Proceedings of the 1988 Winter Simulation Conference
M. Abrams, P. Haigh, and J. Comfort (eds.)

Computer Aided Petri Net Design for decision-making organizations

I. M. Kyratzoglou
Sensors and Surveillance Systems Division
Bolt, Beranek and Newman Systems and Technologies
70 Fawcett Street
Cambridge, MA. 02238

ABSTRACT

A Computer Aided Petri Net Design System is developed for creating decision-making organizational architectures of arbitrary complexity and for computing the structural attributes describing the organization. The Design System has four modes of operation: A Graphics Editor mode, used for the interactive generation of the Petri Net structure of the organization; A Text Editor mode, used to store, modify and retrieve attributes assigned to the Petri Net primitives; A Structural Analysis mode, used to generate the structural properties of the organization, such as the incidence matrix, the interconnection matrix and the information flow paths: and a Hardcopy mode, used to generate graphic images on output devices. In addition a set of auxiliary functions handle the file manipulation, user-software interface, command interpretation and grammar-rule enforcing tasks of the System.

INTRODUCTION

In the context of decision-making organizations a decision-maker is a trained person who can administer a decision by considering the available information. A decision-making organization (DMO) is formed when a single decision-maker does not have the capacity or proficiency to administer a decision. A DMO is a team of trained decision-makers who are coordinated for information processing and sharing, and who are organized according to their skills and authority. Each decision-making process may be partitioned into subprocesses. Each subprocess may be processed by a team member. A distributed decision-making organization (DDMO) is formed when each team member reaches a decentralized decision in his own area of expertise. Decision aids and databases may be introduced to improve the effectiveness of the organization.

The Petri Net representation has been used (Tabak and Levis, 1985) to model DMOs. A model of Timed Petri Nets was introduced in which processing times were assumed deterministic (Ramchandani, 1974). The Timed Petri Nets were used for modeling time critical DMOs (Hillion, 1986). This method of representation is sufficient to capture time delays, information flow and processing, resource contraints, precedence relationships and the sequence of events, interactions between organization members, protocols that govern these interactions and to compute measures of performance and effectiveness. The Timed Petri Net formulation has been used (Grevet, 1988) to implement the discrete event simulation of decision-making processes that require coordination. The execution of a net is carried out through the activation of events that take place at discrete instants in time. A methodology was developed (Andreadakis, 1988) to design distributed DMOs and to analyze their performance and effectiveness given a mission.

A design approach to model, analyze and evaluate a DMO is to create a generic structure and then develop organizational architectures from it. Operational specifications may suggest alternative configurations. This method enables the designer to consider many configurations of similar generic structures. It also enables the classification of each model according to specific structural, data flow or organizational properties, performance, etc.

The Computer Aided Petri Net Design System was developed (Kyratzoglou, 1987) to provide fast and interactive design of DMO architectures. The System provides a set of Petri Net representation tools which assist the designer to generate DMO models, and a set of analytical tools that assign attributes to Petri Net elements and compute structural properties of the Net. It allows the designer to structure the design process. The System may be used as the front end for simulation.

This paper reviews the basic concepts and definition of Petri Net Theory. Then, it describes the operational modes of the Petri Net Design System and their basic features and capabilities. The conceptual phases undertaken in developing a Petri Net design are also described. The application of the conceptual approach for the generation of the Petri Net representation of an organization is presented, in which two 3-person DMOs (one parallel and one hierarchical) are modeled.

THEORETICAL CONCEPTS

Petri Nets are bipartite directed multigraphs (Peterson, 1981). They consist of two types of nodes: places indicated by circles and representing signals or conditions; and transitions indicated by bars

and representing activities or processes. Places must be connected to transitions only, and transitions must be connected to places by a connector. A connector indicated by an arrow represents the directed relationship between the place and the corresponding transition. Each node or connector may have specific inscriptions associated with it related to the physical component it represents. For example, a transition may represent a set of processing algorithms modeling a specific organizational activity. For an activity to take place certain conditions must hold true. The places may represent that set of conditions. Conditions are evaluated according to the incoming information. The information carrier is represented symbolically by a token and is deposited in places. A transition is enabled and it can fire, if all places input to the transition are marked by tokens. Firing times are assigned to each of the transitions of the net.

