Öneri, C.2, S.12.
Haziran 1999, ss.151-157.

DISCRETE EVENT SIMULATION MODEL FOR A COMPLICATED SYSTEM

Dr.Nursel S. RÜZGAR[1] - Dr.Bahadtin RÜZGAR[2]

[1] M.Ü. T.E.F. Bilgisayar Eğitimi Bölümü, Öğretim Görevlisi

[2] M.Ü. İ.İ.B.F. İşletme Bölümü, Öğretim Görevlisi

ÖZET

Bu çalışma kesikli olay simülasyon modellemesini içermektedir. Kesikli olayların sistemi temsil edecek simülasyon modelini kurmak, özellikle karmaşık yapıdaki bir sistem için oldukça zordur. Bu nedenle burada karmaşık yapıda olan bir tanker probleminin simülasyon grafiği ve SLAM II veya SLAMSYSTEM şebeke modelinin kurulması detaylı olarak anlatılmıştır. Bu kurulan modeller bilgisayar desteği ile kolayca simüle edilip sistem yapısı hakkında önceden bilgi sahibi olunmasını sağlar. Alınacak sonuçların güvenilir olması, kurulan modelin güvenirliliği yani modelin sistemi ne derece temsil etmesine bağlıdır. Her sisteme uygun ardından kullanılacak bilgisayar program yapısına göre çeşitli sayıda model geliştirilebilir ve hatta alt programlar yazılarak daha verimli hale getirilebilirler.

I. INTRODUCTION

In this paper, we will formally define a graph structure to represent the event scheduling approach. This structure is not only useful for constructing and analyzing discrete event simulation models, but also sufficiently powerful to represent any computational procedure as we know it today.

In the context of discrete event simulations, graphical representation of models play a crucial role. In fact, the process interaction and activity scanning approches have been stimulated by the availibity of graphical techniques for representing system structures. Block diagrams of GPSS or process networks of SLAM II or SLAMSYSTEM have made the interaction simulation models popular. We will start by reviewing the relevant work that has appeared in literature.

II- LITERATURE REVIEW

II.1 Evans (1967)

In discussing the organization of an event scheduling discrete event simulation model, Evan concentrate on four considerations: the units of the model, the event occurences during simulation, the decision made during simulation, and the routines making up the program.

A unit of the model is a particular kind of component in the model. An event occurence leads to the altering of the state of one or more units of the model, and hence of the model itself. Event occurences can interact with one another in two ways. An occurence can cause the scheduling of further event occurences. It can cause the cancellation of event occurences that were previously scheduled. [1]

II.2-Torn (1981)

Torn introduces a graphical technique which incorporates several extentions of Petri nets. A Petri net can be represented as a net graph, PN=(P,T,I,O), where P is the set of places, T is the set of transitions, and I and O are the functions.

Places, which represent conditions, are represented as circles on the graph. Transitions, which represent events or activities, are represented as bars. The conditions necessary for a transition T to occur are connected to T by directed arcs. The dynamics are represented by black dots traversing the graph. A black dot at a place implies that the corresponding condition holds. A transition may occur only when all input conditions are met. The distribution of black dots on a Petri net defines the state of the net and called its marking.

Petri nets are useful in modeling concurrent system. In addition, they can be used to verify certain desirable structural properties of models. [2]

II.3 Schruben (1983)

The elements of a discrete event simulation are state variables that discribe the system, events that change the values of state variables, and the relationships between events. An event graph is a structure of the objects in a discrete event system that facilitates the development of a correct simulation model.

An event graph may be used to guide the developments of event-scheduling simulation program.

