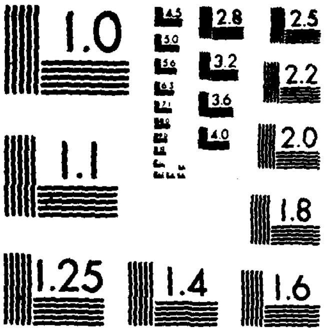
AD-A191 950 DECISION AIDING AND COORDINATION IN DECISION-MAKING 1/2
ORGAINZATIONS(U) MASSACHUSETTS INST OF TECH CAMBRIDGE
LAB FOR INFORMATION AND D.. J M GREVET JAN 88
UNCLASSIFIED LIDS-TH-1737 N00014-84-K-0519 F/G 5/1 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

4

JANUARY 1988

LIDS-TH-1737

Research Supported By:
Office of Naval Research
Contract N00014-84-K-0519
(NR 649 003)

DECISION AIDING AND COORDINATION IN DECISION-MAKING ORGANIZATIONS

Jean-Louis M. Grevet

Laboratory for Information and Decision Systems
MASSACHUSETTS INSTITUTE OF TECHNOLOGY, CAMBRIDGE, MASSACHUSETTS 02139

JANUARY 1988 LIDS-TH-1737

DECISION AIDING AND COORDINATION IN DECISION-MAKING ORGANIZATIONS

by

JEAN-LOUIS M. GREVET

This report is based on the unaltered thesis of Jean-Louis M. Grevet, submitted to the Department of Civil Engineering in partial fulfillment of the requirements for the degree of Master of Science in Technology and Policy at the Massachusetts Institute of Technology in January 1988. The research was conducted at the MIT Laboratory for Information and Decision Systems, with support provided by the Office of Naval Research under Contract Number N00014-84-K-0519 (NR 649 003).

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139

DECISION AIDING AND COORDINATION
IN
DECISION-MAKING ORGANIZATIONS

by

JEAN-LOUIS M. GREVET

Ingénieur de l'Ecole Centrale des Arts et Manufactures
(1985)

SUBMITTED TO THE DEPARTMENT OF CIVIL ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE
IN TECHNOLOGY AND POLICY
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
January 1988

Signature of Author ____

Department of Civil Engineering
January 15, 1988

Certified by ____

Dr. Alexander H. Levis
Thesis Supervisor

Accepted by ____

Professor Richard de Neufville, Chairman
Technology and Policy Program

Accepted by ____

Professor Ole S. Madsen, Chairman
Departmental Committee on Graduate Students
Department of Civil Engineering

DECISION AIDING AND COORDINATION
IN
DECISION-MAKING ORGANIZATIONS

by

JEAN-LOUIS M. GREVET

Submitted to the Department of Civil Engineering
on January 15, 1988
in partial fulfillment of the requirements for the degree of
Master of Science in Technology and Policy

ABSTRACT

A methodology to analyze, model, and evaluate decision-making processes that require coordination is presented. The concept of a team of decision-makers and the issues of inconsistency of information and synchronization are emphasized. Predicate Transition Nets are used as the basic technique to represent organizational structures. Two measures of coordination are introduced: the degree of information consistency and the measure of synchronization. The simulation of Petri Nets is presented as a means of investigating the dynamics of decision-making processes requiring coordination. A model of decision support systems (DSS) is developed. An example of a two-person hierarchical organization aided by a DSS is given as an illustration: The accuracy, time delay and synchronization are computed. The results are that decision aids can alter the coordination of decision-making organizations by modifiying the priority order with which information is processed, and by increasing the number of information flow paths with different processing times.

(Keywords: Systems effectiveness; interactions; information processing)

Thesis Supervisor : Dr. Alexander H. Levis
Title : Senior Research Scientist

ACKNOWLEDGEMENTS

I wish to express my gratitude to:

Dr. Alexander Levis, for his guidance throughout the work on this thesis. Working with him has been a valuable experience.

Dr. John Brode, for his help and advice.

Lisa Babine, for her support and kindness.

My comrades of the Laboratory for Information and Decision Systems as well as my friends, Mathieu and Thierry.

My family and Nicola.

This research was carried at the MIT Laboratory for Information and Decision Systems, with support provided by the Office of Naval Research under Contract No. N00014-84-K-0519 (NR 649-003).

TABLE OF CONTENTS

Page

Page

Page

Page

LIST OF FIGURES

Page

Page

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 PROBLEM DEFINITION

The analysis, design and evaluation of Command, Control and Communications (C^3) systems have been major concerns over the past decade. The development of effective C^3 systems is, indeed, the sine qua non condition for improving the capabilities of military organizations to meet the requirements of the missions they have to accomplish.

The complexity of the battlefield imposes severe constraints on the human beings who participate in the operations. The decision-making and information processing organizations designed to execute these tasks exist in hostile environments where the tempo is fast and the data to be gathered and analyzed are numerous.

Therefore, on the one hand, the distribution of processing between the hardware and human components is a necessity in order to facilitate the task carried out by each organization member. Their activities are synergistic and coordination of the decision-making processes must be achieved in order to improve the effectiveness of the organization. It follows that structures which allow the coordination of the different activities necessary to fulfil the mission must be designed. On the other hand, decision aids, which are part of the C^3 systems, aim at increasing the ability of decision-makers to perform their mission effectively. By offering faster processing capabilities as well as access to databases, they may help the organization members to achieve the requirements of the mission.

However, decision aids also increase the possible alternatives among which to choose in order to process information, and in so doing, modify the nature of the decision-makers' activities. In this context, it is important to evaluate the extent to which decision aids, and more particularly decision support systems, can alter the coordination of the various decision-making processes.

1.2 BACKGROUND

The framework is the quantitative methodology (Levis, 1984) for the analysis and evaluation of alternative organizational structures. The definition and computation of Measures of Performance and Measures of Effectiveness provide a systematic means for assessing the extent to which organizations are effective in meeting the requirements of the missions that they must perform.

The model of the interacting decision-maker (Boettcher and Levis, 1982; 1983) consists of four stages which constitute the successive internal phases that the decision-maker uses to process the information. These stages take into account the interactions between the various organization members.

The Petri Net formalism has been introduced (Tabak and Levis, 1985) to represent organizational forms because it is a convenient framework to model asynchronous and concurrent processes. Petri Nets show explicitly the interactive structure between decision-makers and the sequence of operations within an organizational structure. Furthermore, they prove useful in evaluating the timeliness of the activities carried out by organizations.

The Predicate Transition Net formalism builds on the Petri Net theory to allow the modeling of processes where it is necessary to distinguish specific elements. This framework will be used in this study to develop a model of coordination in decision-making organizations based on the identification of symbolic information carriers in the net.

The modeling and evaluation of decision aids in decision-making organizations has been the subject of previous efforts (Chyen and Levis, 1985; Bejjani and Levis, 1985; Weingaertner and Levis, 1987). In particular, the impact of preprocessors on the workload of organization members assisted by such aids has been investigated. The present study is consistent with the result that a decision-maker has to make meta-decisions with respect to the use of a decision aid.

Therefore, this work builds on the methodology for evaluating organizational structures and integrates the modeling, analysis and evaluation of coordination and decision support systems in decision-making organizations into this framework.

1.3 GOAL AND CONTRIBUTIONS

The goal of this work is to provide some insight on the cohesiveness of organizations carrying out well-defined tasks and the extent to which decision aids can alter it .

In order to do so, the concept of coordination is specified as it relates to static, as well as dynamic, characteristics of the decision-making processes. It embodies three classes of problems, i.e., the extent to which the decision-makers constitute a team, and the issues of inconsistency of information and synchronization. Information processing stages of decision-makers whose interactions require coordination are modeled using the Predicate Transition Net formalism. This model accounts for the fact that, when decision-makers interact, they have some protocol to recognize that they are exchanging information pertaining tu the same event in the environment. Two measures for evaluating coordination are introduced: the degree of information consistency and the measure of performance in synchronization. The latter measure relates to the value of information when the decision-makers actually process it. A generic model of a decision-maker aided by a decision support system (DSS) is presented. It leads to the investigation of an example of an organization aided by the DSS. The simulation of Petri Nets has been implemented through the development of a software system in order to study the dynamics of decision-making activities.

This study makes a contribution to the methodology for the analysis of decision-making organizations from the conceptual, modeling, and evaluation standpoints:

(a) the importance of coordination is emphasized. It is shown that the effectiveness of decision-making organizations depends to a large extent on the level of cohesiveness reached by its various members.

(b) the model of a decision-maker is refined to account for processes that require coordination. The use of Predicate Transition Nets as a modeling tool is shown.

(c) the simulation of Petri Nets is introduced to investigate the dynamics of decision-making processes. It allows to overcome some of the analytical difficulties that are introduced by the model.

(d) the study of examples demonstrate that decision aids can degrade the coordination of decision-making organizations by affecting the dynamics of the activities and by increasing the number of alternatives for processing information.

1.4 OUTLINE OF THE THESIS

The thesis is organized as follows. In Chapter II, an overview of the concepts that constitute the background of the study is presented. The analysis and modeling of coordination in decision-making organizations using the Predicate Transition Net formalism are provided in Chapter III. The evaluation of coordination on the basis of this model, as well as the definitions of the degree of information consistency and of the measure of synchronization, are carried out in Chapter IV. The dynamics of coordinated decision-making processes is investigated in Chapter V using a simulation program for Predicate Transition Nets. In Chapter VI, the model of a decision support system is provided, while an example is presented in Chapter VII. Conclusions and directions for future research are given in Chapter VIII.

CHAPTER II

BACKGROUND

This chapter presents an overview of the basic concepts that constitute the background of this study as well as a survey of the analytical, modeling, and evaluation tools that will be used. The concept of a team is presented as introduction to the concept of coordination in information processing and decision-making organizations. The Measures of Performance and Measures of Effectiveness, which constitute the essential evaluation tools used throughout the work, are then introduced. The Petri Net formalism is subsequently reviewed because it is appropriate for modeling and simulating decision-making organizations. Finally, the framework for the evaluation of organizational structures is reviewed.

2.1 THE CONCEPT OF A TEAM

Many organizations are considered as teams and yet exhibit different structural and functional characteristics, i.e., they have different topologies, protocols and degrees of interdependence between their members. This section provides a review of the diverse definitions that are usually given to this notion.

According to the third Webster's International Dictionary: "*a team is a number of persons associated together in work or activity as:*

- *a number of persons selected to contend one side in a match.*
- *a group of workmen, each completing one of a set of operations.*
- *a group of specialists or scientists functioning as a collaborative unit, each performing several undefined operations.* "

These definitions emphasize two concepts, i.e., the commonality of goal and the association of different persons on the basis of their skills:

(i) The commonality of goal implies that :

- the goal characterizes partially a certain state of the world and can be common to different states of the world. It can be the result of a task or a value.
- all the members of a team have some knowledge of the goal they want to reach.

- they know that this goal is common to all of them.
- they want to attain it together, but do not necessarily know when.

(ii) The association of different persons on the basis of their skills implies that:

- one single person cannot constitute a team.
- the members of the group must be associated in a certain way, i.e., they must be interdependent.

The first meaning stresses the fact that the team's members are chosen on the basis of their aptitudes, or skills, by a designer whose goal is to construct a team superior to any other in the context of a particular game or conflict. Therefore, the designer must:

- evaluate the capabilities of different persons.
- select the team's members on this basis.
- abide by the rules of the game, if they exist, to define the organization structure and a global strategy.
- possibly assign a precise task to each of the team's members, this possibility depending on the uncertainty of the environment.

A sports team illustrates well this type of organization.

The second meaning assumes that the global strategy, the organization structure, and the tasks are already well-defined and that only the team's members must be chosen. It is, for instance, the case of the assembly line of a plant.

Finally, the last meaning addresses another concept. There are no rules of the game and no well-defined task to be performed. The team's members are either selected by a designer or just gather, and each of them can perform different undefined operations in order to attain the common goal: task forces and research teams fall into this category.

In the "Economic Theory of Teams" (Marschak and Radner, 1972), the definition of a team introduces more precisely the concept of information. Actions performed in an organization differ, in general, from those of a single person in two respects:

- the kind of information on the basis of which each member of the organization decides about his actions may differ from one member to another.
- the interests and beliefs of each member of the organization may differ from the interests and beliefs of his fellow members.

A team is defined as an organization in which the first but not the second characteristic is present, i.e., the members of a team have the same interests and beliefs but do not share the same information. In this context, the team problem can be defined as follows: how should the tasks of inquiring, communicating and deciding be allocated among the members of an organization so as to achieve results that would be best from the point of view of their common interests and beliefs, or those of the organizer ?

In the context of corporations (Eddy, 1985) , the following definition of a team is given : " *a team, or small group, is a collection of people who are interdependent; they find it necessary to collaborate to function effectively, think of themselves as belonging to the group, follow a set of rules or guidelines that define appropriate behavior for members and agree about the purposes and goals of the group* . "

Thus, a team consists of two or more people who must coordinate their activities to accomplish a common task. It is not enough for them to want to coordinate. Coordination must be required to accomplish the task. Therefore, if two persons can best perform their task without coordination, they do not constitute a team. If only a portion of their tasks require interdependence, they are a team for only those tasks that require coordination and only during the lifespan of those tasks. The concept of coordination is more stringent than the simple idea of association of different persons. It implies indeed that the team's members must adapt their behavior to the behavior of the other members.

These different considerations will lead in Chapter III to a generic definition of the concept of a team that applies to the class of information processing and decision-making organizations analyzed in this thesis.

2.2 MEASURES OF EFFECTIVENESS

2.2.1 System Effectiveness Analysis

System Effectiveness Analysis (SEA) is a methodology introduced by Dersin and Levis (1981) and developed by Bouthonnier and Levis (1984), Karam and Levis (1985), Cothier and Levis (1986), and Martin and Levis (1987). Its purpose is to provide a framework for measuring the extent to which a system, given its performance, is effective in meeting the requirements of the mission it is designed to accomplish.