A variant of the transition, the switch, indicated by a rectangle with rounded corners is used to represent a decision structure (Tabak and Levis, 1985). The switch models a set of decision rules and tests according to a control law based on the transmitted information, see Figure 1. A decision strategy is chosen based on the decision rule selected. Once a decision strategy is selected a path is enabled, while the rest are disabled.

A supernode is a node that represents a subnet. It might be a superplace or a supertransition. The use of supernodes enables the

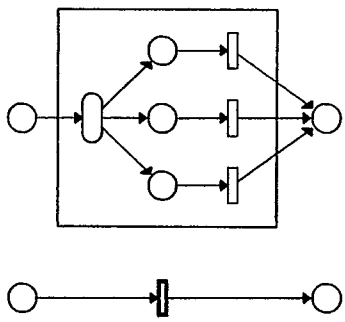

Figure 1: A Switch and its Supertransition Representation

generation of a coarsened version of the Petri Net. It may be viewed as a Petri Net reduction technique. Figure 1, shows a refined and a coarsened version of the Petri Net model of the switch. The refined version contains a switch with three settings. The coarsened version shows the supertransition. Any Petri Net may collapse to a supernode. Figure 2 shows a superplace representation.

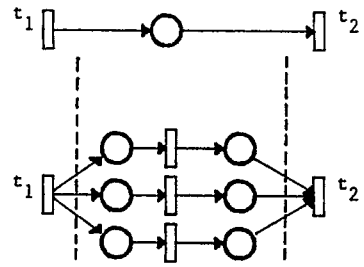

Figure 2: A Superplace Representation

To characterize the structure of a Petri Net, linear algebra representations are introduced. A structural representation is the incidence matrix, which provides description of the flow relations between nodes. For a Petri Net with m places and n transitions, the incidence matrix C is an $m \times n$ matrix, whose elements are defined by the relation,

$$C_{ij} = \begin{cases} -1 & \text{if } p_i \text{ is input place to transition } t_j \\ +1 & \text{if } p_i \text{ is an output place to transition } t_j \\ 0 & \text{no link between } p_i \text{ and } t_j \end{cases}$$

The following definitions are useful:

Definition 1: A Petri Net is connected if there exists a path from any node to any other node.

Definition 2: A directed circuit is a directed path from any node to itself.

Definition 3: A directed elementary circuit is a directed circuit in which no node appears more than once.

Definition 4: A Marked graph is a connected Petri Net, in which each place has exactly one input and one output transition.

Definition 5: An n-dimensional non-negative integer vector X is called an S-invariant if and only if $X^T \bullet C = 0$, (Memmi and Roucairol, 1979).

Definition 6: The set of places whose corresponding components in X are strictly positive is called the support of X and is denoted by $< X >$.

Definition 7: The support of an invariant is said to be minimal if and only if it does not contain the support of another invariant but itself and the empty set.

Definition 8: The S-component associated with $< X >$ is the unique subnet whose set of places is $< X >$ and whose set of transitions consists of all transitions connected to the places of $< X >$ (Sifakis, 1978).

Definition 9: The S-component is said to be minimal if it corresponds to an S-invariant whose support is minimal.

The above definitions in determining the information flow paths of a Marked graph. Information flow or simple paths are the paths that emanate from the source node and terminate at the sink node of a Petri Net. The information flow path indicates the sequential execution of events. The directed elementary circuits of

a Marked graph are exactly the minimal S components of the Marked graph (Hillion, 1986). The Alaiwan and Toudic algorithm (Alaiwan and Toudic, 1985) is used to identify the directed elementary circuits. There maybe more than one flow paths active at the same time. The set of concurrently active paths comprises a complete path. The number of complete paths N for a Petri Net with n switches is $N=\prod_{i=1}^{n} k_i$, where k is the number of branches of switch i. A feedback loop may be used to connect the sink and the source.

