NETWORK MODELLING FOR SYSTEM DYNAMICS

J. Talavage, Purdue University

Joseph Talavage is a Professor of Engineering at Purdue University in West Lafayette, Indiana, Since receiving his PhD in systems engineering from Case Institute of Technology. in 1968, he has been a faculty member at Georgia Tech and then at Purdue. During this time, he has done research and taught courses in the areas of simulation and general systems theory. His current research activities include hierarchical production control system design and automated-manufacturing system simulation.

ABSTRACT
DYNANET is a network modeling language which lays the foundation for interactive graphical-input specification of System Dynamic models. DYNANET is written in FORTRAN to make it portable and to be compatible with other modeling techniques. The special symbols which a DYNANET user would employ to represent System Dynamics concepts are defined. An example is given which illustrates each of the small set of special symbols.

1. INTRODUCTION

The promise for widespread use of simulation modeling is enhanced by the recent developments in interactive computing and computer graphics. Though most of these developments have occurred within the context of mini- and microcomputers, it is expected that mainframe computers will also have such capabilities.

Interactive, graphical-input simulation modeling systems have been in use for several years[1,2,3]. With each of these systems, a simulation modeler defines the model in an interactive manner directly on the computer terminal screen (though it is likely that the overall model design was developed prior to the terminal session). The model itself is expressed in terms of interconnected graphical symbols (each symbol representing a unique simulation concept) selected by the modeler from a menu of such symbols. The selection of a given symbol initiates a computer-conducted "interview" in which the modeler specifies pertinent parameters associated with the model symbol. Some error-checking may be performed during this interaction process, thus providing further aids to the modeler during model development.

This paper describes the development of a FORTRAN program called DYNANET which lays the foundation for an interactive, graphical-input System Dynamics simulation modeling system. Being based on the GASP IV simulation language, DYNANET provides the framework for extending System Dynamics models to include state-initiated event processing and interaction between continuous-time processes and discrete events.

2. SOME ELEMENTS OF SYSTEM DYNAMICS

In postulating the model structure in System Dynamics, the modeler conceptualises the casual relationships between the major elements of the model. The basic building block used in establishing conceptual relationships is the feedback loop[4]. One way of portraying a collection of feedback loops is through a flow diagram[4].

The flow diagram is a pictorial representation of an interconnection of feedback loops which portrays the factors in a model and the information flow between those factors. The substructure of each feedback loop is portrayed in detail in the flow diagram. This substructure consists of:

Levels	– Variables which describe the condition or state of the system at a particular time.
Rates	– Variables which describe how fast the levels are changing
Auxiliaries	– Variables which provide information links between rates or levels
Delays	– Variables which cause rates or information to be delayed a specified amount of time
Constants (Modifiers)	– Variables whose value will not change no matter what the condition of the system

Once a flow diagram has been constructed, the modeler has a device available which he can use to communicate his model, and as a stepping stone to simulating his model's performance. With a flow diagram, postulating changes in the model structure is also much easier.

3. EXAMPLE APPLICATION

To display the use of DYNANET, a simplified model of an apartment construction system is first described[5]. The model is then presented in flow diagram form. After describing DYNANET, the flow diagram can be converted to a DYNANET flow diagram. The complete DYNANET program is shown and the results for one run of the simulation are presented.

Consider an apartment construction system which can be characterised by the current level (number) of apartment rooms available. This level is changed by the rate at which apartments are constructed and older apartments are destroyed.

New apartment construction is seasonal, reaching a peak at the 26th and 78th weeks of a two-year cycle, as shown below;

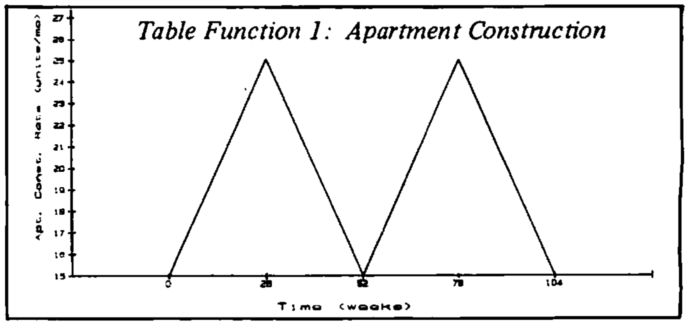
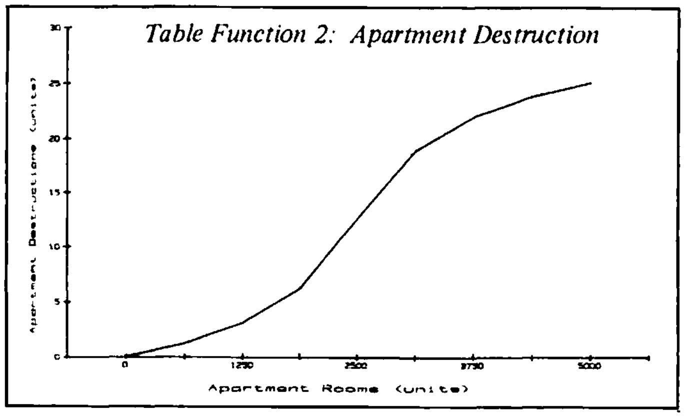