The six basic concepts introduced by the SEA are: the system, the environment, the context, the parameters, the measures of performance (MOP's), and the measures of effectiveness (MOE's).

The **System** consists of components, their interconnection and a set of operating procedures: it can be a data communication network or, more generally, an organization consisting both of human beings and equipment. Thus, it is a set of elements that act together by exchanging information toward the achievement of a particular goal.

The **Environment** is the set of elements of the universe which do not belong to the system, and such that the system can act upon them and they can act upon the system. In the case of a military organization, it would encompass for example the enemy's forces. The mission that the system must perform is the particular state of the environment that has to be achieved by the system.

The **Context** consists of all the other elements of the universe. They are defined as being the elements which can act upon the system but upon which the system cannot act. It can be, for instance, the weather conditions.

The **Parameters** are the independent quantities used to specify the system and the mission requirements. For example, in the case of a communication network, they include such quantities as the probability of failure of links and nodes or the capacities of communication lines.

2.2.2 Measures of Effectiveness

The evaluation of the effectiveness of a particular system can be carried through the definition and measurement of specific quantities called **Measures of Effectiveness**. Since the concept of effectiveness embodies different perceptions of a problem, it is possible to define many MOE's for a given system. For example, when one studies the effectiveness of a decision support system (DSS), one can assess the appropriateness of the DSS from the user's standpoint or from the overall organization's standpoint for different life cycle stages of the DSS, e.g., design, prototype, or operational aid (Riedel and Pitz, 1986). Thus, one must define precisely what is measured and assessed by the MOE's.

The **Measures of Perfomance** are measurable quantities that describe system properties, e.g., system reliability or survivability. If n designates the number of MOP's describing the system and the mission, the system locus and the mission locus are two subsets of the attribute space Ω, itself a subset of $\mathfrak{R}^n$, where $\mathfrak{R}$ is the set of real numbers.

The system capabilities, expressed by the system MOP's, and the mission requirements, expressed by the mission MOP's, are compared in the common attribute space Ω. Therefore, MOE's are quantitative data that result from the comparison of two sets:

- the set of the system measures of performance, represented by the system locus L_s in $\mathfrak{R}^n$.
- the set of the mission requirements, represented by the mission locus L_m in $\mathfrak{R}^n$.

The system MOP's depend on the system characteristics, the environment and the context; in the same way, the mission MOP's depend on the mission characteristics, the environment and the context. It is particularly important to define correctly the system, the mission, the environment, the context, as well as the different parameters which characterize the system and the mission for the given context and environment (Bouthonnier and Levis, 1984; Karam and Levis, 1985; Cothier and Levis, 1986; Martin and Levis, 1987).

The points of the system locus are not necessarily equally likely : for example, in the case of decision-making organizations, there might be more organizational strategies that lead to a certain performance than to another one. The points of the mission locus may also not have the same utility for the system designer (Karam and Levis, 1985). Thus, it is necessary to define a measure of effectiveness that will account for those considerations:

- a probability distribution f is defined over L_s by:

$$\begin{aligned} f \; : \; & L_s \;\text{->}\; [0,1] \\ & X \;\text{->}\; f(X) \\ & \int_{L_s} f(X)\,dX \;=\; 1 \end{aligned} \tag{2.1}$$

- a utility distribution u is defined over L_m by:

$$u : L_m \text{->} [0,1] \qquad\qquad\qquad\qquad$$
$$X \text{->} u(X) \qquad\qquad\qquad (2.2)$$

The MOE E_f, which accounts for the fact that the system will be most effective with regard to the mission if it reaches points of the mission locus that are highly desirable, is then defined by:

$$E_f = \int_{L_s \cap L_m} f(X)\, u(X)\, dX \qquad (2.3)$$

For a given context, environment, and mission the system designer has only one alternative to increase the system's effectiveness, i.e., to modify in a certain way the system itself. As consequences of this modification, there are three possibilities:

(i) the system locus L_s is not altered, but the probability distribution is changed so that the points of $L_s \cap L_m$ that are highly desirable have a higher probability of being reached by the system.

(ii) the shape of L_s is not changed. Each point X of L_s is mapped onto a point X' of Ω by a rotation or a translation such that the probability distribution f verifies f(X) = f(X') and so that $L_s \cap L_m$contains points that are highly probable and desirable.

(iii) the shape of the system locus is altered and consequently the probability distribution f is redefined.

These cases are illustrated in Figure 2.1 in the two-dimensional space. Each system MOP X depends on two groups of parameters which characterize the system in the given context and environment:

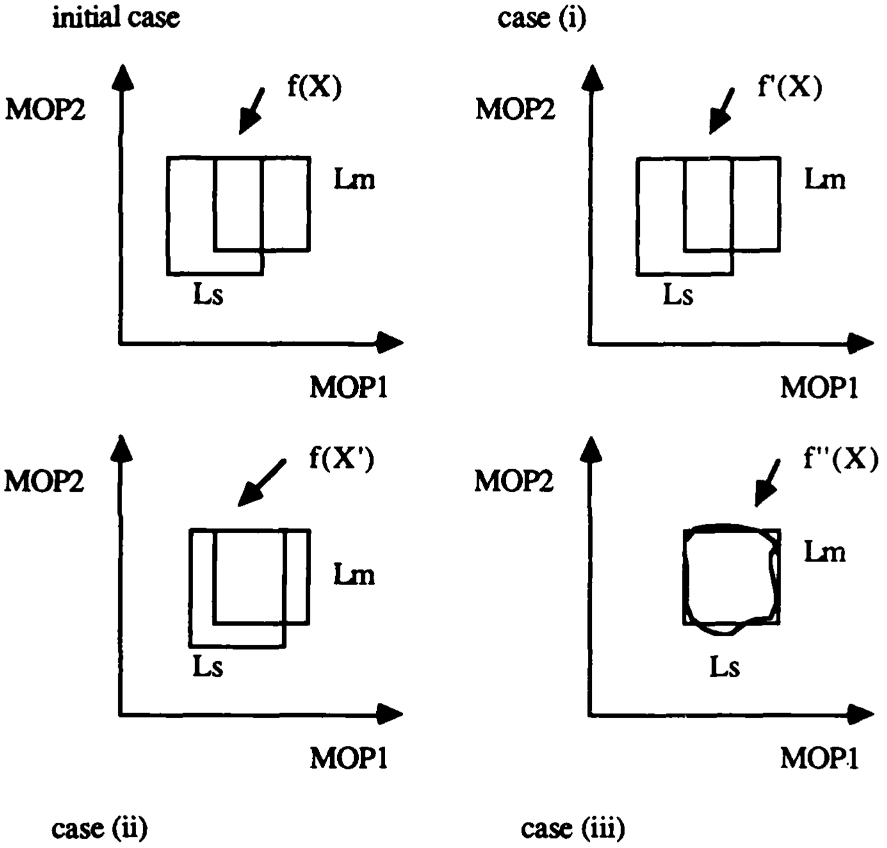


Fig. 2.1 System and Mission Loci

(i) on the one hand, X depends on the components which constitute the system. These components are physical entities such as, for instance, human beings or computer hardware and software. They exhibit certain characteristics which are inherent to them and do not depend on the environment, e.g., the memory capacity of a computer or the information processing rate capability of a human decision-maker. They also have properties that are closely related to the state of the environment or of the context. It can be the perception of the environment or beliefs that decision-makers have, or some models embodied in software.

(ii) on the other hand, X depends on the structure or interrelationships between the system components, i.e., their roles, the protocols, their degree of interdependence. In the same way as above, this structure has characteristics that depend on the environment and characteristics that do not. For example, the roles of the members of a sports team do not change from one game to another. However, the tactics can vary.

Therefore, each MOP depends on parameters which are independent from each other and which describe the physical chracteristics and the structure of the system in the given environment and context. Thus,

$$MOP_i = g_i(\text{structure, components}) \qquad (2.4)$$

The various MOP's chosen to assess system effectiveness can depend on common parameters, which implies that the variation of one parameter can have unforseeable results on the shape and position of the system locus as well as on the probability distribution.

The framework of the System Effectiveness Analysis will be used as a basis for the analysis and evaluation developed in this study for two reasons:

- it is consistent with the methodology used in previous work (Weingaertner and Levis, 1987).
- it proves adequate for the investigation of the coordination of organizations.

2.3 PETRI NET FORMALISM AND THEORY

The functioning of multi-person decision-making organizations involves the execution of concurrent and asynchronous processes. Each of the decision-makers operates indeed in parallel with the other organization members, and they, in turn, coordinate their activities when necessary. These interactions occur asynchronously and not at set times. Petri Nets are, therefore, suitable for analyzing, modeling and simulating decision-making organizations because they allow to show explicitly the structure of the interactions between the decision-makers (Peterson, 1981; Brams, 1983; Reisig, 1985).

2.3.1 Petri Nets

Definition 1: A Petri Net - denoted by PN - is a bipartite directed graph represented by a quadruple PN = $(\mathcal{P},\mathcal{T},I,O)$ such that:

- $\mathcal{P} = \{p_1,...,p_n\}$ is a finite set of places.
- $\mathcal{T} = \{t_1,...,t_m\}$ is a finite set of transitions.
- I is a mapping $\mathcal{P}\times\mathcal{T} \text{ -> } \{0,1\}$ corresponding to the set of directed arcs from places to transitions.

- O is a mapping $\mathcal{T} \times \mathcal{P}$ -> {0,1} corresponding to the set of directed arcs from transition to places.

An example of Petri Net is shown below: places are represented by circles and transitions by bars.

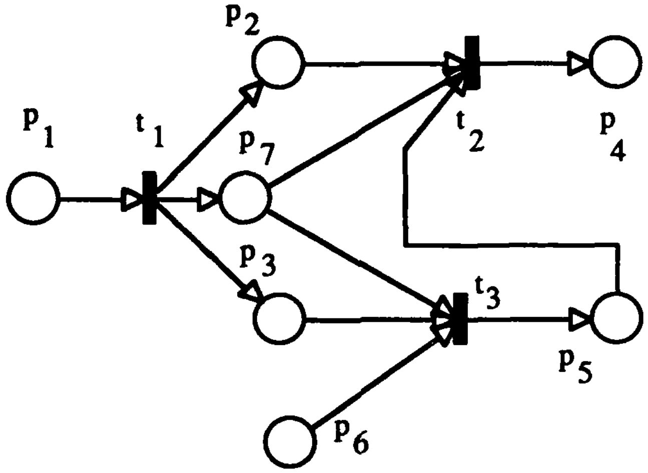


Fig. 2.2 Petri Net PN_1

Definition 2: The postset of transition t, denoted by Pos(t), is the set of all its output places; Pos(t) = $\{p \in \mathcal{P} ; O(t,p) > 0 \}$

The preset of transition t, denoted by Pre(t), is the set of all its input places; Pre(t) = $\{p \in \mathcal{P} ; I(p,t) > 0 \}$

Similarly, the preset and postset of any place can be defined:

- Pos(p) = $\{t \in \mathcal{T}; I(p,t) > 0 \}$
- Pre(p) = $\{t \in \mathcal{T}; O(t,p) > 0\}$

In the example illustrated above, the following relations hold:

- $Pre(t_3) = \{p_3, p_6, p_7\}$
- $Pos(p_7) = \{t_2, t_3\}$

Definition 3: The marking of a Petri Net is a mapping : $\mathcal{P}$ -> $\mathcal{N}$ where $\mathcal{N}$ designates the set of non-negative integer numbers. It is a one-to-one mapping which assigns a number of tokens to each place of the net.

The marking can be represented by a n-dimensional vector M, whose components correspond to the places of the net.

This marking can be illustrated graphically by writing the number of tokens of each place in the circle that represents the place as shown below for Petri Net PN_1; empty places are left blank. Here, $M^0 = (2,3,2,0,1,4,1)$

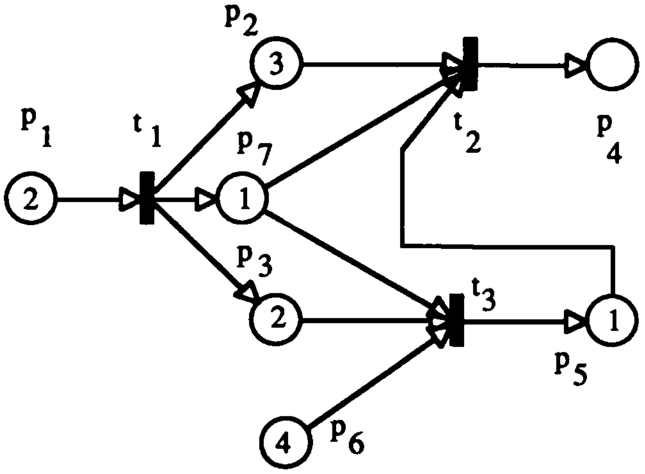


Fig. 2.3 Marked Petri Net PN_1

Definition 4: A transit'on t is enabled by a given marking if and only if for each place $p \in Pre(t)$ the number of tokens in p is greater than or equal to I(p,t). When the transition t is enabled, it may fire; The new marking M' reached after the firing of t is defined by:

$$\forall p \in \mathcal{P}, M'(p) = M(p) + O(t,p) - I(t,p).$$

Definition 5: A transition t is free to fire for a given marking if and only if for each place $p' \in Pos(t)$, the sum of the number of tokens in p' and O(t,p') is less than or equal to the capacity of the place, i.e., the maximum number of tokens that this place can hold at the same time. An infinite capacity means that there is no bound on this value for the corresponding place. When capacities are introduced, a transition may fire if and only if it is enabled and free to fire.

Definition 6: Given an initial marking M^0, the reachability set of the Petri Net PN is the set of all possible markings reachable by firing any sequence of transitions.

