IMACS
International Association for Mathematics and Computers in Simulation

AFCET
ENSM

SYMPOSIUM INTERNATIONAL

LA SIMULATION DANS LES SCIENCES POUR L'INGENIEUR

SIMULATION IN ENGINEERING SCIENCES

9—11 Mai / May 1983

NANTES (France)

PRETIRAGES *PREPRINTS*

EDITEURS / *EDITORS*

Jacques BURGER Yvon JARNY

Laboratoire d'Automatique ENSM — NANTES

TABLE DES MATIERES - CONTENTS

I - OUTILS DE SIMULATION - Logiciels / SIMULATION TOOLS - Software

- A language for real time simulation of processes with Boolean inputs and outputs
E.F. Camacho, L.G. Franquelo, J. Lozano (E) 3

- Computer aided modelling of complex processes : a program package
L. Marcocci, S. Spelta (I) 9

- Flexible software package for railcar design
R.C. White, A.A. Merabet (CDN) 15

- CATPAC : a software package for computer-aided control engineering
M. Barthelmes, P. Bressler, D. Bünz, K. Gütschow, J. Heeger, H.J.Lemke (FRG) 21

II - METHODES DE SIMULATION - METHODES NUMERIQUES
SIMULATION METHODS - NUMERICAL METHODS

- Simulation of on-line state estimation for distributed dynamic systems
A. Maslowski (PL) 29

- The use of numerical simulation to verify the efficiency of new PWM strategies for the feedback control of a D.C. motor drive
L. Fortuna, A. Gallo, M. La Cava (I) 35

- Programmation dynamique différentielle, mise en oeuvre d'algorithmes et applications
J. Lopez Coronado, L. Le Letty (F) 41

- Linear approximation of nonlinear systems based on least squares methods
J.G. den Hollander, J.A. Hoogstraten and G.A.J. van de Moesdijk (NL) 53

- Simulation of engineering problems using boundary elements
C. A. Brebbia (GB) 59

III - OUTILS DE SIMULATION - Matériels / SIMULATION TOOLS - Hardware

- Graphic model building system - GMBS -
Y. Yamamoto, M. Lenngren (S) 71

- The sole simulation package in Pascal
J.E. Rooda, S. Joosten (NL) 77

- UNISYS - Computer assisted modelling and simulation system
K.A. Grabowiecki (PL) 83

- Dynamic system simulation in designing computer peripherals
M.H. Dost (USA) 89

IV - METHODES ET TECHNIQUES DE SIMULATION ET DE COMMANDE
SIMULATION AND CONTROL METHODS AND TECHNIQUES

- Application d'un outil de simulation à la conception des surfaces gauches : notion de processus interpolateur
R. Haj Nassar, D. Meizel, P. Bielec (F) 97

- Simulation - Aid to process interaction
D. de Buyser, L. de Wael, G.C. Vansteenkiste (B) 103

A LANGUAGE FOR REAL TIME SIMULATION OF PROCESSES WITH BOOLEAN INPUTS AND OUTPUTS

E.F.Camacho, L.G.Franquelo and J.Lozano

E.T.S. Ing. Industriales Univ. Sevilla

This paper deals with the problem of real time simulation of processes with boolean inputs and outputs. A language for this purpose and the programs that processes it is presented. The language allows the description of processes with simultaneous evolutions as a timed petri net type of description is used. Random failures can also be introduced in the behaviour of the model. The language allows the control of a semigraphic CRT in order to facilitate the task of following the model behaviour.

1. INTRODUCTION

Simulation is a fundamental tool when developing logical automatas, especially when those automatas are designed to control complex processes or processes where testing is expensive. As an example the starting up and shutting down procedure of a hydroelectric power unit or controlling substation operations, where on-line testing of the automata should be avoided as much as possible due to the risk of damaging expensive equipement while running the experiment.
The simplest of all possible boolean simulators consists of a set of switches, simulating process inputs, and lights, simulating process outputs. The automata is connected to these and a human operator moves the switches as the plant would do according to the sequence of orders received from the automata. The human operator is in this way simulating the behaviour of the plant. This method of testing an automata has tree major drawbacks. The first one is that human operators are very slow; it can take 15 seconds or more for the operator to decide which switches must be changed if the process he is simulating is complex enough. The second disavantage is that due to frequent errors the system is not properly simulated. Finally it is very difficult to carry out systematic tests with this method. Therefore only very slow or simple processes can be simulated with this method in real time.
A hardware model of the plant overrides all these problems but it is normally very costly and unflexible.
This paper presents a software simulator implemented in a PDP 11/23 computer. The simulator consists of a language and a collection of programs for processing it. The language is based on the Petri Nets approach for describing automatas.