Table Function 1: Apartment Construction

Because construction takes time, the rate at which new apartments become available is delayed 25 weeks.

Apartments are destroyed to make room for construction of new apartments because they are assumed to be old and deteriorated. For purposes of this example it is assumed that the rate at which apartments are destroyed is a function of the current number of apartments, and of the difference between the apartment construction rate and apartment finishing rate. The difference accounts for a work crew being released from a finished job and is doubled to account for the fact that the potential number of apartments destroyed in a given time is at least twice the number that can be constructed or finished by the same number of workers. The table function showing this relationship is;



Given a starting level of 2500 apartments, we wish to observe the dynamic behaviour of the system for a period of 104 weeks. The flow diagram for this model is shown in Figure 1.

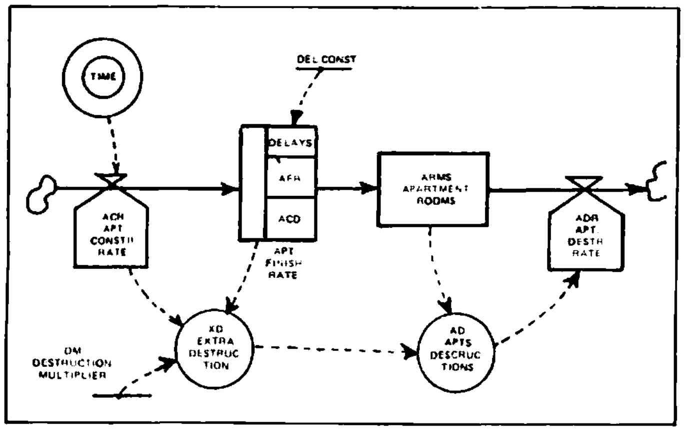


Figure 1: DYNAMO Flow Diagram – Apartment Construction Model.

After introducing DYNANET, we will use it to model this system.

4. THE DYNANET APPROACH TO MODELING

The Systems Dynamics Network Language, DYNANET, is a collection of FORTRAN subroutines specifically designed as a simulator for Systems Dynamics models. The use DYNANET, one ultimately will need only to develop the flow diagram for the system of interest. This diagram is expressed in terms of the five basic substructure variables described above. In particular, each of these variables is represented graphically by a symbol called a node. The description of each variable or node type is as follows:

4.1 Level Node

Levels can be considered the "state" variables of the Systems Dynamics models, and are the basic building blocks of the Systems Dynamics network. A level is used to describe the state or condition of the system at any given time.

The 'LEVEL' node symbol is:

Level	NODUM	NLBL	Describes the state or condition of the system at any given time
	STATS	XIL	

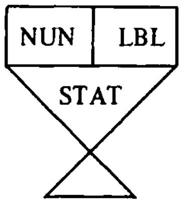
where,

NLBL	–	Label or variable name assigned to the level
NODNUM	–	Node number
STAT	–	Output statistics desired for the level (See Appendix A)
XIL	–	Initial level at the start of the simulation

4.2 Rate Node

Rates are used to change levels. They can cause a level to increase or decrease and are the only means of directly modifying a level. The rate specifies the amount of change to a level in a given time period, and depends on other variables in the system. As a result, no initial rate need be specified. At least one rate must be specified for every level in the system. Rates are changed or modified by feedback from their associated level, (nodes) and/or by delays and modifiers.