Two examples illustrate the graphical design of DMOs. The single interacting DM is modeled as a 4-stage decision process using Petri Nets (Levis, 1984). As the information is received, it is being processed at the first stage for situation assessment (SA). In the second stage there is information fusion (IF) of the SA output with SAs and responses of other DMs. At the fourth stage, all data processing has to be translated into a subset of responses (response selection, RS). The range of responses may be restricted by the command interpretation (CI) stage that precedes the RS stage. Each stage (representing an activity) is modeled as a transition, while the conditions that must be fulfilled for an activity to take place are modeled by places. The Petri Net of a single interacting DM is shown in Figure 3.

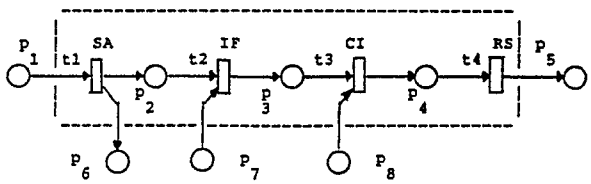

Figure 3: A Single Interacting Decisionmaker

The designer could be greatly assisted if he had at his disposal a design system that could help model and analyze complex organizational structures. The use of such a system will increase the design quality and reduce design time. The architecture of such a System is described next.

SOFTWARE SYSTEM ARCHITECTURE

The Petri Net Design System chart is shown in Figure 4 and it is divided into three levels, the Top Command Level, the Functional Level and the Low Level.

The Top Command Level provides the interactive interface between the user and the System's available modes. A graphics command interpreter evaluates graphics commands and enables the requested mode of operation.

The Functional Level is used for constructing and analyzing

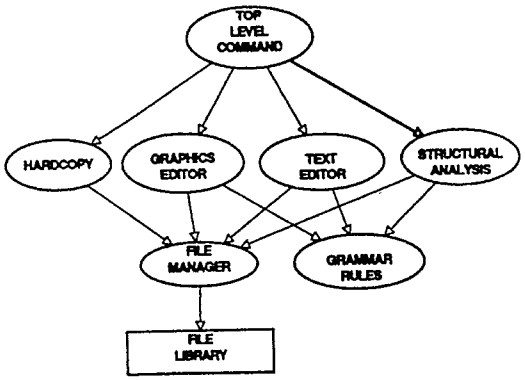

Figure 4: Petri Net Design System Chart

arbitrary organizational architectures. Depending on the specific mode of operation one can create, manipulate and display graphics images interactively, store information represented by graphics primitives and produce specific structural properties of the Petri Net. The Functional Level consists of four modes of operation, the Graphics Editor, the Text Editor, the Structural Analysis and the Hardcopy mode.

The Graphics Editor is used for the interactive generation of the Petri Net structure of the organization. Its purpose is to furnish the designer with a set of graphics tools for developing and modifying Petri Net structures. The unique features of the editor is the interactive development, modification and synthesis of Petri Net structures. One can decompose a large structure into simpler ones, create or resolve supernodes and generate the family tree of an organizational architecture by implementing different switch settings.

While on Graphics Editor mode one can interactively create on the screen a Petri Net structure using the place, transition, switch and connector symbols from the Graphics Commands Menu. In parallel with the graphical design, the underlying data structure is formed. With every graphic element placed on the screen, using the combination of the mouse and keyboard, a set of descriptive information attributes is created and saved in a special formated record. The final structure is stored in a data file containing the text codes and attributes of the graphic symbols . The file is stored in the file library. Subsequent re-editing and modification of the existing structure implies to access the file from the file library, extract, manipulate and restore the contents of records. Record and file manipulation give the designer extra tools when developing a model. A number of operations are available while in Graphics Editor mode.

