TransactionNumber: 1362606

ILL Number:	79026730	**Call #: QA76.5.S893 1984**
MaxCost:	$50IFM	**Location: aa 900000013478**
Billing Category:	RLG	

Article Information

Journal Title: Proceedings of the 1984 Summer Computer Simulation Conference, July 23-25, 1984, The Copley Plaza Hotel, Boston, Massachusetts /

Volume: **Issue:**
Month/Year: July 1984 **Pages:** 967-971

Article Author:

Article Title: EXP - A Software Tool for Experimental Frame Specification in Discrete Event Modeling and Simulation

Loan Information

Loan Title:

Loan Author:

Publisher:
Place:
Date:
Imprint:

Borrower Information

University of Arizona Libraries
Interlibrary Loan
1510 E University Dr.
Tucson, AZ 85720-0055

Odyssey: 150.135.238.6
Ariel: 150.135.238.50
Email: askddt@u.library.arizona.edu

NOTES

/ 6/16/2011 9:23:01 PM (System) Borrowing Notes; For Book Chapter requests, scan the chapter only, do not lend book. If over your page limit, please CONDITIONAL request. Thank you.

RETURN LOANS TO: Penn State University Libraries / Interlibrary Loan / 127 Paterno Library, Curtin Rd. / University Park, PA 16802

EXP – A Software Tool for Experimental Frame Specification in Discrete Event Modelling and Simulation

Jerzy W. Rozenblit
Department of Computer Science
Wayne State University
Detroit, Michigan 48202

ABSTRACT

The design of a software tool to formalize and specify experimental frames in discrete event modelling and simulation is presented in the paper. A concept of expressing objectives of modelling through the notion of experimental frame is discussed. Formal definition and representation of a frame are given. Called EXP, the software tool also facilitates user specification of semantic relations on variables and computes derivability for frames. Interactive application of the system in computer-aided modelling is described.

INTRODUCTION

Modelling and simulation designates a host of activities supportable by computer assistance, associated with constructing models of real world systems and simulating them on a computer. Such activities in a typical modelling process usually comprise the following steps: system decomposition, model construction, model and experimentation specification. The growing complexity of the systems being simulated, on one hand, and the tremendous progress in digital technology, on the other hand, have strongly influenced and motivated the research on development of software tools for modelling and simulation support (3).

While a variety of programming languages are presently available we still lack adequate software support for model construction and development process. In fact there have been no attempts to provide tools for experimentation specification other than the ones that are embodied in some simulation languages like GEST, SIMAN, ISIM or LOBSTER (1,2,3,4).

In the paper we focus on the experimentation aspect of the discrete event modelling methodology and present a prototypic software system for experimental frame specification.

OBJECTIVES-DRIVEN MODELLING METHODOLOGY

The conceptual basis for a methodology of model construction in which the objectives of modelling play the key and formally recognized role (therefore called objectives-driven methodology) was laid down by Zeigler (9). While his approach has been largely theoretical it has been guided by the motivation to implement the methodology in the form of computer assisted modelling.

We shall begin with theoretical foundations of the objectives driven methodology. The basic process in such a methodology is that of defining the experimental frame i.e., a set of circumstances under which a model or real system is to be observed and experimented with. This process comprises the following steps. The purposes (objectives) for which the simulation study is undertaken lead to asking specific questions about the system to be simulated. This in turn requires that appropriate variables be defined so a modeller can answer these questions. Ultimately such a choice of variables is reflected in experimental frames which also express constraints on the trajectories of the chosen variables. The constraints on observations and control of an experiment should be in agreement with the modelling objectives. A choice of relevant variables constitutes the first important stage of experimental frame specification. The next step is to categorize the variables into input, output and run control categories and place constraints on the time segments of these variables. Formally, the experimental frame specifies the following seven tuple:

$$EF=<T,I,O,C,\Omega_I,\Omega_C,SU>$$

where T is a time base and

$I=\{I_i|i=1,2,\ldots,n\}$ is the set of input variables

Let $X=R_I$ be the crossproduct of the ranges of individual input variables. The set X is called input value set .

$$\Omega_I=\{\omega|\omega: T \text{ ----> } X\}$$

denotes the set of all admissible input segments (ω can belong to a certain class of time segments e.g. piecewise constant ,step or discrete event segment (9).