The symbol for the "RATE" node is:

Rate 

Specifies the amount of change to a level in an given time period

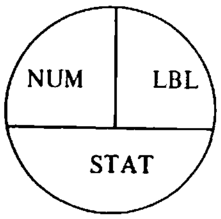
where,

LBL	–	The rate node variable name or label
NUM	–	The rate node number
STAT	–	Desired output statistics on the rate (See Appendix A)

4.3 Auxiliary Node

An auxiliary node is used to describe the "intermediate" variables of the system. Normally, any number of auxiliary variables may be interrelated and an auxiliary may be affected (or modified) by one or more levels and/or other node types, but not by a rate. Although an auxiliary can be modified by a level (i.e., there can be a direct path from the level to the auxiliary), the only way an auxiliary can influence a level is through a rate.

The "AUXILIARY" node symbol is:

Auxiliary 

Links rates or levels and other rates or levels where,

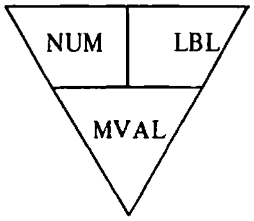
LBL –	The auxiliary node variable name or label
NUM –	The node number
STAT–	Output statistics desired on the auxiliary (See Appendix A)

4.4 Modifier Node

Modifiers are used to indicate constant terms or relationships in the model. As a result, the value represented by the modifier node will not change during the simulation, and hence there are no inputs to a modifier node. However, a modifier may be used to input time-dependent behaviour into the model by specifying a variable name rather than a node value. This is illustrated in the example which follows later. Modifier nodes output only to rates and/or auxiliaries.

The "MODIFIER" node symbol is:

Modifier



Indicates constant terms or input time-dependent behaviour

where,

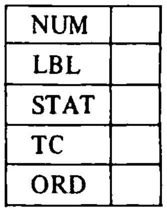
LBL –	The label or variable name associated with the modifier
NUM –	The modifier node number
MVAL –	The numerical value of the modifier or a variable name

4.5 Delay Node

A delay is a special combination of a rate and level. As the name implies, a "delay" in the system causes a delayed response in changes to a level. That is, the changes to a "rate", in response to a new value for a level, are delayed for a specified amount of time. Delays are normally expressed by their "order" (see Goodman)[5], with a third order delay being the most widely used. The average length of a delay time period is given by the delay time constant.

A "DELAY" node is:

Delay



Causes a delayed response in changes to a level

where,

NUM –	The node number
LBL –	The label or variable name associated with the delay
ORD –	The "order" of the delay
TC –	The time constant
STAT –	Output statistics desired on the delay

4.6 Connectors

Nodes are joined to each other, or connected, by connector lines or branches. The branches establish the same casual relationships among the nodes as they do for the original flow diagram.

5. DYNANET MODELING FOR THE EXAMPLE

DYNANET uses the same symbol set as DYNAMO, with additional information about the model included in each symbol. Figure 2 shows a DYNANET flow diagram for the example model.

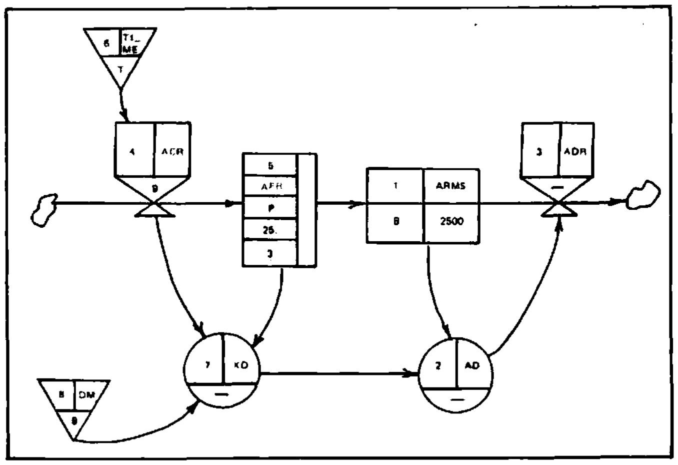


Figure 2: DYNANET Flow Diagram – Apartment Construction Model.

Converting the original flow diagram to a DYNANET flow diagram involves only the addition of a node number (optional), an optional indicator of the type of statistics desired, and for a level node, the initial value of the level. Thus, node 1 is a level node for the state variable ARMS. The initial value of ARMS is shown as 2500. Both a plot and statistics table for ARMS are desired as part of the output. Rate and auxiliary node symbol conversion is also direct, except an initial value is not assigned to rates or auxiliaries. The rate node for ACR is node 4 and a plot of ACR is desired. Rate node 3 for ADR shows that no output for ADR is wanted. The auxiliary nodes 7 and 2 for XD and AD, respectively, also show that no output is to be given for the auxiliary variables. The translation of the delay node into DYNANET involves only a slight modification of the node interior to accommodate information about the delay. In DYNAMO, the delay constant is normally portrayed as an external constant variable that has an information link to the delay. In DYNANET, the delay constant has been included in the node symbol. Also included in the delay node are the node number and optional statistic indicator.