Property 1: A marking M^0 is bounded if there exists a bound on the number of tokens in any place of any reachable marking. If this bound is equal to one, the marking is safe. A Petri Net is structurally bounded, if any initial marking is bounded.

Property 2: A marking M^0 is live if for any transition t and for every reachable marking M, there exists a firing sequence of transitions from M that includes t. A Petri Net is structurally live if any initial marking is live.

Thus, a Petri Net is a formal model of information flow. The tokens can be considered as symbolic information carriers; the places are the nodes where those tokens can stand without being processed; the transitions are the events that perform any kind of transformation on the information: it can be a transmission, i.e., the information is transferred from one geographical point to another geographical point, or a computation, i.e., the information is altered by some process, or a decision.

A conflict arises when two transitions, whose presets are not disjoint, are enabled simultaneously. This type of situation is illustrated in Figure 2.3 for Petri Net PN_1. One can see that t_2 and t_3 are enabled and that p_7 belongs both to $Pre(t_2)$ and $Pre(t_3)$. The two transitions can fire but the firing of one of them will disable the other. The resolution of this conflict is done according to a certain rule, e.g., priority of t_2 over t_3, or probability distribution. Switches have been introduced to model the resolution of conflicts (Tabak and Levis, 1985).

Definition 7: A switch is a transition which resolves conflict situations. It is thus a transition with multiple output places and some decision rule according to which each token is routed toward one and only one of the output places.

The introduction of a switch in Petri Net PN_1 is illustrated in Figure 2.4.

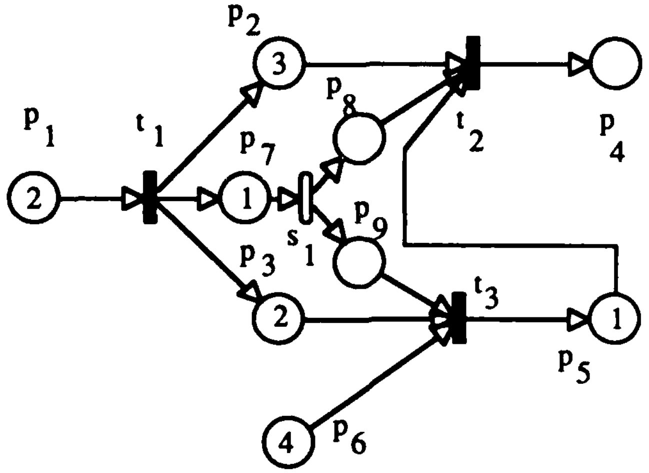


Fig. 2.4 Petri Net PN_1 with switch

2.3.2 Timed Petri Nets

Definition 8: A Timed Petri Net is a Petri Net in which a firing time is associated with each transition of the net.
The firing time is restricted to take values in the set of non-negative rational numbers.

The fact that the firing times are rational numbers allows to discretize the process in units of time and to observe the state of the system at each instant of time. For each transition of the net, any firing will take a certain amount time to be completed. This amount of time can be deterministic or stochastic, and can depend on parameters, e.g., the current marking of the net. When a transition initiates its firing, it removes immediately the tokens from the input places but inserts tokens in the output places only when an amount of time equal to its firing time has elapsed.

During the firing of the transition, if the transition is enabled again by the presence of tokens in all its input places, the transition may fire again. One can put a constraint of the firing rules of a Timed Petri Net by allowing transitions to fire only when they are not already executing. The execution of a Petri Net according to this rule will take more time than when transitions are allowed to process several tokens at a time. However, in the case of decision-making organizations, transitions model algorithms implemented by decision-makers and it is more realistic to assume that a decision-maker processes one item of information at a time in each of his internal stages. This constraint will therefore be used in the remainder of this study because it allows to apply the Timed Petri Net formalism to evaluate the real-time processes that take place in decision-making organizations. Thus, the following definitions will hold:

Definition 9: A transition t is inactive if and only if t has terminated all its previous firings.

Definition 10: A transition may fire if and only if t is enabled, free and inactive.

Firing times can also be assigned to switches since they constitute a special type of transition. The conflict resolution in ordinary Timed Petri Nets cannot be assigned any execution time, whereas switches allow it. The fact that a transition or switch is active will be illustrated by changing the colour pattern of the node representing the transition or switch, as shown in Figure 2.5.

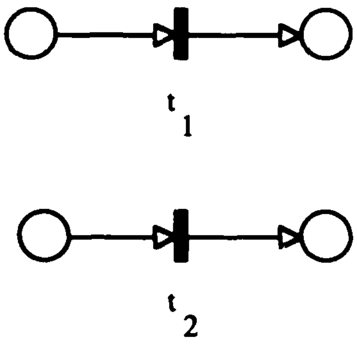


Fig. 2.5 Inactive (t_1) and Active (t_2) Transitions

The introduction of time in Petri Nets allows their use not only as an analytical and modeling tool, but also as an evaluation tool for measuring the performance of decision-making organizations with respect to timeliness.

2.3.3 Predicate Transition Nets

In the previous definitions, an underlying assumption was that the tokens were not distinguishable, i.e., no attribute was associated with the tokens. In the case of information processing and decision-making organizations, the tokens represent information and therefore should have an identity. Predicate Transition Nets (Genrich and Lautenbach, 1981) constitute thus the appropriate framework for studying the processing of information.

Definition 11: Predicate Transition Nets are Petri Nets where:

- each token is an individual of a class; classes of individuals are called variables.
- each place can be associated with a predicate, i.e., a proposition with changing truth value; it is marked with tokens which are the argument of the proposition.
- connectors are labelled with the formal sum of variables to designate the kinds of tokens they can carry. The tokens with no identity are labelled with ¢ .
- each transition can be associated with a function that evaluates whether or not it is enabled depending on the content of its input places and on the labels of its input connectors.

The Predicate Transition Net formalism is useful from many standpoints: first, as mentioned above, it allows to distinguish between the different tokens; Second, it leads to simpler representations of Petri Nets which have some properties of symmetry by folding them. The remainder of this work will make use of this framework in order to model the interactions that occur between different decision-makers in an organization as well as their different strategies for processing information.

Example: the purpose of the following example is to show how Predicate Transition Nets can be represented and to illustrate how the firing process takes place. Consider PN_2 in Figure 2.6.

The connector between p_1 and t_1 carries one token of variable x; the connector between p_2 and t_2 carries one token of variable y. The function of transition t_1 checks whether or not the two tokens are equal; the connector between t_1 and p_3 carries a token of variable x which is inserted in place p_3.

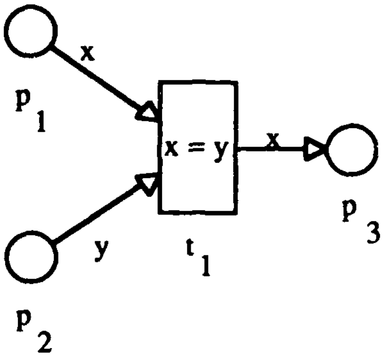


Fig. 2.6 Predicate Transition Net PN_2

Thus, the firing of transition t_1 requires that two tokens that are equal be present in the two input places of t_1. In this case, the function in t_1 acts as a guard over the tokens that are present in its input places. A second possible way of representing the same net is depicted in Figure 2.7.

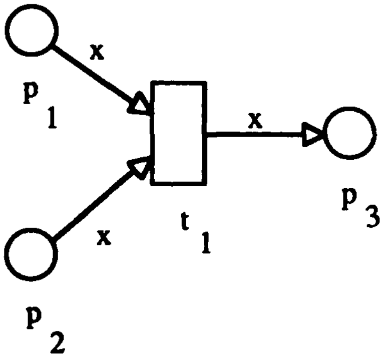


Fig. 2.7 Predicate Transition Net PN_2 - Alternative Representation

2.4 ORGANIZATION THEORY

2.4.1 Introduction

The framework of this study is the quantitative methodology (Levis, 1984) for the analysis and evaluation of alternative organizational structures. The type of information processing and decision-making organizations under consideration consists of groups of well-trained human decision-makers executing well-defined tasks and limited by the constraint of bounded rationality (Boettcher and Levis, 1982; 1983) . This notion refers to the limited ability of the human being to process information. Each decision-maker possesses a set of alternatives from which to choose in order to perform his task: he can choose a specific algorithm to process the information or he can decide to use a decision-aid (Chyen and Levis, 1985; Bejjani and Levis, 1985; Weingaertner and Levis, 1987). The organization exists in a hostile environment where the tempo of operations is fast. It implies that the time constraint has an important effect upon the functioning of the organization. Moreover, the different actions which may result from the decision-making process have various consequences expressed in terms of the cost for the organization.

The analysis of such decision-making organizations has been developed within the information-theoretic framework introduced by Shannon and Weaver (1949) and has been subsequently refined with the Petri Net formalism (Tabak and Levis, 1985; Jin, Levis and Remy, 1986; Hillion and Levis, 1987; Remy and Levis, 1987) .

Boettcher and Levis modeled first a simple one-person organization and, then, the case of a decision-maker interacting with an organization (Boettcher and Levis, 1982; 1983). The assumption that the decision-makers are memoryless was relaxed by Hall (1984). Some studies have addressed the integration of preprocessors and information storage in decision-making organizations (Chyen and Levis, 1985; Bejjani and Levis, 1985; Weingaertner and Levis, 1987).

Some measures of performance for organizations consisting of human beings - workload, accuracy, timeliness - have been defined and assessed in different studies (Jin et al., 1986; Hillion and Levis, 1987; Andreadakis and Levis, 1987) . The bounded rationality constraint has been modeled as a constraint on the total workload or amount of activity of a decision-maker (Boettcher and Levis, 1982; 1983). The computation of the different MOP's and the quantitative evaluation of the corresponding organizations have been carried out through the design and implementation of the CAESAR workstation which allows the user to specify the characteristics of an organization, compute the various MOP's and plot the results.

2.4.2 Model of the Interacting Decision-Maker

The Petri Net representation of the model of the decision-maker interacting with an organization is shown in Figure 2.8.

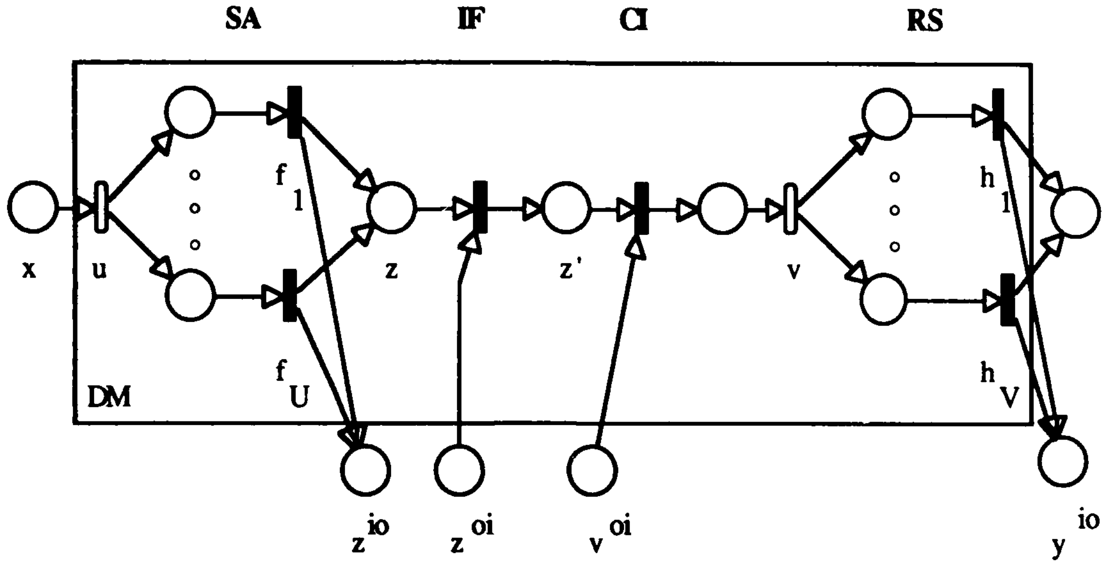


Fig. 2.8 The Interacting Decision-Maker

It consists of four stages:

(i) in the **Situation Assessment (SA)** stage, the decision-maker receives an input x from the environment and processes this information through the use of one of the U algorithms $f_1,...,f_U$. The choice of the particular algorithm to be used is done by the decision switch on the basis of a certain decision rule; this decision rule can take various forms, e.g., it can be a probability distribution prob(u) or a probability distribution prob(u|x) where the knowledge of x conditions the selection of the algorithm. The result of the SA stage is z; the decision-maker can also transmit some information z^{io} to the rest of the organization.

(ii) in the **Information Fusion (IF)** stage, the decision-maker can merge his own situation asessment z with some information z^{oi} communicated by some members of the rest of the organization. The protocol of interaction can take various forms depending on the task at hand and on the roles of the different decision-makers interacting in this stage. The resuit of this stage is z'.

(iii) in the **Command Interpretation (CI)** stage, the decision-maker can receive commands

v^{oi} from decision-makers hierarchically superior to him. These commands are used in the decision rule v on which the selection of a final response is based.

(iv) in the **Response Selection (RS)** stage, the decision-maker selects a response y by processing the input z' through one of the v algorithms $h_1,...,h_V$. The desired algorithm is chosen by the decision rule v. The decision-maker can send some information y^{io} to the rest of the organization.

The decision strategy of each decision-maker corresponds to the choice of the algorithms f_i and h_k. In a multi-person environment, the organizational strategy corresponds to the choice of the SA and RS algorithms for each decision-maker of the organization.