The designer may delete an existing element by invoking the

delete command. He can define a set of generic structures, used as building blocks of a much larger structure. The building blocks can be developed, analyzed and stored. The final structure is constructed by retrieving the files, and inserting their contents, using the insert command, at a user-specified location and synthesizing the blocks. The composite Petri Net model may be defined as a single unit. Regions of the structure may be specified and copied, using the copy command, to another location. The designer can bound a region of the Petri Net structure and define it as a supernode using the coarsen command. The bounded region collapses into a standard element (superplace or supertransition) with special features. This method permits the existence of different aggregation levels. The reverse operation of unfolding the structure can be performed, using the refine command. If the structure has a switch, one can implement different switch settings and generate the family tree of an organization by using the switch command. To display the contents of other graphic files the show command can be accesed. On line help is available by invoking the help command. The designer may exit the graphics editing session and save the graphic edits using the exit command or delete the edits with quit command.

The Text Editor is used to store, modify and retrieve attributes represented by places, transitions, switches and connectors. The Editor works only with text files that contain specific text editing and formatting capabilities. Four text files are used to store information about each element of a Petri Net structure. Each record of a file may contain information relative to capacity, algorithms, probabilities, logical or mathematical expressions, comments, existence of subnets etc. Each record consists of data fields, where the attribute is stored. Each attribute entry is identified by its type code and data. Different data fields are separeted by a semicolon. For example, the third record of the place text file may look like `C:2;T:comments`, denoting that the current token content of the place with identification number three is two. The T stands for text and following that are comments. Each element has its own list of type codes. In some cases the information stored in text files is used for syntactical purposes. In other cases, for simulation or performance evaluation.

The Structural Analysis mode is used to provide a set of descriptive tools for the analysis of the Petri Net structure. The Structural Analysis mode generates the interconnection matrix, the incidence matrix, computes the minimal supports matrix of S-invariants and displays graphically the information flow paths of a Marked graph. The interconnection matrix is an algebraic representation of the interconnectivity of the nodes and is computed by invoking the matrix command. A Petri Net with M places, N transitions and L switches has an $M\times(N+L)$ matrix representation. The nonzero elements indicate the connectors number. For example, if connector number k, connects the $i'th$ place with $j'th$ transition, then element (i,j) of the matrix has a value $-k$. The construction of the algorithm is simple. The algorithm checks to find what nodes the two ends of the connector are connected, provided that valid connections exist. The incidence matrix is generated by changing every positive element of the interconnection matrix to +1 and every negative one to −1.

The minimal supports of S-invariants are generated using the Alaiwan and Tudic algorithm and implemented using the s-inv command. The algorithm operates on the incidence matrix of a special type of Petri Nets called Marked graphs. The minimal supports matrix is used to determine the information flow paths. The paths may be displayed upon user request. A numeric file is created and stored in the file library with the identification numbers of the elements in the path.

The Hardcopy mode is used to produce a hardcopy of the graphic image of the organizational architecture and its information flow paths on a standard output device such as the printer and plotter.

The Low Level consists of a set of functions performing auxiliary tasks such as the File Server and Grammar-Rule Checker. The File Server consists of a set of utility functions provided to manage the data in files and the files themselves. File management includes file request, verification, manipulation and processing. A file library exists to store the latest version of the files.

The Grammar-Rule algorithms check for the proper graphical construction and enforce the constraints during analysis of organizational architectures. The Grammar-Rule algorithms are observed during a session and alert the user of any implications with a message. The algorithms check for valid connections, for a connected Petri Net, for graphics elements overlap, for elements or subnets placed outside of window boundaries, for existence of self loops and enforce the Marked graph constraint when the minimal supports matrix is computed.

A design methodology is proposed and two examples, demostrating the capabilities of the Petri Net Design System are presented next.

DESIGN METHODOLOGY

The interactive organizational design process involves four phases. The first phase involves the conceptual and graphic design stages. A set of generic structures is identified from the organizational model and constructed graphically by invoking the Graphics Editor. These structures are used as building blocks for composing the overall architecture. For example in the case of

developing a DMO architecture, two, three or four stage Petri Net models, may be constructed, stored in files and recalled during the design. This automates the repetitive operations. Structural decomposition is used to resolve an existing architecture into its constituent components. Then the notion of graphical abstraction is introduced for representing a subnet as a single entity. A subnet, under certain conditions, may be defined as a supernode. In such a case, supernodes replace subnets in the graphical structure, thus creating a coarsened version of the organizational model. One can undo the operation to produce the refined version. Switch settings may be set, enabling the creation of a child structure. The Design System has its own metafiles to store the graphics structure of the organization.