The overall system structure is described in the next section. The language is treated in section 3 and some examples are given in section 4.

2. SYSTEM STRUCTURE AND FUNCTIONING

The structure chosen for the simulator can be seen in figure 1. It consists of a computer connected to the automata to be tested and to a semi-graphic CRT that allows an interactive simulation. The automata is connected to the computer via parallel input-output digital ports.

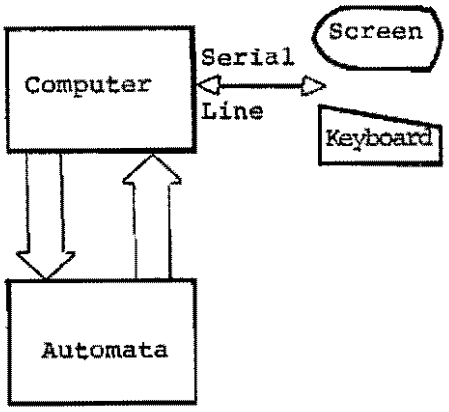

Figure 1. System structure

Once the model is running, manual operations can be introduced easily using the keyboard attached to the CRT. As an example, when simulating an electrical substation a breaker can be manually opened or closed by the operator while the model is running.
A mimic can be related to the model and it will appear on the screen as soon as the simulation program begins. Up to 128

active points can be defined for each -- application. These points are related to 4 boolean variables two of which are associated to the keyboard (inputs for the simulator) and the other two to the screen (outputs for the simulator).
The keyboard signals are activated positioning the screen cursor on the active position and pressing one of the four -- predefined keys that are associated with the values 00, 01, 10, 11 for the two -- keyboard signals mentioned above. As was mentioned before, these keyboard variables will simulate manual operations and will be considered as input signals for the model. In the example of the circuit breaker mentioned above, an active point could be related to it and the open and close would be associated to the two keyboard signals corresponding to that active point with values 10 and 01 respectively. The non logical conditions (11) of these signals can be used within the model to declare a defective breaker -- which can be useful to test the automata under malfunctioning of the process.
The two screen signals mentioned before are output variables for the model. Up to four symbols can be associated to the four possible values of these two signals. The simulator will represent in -- the related active point the symbol corresponding to the values of the variables. These variables are very useful -- for an interactive simulation of the process, as the model behaviour can be easily followed on the screen.
Besides the keyboard and screen variables mentioned above, the simulator allows the use of 512 input-output signals connected to the automata. Internal boolean variables (up to 256) can also be used. These internal signals are useful for connecting Petri Nets.
In order to simulate stochastics failures or evolutions in the model, up to 32 randomly generated boolean signals are provided. The first 16 of these signals are generated by a 1 second clock whilst the other 16 with a 1 minute clock.
The way of operating the simulator can be seen in figure 2. The model is defined in a simulation language that will be described in the next section. The model is compiled and the tables and code necessary for the simulation program are produced as is shown in figure 2.
The simulation program reads the output data of the compiler and the graphic representing the process from a disk.
With this information and the input signals (external , internal , keyboard and random) the simulator moves the output signals (external, internal and screen) according to the process description and its actual state.
A matrix method (2) is used to compute - the marking of the nets.
The amount of memory and the computation required decrease considerably if instead of using a single big net for modelling the system, various small nets are used.