These notions are illustrated below for a two-person hierarchical organization:

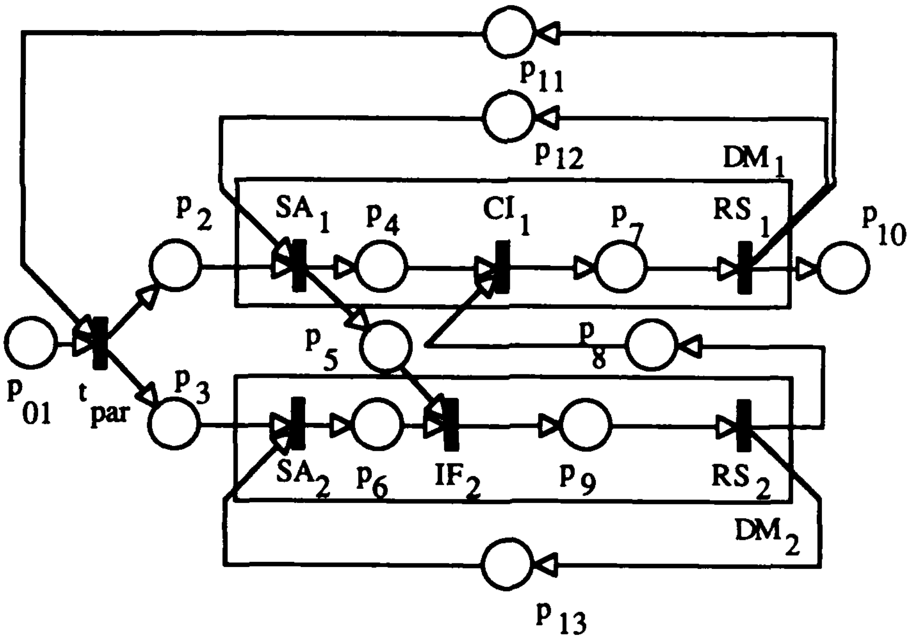


Fig. 2.9 Petri Net Model of Hierarchical Organization

The transition t_{par} models the partitioning of the input into two items of information sent to each decision-maker.

Two types of places are distinguished (Hillion and Levis, 1987; Remy and Levis,1987):

(i) memory places correspond to the places that carry information internally processed by each decision-maker. These places are both input and output places of the same decision-maker. They model his internal memory where the information has to be stored temporarily until the algorithm of the corresponding stage is available.

(ii) structural places carry the information exchanged between the decision-maker and the environment or other organizational members. They model the interactions between the organization and the environment and the interactions between the different decision-makers.

Furthermore, the limitation of resources that constrain the processing of information are modeled by adding resource places to the previous model.

In the example of Figure 2.9, the different places are separated in the following classes:

- memory places: p_4, p_6, p_7, p_9
- structural places: p_2, p_3, p_5, p_8
- resource places: p_{11}, p_{12}, p_{13}
- source: p_{01}
- sink: p_{10}

The resource place p_{14} models the limit on the number of inputs that the organization can handle at once; the resource places p_{12} and p_{13} model the limit on the number of inputs that each decision-maker can process at the same time, i.e., the bounded rationality constraint.

2.4.3 The Interacting Decision-Maker Assisted by a Decision Aid

The introduction of a decision-aid in the organization changes the strategies that can be used by each decision-maker who has access to it (Weingaertner and Levis, 1987). In general, for any input and any stage of his decision-making process, the decision-maker has three alternatives with respect to the use of the decision-aid:

- he does not use the decision-aid and performs the processing by himself. This alternative can be the result of a lack of trust in the decision-aid.
- he queries the decision-aid and relies totally on the answer. In this case, the decision-maker completely trusts the decision-aid.
- he queries the decision-aid and compares its answer to his own perception of the issue. This alternative represents an intermediate possibility when the decison-maker

has no definite opinion concerning the quality of the decision-aid.

The meta-decisions that the decision-maker has to make concerning the use of the decision-aid can increase significantly the number of alternatives that he can consider in order to process any input.

2.4.4 Accuracy, Timeliness, Workload

The decision-making organization DMO can be considered as a system which must perform a certain task. This task is modeled by a finite alphabet X where the input x takes its values with a discrete probability distribution prob(x) and a cost function c(x) for the organization. It implies that the real probability that the DMO receives the input x_i is prob($x=x_i$); moreover, c(x_i) is a function which assigns a cost to each organizational response to the input x_i: one possible way to define c(x) is to map x_i into an ideal response y_{di} and, then, to assign a cost c(y,y_{di}) to the discrepancy between the actual organization response y and the desired response y_{di}. The prob(x) accounts for the fact that all the inputs are not equally likely; in the same way, c(x) accounts for the fact that the processing of different inputs may have different utilities for the organization designer.

In this context, one can define different measures of performance (MOP's) in order to assess the effectiveness of the organization in performing its task :

(i) the accuracy of the response, or performance, J, which measures how well the organizational responses correspond to the desired responses (Andreadakis and Levis, 1987). J is defined for each organizational strategy; for each input x_i, the cost c(y_h,y_{di}) is computed. When the algorithms used in the strategy are deterministic, there is one response y_h provided to each input x_i; otherwise, several responses can be given to the same input x_i. The accuracy of the organization for the given strategy δ is then defined by:

$$J(\delta) = \sum_i \text{prob}(x_i) \sum_h c(y_h, y_{di})\, \text{prob}(y_h \mid x_i) \qquad (2.5)$$

(ii) the timeliness of the response,T, which measures the extent to which these responses are provided at the right instants. It is possible to define several measures of timeliness. It can be, for example, the probability that the response time lies within a certain interval $[T_{min}, T_{max}]$. It can be also the expected delay, which is defined as follows for each organizational strategy: for each input x_i, the time delay $r(x_i)$ for processing is computed. The expected delay is then determined by:

$$T = \sum_i \text{prob}(x_i)\ r(x_i) \tag{2.6}$$

(iii) the workload G^i of each decision-maker DM_i, i.e., his information processing activity in carrying out his task. It is obtained by computing the entropy of all the internal variables of DM_i in accordance with the Partition Law of Information (Conant, 1976).

The mission's requirements are expressed in terms of constraints on the value that these MOP's can take.

The bounded rationality constraint expresses that the workload G^i verifies:

$$G^i \leq F^i \tau \tag{2.7}$$

where F^i characterizes the maximum activity rate of decision-maker DM_i and τ, the mean input interarrival time. In the steady-state, τ is a constant for the task.

Another principle is that the decision-maker exhibits a satisficing and not optimizing behavior. It means that any performance better than a certain threshold J_0 is satisficing. In this case, the decision-maker adopts strategies that do not necessarily lead to the best performance but that are accurate enough and that do not lead to an overload. If the measure of accuracy J is computed as the expected value of the cost of the organizational response, then the satisficing behavior constraint is expressed by:

$$J \leq J_0 \tag{2.8}$$

The feasible organizational strategies are the strategies which satisfy these constraints. Therefore, a possible measure of effectiveness, the measure of consistency Q, computes in the strategy space the ratio of a measure on the set of feasible strategies over the same measure on the set of all organizational strategies.

2.5 CONCLUSION

This chapter has presented the analytical framework and evaluation tools that will be used in the remainder of this study. The investigation of coordination in decision-making organizations will be carried out by modeling processes that require coordination using the Predicate Transition Net formalism and by defining a new Measure of Performance for evaluating alternative organizational structures from this standpoint. Furthermore, a generic model of decision support system using the Petri Net framework will be proposed.The next chapter presents the concept of coordination in decision-making organizations.

CHAPTER III

COORDINATION IN DECISION-MAKING ORGANIZATIONS: ANALYSIS AND MODELING

This chapter investigates the concept of coordination in decision-making organizations and introduces a methodology to analyze and model the interactions of different decision-makers who constitute a coordinated unit. The concept of a team of decision-makers, and the issues of inconsistency of information and synchronization are emphasized. The Predicate Transition Nets formalism is used to model the information processing stages of decision-makers whose interactions require coordination.

3.1 THE CONCEPT OF COORDINATION

3.1.1 Introduction

From a conceptual standpoint, the idea of coordination in decision-making organizations embodies three classes of issues:

- the extent to which the decision-makers constitute a team.
- the synchronization of the activities during the decision-making process.
- the consistency of the information processed by the different members of the organization.

The underlying assumption of the concept of coordination is that the organization members have activities that are synergistic. It is necessary for the organization to combine in some way the results of the different processes that are implemented in order to perform its mission. This is the case when the organization must produce a unique response for each input originating from the environment and when this input is partitioned into groups of components that are assigned to different organization members. Each decision-maker processes information that he receives from the environment or from other decision-makers; he can also send information to other organization members and produce a response. The organizational response can be produced by a decision-maker alone or can result from the fusion of responses provided by different decision-makers.

The first issue is that different decision-makers may have different perceptions of the environment even though the mission requires that they interact in some way to produce the organizational response. Because of these different perceptions, they might adopt strategies that lead to a degradation of the effectivenesɛ ɔf the organization.

The second issue characterizes the dynamics of the decision-making process. The synchronization is a concept related to the value of information when the decision-makers actually process it.

The last issue is related to the fact that decision-makers do not necessarily exchange data that are consistent because they have different perceptions of the task and use different models or databases.

3.1.2 Definition of a Team

The different definitions introduced in Chapter II emphasized the commonality of goal , the commonality of interests and beliefs, and the interdependence of the team's members. The latter notion means that the actions or decisions of any one member of the organization impact upon the activities of the other members. Thus, to determine if a group is a team, one must first identify the goals or tasks to be accomplished, then compare the interests and beliefs of the members, and finally examine to what extent the members are interdependent, i.e., how great their need is to coordinate efforts to achieve goals.

Interdependence does not necessarily mean communication. Indeed, t'vo decision-makers may have tasks that are independent from each other and that are both necessary for the achievement of the mission: they do not need to communicate. However, their activities are coordinated from the organizational standpoint because each of them accomplishes only the task for which he has been trained. In this case, the coordination of activities is achieved through the design of a structure that partitions the tasks to be accomplished in accordance with the organizational objectives.

Definition 1: A team is an organization in which the members:

- have a common goal;
- have the same interests and same beliefs; and
- have activities that must be coordinated so as to achieve a higher

effectiveness.

The overall team's performance will be, in general, greater than the sum of the individuals' performance. This synergy is the result of coordination of the decision-makers' activities, which is achieved through the coordination of skills and the coordination of information. It is possible to conceptualize the team's organization as in Figure 3.1.

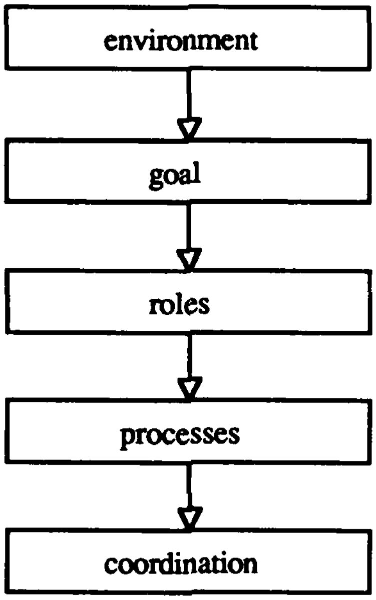


Fig. 3.1 Concept of a Team

The coordination of the team's members is influenced by the environment, the goal, the roles, and the processes:

- the state of the environment has an effect upon the definition of the goal.
- the goal determines the roles in the team, i.e., who does what, given the skills of the various members.
- the roles influence the definition of the processes, i.e., the functioning of the team.
- the processes determine the coordination between the team's members, i.e., the quality of their interactions.

In this context, the coordination of the team's members can be modified by acting on one or several of the layers defined above. It means that by altering the state of the environment, by refining the definition of the goal, by changing the roles of the members or by modifying the processes of interaction, the designer can improve the cohesiveness of the organization.

The design of an organizational structure with appropriate roles and processes can allow for increased coordination of the activities of the decision-makers. For example, the definition of protocols of interactions that require from decision-makers that they check whether or not they speak of the same threat will increase the quality of their interactions.

3.1.3 Synchronization

The concept of synchronization in decision-making organizations is related to the fact that some of the activities that take place during the decision-making process are asynchronous and concurrent. Therefore, the various decision makers work in parallel and interact at instants that are not predetermined. The protocols of interaction can take various forms: for example, decision-maker DM_1 may be requested to always wait for the command from DM_2 before proceeding with any further action; on the other hand, DM_3 may be allowed to decide whether or not he will wait for this command depending on the context of operations.

In the context of decision-making organizations, synchronization is an important concept because the processing of information introduces three types of biases:

- biases due to the uncertainty embodied in the information processed.
- biases due to the models used.
- biases due to the value of the information when the decision-maker actually processes it.

Biases of the first type occur when a decision-maker does not have all the correct information concerning the input that he must process. Then, for instance, he might choose the wrong protocols when he performs his task. This situation occurs when the information that he receives contains some noise or represents only partially the state of the environment. For example, the information transmitted by a sensor to a decision-maker is noisy and accounts only for a limited aspect of the state of the environment.

Biases of the second type originate from the fact that the protocols implemented to process the information use models and, therefore, only approximate reality. For example, the computation of the speed of a warship through some algorithm introduces necessarily a certain error.

Synchronization is related to biases of the third type. These biases are due to the fact that, if a piece of information remains for a long time in memory, the decision-maker might well attach less value to it when he actually processes it. Therefore, the quality of his processing might degrade.

In particular, if a decision-maker cannot process a piece of information because he must wait to receive some information from other organization members in order to proceed, the value of this information might decrease if the environment changes or if other tasks must be performed. This effect might occur because the decision-maker does not know when he will receive the necessary information.Therefore, it appears that the interactions between different organization members can be the cause of these biases. Consequently, if the organization members never wait for the information that they need in order to perform the processing of the information that is in memory, no bias of the third type will be introduced in the decision-making process.

These considerations motivate the following definition:

Definition 2: An organization is perfectly synchronized, for the whole decision-making process, when the decision-makers do not have to wait to receive the information that they need in order to continue processing the information that is in memory.