Modifier nodes in DYNANET serve two purposes. The first is to represent the constants of the model. Node 8 is an example of a modifier used to symbolize the constant DM. Note also that, in addition to the optional node number, the value of the constant is also specified in the node.

The second function of the modifier node in DYNANET is to induce a time-dependent value in the system. Notice that in the DYNAMO flow diagram, the "exogenous" input node 6, indicates that the construction rate is dependent on some exogenous variable. In DYNANET, the modifier node is used to show that ACR depends on the current simulation time. In this case, instead of assigning a numeric value to the modifier, any alphabetic character or label is used.

A complete interactive, graphical-input procedure for DYNANET is not yet available. Therefore, preparing the model for simulation in DYNANET currently involves writing one data statement to describe each node. The information used in preparing the statement is taken directly from the DYNANET flow diagram, since, except for table functions, it completely describes the model. (In any input statement, commas are used to separate fields of information, and a semi-colon to denote the end of the statement).

The complete listing of the DYNANET statements for this model is in Figure 3. In an interactive graphics modeling system, nearly all of these statements would be automatically generated by the input processing program. In addition to the model statements, two statements are also prepared to describe each table function. The TBL statement specifies a table number, the independent variable, the dependent variable, the minimum and maximum value of the independent variable, and its increment. The DAT statement specifies, for a given table (number), the value of dependent variable at each increment value of the independent variable. In any model statement that expresses a relationship between variables and a table function, the name of the table function is preceded by an ampersand (see the statements for the rate ACR and the auxiliary AD in the listing of Figure 3.

Along with model and table statements, control statements are used to specify the model author and date (via GEN statement), title for the output (via the TTL statement) and simulation run control (via the RUN statement). The RUN statement specifies start time, end time, and time increment for the simulation.

In calculating the value of algebraic expressions, DYNANET performs the algebraic operations as they are read. To illustrate this, notice the XD could be determined by

$$XD = (ACR - AFR) * DM$$

or by

$$XD = ACR - (AFR * DM)$$

which are not equal in general. In this example, the first relationship is the appropriate one. The output for this model is discussed in the next session.

5.1 DYNANET Output

The DYNANET modeler is provided with the standard graphical plots of dynamic behaviour of the model versus time. With DYNANET, up to two sets of plotted outputs are allowed, each with a single independent variable (generally simulation time). Up to nine dependent variables may be plotted on each output, with a user-specified symbol denoting each variable plotted. DYNANET provides the option of adjusting the scale of plots to obtain "pleasing" results. Also, the modeler may request a table of the plotted variables in addition to, or in place of, the graphical plot. The flexibility is provided by use of plot control statements which must be specified

```
GEN, TALAVAGE, APARTMENT EXAMPLE, 15,
MARCH 84;
TTL, APT CONSTRUCTION EXAMPLE;
RUN, 0, 104, 1;
MOD, 6, TIME, T;
TBL, 1, TIME, ACR, 0, 104, 26;
DAT, 1,15,25,15,25,15;
RAT, 4, ACR, & TIME, P;
AUX, 7, XD, = ACR - AFR * DM;
MOD, 8, DM, 2.;
DLY, 5, AFR, ACR, P, 25., 3;
LEV, 1, ARMS, + AFR - ADR, B, 2500;
RAT, 3, ADR, = AD;
AUX, 2, AD, & ARMS + XD;
PLO, 1, TNOW, 3,2,1;
VAR, 1,2,C, ACR, 1,1,15, 35;
VAR, 1,3,F,AFR,1,1,5,25;
FIN;
```

Figure 3: DYNANET Equations (would be automatically generated in interactive graphical-input modeling procedure).

if plotted output is desired. In the above example, the PLO and two DAT statements specify that a plot of ACR and AFR is desired, with the plot symbol for ACR being C, and F for AFR. The scale of the plot for ACR is to be between the values of 15 and 35, while the scale for AFR is to be between 5 and 25.

DYNANET produces statistical output as well as plot/tables. The user may indicate on all nodes except modifier nodes the type of statistics he desires, as indicated in the input card description. The necessary statistical array initialising is done in the DYNANET executive routine.