Ω_I is a subset of all time segments over the crossproduct of the input variable ranges i.e. $\Omega_I \subset (T,X)$.

$O=\{O_i|i=1,2\ldots,k\}$ is the set of output variables

Let $Y=R_O$ be the crossproduct of the ranges of individual output variables. The set Y is called output value set.

Given Y we can define (T,Y) as the set of all segments over the output space. The I/O data space defined by the frame is the set of all pairs of I/O segments:

$$D=\{(\omega,\rho)\mid \omega\epsilon\ (T,X)\ ,\ \rho\epsilon\ (T,Y) \text{ and } dom(\omega)=dom(\rho)\}.$$

Any input/output data acquired in an experiment on a model within the frame lies in D.

$C=\{C_i|i=1,2\ldots,n\}$ is the set of run control variables

Let $Z=R_C$ be the crossproduct of the ranges of individual control variables. The set Z is called control value set

Finally, SU is a set of summary mappings that have as domain the I/O date space D defined by the frame (9).

The concept of run control variables needs explanation. In the case of experimentation on a real system, there is no concept of initial state. Thus, specifying the input segment in the frame is not sufficient to determine the output of the system. Since experimental frames should have an interpretation for both the model and the real system, we should provide a meaningful concept of restricting the initial state for the model. The notion of run control variables serves this purpose. Not only do the run control variables initialize the experiments, they also set up the conditions for continuation and termination. The set of initialization conditions constitutes a subset of the control space called INITIAL. Similarly, the subset of the control space defined by the termination conditions is called TERMINAL. These two sets have the following impact on the experimentation. An experiment starts with the control variable values in the INITIAL set and terminates as soon as the TERMINAL subset is entered. In other words, it is continued as long as the values of the control variables stay in the subset called CONTINUATION. Thus, we arrived at the definition of the set of run control segments

$$\Omega_C=\{\mu|\mu:<t_i,t_f>\to Z$$

and $\mu(t_i)\in$INITIAL $\mu(t)\in$CONTINUATION for $t\in[t_i,t_f)\}$.

CONCEPTS OF DERIVABILITY

In the previous section we presented the experimental frame concept. This section places the theory of objectives-driven methodology in the context of computer assisted modelling.

As depicted in Figure 1 the following tools and procedures should be available to a modeller (7,8).

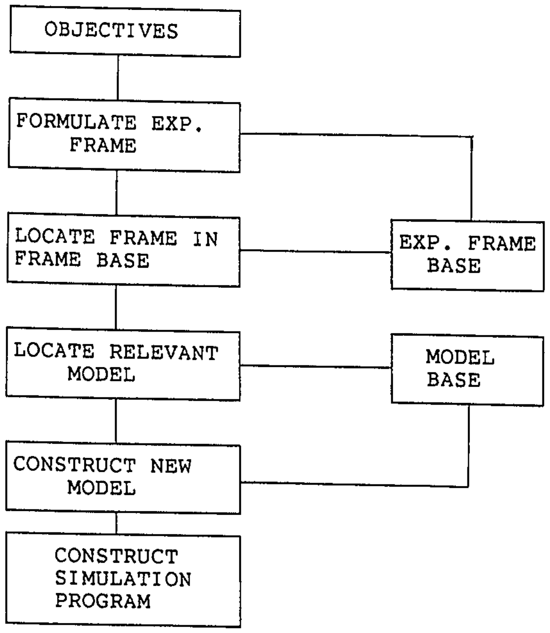

Fig.1 Computer assistance to modelling.

The modeller should have assistance in specifying an experimental frame that represents the objectives as well as have assistance in locating this frame in the frame base. Also the models relevant to the new frame should be located in the model base. There actually may exist a model to which the new frame is applicable i.e., a model capable of answering the questions that resulted from the objectives, and were posed in the new frame. If such a model does not exist in the base the procedure for its construction should be provided as well. Constructing a new model may require the modeller to start from scratch or to simplify, modify or enhance the models from the model base to obtain the composite model. This implies that the model as well as the frame base should be somehow organized so the procedures of model and frame retrieval can be automated in the algorithmic manner. The concept of derivability significantly facilitates the above mentioned features.