Furthermore, an analysis of an event graph can aid in the following modeling tasks;

i) Identifying needed state variables.

ii) Determining a minimal set of events that must be scheduled at model initiation.

iii) Anticipating logic errors due to simultaneously scheduled events.

iv) Eliminating unnecessary event routines. Schruber presents several rules of thumb to resolve these issues. [3,4]

II.4-Pritsker (1986)

A fundamental contribution of Q-GERT, SLAM II and SLAMSYSTEM are their method for graphically modeling systems in a manner that permits direct computer analysis. Q-GERT, SLAM II and SLAMSYSTEM have been developed to provide this computer analysis. Among them only SLAMSYSTEM has animation capability. [5]

Basically, Q-GERT, SLAM II and SLAMSYSTEM support a system approach to problem resolution consisting of four steps. First, a system is decomposed into its significant elements. Second, the elements are analyzed and discribed. Third, the elements are integrated in a network model of the system. Fourth, system performance is accessed through the evaluation of the network model. These programs can be viewed as a simulation language, much like GPSS.

In the following section, a more complicated system, mainly tanker problem, will be represented and event graph and network model of that system will be constructed.[5,6,7]

III-THE EVENT SIMULATION GRAPH MODEL OF A TANKER PROBLEM

A sample study problem from the texts by Schriber, and Law and Kelton is used next to illusrate the construction of an event graph for a more complicated system. This system is presented as a process block diagram in Schriber, as a process network in Pritsker and as a modified Petri net graph in Torn. The system description is adapted from Law and Kelton (problem 2.23) [8,9]

This problem is extracted from Schriber. A port consists of three berths and one tugboat. Tugboats are used to berth tankers so that tankers can be loaded. Tugboats will only begin to berth a tanker if a berth is avaliable. When tanker is loaded, a tug is required to deberth the tanker before the berth can be reemployed. The berthing and deberhing operations one hour of tugboat time. Top periority is given to the berthing activity. The port currently services three types of tankers, each of which requires a different amount of time to load. All loading times are uniformly distributed. 25 percent of arriving tankers require a loading time that is uniformly distributed between 16 and 20. 55 percent of the tankers require between 21 and 27 hours,uniformly distributed (type 2). Type 3 tankers, representing the remaining 20 percent, require between 32 and 40 hours to load, uniformly estimated. The interarrival time between tankers of all three types is between 4 and 18 hours, uniformly distributed. A proposal is being considered that would contract for the port to service five additional tankers that require between 18 and 24 hours to load. After loading and deberthing, they would travel offload the oil, and return to the port for reloading. Their round trip-travel time, including offloading, is estimated to be between 216 and 264 hours, uniformly disributed. [8,9]

A complicating factor is that the port experiences storms. The time between the onset of storms is exponentially distributed with a mean of 48 hours. The duration of a storm is uniformly distributed between 2 and 6 hours. No tug can start an operation until a storm is over.

III.1- Event Graph Of The Problem

This model is taken from Schruben. According to Schruben, events are represented on the graph as vertices. Each vertex is associated with a set of changes to state variables. These variables are used for describing system entities. From this aspect, elements of the problem can be given in the following form. [4]

Parameters of the model;

$T_a(j)$: the distribution for random time between arrivals of type j tanker (known).

$t_l(j)$: the random time required to load a type j tanker (known).

t_s : the time between storm occurences.

t_d : the duration of storm.

$t_l(4)$: loading time.

$t_a(4)$: round-trip time.

The state variables selected for this model;

B : the number of tankers waiting for berthing.

(B>0 : berthing queue is nonempty, B=0 : berthing queue is empty.)

D : the number of tankers waiting for deberthing.

(D>0 : deberthing queue is nonempty, D=0 : deberthing queue is empty.)

P : the number of empty docks at the port, The range of P is the set {0,1,2,3}.

T : the status of the tug. The range of T is the set {-2,-1,0,1}. (–2 : berthing aborted by storm; -1 : deberthing a tanker; 0 : berthing a tanker; 1 : avaliable.)

S : the status of storm. The range of S is the set {0,1}. (0 : storm in progress; 1 : no storm.)

QB, QD : list of tankers requesting berthing and deberthig, respectively.