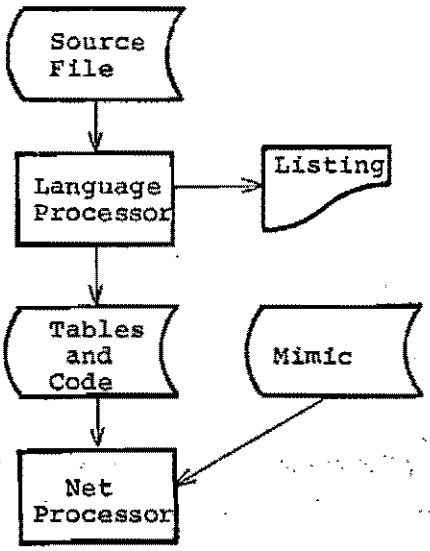

Figure 2. System operation.

3. SIMULATION LANGUAGE.

To facilitate the task of modelling, a simulation language has been defined -- and its processor implemented. The language uses the Petri Nets approach allowing a model to contain several Petri Nets.
Each net defined in the model begins -- and finishes with reserved words. Labels can be associated with the booleans variables described in the previous section. The labels can be then used as outputs associated to the marking of any place or can be used as part of an expression in a transition.
Labels can be global, valid for all --- nets, or local to a net. Labels can be associated with a boolean signal specifying type and number of signal or indicating only type. The processor will associate the next free signal of that - type in this last case. This and the possibility of using local labels allows the effective use of a macro processor, which is implemented within the program, thus facilitating the modelling of systems with repetitive parts (see example of substation given below).
Four labels can be assigned to a screen active point, giving the coordinates. - The first two correspond to the keyboard and the last two to the screen signals. The four graphic simbols associated with the screen labels must be given in the asignation instruction.

Transitions are defined indicating their number (within the net), the places entering and leaving the transition and a boolean infix expression associated with it. The expressions can contain any label previously defined, the boolean operators NOT, AND, OR and parenthesis.
The net structure is defined once all the transitions have been specified. Outputs are, in this model, associated to the marking of the places. Therefore it is necessary to use another type of instruction specifying the initial marking of the places and the outputs related to them if any. It is also possible to introduce timed Petri Nets, this is achieved by specifying the time delay associated with each place if any. The marking of a place is not effective (for outputs or validating transitions) until this delay time has elapsed.

4. EXAMPLES

To illustrate the scope of the language, two of the applications where the simulator has been used are described. The first one is a model for an electrical substation. The second example is a hydroturbine generation unit. Automatas to control some functions of these two systems are being developed and the simulator is being used to test the behaviour of the automatas.

4.1 Electrical substation

This example shows how easily systems with repetitive parts can be described with the language presented.
As was mentioned in the previous paragraph, an automata for controlling certain aspects of the operation of electrical substations is being developed. This automata is based on microprocessors and has three main functions: load shedding, automatic reclosure and faulty ground detection. The model of the substation should therefore reproduce in its behaviour those aspects of the behaviour of the substation which are relevant to the functions mentioned above.
The two main elements in a substation are circuit breakers and line switches as a substation contains several of these elements, they will be defined as macros.
The program listing for the macro describing the circuit breaker is the following:

```
01   .MACRO BREAK XX,YY
02  NET
03  &I=X+
04  &SA=0+
05  &SC=0+
06  &EA=I+
07  &EC=I+
08  &TA,&TC,&PA,&PC=XX,YY 96,97,0,X
09  TR 1 FROM 1 TO 2 EX &TC*~&TA+&EC*~&EA
10  TR 2 FROM 2 TO 3 EX  ~(&TC*&TA)
11  TR 3 FROM 3 TO 4 EX ~&TC*&TA+~&EC*&EA
12  TR 4 FROM 4 TO 1 EX ~(&TC*&TA)
13  PL 1 M 1 &PA,&SA
14  PL 2 T S 2
15  PL 3 &PC,&SC,&I
16  PL 4 T S 2
17  ENDNET
18  .ENDM
```

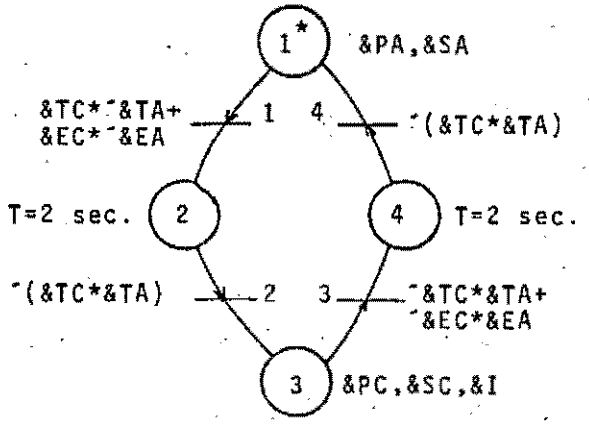

Figure 3 Circuit breaker Petri Net.