Derivability is a relation that partially orders frames according to the extent of experimentation they embody (9). To provide the formal definition of derivability relation requires introduction of the semantic relation on variables, which we herewith present.

The semantic structure of variables

Recall Figure 1 and the steps one should follow in the modelling methodology involving model and frame bases. In the case of experimentation with this type of software assistance it is natural to augment the tools with the base of generic variable types. Then, the modeller faced with a problem of modelling a certain class of systems would refer to an appropriate variable base (that would consist of generic variable types suitable for the class his problem belongs to) and would formulate a set of experimental frames using variables from this base. Within such a variable base we expect certain relations to hold among variables. More precisely, they are relations on variable ranges and thus are termed the semantic relations (9). The set of all such relations in a variable base is called the semantic structure. If the variables related by the semantic structure appear in different frames, this constitutes a correspondence allowing us to determine the derivability relation.

Formally the semantic relation between variables V_1 and V_2 is a relation:

$$R \subset V_1 \times V_2$$

To employ the semantic structure to the concepts of derivability we seek a functional relationship on variable ranges that results from the semantic relation between the variables. Assume that $\mathcal{VB}$ denotes the variable base and $\mathcal{R}$ the set of all semantic relations in this base. Then we say that the variable V is derivable from the variable W, where $V,W \in \mathcal{VB}$ if a functional relation $f\in\mathcal{R}$ holds between V and W, such that

$$f:\ W.\text{range} \dashrightarrow V.\text{range}$$

and f is an onto mapping. Every variable is derivable from itself. The finite composition of onto functional semantic relation is an onto functional semantic relation. Hence derivability is a transitive relation.

We utilize the semantic relation concept in order to define derivability of input and run control segments of a frame. Let I and I' be the set of input variables such that I' is derivable from I in the semantic structure. Let $f:\ (T,X) \dashrightarrow (T,X')$ be a segment to segment mapping such that $f(\omega)(t)=f(\omega(t))$ for each $t\in \mathrm{dom}(\omega)$. Then the input segment set Ω' is derivable from Ω if the following conditions hold:

$$i.)\ f(\Omega_I) \subset \Omega_I'$$

$$ii.)\ \text{for all } \omega \in \Omega_I' \quad f^{-1}(\omega) \subset \Omega_I$$

To illustrate this concept let us consider an example of a queuing systems in which jobs arrive at a processor. We define an input variable PROCESSOR.JOB with the range set X={ non-negative integers } (e.g. task numbers). Assume now that in another frame a variable PROCESSOR.ARRIVAL with the range set X'={0,1} has been defined. Let ϕ be a point-wise extension of f and extract only the fact of arrival or non-arrival of a job i.e., $\phi(x)=1$ if $x<>0$ and $\phi(x)=0$ if $x=0$. It is clear that the variable PROCESSOR.ARRIVAL is derivable from PROCESSOR.JOB. The derivability relation also holds for the input segments defined over these two variables and the segment-to-segment mapping ϕ.

The case of run control segments is similar. However, since the control variable trajectories do not appear in the I/O data we require a stronger notion of derivability namely the condition ii.) should hold in the following form

$$iii.)\ f^{-1}(\Omega_C') = \Omega_C$$

This means that the restrictions placed on the sets of segments are the same when translated by the mapping f. The formal definition of experimental frame derivability is given by Zeigler in (9).

Having laid the theoretical background for experimentation methodology , in the ensuing sections we demonstrate its implementation in the form of an interactive software tool for experimental frame specification called EXP.

EXP - A TOOL FOR EXPERIMENTAL FRAME SPECIFICATION

The design of EXP was driven by the following requirements. We wanted the system to perform two functions: allow for specification of experimental frame bases and for ordering the frames with respect to the derivability relation. The second objective implied the need for specification of the generic variable base and the corresponding semantic structure. These preliminary requirements resulted in the architecture depicted in Figure 2.