The edge conditions for the model;

1) S=1, P>0, T=1, i.e., (S*P*T>0),
2) S=1, P>0, B>0, i.e., (S*P*B>0),
3) S=1, D>0, B=0, i.e., (S*D>0, B=0),
4) S=1, T=1, B=0, i.e. , (S*T=1, B=0),
5) S=1, B=0, D>0, i.e., (S*D>0, B=0),
6) S=1, B>0, i.e., (S*B>0),
7) T=0, the tug is berthing a tanker,
8) T=-2, berthing is aborted by a storm,
9) T=1, P>0, B>0, i.e., (T*B*P>0),
10) T=1, D>0, B=0, i.e., (T*D>0, B=0),

EVENT DESCRIPTION			
Event type	Event Description	Parameters	State Changes
AR (j)	Type j tanker arrival	j=k	B=B+1, QB [tail]=j
BB (j)	Begin berthing tanker type j		T=0, B=B-1, k= QB [head]
EB (j)	End berthing tanker type j	j=k	P=P-1, T=1
BL (j)	Begin loading tanker type j	j=k	
EL (j)	End loading tanker type j	j=k	D=D+1, QD [tail]=j
BD (j)	Begin deberthing tanker type j		T=-1, D=D-1, k= QD [head]=j P=P+1
ED (j)	End deberthing tanker type j	j=k	T=1
BS	Begin storming		S=0
ES	End storming		S=1
BS2	Abort berthing tanker		T=-2

The associated graph is presented in Figure 1. In fact, there may be several possible representations of this model as an event graph. Variations of these graphs are used to construct event-scheduling discrete event simulation models.

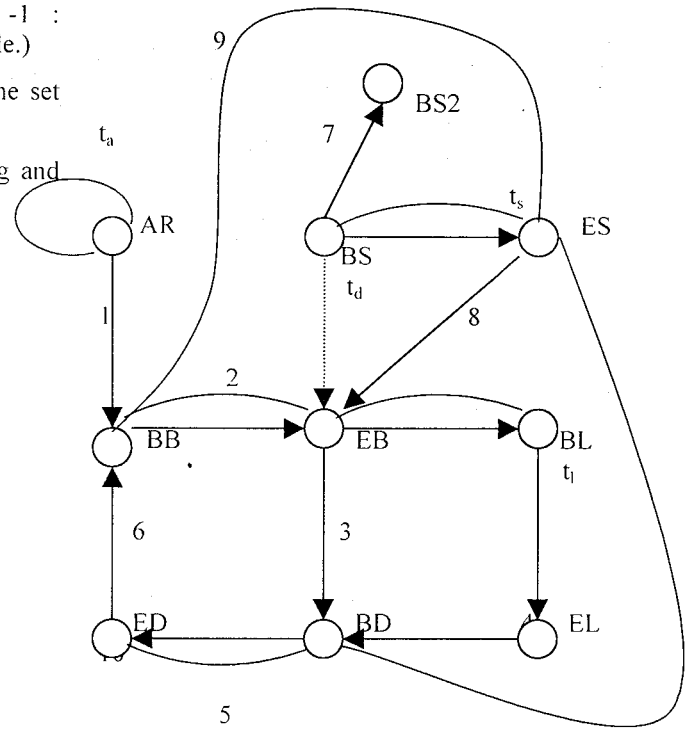

Figure 1. Simulation Graph For The Tanker Problem.

More recently, besides new rules for detecting simultaneously scheduled events and event reduction have been introduced about extention of event graph analysis, rules for identifying simultaneously scheduled events and assigning execution order periorities have been defined and also an extensive algorithm for event reduction have been introduced. In addition , the algorithm can result in an infinite graph when applied to certain subgraphs.[10,11]

The main idea here is that a simulation graph specifies the relationships between the elements of the sets of objects in a simulation model. In a simulation graph model, parameter strings can be passed from one vertex to another through vertex and edge attributes. These lists are useful in scheduling or cancelling specific instances of system events. A vertex attribute list is a string of state variables associated with a particular vertex. An edge attribute list is a string of expressions associated with a particular edge. When the origination vertex of an edge is executed, the expressions in the edge attribute list are evaluated. When the destination vertex is subsequently executed, the state variables in its attribute list take on the values that had been computed for the expressions in the scheduling edges attribute list. At this point, we will use tanker problem for a concrete illustration of the sets in a simulation model. The following objects make up the model.