The synchronization degrades when the processing of a certain input leads one decision-maker to wait to receive the information that he needs in order to continue the processing of the item of information that he knows about the corresponding input.

However, perfect synchronization of the organization does not imply necessarily that the delay for the processing of one input will be low. Two decision-makers can be perfectly synchronized but very slow; they can also be not perfectly synchronized but very fast. In the same way, good synchronization will not ensure that the organizational response will be perfectly accurate: indeed, in order to be well synchronized, one decision-maker might use algorithms which introduce important biases in his response because he wants to be faster in order to keep up with another decision-maker; a decision-maker can also take more time to do his processing in order to provide better response but, then, either his message might be of no value to another decision-maker because it arrived too late or the organizational response might be not timely. Thus, the issue of synchronization has important ramifications; its relationship to the effectiveness of the organization must be described.

The concept of synchronization characterizes, from a certain standpoint, the timeliness of the organization. It provides some information on the evolution of the decision-making process and on the dynamics of the interactions. As it will be demonstrated subsequently, the following have an effect upon the synchronization:

- the organizational structure and protocols that define the information flow path and the internal interactions.
- the strategies adopted by the various decision-makers for the processing of information.

Thus, the characterization of decision-making organizations on the basis of their synchronization must take into account these two sets of factors. It implies that the designer should be able to evaluate the impact of modifying any of these factors on the synchronization of the organization. The framework of the methodology introduced in Chapter II is suitable for this type of evaluation. In consequence, a new Measure of Performance (MOP) will be introduced in section 3.2.

3.1.4 Consistency of Information

When a decision-maker receives some data from another decision-maker or from a database, it may be the case that these data are inconsistent with what he knows or with what he receives from another part of the organization. In particular, this can be due to the fact that he receives data that have different geographical or temporal origins, e.g., different databases or the same database accessed at different instants, or data that have been sent by decision-makers with different perceptions of the environment.

Consistency of information shows the extent to which different items of information can be fused together without contradiction. Therefore, it is essentially mission dependent. One can evaluate the extent to which two data are inconsistent if the subsequent processing or actions that will take place for each item of information are not the same. Thus, one cannot speak of the inconsistency of information in general, but must refer to the context and the mission at hand. For example, in the case of an air defense task, the fact that one decision-maker identifies an object as being a missile and communicates this information to a decision-maker who has identified it as being a war plane does not imply that their interaction is inconsistent. To the extent that missiles and war planes are handled in the same manner, it is perfectly consistent. However, if a specific course of action applies to missiles and another set of actions to war planes, the inconsistency must be overcome. Decision-aids, such as expert systems, can be used to resolve these conflicts (Perdu, 1987).

3.2 A CHARACTERISTIC OF TEAMS

The definition of the concept of a team introduced the notion that the members of the team must have the same interests and same beliefs. It does not imply that they have the same knowledge of the state of the environment and of the task at hand. Intuitively, the commonality of interests and beliefs can be understood as follows: if decision-makers DM_i and DM_k had the same capabilities, each of them would be able to carry out the role of the other without degrading or improving the effectiveness of the organization.

The organization's task is defined as the processing of inputs x_r to produce output symbols. The inputs x_r belong to the alphabet of inputs X. The probability of having a certain input x_r is defined by the probability distribution prob($x=x_r$). Furthermore, the desired response, y_{dr}, for any given input x_r is known by the organization's designer. Therefore, a cost is associated with the production of a response which is not desired. This cost is denoted by the cost function c(x) . For each input x_r, $c(x_r)$ is a function associating with each possible response y_h a certain cost which depends on y_h and on the desired response, y_{dr}, that should be provided for x_r. The cost is denoted by $c(y_h, y_{dr})$. It is zero when the response y_h is desired.

The model of the interacting decision-maker with bounded rationality presented in Chapter II shows that a decision-maker DM_i is characterized by several parameters related to the task that he must perfom and to his own capabilities. These parameters are:

(i) the algorithms that he can use.

(ii) the activity rate F^i.

(iii) the perception of the task :

- $\mathrm{prob}_i(x)$ accounts for the beliefs of DM_i; for each element x_r of the alphabet of inputs X, $\mathrm{prob}_i(x=x_r)$ is the probability which DM_i associates to the input x_r.
- $c_i(x)$ accounts for the interests of DM_i; for each element x_r of the alphabet of inputs X, $c_i(x_r)$ represents the cost function for x_r as perceived by DM_i.

(iv) the strategies that the decision-maker can adopt to perform his processing.

The design of an organization can be a complicated problem when the decision-makers are numerous and do not constitute a homogeneous team. Therefore, it is not enough to request that the decision-makers have the same interests and beliefs; one must also request that they account well for the organizational objectives.

Certain decision-makers can have different interests and assign some high value to the processing of input x_r whereas, from the organization's standpoint or from other decision-makers' standpoint, x_r is not as valuable. In the same way, the decision-makers can have different beliefs concerning the state of the environment and the task to perform and, thus, assign different probabilities to x_r. This leads to the conclusion that each decision-maker DM_i has his own performance function J_i for the organization defined on the basis of his interests and beliefs. For each organizational strategy δ:

$$J_i(\delta) = \sum_r \mathrm{prob}_i(x_r) \sum_h c_i(y_h, y_{dr})\, \mathrm{prob}(y_h \mid x_r) \qquad (3.1)$$

where $\mathrm{prob}(y_h \mid x_r)$ denotes the probability of providing the response y_h for the input x_r.

During his processing of information, decision-maker DM_i makes his decisions on the basis of his interests and beliefs. It does not mean that he computes his index of performance J_i since he does not know what will be the outcome of the overall decision-making process; however, if he were measuring the accuracy of the organization as the organization's designer does it, he would assign the value J_i to the organization's accuracy measure.

In this context, the concept of a team takes a particular meaning. The ideal team consists of people who have a common goal, and who have the same interests and beliefs.The commonality of goal expresses that all of them want the organization to be accurate.

Furthermore, suppose that the organization consists of n decision-makers $DM_1,...,DM_n$ whose perceptions are $((prob_1(x),c_1(x)),..., (prob_n(x),c_n(x)))$. They are a team if

$$\forall (i, j) \in \{1,..., n\}\times\{1,...,n\}, \text{prob}_i(x) = \text{prob}_j(x), \; c_i(x) = c_j(x) \qquad (3.2)$$

The team's perception of the task is denoted by $(prob_t(x), c_t(x))$. The common $prob_t(x)$ means that all the team's members have the same beliefs; the common $c_t(x)$ implies that they have the same interests. Thus, if the decision-makers constitute a team, we also have:

$$\forall (i, j) \in \{1,..., n\}\times\{1,..., n\} \quad J_i = J_j = J_t \qquad (3.3)$$

where J_t represents the team's performance index defined, for each organizational strategy δ, by:

$$J_t(\delta) = \sum_r \text{prob}_t(x_r) \sum_h c_t(y_h, y_{dr}) \, \text{prob}(y_h \mid x_r) \qquad (3.4)$$

If the designer can design a team which has interests and beliefs that match exactly the organizational objectives, then the team's effectiveness will be higher. If we assume that the real probability distribution for the task x is prob(x) and the cost function for the organization is c(x), the team's perception of the task will be optimal from the organization's standpoint when $prob_t(x)$ is equal to prob(x) and $c_t(x)$ is equal to c(x), so that J_t is equal to J.

3.3 PREDICATE TRANSITION NET MODEL OF COORDINATION

3.3.1 Introduction

The organizations under consideration consist of groups of decision-makers processing information originating from a single source and who interact to produce a unique organizational response for each input that is processed. In terms of Petri Nets, it implies that there exists a source place, p_{so}, and a sink place, p_{sk}: the source place models the source from where the inputs originate; the sink place models the place where the organizational response appears at the end of the process. A resource place, p_{rs}, is introduced to model the limited organizational resources.

A transition t_{par} models the partitioning of the inputs. Furthermore, if several decision-makers provide responses that must be fused in order to obtain the organizational response, this stage of response fusion is modeled by the transition t_{rf} (see Figure 3.2).

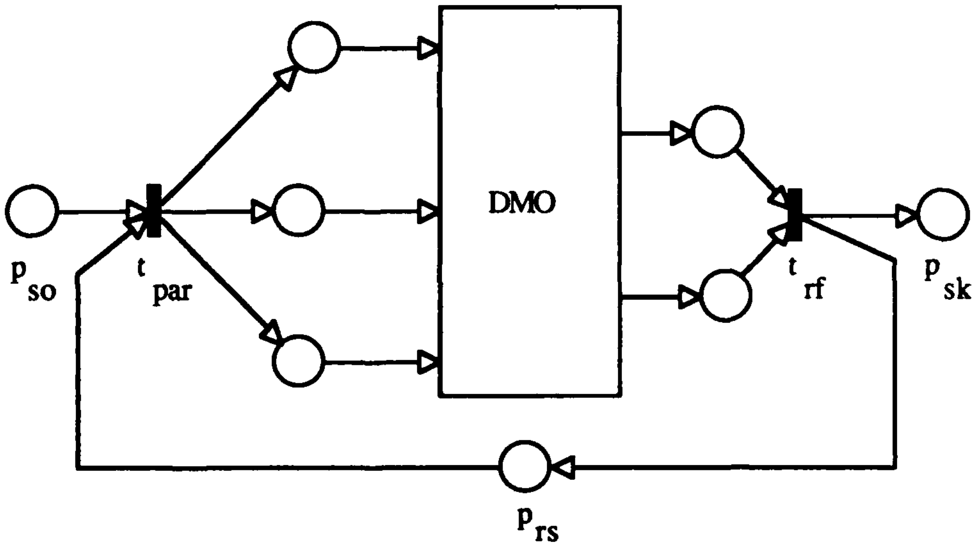


Fig. 3.2 Petri Net PN: Interactions DMO - Environment

The source generates inputs that arrive sequentially and one at a time. It implies that each event is uniquely identified by the time at which it was detected by the sensors. In the Petri Net representation, it corresponds to the time of appearance of a token in the source place p_{so}. Furthermore, it is assumed that the different decision-makers have some means of recognizing that, when they interact, they are referring to the same input: for example, their protocols of interaction can require that they communicate the time at which the input they are processing entered the organization.

3.3.2 Definitions and Notation

This section recalls some basic definitions and notation that are used in the model of coordination introduced in this study.

The task is modeled by the alphabet $\mathcal{X} = \{x_1,...,x_n\}$. A probability distribution is defined on $\mathcal{X}$. The probability that the input x is equal to x_i, i.e., prob($x=x_i$), is denoted by prob(x_i). The set of subsets of $\mathcal{X}$ is denoted by $\Pi(\mathcal{X})$; then:

$$\Pi^*(\mathcal{X}) = \Pi(\mathcal{X}) - \{\emptyset\} \tag{3.5}$$

where $\emptyset$ denotes the empty set.

The organization is modeled by a Petri Net, PN, which consists of places and transitions:

- the set of places of PN is denoted by $\mathcal{P}$.
- the set of transitions of PN is denoted by $\mathcal{T}$.

The set of places, $\mathcal{P}$, can be partitioned into three classes:

- $\mathcal{M}$ denotes the set of memory places of PN.
- $\mathcal{S}$ denotes the set of structural places of PN.
- $\mathcal{W}$ denotes the set of resource places of PN.

In the same way, the set of transitions, $\mathcal{T}$, can be partitioned into three classes:

- $\mathcal{A}$ denotes the set of interactional transitions, i.e., transitions which model interactions between different decision-makers. These transitions are such that their preset contains at least one structural place.
- $\mathcal{H}$ denotes the set of internal transitions, i.e., transitions which model algorithms performed by decision-makers with no interaction with any other organization member. These transitions are such that their preset contains one, and only one, place that is not a resource place: it is a memory place.
- $\mathcal{E}$ denotes the set of external transitions, i.e., transitions that model processes not performed by any decision-maker, e.g., t_{par} or t_{rf}.

A clock is assumed to exist. This clock is used to follow the instants at which the process is observed. In accordance with the formalism of Timed Petri Nets, this clock provides non-negative rational numbers. Therefore, the current time is denoted by T_c, such that $T_c \in \mathbf{Q}^+$ where $\mathbf{Q}^+$ denotes the set of non-negative rational numbers.

3.3.3 Distinguishability of Tokens

The fundamental assumption of the model is that a decision-maker can process only one input at a time in any of his internal stages: it follows that if the decision-maker is already processing an input in one of his stages, any other input that is ready to be processed by the same stage waits in memory.

The Petri Net model of decision-maker presented in Chapter II does not provide any means of distinguishing tokens in the net. It implies that when several tokens are in any particular place, they are considered as being all equivalent since they are at the same instant in the same place. In other words, the examination of the net at this instant does not provide any possibility of identifying any token's identity and history in the organization.

This model is valid as long as one can consider that no queue will form in the system. This is achieved under two circumstances:

- one item of information is processed by the organization during each time interval.
- the organization can process several items of information at once, but does it in such a way that there is always at most one item of information waiting to be processed in any stage. This is achieved, for example, when inputs have a constant interarrival time and all the stages have a processing time less than this interarrival time.

If one wants to analyze and model information-processing and decision-making organizations in a more general framework, one needs to be able to give attributes to the tokens so that they will be distinguishable. The formalism of Predicate Transition Nets is suitable for this kind of modeling.

The basic assumption of the model is that, at any internal stage of the decision-making process, a decision-maker can discriminate between different items of information on the basis of three characteristics:

- the time T_n at which the inputs that these items of information represent entered the organization.
- the time T_d at which the item of information entered the internal stage where it is currently.
- the class C associated with any item of information by the previous processing stage.

The definition of the attributes T_n, T_d and C derives from the following considerations:

(i) the task is modeled such that inputs originate from a single source, one at a time. For each token with an identity, i.e., each information carrier, the attribute T_n corresponds to the time at which the input represented by this token entered the organization. In accordance with the formalism of Timed Petri Net the first attribute T_n is a non-negative rational number, i.e., an element of $\mathbf{Q}^+$.