The second phase involves the construction of an alphanumeric data structure for the organizational model by invoking the Text Editor. The data structure contains attributes which correspond to representations of places, transitions, switches, connector or supernodes in the graphics image. Attributes may include processing algorithms and delays for transitions, token capacity for places, functional relations and probabilities for connectors, file names for subnets, comments, codes, etc .

The third phase involves the creation of an analytical data structure of the organizational model by enabling the Analysis mode. Data processing algorithms access the metafile, process its contents and generate the incidence matrix. The incidence matrix serves as an input to an algorithm which produces a matrix containing the minimal supports of S-invariants in the case of an Event graph. The minimal support matrix is used to determine the information flow paths of a Petri Net.

The fourth phase involves the production of a hard copy of the Petri Net graphical image of the organization as well as the information flow paths, by enabling the Hardcopy mode.

APPLICATIONS

Two examples are presented to illustrate some of the graphical and analytical capabilities of the Design System. Two Petri Nets each representing three decisionmakers of a DMO, one with hierarchical and the other with parallel structure ares constructed. The background information is introduced briefly and then the construction of the organization is shown according to the design methodology presented earlier. Both decisionmaking organizations implement an air defence task (Andreadakis and Levis, 1987). In the hierarchical organization, the airspace is divided into two sectors and each sector is assigned to one decisionmaker. A center region is defined that straddles the two sectors and a supervisor is introduced, which does not observe the airspace directly, but receives information about threats in the central region from the other two decisionmakers. He then processes the data and allocates the threats in the central region to either one of the decisionmakers, depending on the trajectory of the threat. In the parallel organization the airspace has been divided into three sectors, with each decisionmaker assigned to one sector. Each decisionmaker can observe and engage threats in his own sector. However, threats may move between sectors, therefore there is a need for communications and information sharing between decisionmakers.

In the first session, the hierachical organization, Figure 5, is constructed. All supernodes are identified and overall structure is simplified. Next the Analysis mode is invoked to identify the structural properties of the net and the flow paths. Similarly the parallel organization is constructed.

Hierarchical Organization. The designer invokes the Graphics Editor from the Top Command Level. The Hierarchical architecture is created using two generic architectures. Both

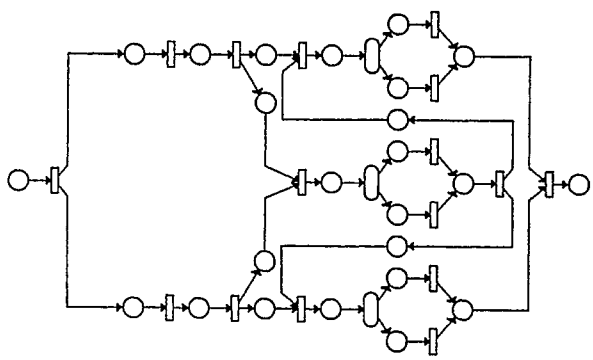

Figure 5: Hierarchical Organization

architectures are used as building blocks to compose the organization's frame. The first generic architecture, is the Petri Net representation of the four stage DM, see Figure 6. The RS stage is modeled by a 2-decision switch. The second generic architecture is the Petri Net representation of a two stage DM, see Figure 7. The two generic architectures are merged to create a single structure, see Figure 8. Elements are selected from the Graphics Commands Menu to complete the structure, Figure 5. The RS stage of each of the three DMs is a switch with two switch settings and can be modeled as a supernode using the coarsen command. The designer specifies the boundaries of the candidate supernode. The aggregated structure is shown in Figure 9.