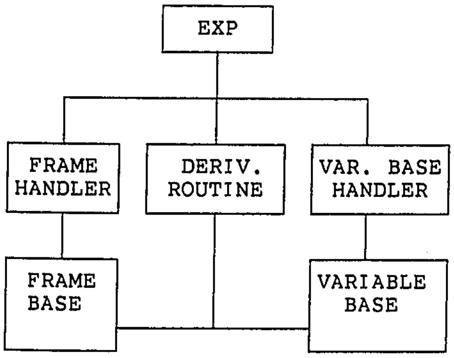

Fig.2 System architecture

A user is able to perform the following operations while using EXP:

1.) Display frames stored in the frame base.

2.) Add a frame to the base

3.) Remove a frame from the base

4.) Order the frames with respect to the current semantic structure

5.) Edit a frame.

The functions 1. trough 3., and 5. are performed by the frame handler module. The function 4. is carried out by the derivability routine and the variable handler.

The actual specification of a frame takes place under the frame editor control which enables the below listed operations:

1.) Add an <input| output| control<variable>| input| control<segment>>.

2.) Remove an <input| output| control<variable>| input| control<segment>>.

3.) Display the frame being edited.

The functions associated with the derivability routine and the variable handler fall into the following categories:

1.) Initialize new generic semantic structure

2.) Update the current semantic structure

3.) Order frames with respect to the current structure and observe global and/or partial derivability.

The updating of the current structure consists in adding a new generic variable to the variable base and specifying all the semantic relations that hold between the added variable and the variables that are already in the base.

In the next section we present the EXP representation of experimental frames and the description of the procedures governing the specification of the generic semantic structure. The method of ordering the frame base is also given.

REPRESENTATION OF EXPERIMENTAL FRAMES IN EXP

The frame base is a file containing all the experimental frames defined by the user. As we mentioned before the procedure of a frame specification is supervised by the frame handler (recall Figure 2). The frame base is a set:

$$\text{FRAME-BASE}=\{\text{FRAME}_1,\ldots\ldots,\text{FRAME}_n\}$$

Each frame in the base has the following representation:

FRAME=<name,{I},{O},{C},{ISEG},{CSEG},{PARAMETERS}>

where:

{I} is the set of input variables

{O} is the set of output variables

{C} is the set of run control variables

{ISEG} is the set of input segments

{CSEG} is the set of run control segments

{PARAMETERS} is the set of variables enabling parametrization of the frame .

Each variable in the frame has its own representation:

FRAME.VARIABLE=ENTITY.VARIABLE

where VARIABLE is one of the possible generic variables offered by the generic variable base, while ENTITY is the name of the entity the variable pertains to.

What is the motivation behind such a representation? First, EXP should be an extension of the Entity Structuring Program (ESP) (6). In such a mode of operation the modeller uses the variable base to define variables in both the entity structure and experimental frames. Thus, the variable base serves as an interface and a resource shared by the two systems. The variables are differentiated by the entities name. This is obvious since a variable may pertain to many entities. Secondly, having given a generic variable base the user can order a number of different frames (all having the same variable types but maybe different components) with respect to the semantic structure over the variable base.

The segment representation is rather simple

SEGMENT=<name,characteristic variable>

The characteristic variable must be one of the variables already specified in the {I} or {C} set. (It simply represents the range set of a given segment). We plan to augment the representation of segments with elements that will allow a user to define the actual dynamics and discrete event nature of a segment.

It is now important to show how the concepts of the semantic structure and derivability are realized in EXP.

INTERACTIVE APPLICATION OF EXP IN COMPUTER-AIDED MODELLING

Recall from Figure 2 the variable handler and the variable base. There are two ways the modeller can utilize these tools. Assume that he begins from scratch and there are no frames in the variable base. However, he is faced with the task to develop an experimental frame base for a certain class of models , say, the queuing models. First he may want to determine the generic variables for this particular class. Then, the variable handler prompts him to determine all the pairwise semantic relations among these variables. He does so, having the knowledge of the variable ranges. (Notice that it is not required by the system to specify the variable ranges). The variable handler automatically computes the transitive closure of the semantic relations applying the Warshall algorithm (5). This is done to ensure that all the variable pairs are properly related.