Lines 2 and 17 correspond to the instructions specifying the begining and end of a Petri Net.
Lines 3 to 7 are label definitions, the symbol & specifies that these labels are local to the net. The label &I will be associated with the next available internal signal, the labels following,&SA,&SC,&EA and &EC will be associated with the following two available output and input signals respectively.
Signals &EA and &EC are associated to the open and closure orders to the circuit breaker,whilst the local labels &SA and &SC are associated to the open and closed position switches of the circuit breaker. The internal signal &I will be used inother nets where the circuit breaker position needs to be known, as is the case of the faulty ground detection where topological considerations are needed. Line 8 of the listing corresponds to the definition of an active point on the screen. In this case only one active point is associated with the circuit breaker. The signals &TA and &TC correspond to the keyboard signals associated with the breaker, simulating as was mentioned before, manual operation on the circuit breaker. &PA and &PC correspond to the screen variables for the active point and the four parameters at the end to the symbols that will be associated with the four possible values of the signals

&PA and &PC.The parameters XX and yy are the relative x-y position of the active point on the screen.

Lines 9 to 12 describe the transitions - of this particular net. As it can be seen, they are defined by the input and output places and by the associated boolean expression.

Lines 13 to 16 define the initial marking of the places, the associated output and time-lag. Line 14 specifies that a 2 second time delay should be observed for - place number 2.

A line switch can be modelled using an identical net to the one described above except for the external input signals , which are non existent as the automata is not going to alter line switches,and the screen representation which is also di - fferent.

Figure 4 shows the screen representations of a coupling cell and a line cell.These types of cells can be defined as follows

```
.MACRO CELLIN OX,OY
;       LINE CELL POS. OX,OY
X=OX
Y=OY
;       LINE BREAKER POS. OX+5,OY+4
BREAK 5,4
;       LINE SWITCH BUS 1 POS. OX+7,OY+2
SWITCH 7,2
;       LINE SWITCH BUS 2 POS. OX+3,OY+2
SWITCH 3,2
;       SWITCH BYPASS POS. OX,OY+2
SWITCH 0,2
;       LINE SWITCH POS. OX+5,OY+6
SWITCH 5,6
;       LINE VOLTAGE  POS. OX+5,OY+9
VL 5,9
.ENDM


.MACRO COUPL OX,OY
;       COUPLING CELL POS. OX,OY
X=OX
Y=OY
;       COUPLING BREAKER POS OX+2,OY+4
SWITCH 2,4
;       SWITCH BUS 1  POS. OX+4,OY+2
SWITCH 4,2
;       SWITCH BUS 2  POS. OX,OY+2
SWITCH 0,2
.ENDM
```

The program line with VL is a macro call to a Petri Net where the line voltage is simulated. The general program of a mo - del that only takes into account the behaviour of the switches and breakers is given in the following listing.

```
0074  ;
0075  ;       PROGRAM
0076  ;
0077  COUPL 3,12
0078  CELLIN 11,12
0079  CELLIN 22,12
0080  CELLIN 33,12
0081  CELLIN 44,12
0082  CELLIN 55,12
0083  CELLIN 66,12
0084  COUPLU 3,10
0085  CELLIU 11,10
0086  CELLIU 22,10
0087  CELLIU 33,10
0088  CELLIU 44,10
0089  CELLIU 55,10
0090  CELLIU 66,10
0091  X=0
0092  Y=0
0093  ;       SWITCH BUS 1
0094  SWITCH 76,11
0095  ;       SWITCH BUS 2
0096  SWITCH 78,11
0097  END
```