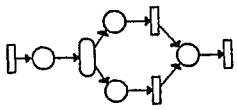

Figure 6: First Generic Architecture

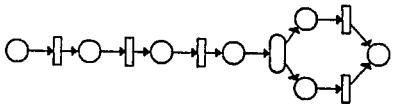

Figure 7: Second Generic Architecture

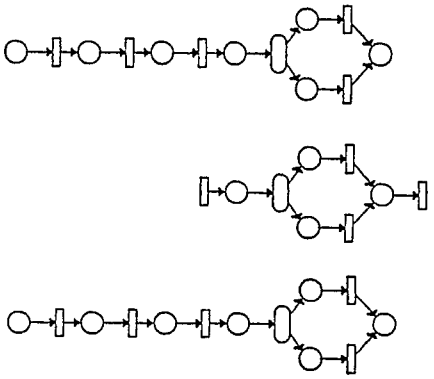

Figure 8: Construction of Hierarchical Model

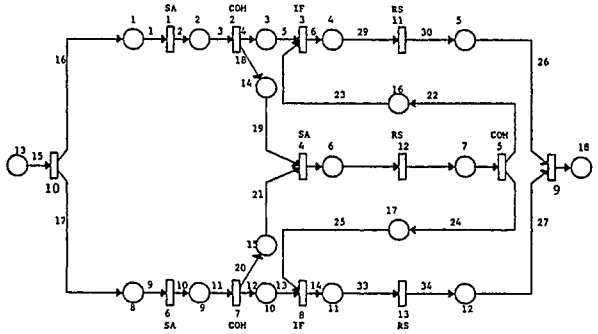

Figure 9: Simplified Hierarchical Organization

-1	0	0	0	0	0	0	0	0	16	0	0	0
2	-3	0	0	0	0	0	0	0	0	0	0	0
0	4	-5	0	0	0	0	0	0	0	0	0	0
0	0	6	0	0	0	0	0	0	0	-29	0	0
0	0	0	0	0	0	0	0	-26	0	30	0	0
0	0	0	7	0	0	0	0	0	0	0	-31	0
0	0	0	0	-8	0	0	0	0	0	0	32	0
0	0	0	0	0	-9	0	0	0	17	0	0	0
0	0	0	0	0	10	-11	0	0	0	0	0	0
0	0	0	0	0	0	12	-13	0	0	0	0	0
0	0	0	0	0	0	0	14	0	0	0	0	-33
0	0	0	0	0	0	0	0	-27	0	0	0	34
0	0	0	0	0	0	0	0	0	-15	0	0	0
0	18	0	-19	0	0	0	0	0	0	0	0	0
0	0	0	-21	0	0	20	0	0	0	0	0	0
0	0	-23	0	22	0	0	0	0	0	0	0	0
0	0	0	0	24	0	0	-25	0	0	0	0	0
0	0	0	0	0	0	0	0	28	0	0	0	0