The system then sets up a structure of all generic variables with the corresponding semantic relations. This constitutes the generic semantic structure (for each variable in the base the system determines all the variables that a given variable can be derived from).

The next step the modeller may undertake is to define experimental frames with respect to his modelling objectives and the class of models. While specifying the frames he should use the generic variables and must follow the syntax ENTITY.VARIABLE while defining the frame variables. If he decides that a new generic variable is indispensable for model development then he can update the semantic structure at any moment, by adding a new variable and specifying all the semantic relations.

After he has completed this stage he can order the frames with respect to the current structure. An option of observing the global and/or partial (i.e. input, output etc.) derivability is given and the system performs this task by checking all the frames that are stored in the base.

The modeller can use the EXP system in another way. Faced with a class of models and a set of experimentation objectives he can retrieve a generic variable base that corresponds to the class of systems to be modelled. Then he can search a frame base for a frame that would suit his objectives. If such a frame cannot be found the modeller should specify a new frame and order the frame base with respect to the current generic semantic structure. This will locate the new frame in the base and point to all the frames related to it.

Computation of derivability

The derivability routine handles the problem in three separate steps. First, it determines the input semantic structure. All the input variables from the frame base are ordered with respect to the generic semantic structure. Then, every pair of frames is checked to determine if it satisfies the derivability condition and if it does this fact is recorded as the input variable derivability. At the same time the input segment derivability is determined. The same input semantic structure is used due to the fact that the segments' characteristic variables are frame input variables.

The next step is very similar. The routine sets up the output semantic structure for all the output variables. Then all the frames are checked for output derivability. A slightly different procedure is applied to the run control variables. In this case the control semantic structure must contain all the run control, input and output variables. The semantic relations, however, are defined only on the following crossproduct:

$$\text{Control Semantic Relation} \subset \{C\} \times (\{C\} \cup \{I\} \cup \{O\})$$

Then, the frame control derivability is determined with respect to control variables and segments. Finally, the global derivability relation is defined as a product of the partial relations.

FURTHER DEVELOPMENT OF THE SYSTEM

Our ultimate goal is to fully implement the concept of experimental frame in the context of computer assisted modelling. With this motivation, we shall consider a design of a high level procedural description of frames. Although the EXP system is a good experimental frame base management tool it does not allow the user to generate a meaningful experimental module which could be used in a simulation program design. We shall follow the path suggested by Javor (2) and design a high level language allowing for specification of frames applicable to a general class of discrete event systems. Such a language should provide a means for a dynamical representation of experimental frames reflecting the time-event profiles of the input and control segments.

ACKNOWLEDGMENTS

I wish to thank Prof. Bernard P. Zeigler for his advice, encouragement and many informative conversations which helped to improve the presentation of this work.

REFERENCES

1. Crosbie R.E., Hay J.L., "ISIM - A Simulation Language for CP/M Systems", Proc. of the Summer Computer Simulation Conference, Vancouver, July 1983.

2. Javor A., "Proposals on the Structure of Simulation Systems", in:Discrete Simulation and Related Fields, (ed. Javor, A.), North-Holland, Amsterdam, 1982.

3. Oren T.I., "Computer Aided Modelling Systems", in Progress in Modelling and Simulation, Academic Press, London, 1982.

4. Pegden D.C., "Introduction to SIMAN", (manual) Systems Modeling Corporation, 1982.

5. Prather R., "Discrete Mathematical Structures for Computer Science", Houghton Mifflin Publ. Comp., Boston 1976.

6. Zeigler B.P., Belogus D., Bolshoi A., "ESP: An Interactive Tool for System Structuring", in Proc. European Meeting on Cybernetics and System Research, Vienna, Hemisphere Press, 1980.

7. Zeigler B.P. "Structures for Model Based Simulation Systems", in Simulation and Model-Based Methodology: An Integrative View, Springer-Verlag, New York, 1984.

8. Zeigler B.P. "Modelling and Simulation Methodology: State of The Art and Promising Directions", Simulation of Systems '79, North Holland, Amsterdam, 1980 pp. 819-835.

9. Zeigler B.P., "Multifacetted Modelling and Discrete Event Simulation", to be published by Academic Press, London, 1984.