.107. Programming theory and practice. Kishinev: Shtiintsa.1989. P.79-85.(Russian)

[5] Ferrari D. Computer systems performance evaluation. Prentice-Hall. Englewood Cliffs. N.J. 1978.

[6] Halstead M.H. Elements of Software Science. Elsevier North-Holland. New York. 1977.

[7] Madan V.I., Ungureanu M.M. Control of Software Quality. J.Studies in Informatics and Control- With Emphasis on Useful Applications of Advanced Technology. Bucharest.1992.

G.Magariu, V.Madan, L.Burtseva
Institute of Mathematics,
Academy of Sciences of Moldova,
5 Academiei str.,
Chişinău, 277028, Moldova
e-mail: 24*gal*@*math.moldova.su*

Received April 5, 1993