Figure 10: Interconnection Matrix

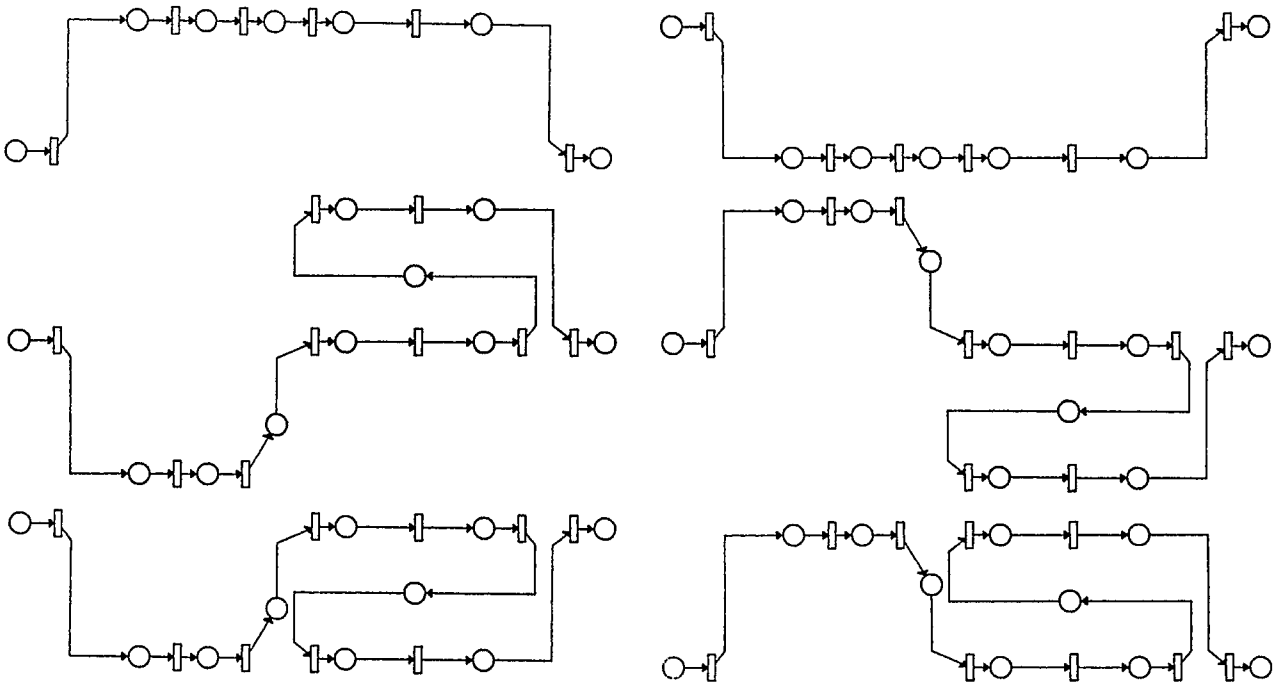

Figure 11: Information Flow Paths of Hierarchical Organization

Next, the Analysis mode is invoked. The interconection matrix is generated by invoking the matrix command and is shown in Figure 10. The coarsened Petri Net representation of the hierarchical organization is a marked graph and the minimal supports are computed, using the s-inv command. Then, the information flow paths can be displayed upon user's request, see Figure 11.

Parallel Organization. The Petri Net representation of the parallel organization is constructed in a similar fashion. The same generic architectures used in the construction of the hierarchical organization are used. The Petri Net shown in Figure 12 has decision switches. It has three switches with two switch settings each. In the hierarchical case, the decision switch and its possible paths were modeled as a supernode. In the parallel case a switch setting is selected and a particular decision strategy is implemented, by invoking the switch command. The selection of switch setting is done by the System if the proper information (probability assigned to each connector) is stored in the connector attribute text file. If the probability of instantiating path A is greater than the probability of instantiating path B, then path A is selected. If such information does not exist, then the settings are selected by the designer. The structure created after all settings are set is stored in a file and is shown Figure 13.