6. SOME DETAILS OF DYNANET OPERATION

DYNANET is a collection of FORTRAN subroutines which interpret and operate on user-provided data about Systems Dynamics models. More specifically, DYNANET performs the following functions:

1. Initialisation – Read and interpret data input statements (future modeling systems would accomplish this via interactive graphics computing)

   Check input data for errors

   Initialise data storage arrays based on data input

2. Organisation – Associate terms of the system equations with the proper nodes

   Order the system equations into the proper computational sequence

   Associated data arrays for delays, table functions and statistics with nodes and equations

3. Execution – Execute step-by-step simulation of the model with the prescribed time interval
4. Statistics Collection – Store desired statistical information
5. Reporting of Results – Print requested input echo
   Statistical output
   Check and statistical output

As the input data is read, DYNANET performs error checks and prints error messages if necessary. While performing this check DYNANET orders the equations into the proper computational sequence.

Prior to executing the simulation, if it has been requested, DYNANET will print an Echo Check of the input data. The Echo Check is formated to be easily read and can be used to check input data, the model structure, and simulation parameters.

DYNANET then executes the step-by-step simulation with a fixed time increment. After incrementing the simulation time, it computes values for each equation in the proper order. Upon completion of one computational sequence, values are collected and stored for variables on which statistics are collected. If, during the simulation, DYNANET detects a mathematical or execution error, it will terminate the simulation at that point and print an error message.

At the end of the simulation time, DYNANET terminates the simulation and prints a summary report, if requested. The summary report includes a table of statistical values on desired variables and a plot and/or table of values for those variables for which they were requested.

7. CONCLUSION

DYNANET should prove an invaluable aid as a Systems Dynamics model simulation tool, especially when interactive, graphical-input procedures are completed development. Being written in FORTRAN, and of relatively small size, DYNANET is portable and can be implemented onto almost any machine containing a FORTRAN compiler[6]. The simplicity of the programming requirements makes it easy to learn for anyone familiar with the approach of Systems Dynamics. Because of this simplicity, DYNANET also should prove an excellent classroom aid in teaching.

REFERENCES

1. WILLIS, R. and ANSTELL, W., "GMSS-Graphic Modeling and Simulation Systems," *Proc. of 1983 Simulation Symposium,* Tampa.
2. YANCEY, D., HOWELL, E., and MAYER, R., "Integrated Decision Support for Aerospace Manufacturing," TIMS/ORSA Conference, San Diego, 1982.
3. PEGDEN, C.D., *Introduction to SIMAN,* Systems Modeling Corporation, 1982.
4. FORRESTER, J., *Principles of Systems,* Wright-Allen Press, 1968.
5. GOODMAN, M., *Study Notes in System Dynamics,* Wright-Allen Press, 1974.
6. MATWICZAK, K. and TALAVAGE, J., "DYNANET – Network Modeling for System Dynamics", *Proc. of Summer Comp. Sim. Conf.,* Seattle, 1980.

APPENDIX A

Card Type	Field Number	Max Length of Field	Definition	Default
Level Node Card:				
LEV	1	–		
	2	4	Node number	(Blank)
	3	4	Node label	(Blank)
	4	–	Signed input and output rate nodes which affect the level node	Must specify
	5	–	Statistics desired on this level 0 = none P = Plot only S = Time persistent statistics only B = Both plot and "S" statistics A = Time persistent statistics based on average value in time interval	0
	6	–	Initial value of level	0
Rate Node Card:				
RAT	1	–		
	2	4	Node number	(Blank)
	3	4	Node label	(Blank)
	4	–	Equation which determines rate value (i.e., inputs to rate node, listed in the order in which they are to be included in the calculations, as noted in text).	Must specify
	5	–	Statistics indicator (same as field 5 of Level Card)	
Auxiliary Node Card:				
AUX	1			
	2-5	–	Same as Rate Node Card	
Modifier Node Card:				
MOD	1			
	2	4	Node number	(Blank)
	3	4	Node label	(Blank)
	4	–	Value of the time constant	0
	or			
	(4)	4	Alphabetical Table. (This sets the value of the modifier to the current simulation time).	
Delay Node Card:				
DLY	1			
	2	4	Node number	(Blank)
	3	4	Node label	(Blank)
	4	4	Label or number of node which is input to the delay	0
	5	–	Statistics indicators (same as for other node cards)	
	6	–	Delay constant	0
	7	–	Order of the delay	0