V (ϑ) ={$\nu_1,\nu_2,\nu_3,...,\nu_{10}$}={AR, BB, EB, BL, EL, BD, ED, BS, ES, BS2} (vertex of set ϑ)

εs(ϑ)={$e_1,e_2,e_3,...,e_{17}$}={(AR,AR),(AR,BB),(BB,EB), (EB,BB),(EB,BL),(EB,BD),(BL,EL),(EL,BD),(BD,ED), (ED,BD),(ED,BB),(BS,ES),(ES,BS),(BS,BS2),(ES,BB), (ES,EB),(ES,BD)} (set of scheduling edges of ϑ)

εc (ϑ) ={e_{14}}={(BS,EB)}(set of cancelling edges of ϑ

ψ ϑ : the incidence function.

ℑ : the set of transition functions associated with (event) vertex ν.

f_{AR} ={j=k, B=B+1, QB [tail]=j },

f_{BB} ={ T=0, B=B-1, k= QB [head]}

f_{EB} ={j=k, P=P-1, T=1}

f_{BL} ={j=k}

f_{EL} ={j=k, D=D+1, QD [tail]=j }

f_{BD} ={T=-1, D=D-1, P=P+1, k= QD [head]}

f_{ED} ={j=k, T=1}

f_{BS}={S=0}

f_{ES} ={S=1}

f_{BS2} ={T=-2}

C : the set of edge conditions;

$C_{AR,BB}$={ S*P*T>0}

$C_{EB,BB}$={ S*P*B>0}

$C_{EB,BD}$={ S*D>0, B=0}

$C_{EL,BD}$={S*T=1, B=0}

$C_{ED,BD}$={ S*D>0, B=0}

$C_{ED,BB}$={S*B>0}

$C_{BS,BS2}$={T=0}

$C_{ES,EB}$ ={T=-2}

$C_{ES,BB}$={T*B*P>0}

$C_{ES,BD}$={T*D>0, B=0}

ℑ : the set of edge delay times;

$t_{AR,AR}=t_a$

$t_{BB,EB}=1$

$t_{BL,EL}=t_l$

$t_{BD,ED}=1$

$t_{BS,ES}=t_d$

$t_{ES,BS}=t_s$

$t_{ES,EB}=1$

Γ={γ_{AR}, γ_{BB}, γ_{EB}, γ_{BL}, γ_{EL} ,γ_{BD}, γ_{ED}, γ_{BS}, γ_{ES}, γ_{BS2}}=

{2,1,2,2,2,2,2,1,2,1}

Execution of the simulation graph model can be given as the following form by using the presented algorithm above. This operation requires event scheduling function which maintains two crucial variables : τ, the global simulation clock, and Z the list of scheduled events. The event list is an ordered set of triples. That is, Z={(t_1, γ_1, ν_1), (t_2, γ_2, ν_2),....} where ti, γi and νi represent the event execution time, the event execution periority, and the associated event vertex, respectively. For the execution of a simulation graph model, steps for initialization and execution are to be followed. The execution of a simulation graph model is carried out in the following manner. (the symbol := denotes an assigment) [10,11]

Initialization:

Step1 : Initialize global simulation clock, τ:=0.

Step2 : Insert the first event record into the event list, Z := Z∪{(0, γ_0, ν_0)}.

Execution:

Step1 : Remove the fist event record from Z, Z := Z/{(t_i, γ_i,ν_i)}.

Step2 : Update the simulation clock, τ:= t_i.

Step3 : Assign the values of the state variables in the vertex attribute list, Aνi, if the list is not empty.

Step4 : Evaluate the state variables in $S\nu_i$, $S\nu_i = f\nu_i (S\nu_i)$.

Step5 : Schedule and/or cancel further events: for all edges emanating from vertex νi : if $C\nu_i\nu_j$ ($E\nu_i$) =1, then compute values of expressions in the corresponding edge attribute list, generate the inter-event time t_j and Z := Z∪{(j+ t_j, γ_j,ν_j)}.

