TOOLS OF AN INFORMATION DATABASE CREATION ON A SIMULATION EXPERIMENT PERFORMANCE BY MICIC4 ENGINE

V.D. Liauchuk, A.A. Liauchuk, U.U. Starchanka,
P.L. Chechat, S.F. Maslovich, A.S. Pomaz
Gomel State University named by Francysk Skaryna
Gomel, BELARUS
e-mail: lv@gsu.unibel.by

Abstract

Original approach to the information database creation on simulation experiment performance used in the simulation engine MICIC4 designed by authors is considered.

1 Introduction

Making the information database on simulation experiment performance is an important stage of the simulation project implementation. Standard statistics is used in many simulators because of a method-oriented scheme formalization. For instance in simulators of a transaction type these are usage factors, queues lengths, service times and so on. The report on the experiment is also standardized and usually contains an unnecessary information. The researcher is required to interpret the standard statistics in notions of a problem branch, as well as to use an external software for the transformation of the standard report to suitable type for analysis.

Another approach to the information database creation on simulation experiment performance is used in the simulation system MICIC4 designed by authors Liauchuk (2006). It is founded on the following principles:

- the developer must form contents of the information database in terms of a problem branch;
- the researcher must get only interesting in simulation experiment information i.e. processor time must be not spent on calculation unnecessary characteristics;
- the researcher can get the report in the standard or unique form;
- the information database in the run is intended basically for the simulation model verification, but as a whole on experiment for the decision tactical and strategic problems.
- it is important in run to fasten the numeric attributes of the state to activities, in which the state is changed.

The realization these principles in the simulation model program on MICIC4 is considered below.

2 Arrays of statistical objects

In the narrow sense MICIC4 is a C++ library for a creation of a simulation model program and an implementation of simulation experiments with it. Therefore the solution of the above problem is to present an appropriate set of classes allowing to a model developer to use their interfaces by effective way. Let us consider these classes.

Varied input variables in the theory of an experiment designing are called factors. Corresponding class *Factor* is intended for variation of factors values on several levels. Besides, given class allows to present automatically factors values in report, created on results of the experiment realization. This class has no constructors. The elements of the factors array are initialized through the method *createFactor* of the class *Experiment* , intended for a simulation experiment implementation.

The output variables of an experiment are called responses. Unlike many simulation systems a set of responses in MICIC4 is not fixed. It is formed by the developer of a simulation model. An automation of a calculation of responses and their print in the report is provided through the notion statistics. Statistics is an ensemble of responses, calculated on one and the same expression, but differing by conditions of the calculation. For instance, whole amount of served transactions of all types, an amount of served transactions of each type, an amount of served transactions with the service time smaller than given one and so on - different responses, belonging to the same statistics. MICIC4 provides the automatic renovation of all responses, belonging to one statistics through calling the method *add* of the class *Stat* .

Thereby, a statistics can be considered as the server for elementary responses. In a model program it is needed to ensure one-to-one correspondence between arrays of statistics and responses. It is realized by an enumeration of statistics identifiers to be described originally. Then the array of statistics *Stats* is defined in the same order as an enumeration of statistics identifiers. Array elements are initialized by the call of constructor of the class *Stat* .

In MICIC4 responses are divided into five types: in discrete time, in continuous time, additive, final, resulting. Standard classes correspond to first four types. A developer must create inherited classes for resulting responses from the base class *Response* . Regardless of type all responses are collected in one array of the pointers *Responses* . For efficiency of a simulation experiment implementation the given array is automatically ordered by an enumeration of statistics identifiers. The responses can be brought in the array in free order. Since a renovation of responses is realized through an interface of the class *Stat* it is enough to insert in array *Responses* only that responses to be required for concrete simulation experiment. The code of model activities remains unchangeable.

Constructor of the class *Stat* is the following: *Stat(char *name, int num_ dec=2, int num_pos=10)* . Its arguments are intended for presenting in standard report on a simulation experiment implementation.

The only method of the class Stat is the following: *void add(float value)* . It updates all responses, falling into statistics and satisfying given condition, adding the value *value* in calculated expression of the response.

Constructors of responses classes are the following:

- *Response(float sum=0.0)* . It is a constructor of the base class, where *sum* is an initial value of the response (as in other constructors). The given constructor is necessary to use for the creation resulting responses.
- *DiscreteResponse(float sum=0.0, long num=0)* . Creates a response in discrete time, where *num* is an amount of measurements of the response.
- *ContinuousResponse(float sum=0.0, float prev_ time=0.0, float sum_ time=0.0).* Creates a response in continuous time, where *prev_ time* is the time of the previous change of the response, *sum_ time* is the general time of the measurement of the response.
- *AdditiveResponse(float sum=0.0)* . Creates an additive response.
- *FinalResponse(float sum=0.0)* . Creates a final response.

Methods of responses classes are to be overridden only for resulting responses. They are the following:

- *virtual void add(float x)* . Adds in the calculated expression of the response the value *x* .
- *virtual float mean(void)* . Returns the (average) value of a response in the run for the current moment of a model time.
- *virtual void reset(void)* . Restores an initial value of the response.
- *virtual int condition(void)* . Returns a nonzero value if a condition of a response calculation is fulfilled.

The following set of functions is used for an array *Responses* performance:

- *NewRsp(idStat)* . Reserves the index in array *Responses* for a response from the statistics with a number *idStat* . It returns the pointer on free element in responses array.
- *int Rsp(int stat_ num, int local=0)* . Returns the index of the response from the statistics *stat_ num* with relative number *local* in the array *Responses* . The function on the information from the array *Stats* converts a relative number into an absolute index.
- *int RspNumber(int stat_ num)* . Returns the amount of responses in the statistics with number *stat_ num* .
- *void FreeResponses(void)* . Cleans the array *Responses* , allowing to implement the following simulation experiment with the new set of responses.
- *void ResetResponses(void)* . Restores initial values of all response. It is used at termination of a transition period in a simulation model on the event occurrence.

3 Array of traces

The trace is intended for the verification of a simulation model. It is a text file, in which the simulator manage program adds a record after completion of each activity. It is created automatically if macro DEBUG is defined in header file of a system module micic4.h.

The array of traces *Traces* contains an information about elementary data of a simulation model state to output in the trace file. One part of data is accumulated automatically by simulator and the other one Ц by a developer of a simulation model by means of corresponding functional calls. The simulator manage program always outputs in trace:

- an activity name, model time of its initialization,
- name of the static element, its version number and the number in the array of static elements, an amount of occupied channels on device before an activity call,
- a transactions name and its number in the array of transactions, an amount of occupied channels by transactions,
- a pointer of the direction of a transaction motion after an activity performing.
- interacting device and transaction priorities,
- a transaction state at the beginning and on the termination of an activity.

An interface of the class *Trace* is used for including in the trace unique data. Constructor of the class *Trace* is the following: *Trace(char *header)* . Its argument *header* is a name of some characteristic added in the trace. A set of functions *void FillTraceType(char *header, type value)* allows to save the *value* of the trace element with the name *name* in the trace. As it was noted above an immediate record into the trace file is fulfilled on an activity termination.

4 Conclusions

Thus, the researcher of a simulation model designs an information database of an experiment performance accordingly to the problem branch tasks. Further processing and analysis of collected data is realized in the environment of the most suitable for the researcher statistical software.

References

[1] Liauchuk V.D., Maximey I.V. (2006). *Program Technological Engines of Complex Discrete Systems Simulation.* Gomel State University, Gomel (in Russian).