(ii) the firing of any token in the net takes an amount of time that is known since it characterizes the processing time of the corresponding transition. It is thus possible to identify any token in a place p with the time T_d at which it entered this place. The second attribute T_d is also a non-negative rational number, i.e., an element of $\mathbf{Q}^+$.

(iii)the task is modeled by the alphabet $\mathcal{X} = \{x_1,...,x_n\}$. It is assumed that each structural or memory place p is associated with a partitioning $\mathfrak{D}(p)$ of this alphabet. The number of elements of this partitioning is denoted by e(p). This partitioning is such that $\mathfrak{D}(p) = \{D(p,1),..., D(p,e(p))\}$ where D(p,i) denotes an element of $\Pi^*(\mathcal{X})$. Thus, the third attribute C of each token belongs to a certain partitioning $\mathfrak{D}(p)$ of $\mathcal{X}$, this partitioning depending on the place p where the token stands.

On the other hand, the different resources that the organization has are not distinguishable because it is assumed, in this study, that any organizational resource can be used to process any input. This might not be the case when organizational resources are allocated to different inputs in accordance with some doctrine. In the same way, the resources that represent the decision-makers' processing capacities are not distinguishable.

Following the definition of Predicate Transition Nets provided in Chapter II, two variables χ and ϕ are introduced.

<u>Definition 3</u>: The variable ϕ takes its values in the set Φ such that $\Phi = \{¢\}$. All the tokens with no identity are denoted by the color ¢.
The variable χ takes its values in the set X such that $X = \mathbf{Q}^+ \times \mathbf{Q}^+ \times \Pi^*(\mathcal{X})$. Each element of X is a color that is represented by (T_n, T_d, C). A token with an identity is an individual that is assigned a color.

Furthermore, memory and structural places contain tokens that have an identity since they model information carriers. Resource places contain tokens with no identity since they model resources. Thus, each place is associated with one of the variables χ or ϕ.

<u>Definition 4</u>: The application v associates with each place, p, one of the variables χ or ϕ.

$$\begin{aligned} v:\ \mathcal{P} &\to \{\chi, \phi\} \\ p &\to v(p) \end{aligned} \qquad (3.6)$$

The following relations hold:

$$\forall\, p \in \mathcal{M} \cup \mathcal{S},\quad v(p) = \chi \qquad (3.7)$$

$$\forall\, p \in \mathcal{W},\quad v(p) = \phi \qquad (3.8)$$

For any place p, $\underline{v}(p)$ denotes the set in which v(p) takes its values. Thus, if p is a resource place, $\underline{v}(p)$ is equal to Φ. If p is a memory or structural place, $\underline{v}(p)$ is equal to X.

Each place can contain a certain number of tokens. In the case of resource places, all these tokens have the color ¢. In the case of memory or structural places, each of these tokens has a color: a color can correspond to several tokens in a same place or to several tokens in the entire Petri Net PN. The marking of PN is defined as follows:

<u>Definition 5</u>: For each place p, $\mathcal{Z}(p)$ designates the set of applications from $\underline{v}(p)$ into $\mathcal{N}$ where $\mathcal{N}$ denotes the set of non-negative integer numbers. $\mathcal{Z}_n$ is the set of $\mathcal{Z}(p)$ for all p.
The marking M of the Petri Net PN is defined as an application from the set of places, $\mathcal{P}$, into the set $\mathcal{Z}_n$.

$$\begin{aligned} M:\ \mathcal{P} &\to \mathcal{Z}_n \\ p &\to M(p) \end{aligned} \qquad (3.9)$$

Therefore, for each place p, M(p) is an application from $\underline{v}(p)$ into $\mathcal{N}$. It assigns to each value of the variable associated with p a non-negative integer number. It represents the number of tokens in the place that have the corresponding color. If m designates a certain color, M(p)[m] will denote this number.

Since each color m corresponds to a 3-tuple (T_n, T_d, C), this number will be also denoted by M(p)[(T_n, T_d, C)]. In the case of a resource place, the tokens can have only the color ¢. M(p)[¢] will be denoted by M(p).

The following example (Figure 3.3) illustrates these definitions:

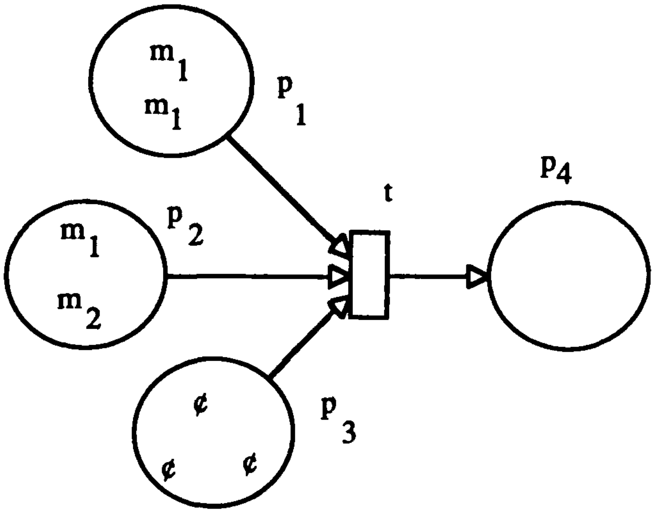


Fig. 3.3 Example of Marking

In this example, the following relations hold:

- $p_1 \in \mathcal{S}$, $p_2 \in \mathcal{M}$, $p_3 \in \mathcal{W}$, $p_4 \in \mathcal{M}$ and $t \in \mathcal{A}$.
- $m_1 \in X$, $m_2 \in X$.
- $v(p_1) = \chi$, $v(p_2) = \chi$, $v(p_3) = \phi$, $v(p_4) = \chi$.
- $M(p_1)[m_1] = 2$; $\forall\ m \in X$-$\{m_1\}$, $M(p_1)[m] = 0$.
- $M(p_2)[m_1] = 1$; $M(p_2)[m_2] = 1$; $\forall\ m \in X$-$\{m_1, m_2\}$, $M(p_2)[m] = 0$.
- $M(p_3) = 3$.
- $\forall\ m \in X$, $M(p_4)[m] = 0$.

Some properties of the model can be derived from the characteristics of the organizations under consideration in this study.

Hillion (1987) has shown that the Petri Nets that model the type of organizations under consideration in this study are Event-Graphs, i.e., Petri Nets where each place has only one input transition and one output transition. Furthermore, if one considers the Petri Net PN_d which corresponds to the Petri Net PN, as shown in Figure 3.2, from which the source place and the sink place are deleted, this net is strongly connected, i.e., there exists a directed path from any node to any other node. Since an Event-Graph is bounded if and only if it is strongly connected (Hillion and Levis, 1987), the Petri Net PN_d is bounded.

Property 1: The number of tokens in the Petri Net PN_d remains bounded.

The number of tokens in the net is finite and it is possible to observe the state of the process by discretizing it in units of time and by examining the tokens that are in the places.

The firing of a transition t, as illustrated in Figure 3.4, is characterized by the following:

- if p' is a resource place, m' is the color ¢.
- if p and p' are memory or structural places, m and m' are elements of X. Therefore, there exist two 3-tuples (T_n, T_d, C) and (T_n', T_d', C') such that m is equal to (T_n, T_d, C) and m' is equal to (T_n', T_d', C').

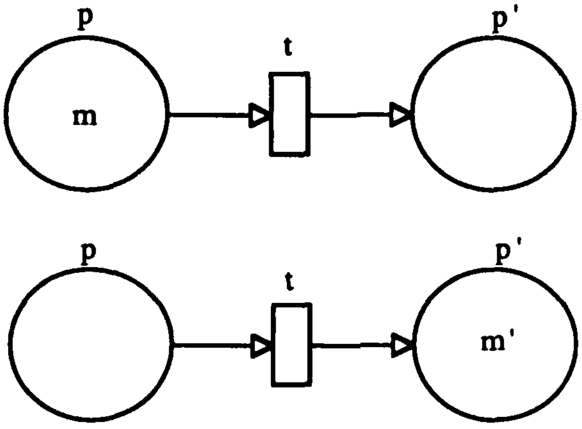


Fig. 3.4 Firing of a Transition

If M denotes the marking of PN before t fires and M', the marking of PN after t has fired, it is possible to write:

$$p \in \mathcal{W} \Rightarrow M'(p) = M(p) - 1 \qquad (3.10)$$

$$p' \in \mathcal{W} \Rightarrow M'(p') = M(p') + 1 \qquad (3.11)$$

$$p \in \mathcal{S} \cup \mathcal{M} \Rightarrow M'(p)[m] = M(p)[m] - 1 \qquad (3.12)$$

$$p' \in \mathcal{S} \cup \mathcal{M} \Rightarrow M'(p')[m'] = M(p')[m'] + 1 \qquad (3.13)$$

Moreover, if f(t) denotes the firing time of transition t, the following relations hold:

$$T_n' = T_n \qquad (3.14)$$

$$T_d' = T_d + f(t) \qquad (3.15)$$

The relation (3.14) derives from the fact that the firing of a token from a structural or memory place does not modify the time at which the input represented by this token entered the net.

Property 2: The firing of a transition does not modify the attribute T_n.

The attribute T_n characterizes one and only one input since the source generates one input at a time. Furthermore, two representatives of the same input cannot stand in the same place. Indeed, the net is an Event-Graph and, so, each place has only one input transition which produces in each of its output place only one token per firing.

Property 3: A place cannot contain two tokens which have the same attribute T_n.

In the same way, since each place p has only one input transition t, p can contain two tokens with the same attribute T_d only if the firing time f(t) of the transition t is zero. Indeed, if the firing time f(t) is different from zero, two firings of t cannot end at the same instant.

Property 4: A place p can contain several tokens with the same attribute T_d only if the firing time f(t) of its input transition is zero.

It results from Property 4 that two tokens in any structural or memory place cannot have the same color m. Indeed, if they had the same color, the attribute T_n would be the same for both. It follows that there is a one-to-one correspondence between tokens and their color in any structural and memory place. In the remainder of this study, a token of color m will be also called token m.

The grammar of execution will be developed later in this chapter. Some precise statements must be made concerning the attribute C. In the context of decision-making organizations, each decision-maker is assigned by the organization's designer specific algorithms that he can use to process the information. By using these algorithms, at each stage of his internal process, the decision-maker associates a class of inputs to the information he is handling.These classes are defined by the organization's designer so that the processing in the next stage of all the items of information that are associated with a class is identical. Consequently, each decision-maker is assigned a finite set of classes for each stage of his process: when processing an input in a stage, he must associate a class to it.

A set of classes for a given decision-maker and a given stage corresponds, as introduced in this section, to a partitioning $\mathfrak{D}(p)$ of $\mathcal{X}$ into e(p) classes. However, before the situation assessment stage, the decision-maker has not assessed the information yet and, in this case, the attribute C can be defined more precisely.

Two cases must be considered:

- the decision-maker receives data that have not been preprocessed.
- the decision-maker receives data preprocessed by a decision-aid.

The first case is illustrated in Figure 3.5 (a).

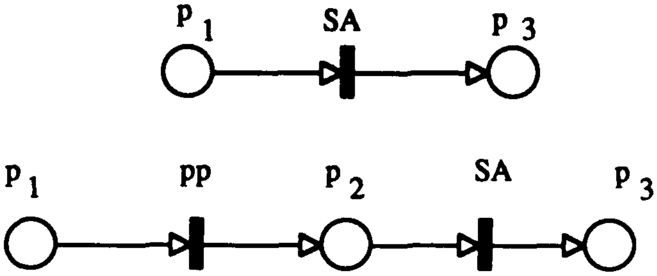


Fig. 3.5 Situation Assessment (a) with no Preprocessor and (b) with Preprocessor

The tokens that stand in place p_1 are the representations of different inputs that entered the organization at different instants in time. Thus, each one characterized by a unique T_n. All these tokens are also characterized by the time T_d at which they entered the place p_1.

As far as the third attribute C is concerned, the situation is different. This attribute is the same for all of them and corresponds to the alphabet $\mathcal{X}$ itself. Indeed, the decision-maker has not processed the information yet and, therefore, we must consider at this stage that the token can be a representation of any possible input.

A preprocessor can be inserted between the source and the decision-maker as illustrated in Figure 3.5 (b).

As previously, the tokens in the place p_1 are uniquely defined by the attributes T_n and T_d; the attribute C is the same for all of them and corresponds to the alphabet $\mathcal{X}$. Now, if the preprocessor aggregates the inputs in zones of indifference (Chyen and Levis, 1985), the attribute C of a token in place p_2 will correspond to the zone associated with the input from which the token originates. Since this zone of indifference depends on the input considered, the tokens in place p_2 might not have the same attributes C.

3.3.4 Protocols of Interaction

The transitions that model protocols executed by organization members are interactional transitions and internal transitions. Indeed, the external transitions represent processes executed by components outside from the decision-making organization, as shown in Figure 3.2. This section defines rules that apply for all transitions, be they interactional, internal, or external.

The present model considers that decision-makers can interact only in the Situation Assessment stage, Information Fusion stage and Command Interpretation stage. However, fusion of different pieces of information can also occur in the Response Fusion stage, represented by t_{rf} in Figure 3.2. The original model of interactions between different decision-makers must be refined to account for the facts that tokens are distinguishable. In order to do so, a characterization of the set of input places of each transition according to its type, i.e., interactional , internal or external, is provided.