Step6 : Terminate the execution of the simulation if any of the following situation is reached;

a) $\tau \geq T_{STOP}$.

b) (t_{end}, γ_{end},ν_{end}) has just been executed. Otherwise, go to Step1 of execute. Here, T_{STOP} represents

a pre-determined stopping time for the simulation and v_{end} represents an end-of-simulation event.

Execution of simulation graph model is presented above. In the following section , the SLAM II or SLAMSYSTEM network model of the same system will be represeted.

IV. SLAM II NETWORK MODEL OF TANKER PROBLEM

SLAM II is an advanced simulation language with both FORTRAN and C versions that allows models to be build based on three different world views. It provides network symbols for building graphical models that can be automatically translated into input statements for direct computer processing. It contains subprograms that support both discrete event and continuous model developments.

The network model of this eample is illustrated in Figure 2. The explanation of the model will be given in terms of network model. [7]

IV.1. Model Description

The RESOURCE block shows that resource type 2 is for the TUG, and resource type 1 defines the BERTH. The resource BERTH is assigned a capacity of 3 and entities waiting for a BERTH reside in file 1. The reseource TUG, however, has capacity of 1. Entities waiting for the TUG reside in either file 2 or 3. The network model can be divided into three major segments; such as tanker arrival segment, port operation segment and storm segment.

The arrival process for this problem is composed of two classes of arrivals. The fist arrival class represents the existing tanker traffic consisting of tanker types 1, 2, and 3. These entities are generated by CREATE node and are routed probabilistically by the three emanating ACTIVITY's to either ARV1, ARV2, or ARV3 ASSIGN nodes.At these nodes, ATRIB(1), and ATRIB(2) are set equal to the appropriate loading time and the appropriate tanker type, respectively. Following any of these ASSIGN nodes, the entity is routed to another ASSIGN node labeled PORT. The second arrival class involves inserting five entities representing the proposed type 4 tankers into the network. The entities are created by the CREATE node which generates an entity every 48 time units with the first entity at time 0, and a maximum of five entities created. At the ASSIGN node labeled ARV4, ATRIB(1) and ATRIB(2) are set equal to the loading time and the tanker type, respectively. The entities then are routed to the PORT ASSIGN node.

The second major segment in the model represents the port operations begins with ASSIGN node labeled PORT. This node records the time of arrival to the port as ATRIB(3) of the entity. Entities then arrive to the AWAIT node when no BERTH is avaliable reside in file 1. When a BERTH is avaliable, the entity is routed to the next AWAIT node where it waits in file 2 for the TUG. The ACTIVITY following this AWAIT node represents the berthing operation and has duration of one hour. Following berthing, the entities arrive at a FREE node which frees one unit of the resource TUG. The ACTIVITY following this FREE node represents the tanker loading activity. At the next AWAIT node entity waits in file 3 for a TUG. Since file 3 is listed after file 2 in the RESOURCE block for the TUG, the TUG will be allocated to the deberthing operation. When a TUG is finished deberthing, the BERTH and the TUG are freed. After freeing the TUG, the tanker entity is conditionally branched based on tanker type by four ACTIVITY's to the appropriate departure COLCT (COLLECT) node where interval statistics on port residence time are recorded. The entities corresponding to the existing tanker traffic of types 1, 2, and 3 are terminated. The round trip travel time for tankers of type 4 is represented by the ACTIVITY which routes the entity back to the ARV4 ASSIGN node. Therefore, the five type 4 tankers continue to cycle through the model until the simulation is terminated after 8640 hours (one year of time) of operations.

The last segment of the model is the storm segment which starts with the creation of a storm entity at CREATE node. The first storm is delayed by an exponentially distributed time with a mean of 48 by an ACTIVITY. The entity continuous to the STORM node where the TUG resource is requested to be altered by 1 unit. If the tug is not in use or at the end of the tug's current operation this decrease in capacity will occur immediately. The storm duration is uniformly distributed between 2 and 6. Following the storm, the TUG resource capacity is increased by 1 at an ALTER node. The next storm is scheduled by an ACTIVITY and the storm entity is routed back to node STORM. This completes the description of the model.