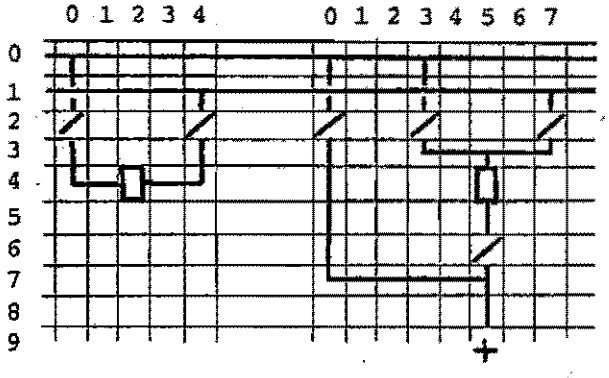

Figure 4. Coupling and line cells screen representations

The mimic representation associated to this program can be seen in figure 5.As it can be observed this model only considers the isolated functioning of the main elements of the substation. More Petri Nets have been developed to simulate other aspects of the behaviour of the process. As an example, let us consider the faulty ground signals generation.

These signals are obtained at transformers and should be computed (by the simulator) taking into account the faulty lines and busbars (declared by the operator via keyboard signals) and topological considerations (depending on the status of the circuit breakers and line switches). For this purpose four internal signals are defined. These signals represent a faulty ground transmited to one of the four busbars, and are set if one or more faulty grounded lines are -

connected to the respective busbar or if a busbar ground signal has been set by the operator. To obtain these signals a trivial Petri Net with four places is associated with each line cell.

The faulty ground signals on the transformers are then easily computed from the busbars faulty ground signals and very simple topological considerations on the transformer cells status.

4.2. Hydroturbine power unit.

A model of a hydroturbine power unit has been developed to test on automata controlling its starting up and shutting down procedures. To show the scope of the model let us sumarize the starting up procedure taken from (3).

After a local or remote start up order has been received and once the security conditions have been checleed the starting pilot valve is operated. After that, the limiter of the hydraulic actuator is reset to the no-load position, the gate apparatus is opened and control is transfered to the automatic synchronization equipment once the generator speed is over a porcentage of its nominal value.

The secuence of operations in the normal shutting down procedure is practically the same but in reverse order. In the case of an emergency shut down the secuence is simpler as most of the elements of the unit are turned off at the same time.

From the automata point of view, the hydroturbine power unit consists of a set of elements that behave as logical systems. For example the oil system with two inputs (activating and disactivating the oil pumps) and one output that is set after a time lag which depends on the oil preassure is adecuate. This behaviour is very easily modelled with a two places Petri net.

The behaviour of the hydroturbine is modelled with a Petri Net composed of a doubled linked chain of timed places. The outputs of these places correspond to the speed relays of 0.5, 20, 90 % , and overspeed and the evolution of the marks depending on the valves positions. Various types of failures can be introduced in the behaviour of the model via keyboard signals. For instance keyboard signal is used in the transition between places with outputs associated with 20 and 90 %. When this keyboard signal is set the transition is not validated and a time out alarm should be produced in the automata. Other aspects of the unit are modelled this way. The