(i) if t denotes an interactional transition, i.e., $t \in \mathcal{A}$, the set of input places of t contains at least one structural place. Thus:

$$\forall\, t \in \mathcal{A} \,,\; \exists\, p \in \mathrm{Pre}(t) \,,\; p \in \mathcal{S} \tag{3.16}$$

(ii) if t denotes an internal transition, i.e., $t \in \mathcal{H}$, the set of input places of t contains one and only one memory place and no structural place. Thus:

$$\forall\, t \in \mathcal{H} \,,\; \exists!\, p \in \mathrm{Pre}(t) \,,\; p \in \mathcal{M} \tag{3.17}$$

$$\forall\, t \in \mathcal{H} \,,\; \forall\, p \in \mathrm{Pre}(t) \,,\; p \notin \mathcal{S} \tag{3.18}$$

where ! denotes the uniqueness.

(iii) if t denotes an external transition, i.e., $t \in \mathcal{E}$, the set of input places of t contains no memory place. Thus:

$$\forall\, t \in \mathcal{E} \,,\; \forall\, p \in \mathrm{Pre}(t) \,,\; p \notin \mathcal{M} \tag{3.19}$$

In addition to these considerations, one must recall that the set of input places of any transition t can contain a resource place. The rule according to which the resource place must contain at least a token in order for t to be enabled will apply. However, since resource places do not constrain the rule of enablement of a transition, but by requiring the presence of a token, the discussion on enablement that follows focuses on structural and memory places.

If t denotes a transition modeling a stage where no fusion of data is performed, the set of input places contains only one place that is not a resource place. If the transition is an external transition, e.g., t_{par}, the place is strutural ; if the transition is internal, the place is a memory place; if the transition is interactional, the place is structural. The rule of enablement of such transitions will be that this place contains at least a token.

The Petri Net model of transitions where fusion of data is done is shown in Figure 3.6.

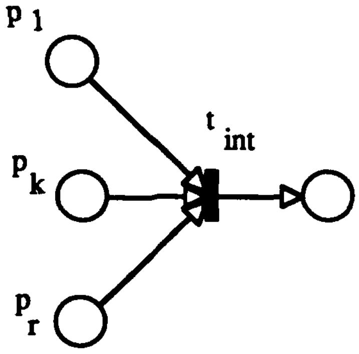


Fig. 3.6 Petri Net Model of Interaction with Fusion of Data

In the case of Information Fusion and Command Interpretation, t_{int} is an interactional transition because this fusion is performed by the decision-maker who receives the information from other decision-makers. Only one of the places $p_1,..., p_r$ is a memory place. We denote it by p_k . In the case of Response Fusion, all these places are structural. In accordance with relation (3.6), the variable associated with all these places is χ and the colors of the tokens that they contain belong to X.

Any rule of enablement can be introduced at this point. It can take explicitly into account the attributes of the tokens in the input places, but this is not necessary. For example, here are three possible rules. M denotes the marking of the net:

- rule 1: t_{int} is enabled, if and only if there exists a token in all input places:

$$\forall\, p \in \mathrm{Pre}(t_{int})\ ,\ \exists\ m \in X\ ,\ M(p)[m] \neq 0 \qquad (3.20)$$

- rule 2: t_{int} is enabled, if and only if all its input places contain a token with the same value of the attribute T_n:

$$\forall\, p \in \mathrm{Pre}(t_{int})\ , \exists\ (T_n, T_d, C) \in Q^{+} \times Q^{+} \times \Pi^{*}(\chi)\ ,\ M(p)[(T_n, T_d, C)] \neq 0,$$

$$\forall\, p' \in \mathrm{Pre}(t_{int})\ ,\ p' \neq p\ , \exists\ (T_n', T_d'\ ,\ C') \in Q^{+} \times Q^{+} \times \Pi^{*}(\chi)\ , M(p')[(T_n', T_d', C')] \neq 0,$$

$$T_n = T_n' \qquad (3.21)$$

- rule 3: interactional transition t_{int} is enabled if and only if rule 2 applies or there exists a token in the memory place p_k which has been in it for more than d units of time.

$$\text{(rule 2) or}$$

$$(\exists (T_n, T_d, C) \in \overset{+}{\mathbf{Q}} \times \overset{+}{\mathbf{Q}} \times \Pi^*(\mathsf{X}), M(p_k)[(T_n, T_d, C)] \neq 0, (T_c - T_d) \leq d) \quad (3.22)$$

Rule 1 is the usual rule of enablement in ordinary Petri Nets. It is, however, not realistic, since it does not take into account the fact that, when two decision-makers exchange information, they check that they are referring the same thing.

Rule 2 means that the transition t_{int} is enabled if and only if all the places of its preset contain at least a representation of the same input. Indeed, it results from the fact that memory and structural places contain only tokens of the (T_n, T_l, C) type, and that tokens having the same attribute T_n represent the same input. From the organizational standpoint, it means that, when decision-makers interact, they must refer to the same input. Rule 2 will hold in the remainder of the study because it is the kind of rule that applies most often for command and control organizations. This is consistent with what happens in most situations where decision-makers have some protocol to recognize that they are exchanging information pertaining to the same event in the environment.

Rule 3 means that rule 2 applies when a token has been in the memory place for less than d units of time. This rule models the interactions where decision-makers wait for information from other parts of the organization only for a certain amount of time. Rules of this category can be implemented when the organization has to cope with an environment where the tempo of operations is fast. In this case, certain decision-makers may have the possibility to wait for relevant data from other members of the organization only for a limited period. After then, they must continue their processing.

In the remainder of this study, the following definition will hold:

Definition 6: A transition is enabled if and only if rule 2 is verified.

The representation of such a rule is illustrated in Figure 3.7.

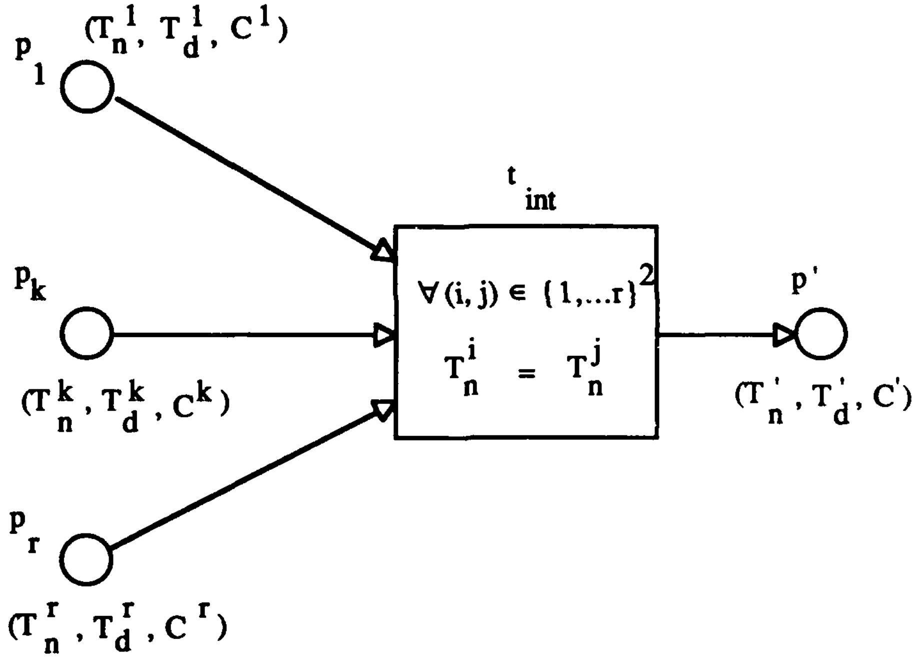


Fig. 3.7 Predicate Transition Net Model of Interaction

This definition holds also for all transitions, i.e., interactional , external and internal. In the case of internal transitions, rule 2 is always verified when all its input places have a token since, in this case, the preset contains only one place that is not a resource place.

Rule 2 of enablement can be expressed formally as:

$$\forall (i,j) \in \{1,\dots,r\}\times\{1,\dots,r\}, \quad T_n^i = T_n^j \tag{3.23}$$

It means that the attributes T_n^i of the colors $m_1, \dots, m_r$ must have the same value.

When a transition fires, it produces tokens in all its output places. The attributes of these tokens are set according to the following rules:

(i) if the place is a resource place, a token ¢ is inserted in it.

(ii) if the output place is a structural or memory place:

- its attribute T_n' corresponds to the attribute T_n of the tokens that were fired. By definition of the grammar, all these tokens have the same T_n.
- its attribute T_d' corresponds to the time at which the firing is terminated.
- its attribute C' depends on the algorithm implemented by the transition and the set of classes associated with this stage.

Thus, an operator OP_t is assigned to each transition t of the net . If we assume that r is the number of input places of t that are not resource places, this operator can be defined formally as follows:

Definition 7: OP_t is a mapping such that :

$$\begin{aligned} OP_t :\ & X^r \times Q^+ \times Pos(t) \text{ -> } X \\ & (m_1, \ldots, m_r, T_c, p) \text{ -> } m' \end{aligned} \qquad (3.24)$$

The operator assigns to each output place p of t a token of color m', m' being determined on the basis of:

- the colors $m_1, \ldots, m_r$ of the r tokens enabling t.
- the time T_c
- the algorithm performed by t.
- the classes of inputs associated to the place p.

3.3.5 Token selection

The fact that queues can build in the system implies that, at any stage of the decision-making process, the decision-maker must apply some rules to decide what set of data he will process next.

In the previous models, the problem of token selection did not arise since the tokens were not distinguishable, and, thus, the firing of a transition could operate on any of the tokens present in the input places.

When tokens are distinguishable, rules of selection must be applied to select the tokens that will be fired for any firing of a transition t. These rules operate on the tokens of the input places that enable the transition t. This is illustrated by the example of Figure 3.8 where rule 2 of enablement applies.

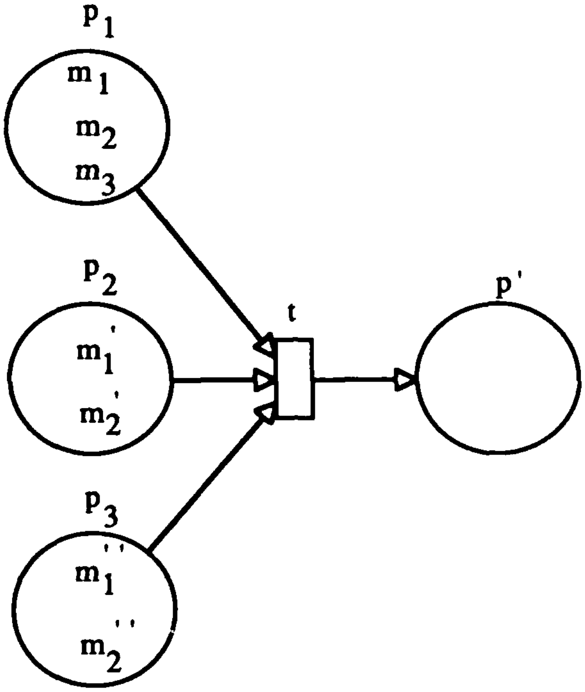


Fig. 3.8 Token Selection

In this case, we suppose that:

- $m_1 = (T_n^1, T_d^1, C^1)$; $m_1' = (T_n^1, T'^1_d, C'^1)$; $m_1'' = (T_n^1, T''^1_d, C''^1)$.
- $m_2 = (T_n^2, T_d^2, C^2)$; $m_2' = (T_n^2, T'^2_d, C'^2)$; $m_2'' = (T_n^2, T''^2_d, C''^2)$.

Since the enabling condition is (3.23), it follows that the transition t is enabled by the set $\{m_1, m_1', m_1''\}$ and by the set $\{m_2, m_2', m_2''\}$. Therefore, a rule must decide what tokens will be removed by the next firing of transition t.

It is assumed that this rule works as follows: it selects a token in a certain place p of the preset of transition t ; then the set of tokens removed is the one to which the token selected belongs. It is shown further in this section (Property 5) that a token can belong to one and only one such set.

Therefore, before applying the rule to the place p, it is necessary to decide in which place p the selection will be done. One can see on the example of Figure 3.8 that p_1, p_2 and p_3 contain each two tokens that enable transition t. This means that the selection of the tokens that will be fired next can be done in place p_1 or place p_2 or place p_3. Thus, a choice must be made to decide if the token selection rule will apply on p_1 or p_2 or p_3.

In the general case, one must distinguish between internal transitions and interactional or external transitions. In the case of internal transitions, the token selection rule will apply to the memory place since it is the only place of the preset that is not a resource place. In the case of external or interactional transitions, the preset can contain several places that are not resource places and, consequently, a choice must be made in order to know to what input place the token selection rule will apply.

Different strategies can be applied to choose the place on which the token selection rule will operate; for example:

- the decision-maker considers only his own information in order to discriminate between the various items of information that he can continue to process. In such a case, the token selection rule would apply to the memory place.
- the decision-maker relies more on the information sent by another member, and, therefore, chooses the next piece of information to process on the basis of the data that he receives from this decision-maker.

The remainder of this chapter considers that the choice of the place on which the token selection rule will apply is done according to some well-known rule PS(t), for each transition t, given the state of the system.

Once the rule PS(t) has been applied, the place p on which the token selection rule will apply is determined. Then, the selection of a token in this place determines an attribute T_n. The knowledge of this attribute allows to select the corresponding tokens in the other places.

In the example above, if PS(t) selects p_1, the token selection rule must discriminate between m_1 and m_2 and m_3. If m_2 is selected, then m_2' and m_2" are automatically selected in places p_2 and p_3.

Property 5: The selection, in the place p, of a token among the tokens that can be fired by transition t determines uniquely the tokens that will be fired in the other places. It results from the fact that once a token has been selected in the place p, its attribute T_n corresponds to one and only one token in any other place of the preset of the transition t.

Thus, it must be decided what will be the possible strategies that the decision-makers will use in order to choose between the several pieces of information that they can continue to process in a given stage. Four types of rules of selection can be thought of:

(i) rules that discriminate with respect to the attribute T_n.

(ii) rules that discriminate with respect to the attribute T_d.

(iii)rules that discriminate with respect to the attribute C.