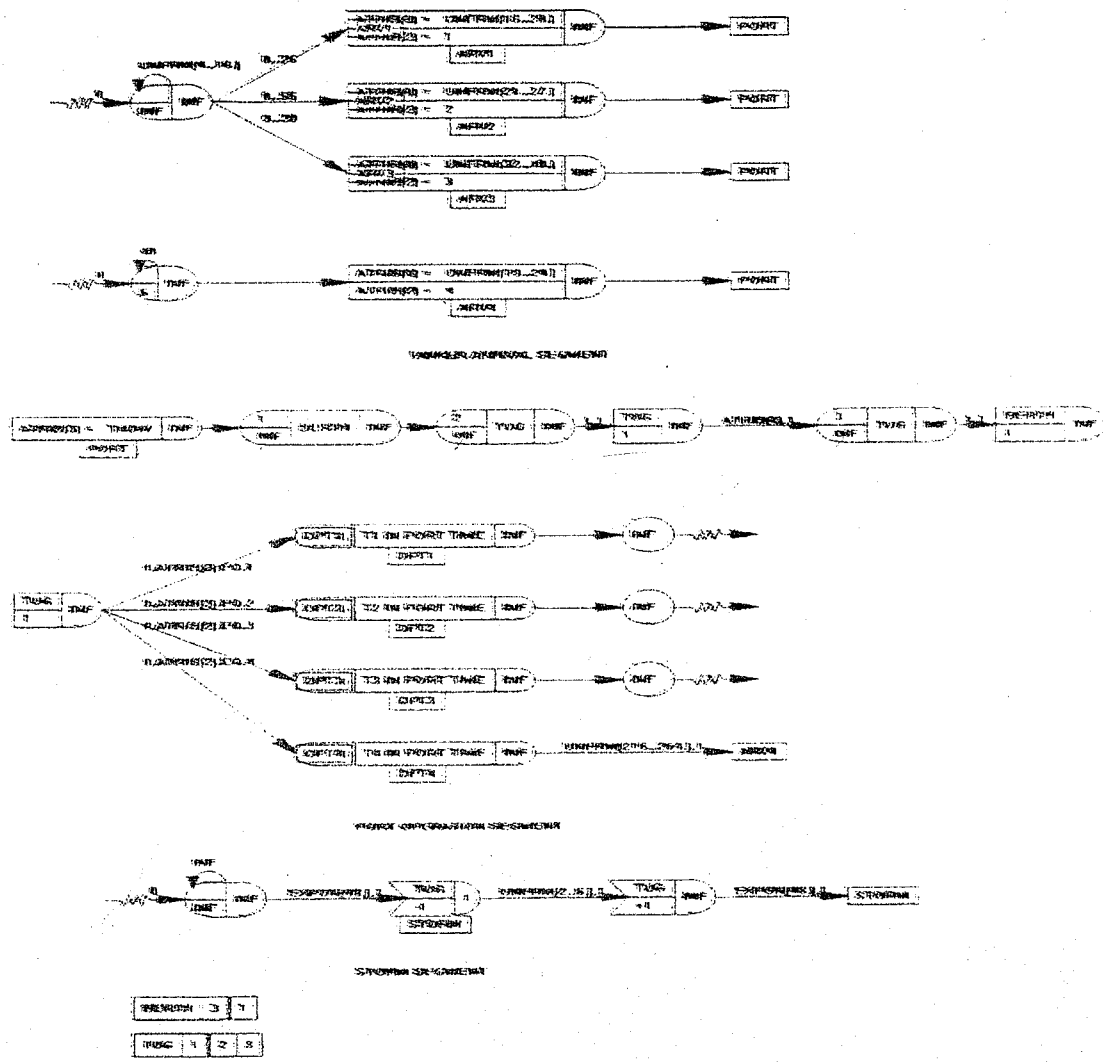

Figure 2. SLAM II Network Model Of Tanker Problem.

Up till now, we presented two types of model graphics for a complicated discrete event system. In fact, there are other representations but they leave to reader's interest for further researh.