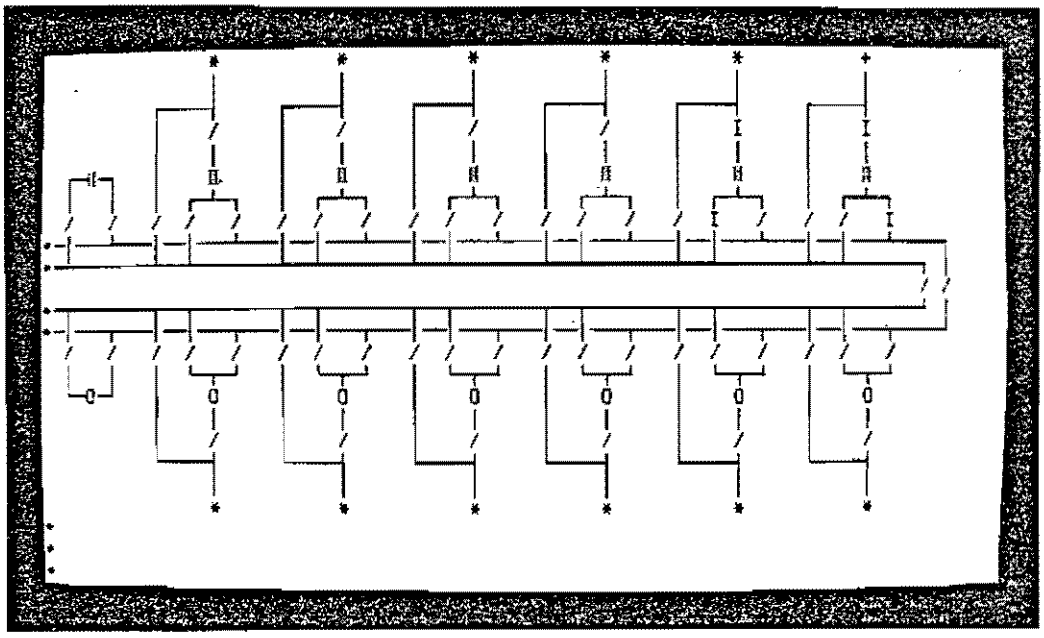

Figure 5. Substation Screen Mimic.

screen mimic associated with the model - can be seen in figure 6.

For these two examples presented, the cycle time has been smaller then 0.2 seconds. Both of them containing over 400 places and about the same number of transitions.

CONCLUSIONS.

A software simulator for modelling boolean systems bases on the Petri Nets - description approach has been implemented. The simulator allows an interactive operation via a CRT, which has proved very helpful in the cases simulated to increase the description power of the lenguage, output functions are now being implemented.

ACKNOWLEDGEMENT.

The authors would like to acknowledge - the help of the Cía. Sevillana de Electricidad for supporting this project. Comments by Mr. J.C. Serrano, J. Colmenero, F. Mateo and J. Montaner were especially helpful.

REFERENCES:

1 Peterson, J.L. Petri Nets, A.C.M. Comp. Surveys, vol. 9, n°3 (1977), 223-251.

2 Daclin, E. and Blanchard, M., Synthese des Systémes Logiques (Cepadues, 1976).

3 Barzan A., Automation in Electrical Power Systems. (Mir, Moscow, 1981).

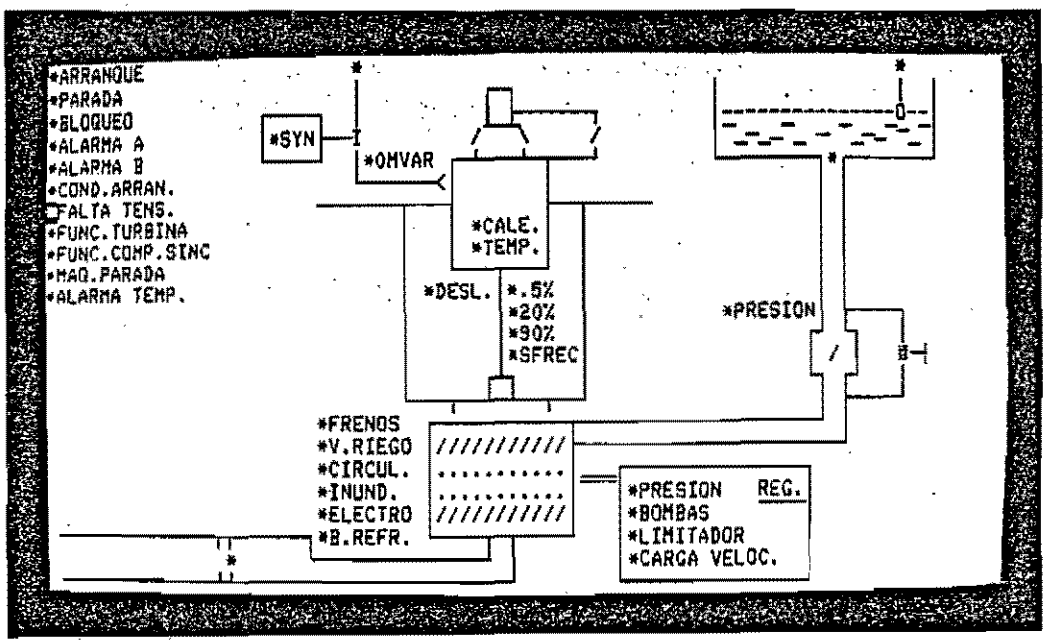

Figure 6. Hydroturbine Screen Mimic.