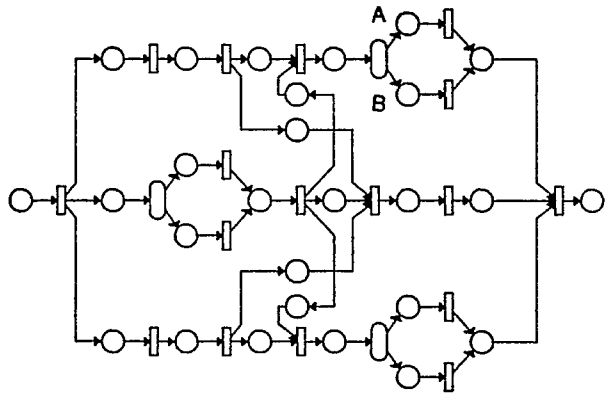

Figure 12: Parallel Organization

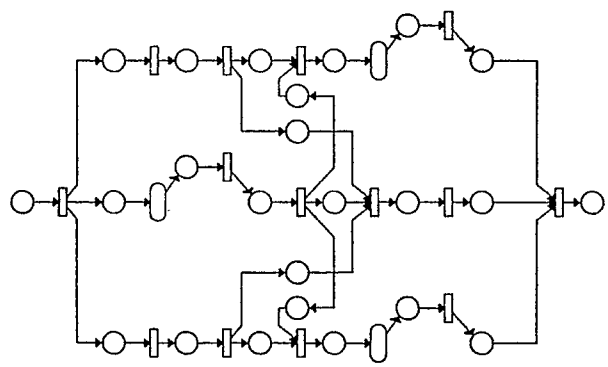

Figure 13: Parallel Organization: Instantiation One

CONCLUSIONS

The Petri Net Software System is used for the graphical and analytical description of DMO architectures. The System has four modes of operation: A Graphics Editor, a Text Editor, a Structural Analysis and a Hardcopy mode. The Graphics Editor mode capabilities enable the construction and modification of organizational architectures. In Graphics mode the designer may coarsen or refine a structure, may merge generic architectures to form an aggregate structure, may copy subnets and may generate the family tree of an organization by implementing different combinations of switch settings. The Text Editor mode provides attribute management. The Structural Analysis mode provides the tools for analysis. The Hardcopy mode produces output on a printer or plotter. The System may be used as the front end in a simulation.

REFERENCES

Alaiwan, H. and J.M. Toudic (1985). Recherche des Semi-Flots, des Verrous et des Trappes dans les Reseaux de Petri. *Technique et Science Informatiques*, 1985, *4*(1), 103-112. Dunod, France.

Andreadakis, S.K. (1988). *Analysis and Synthesis of Decision-Making Organizations.* PhD thesis, MIT, 1988. Laboratory for Information and Decision Systems, Cambridge, MA.

Andreadakis A.K and A.H. Levis (1987). *Accuracy and Timeliness in Decision-Making Organizations.* Technical Report LIDS-P-1650, MIT, 1987. Laboratory for Information and Decision Systems, Cambridge, MA.

Grevet, J.L. (1988). Decision Aiding and Coordination in Decision-Making Organizations. Master's thesis, MIT, 1988. Laboratory for Information and Decision Systems, Cambridge, MA.

Hillion, H.P. (1986). Performance Evaluations of Decision Making Organizations Using Time Petri Nets. Master's thesis, MIT, 1986. Laboratory for Information and Decision Systems, Cambridge, MA.

Kyratzoglou, M.I. (1987). Computer Aided Design for Petri Nets. Master's thesis, MIT, 1987. Laboratory for Information and Decision Systems, Cambridge, MA.

Levis A.H. (1984). Information Processing and Decisionmaking Organizations: A Mathematical Description. *Large Scale Systems*, 1984, *7*, 151-163.

Memmi, G. and G. Roucairol (1979). *Lecture Notes in Computer Science.* Volume 84: Linear Algebra in Net Theory. In *Net Theory and Application,* Hamburg, FRG: Springer Verlag, 1979.

Peterson, J.L. (1981). *Petri Net Theory and the Modeling of Systems.* Englewood Cliffs, New Jersey:Prentice Hall, 1981.

Ramchandani, C. (1974). *Analysis of Asynchronous Concurrent Systems by Timed Petri Nets.* PhD thesis, MIT, 1974. Laboratory for Computer Science, Cambridge, MA.

Sifakis, J. (1978). *Lecture Notes in Computer Science.* Volume 64: Structural Properties of Petri Nets. In *Mathematical Foundations of Computer Science,* Berlin, FRG: Springer Verlag, 1978.

Tabak, D. and A.H. Levis (1985). Petri Net Representation of Decision Models. *IEEE Transactions on Systems, Man and Cybernetics*, 1985, *SMC-15*(6), 812-818.

AUTHOR'S BIOGRAPHY

IOANNIS M. KYRATZOGLOU (S'81-M'84-ME'87) was born in Athens, Greece, on August 3, 1956. He received the B.S. degree from City College of New York, in 1981, the M.S. and Mechanical Engineer's degrees from Massachusetts Institute of Technology, Cambridge, in 1984 and 1987, respectively, all in Mechanical Engineering.

He has been employed for a year at BBN Systems and Technologies and engaged in research and development with the Sensors and Surveillance Division on sonar applications of real time systems. His research interests are in the sonar applications of digital signal processing techniques, and in the theory and applications of command and control organizations.

Mailing Address: Mr. Ioannis M. Kyratzoglou
BBN Systems and Technologies
70 Fawcett Street
Cambridge, MA. 02238
(617) 873-4232