In this paper, we propose graph theory as an effective base for representing discrete event simulations. This is not suprising, since "any system or structure which may be considered abstractly as a set of elemens, certain pair of which are related in a specified way, has a representation as a graph or digraph. Thus, graph theory is

really a theory of relations, with graphs representing symmetric relations and digraphs asymmetric relations." [12]

Recently a computerizable system that is capable of providing a useful environment for working with any graph based model. Implementen in Prolog, graph based modeling system uses directed graph with attributed nodes and edges in addition to certain structural consraints for formulating different classes of problems. The appealing aspect of graph based modeling system is that it provides tools, not only for representing graph based models, but also for characterizing a solution procedure to analyze them.

"Simulation graph models can be implemented by using a high level programming language or general purpose simulation modeling language by possibly coding subgraphs into separate procedures." [10] Alternatively, these graph models can be directly implemented using Σ. [13] Σ is an interactive graphics program specifically designed to build, tet, and experiment with discrete event dynamical systems on personal computers using simulation graphs. Σ is intended to faciliate model implementation by allowing the user to construct executable models by drawing their graphs with a mouse. These graphs can be executed interpretively for debugging a model or translated into ANSI standard C for compilation. SLAM II or SLAMSYSTEM, however, provides network symbols for building graphical models that can be automatically translated into input statements for direct computer processing.

V-CONCLUSION

Model formulation is the abstraction of the system into mathematical/logical relationships which are relavant within the scope of the study and consistent with our problem solving objectives. "The actual process of formulating a model is one which is largely an art. The modeller must understand the structure and operating rules of the system, and be able to extract the essence of the system without including unnecessary detail. The crucial decisions concern what simplifying assumptions are valid, what elements should be included in the model, and what interactions occur between the elements."[14]

KAYNAKLAR

[1]Evans, G.W.; Wallace, G. F.; Sutherland, G. L.."Simulation Using Digital Computers", Prentice Hall, Inc., New Jersey, 1967.

[2]Forn, A. A., "Simulation Graphs: A General Tool For Modeling Simulation Designs", Simulation, Vol.37 (December), s. 187-194, 1981.

[3] Schruben, L. W.,"Simulation Modeling with Event Graphs", Conm. ACM, Vol. 29.11, s. 957-963, 1983.

[4]Schruben, L. W.,"Simulation Modeling with Event Graphs", Technical Report, no.498, School of Operation Research and Industrial Engineering, Cornell University, Ithaca, N.Y., 1982.

[5]Pritsker, A. A. B., "Modeling and Analysis Using Q-GERT Networks", John Wiley and Sons, New York, 1979.

[6]Pritsker, A. A. B.; C. D. Pedgen, "Introduction to Simulation and SLAM", Halsted Press, New York, 1979.

[7]Pritsker, A. A. B.,"Introduction to Simulation and SLAM II", Halsted Press, New York, 1986.

[8]Schriber, T. J., " Simulation Using GPSS ", John Wiley and Sons, New York, 1974.

[9]Law, M. Averill; Kelton W. David, "Simulation Modeling and Analysis", Mc Graw Hill Book Company, Inc., New York, 1991.

[10]Som, T. K.; Sargent, R. G.,"A Formal Development of Event Graphs as an Aid to Structured on Efficient Simulatuon Programs", ORSA Journal and Computing, Vol. 1.2, s. 107-125, 1989.

[11]Sargent, R. G.,"Event Graph Modeling For Simulation with an Application to Flexible Manufactoring Systems", System Management Science, Vol. 34.10, s. 1231-1251.1988.

[12]Lawler, E." Combinatorial Optimization", Networks and Matroids Holt, Rinehart and Winston, N.Y.,1976.

[13]Schruben, L. W.; Briskman, D.,"Teaching Simulation with SIGMA ", Proceeding of the 1988 Winter Simulation Conference (Abrahms, Haigh, Comfort, Edts.), San Diego, CA, s. 869-874, 1988.

[14]Pritsker, A. A. B.; Standridge, C. R.," The Extended Simulation Support System", John Wiley and Sons, New York, 1987.