(iv)rules that combine different rules of the previous types.

Some example of possible rules are the following:

1/ FIFO: the decision-maker can decide to process first the inputs that entered the organization first. In this case, the token with the lowest T_n is selected.

2/ LIFO: the decision-maker decides to process first the inputs that entered the organization last. Then, the token with the highest T_n is selected.

3/ LOCAL FIFO (LFIFO): the decision-maker decides to process first the inputs that entered the internal stage where they currently are first. The token with the lowest T_d is selected.

4/ LOCAL LIFO (LLIFO): the decision-maker processes first the inputs that entered the internal stage where they currently are last. The token with the highest T_d is selected.

5/ PRIORITY: the decision-maker can assign priorities to certain classes of inputs, i.e., can set priorities on the basis of the attribute C. He selects first the items of information with the highest priority.

6/ MIXED: if several pieces of information have the same highest priority, the decision-maker can then decide to apply some rule of the type (i) to (iv) to discriminate between them.

The LFIFO and FIFO rules are different. Indeed, consider the following case:

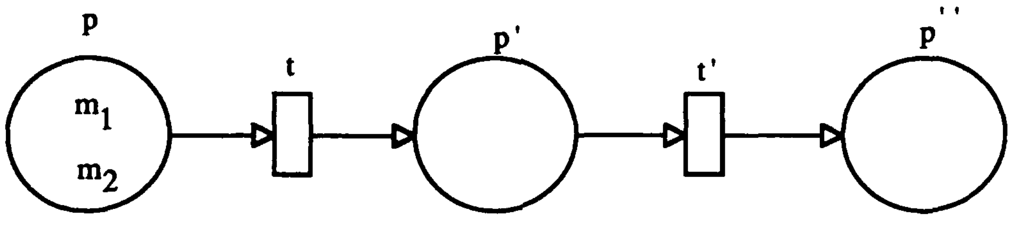


Fig. 3.9 Example LFIFO - FIFO

We assume that $m_1 = (T_n^1, T_d^1, C^1)$, $m_2 = (T_n^2, T_d^2, C^2)$ and $T_n^1 < T_n^2$; it is also supposed that t' is firing when t fires the tokens with the colors m_1 and m_2 . Therefore, a queue with these two tokens forms in p'.

If t applies the LLIFO rule and t' the LFIFO rule, then the token with the color m_2 will enter first p". However, if t' uses the FIFO rule, the token with the color m_1 enters first p".

3.4 CONCLUSION

This chapter has investigated the concept of coordination in decision-making organizations. The concept of team of decision-makers, and the issues of inconsistency of information and synchronization of activities have been emphasized. The Predicate Transition Net model that was introduced allows to distinguish between tokens in the net and to define rules of enablement and firing that account for what occurs in command and control organizations. In the next chapter, the coordination of decision-making organizations is characterized on the basis of this model. Furthermore, measures to evaluate the degree of information consistency and the synchronization of activities are introduced.

CHAPTER IV

COORDINATION IN DECISION-MAKING ORGANIZATIONS: EVALUATION

This chapter introduces a Predicate Transition Net characterization of coordination in decision-making organizations. It bears on the issues of consistency of information and of synchronization of decision-making processes. Two measures are introduced, i.e., the degree of information consistency and the measure of synchronization. The degree of information consistency evaluates the extent to which data exchanged by different decision-makers are consistent. The measure of synchronization evaluates how well the various activities are synchronized, i.e., whether or not the decision-makers have to wait for long periods of time before receiving the data that they need.

4.1 INTRODUCTION

The previous chapter has introduced the framework that will be used to carry out the evaluation of coordination in decision-making organizations.

This evaluation can be done at three levels:

- for each interaction that occurs during the decision-making process.
- for all the interactions that take place during the processing of a particular input.
- for the overall decision-making process, given the alphabet of inputs X.

The first level characterizes locally the coordination of the organization. That is, it measures the extent to which certain organization members are correctly coordinated when they interact for a given input. Therefore, it is useful when different protocols of interaction must be compared for the processing of a certain input of a given scenario.

The second level characterizes the coordination of the organization with respect to the different inputs that it can process for a given scenario. Therefore, it is useful to analyze the evolution of the overall coordination of the organization for a certain scenario.

The third level characterizes the coordination of the overall decision-making process for the mission that it must perform. It is independent of the dynamics of the process, i.e., of the sequence in which the inputs are processed by the organization.

4.2 PREDICATE TRANSITION NET CHARACTERIZATION OF COORDINATION

4.2.1 Introduction

In accordance with the considerations developed in Chapter III, the coordination of different decision-makers shows the extent to which their activities are synchronized and the information that they exchange consistent. Therefore, the coordination bears only on the protocols that decision-makers use when they interact.

It follows that the definition of coordination that will be introduced subsequently applies only to interactional transitions, i.e., to elements of the set $\mathcal{A}$. In the remainder of this section, we shall consider only such interactional transitions. Internal transitions are not involved since they model protocols performed by decision-makers when they do not interact with other organization members. External transitions represent processes which are not performed by decision-makers and, therefore, that are not part of the organizational protocols. The Petri Net representation of the transitions considered in this section is shown in Figure 4.1.

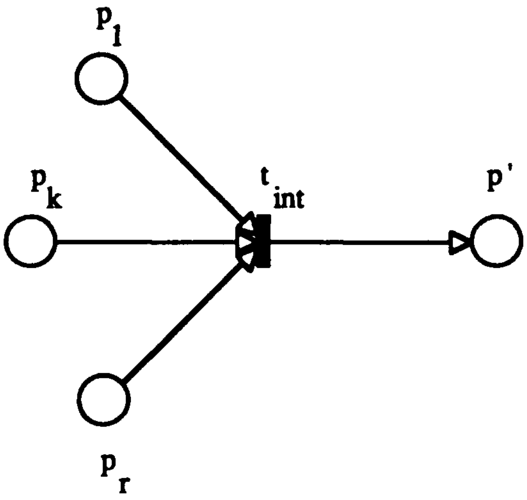


Fig. 4.1 Interaction Stage

In this representation, p_1, ..., p_r represent only structural and memory places. Since resource places do not constrain the enablement except for requiring the presence of a token ¢, their consideration is omitted in the remainder of this chapter. Furthermore, the unique memory place is denoted by p_k. DM_1,..., DM_r designate the decision-makers from whom the data in p_1, ..., p_r originate. DM_k is, therefore, the one who has the protocol t_{int}.

The characterization of the coordination for an interaction t_{int}, using the Predicate Transition Net model introduced in Chapter III, derives from the definition of an order relation on the set of tokens fired by transition t_{int}. The next section reviews some basic definitions.

4.2.2 Order Relation

Definition 1: A binary relation Ψ is an application from $\mathfrak{B}\times\mathfrak{B}$ to {TRUE, FALSE} which assigns to each 2-tuple (x, y) a boolean value. Therefore:

$$\begin{aligned} \Psi : \ & \mathfrak{B}\times\mathfrak{B} \text{ -> } \{\text{TRUE, FALSE}\} \\ & (x, y) \text{ -> } \Psi(x, y) \qquad (4.1) \end{aligned}$$

When $\Psi(x, y)$ is TRUE, we denote it by $x \Psi y$; when $\Psi(x, y)$ is FALSE, we denote it by $x \text{ not}\Psi y$.

Definition 2: A binary relation Ψ defined on a set $\mathfrak{B}\times\mathfrak{B}$ is an order relation if, and only if:

$$\forall x \in \mathfrak{B} , x \Psi x \qquad (4.2)$$

$$\forall (x, y) \in \mathfrak{B}\times\mathfrak{B}, ((x \Psi y) \text{ and } (y \Psi x)) \Rightarrow (x = y) \qquad (4.3)$$

$$\forall (x, y, z) \in \mathfrak{B}\times\mathfrak{B}\times\mathfrak{B}, ((x \Psi y) \text{ and } (y \Psi z)) \Rightarrow (x \Psi z) \qquad (4.4)$$

In this case, if $\Psi(x, y)$ is TRUE, we shall say that x is smaller than y.

Definition 3: A binary relation Ψ defined on a set $\mathfrak{B}\times\mathfrak{B}$ is a total order relation if, and only if, it is an order relation, and for all (x, y) $\Psi(x, y)$ or $\Psi(y, x)$ is TRUE.

A binary relation Ψ defined on a set $\mathbf{B}\times\mathbf{B}$ is a partial order relation if, and only if, it is an order relation and there exists a 2-tuple (x,y) such that $\Psi(x, y)$ and $\Psi(y, x)$ are both FALSE.

Definition 4: g is the greatest element of $\mathbf{B}$ if, and only if, for all x, $\Psi(x, g)$ is TRUE, i.e.,

$$(\forall x \in \mathbf{B},\ x \Psi g) \tag{4.5}$$

4.2.3 Definitions

On the basis of the Predicate Transition Net model introduced in Chapter III and of the order relation, it is possible to introduce a characterization of coordination in decision-making organizations. First, the following binary relations are defined:

Definition 5: Ψ_1 is a binary relation defined on $Q^+\times Q^+\times \Pi^*(\mathcal{X})$ by :

$$((x, y, z)\ \Psi_1\ (x', y', z')) \Leftrightarrow ((x = x') \text{ and } (y \leq y')) \tag{4.6}$$

Ψ_2 is a binary relation defined on $Q^+\times Q^+\times \Pi^*(\mathcal{X})$ by :

$$((x, y, z)\ \Psi_2\ (x', y', z')) \Leftrightarrow ((x = x') \text{ and } (z = z')) \tag{4.7}$$

Ψ_3 is a binary relation defined on $Q^+\times Q^+\times \Pi^*(\mathcal{X})$ by :

$$((x, y, z)\,\Psi_3\,(x', y', z')) \Leftrightarrow (((x, y, z)\,\Psi_1\,(x', y', z')) \text{ and } ((x, y, z)\,\Psi_2\,(x', y', z'))) \tag{4.8}$$

Property 1: The relation Ψ_3 defines an order relation on the set $X = Q^+\times Q^+\times \Pi^*(\mathcal{X})$.

It derives from the fact that the relation $\leq$ defines an order relation on the set Q^+.

We denote by $m_1, ..., m_r$ the elements of X which represent the colors of the r tokens removed respectively from places $p_1, ..., p_r$ by transition t_{int}. m_k denotes the color of the token removed from the memory place p_k. Furthermore, each color m_i corresponds to $(T_n{}^i, T_d{}^i, C^i)$, element of $\mathbf{Q}^+ \times \mathbf{Q}^+ \times \Pi^*(\mathbf{X})$.

In accordance with the conceptual considerations developed in Chapter III, it can be said that different decision-makers $DM_1, ..., DM_r$ are synchronized for the interaction t_{int}, if DM_k does not wait for any data from the other decision-makers. It motivates the following definition :

Definition 6: The firing of t_{int} is **synchronized** if, and only if:

$$\forall i \in \{1,...,r\}, \ (T_n^i, T_d^i, C^i) \ \Psi_1 \ (T_n^k, T_d^k, C^k) \qquad (4.9)$$

This definition allows to discriminate between firings that are synchronized and firings where one or several tokens m_i arrive in their respective places later than m_k in p_k. The measure of the degradation of synchronization in the latter cases will be evaluated in the section 4.4.

In the same way, the data fused by DM_k are consistent if they correspond to the same class C. It leads to the following definition:

Definition 7: The firing of t_{int} is **consistent** if, and only if:

$$\forall (i, j) \in \{1,...,r\} \times \{1,...,r\}, \ (T_n^i, T_d^i, C^i) \ \Psi_2 \ (T_n^j, T_d^j, C^j) \qquad (4.10)$$

The evaluation of the degradation of information consistency will be introduced in section 4.3.

On this basis, the following definition for the coordination of an interaction is provided:

Definition 8: The firing of t_{int} is **coordinated** if, and only if, it is synchronized and consistent.

It is possible now to characterize a coordinated transition firing by the order of arrival of the tokens in the places of its preset.

Property 2: When the firing of $m_1,..., m_r$ by t_{int} is coordinated, the relation Ψ_3 induces an order relation on the set $\{m_1,..., m_r\}$ for which m_k, token of the memory place, is the unique greatest element.

Proof: Since $\leq$ is a total order relation on $\mathbf{Q}^+$, i.e, an order relation which allows to compare all rational numbers two by two, we have:

$$\forall\ (m_i, m_j) \in X{\times}X,\ m_i\ \Psi_1\ m_j \text{ or } m_j\ \Psi_1\ m_i\,.$$

Since the firing of t_{int} is consistent:

$$\forall\ (m_i, m_j) \in X{\times}X,\ m_i\ \Psi_2\ m_j.$$

It implies that Ψ_3 is a total order relation for $\{m_1, ..., m_r\}$ since:

$$\forall\ (m_i, m_j) \in X{\times}X,\ m_i\ \Psi_3\ m_j \text{ or } m_j\ \Psi_3\ m_i\,.$$

Furthermore, since the firing is synchronized:

$$\forall\ m_i \in X,\ \ m_i\ \Psi_3\ m_k$$

It follows that m_k is the greatest element.

This property characterizes coordinated transition firings with the Predicate Transition Net formalism. It means that the token of the memory place of the preset of the transition arrives in this place after all other tokens have arrived in their respective places.

Definition 8 bears on the coordination of a single interaction. The definitions of the coordination of a single task, i.e., for a sequence of interactions concerning the same input, as well as for all tasks executed are derived as follows.

Definition 9: The execution of a task is coordinated if, and only if, it is coordinated for all interactions that occur during the task.
The execution of a Petri Net PN is coordinated if, and only if, it is coordinated for all the tasks performed.

This definition of coordination does not encompass the definition of team that was presented above. Indeed, whereas the latter is essentially a static characteristic of the organization independently of how the decision-making process actually takes place, the former addresses the dynamics of this process and the interactions that occur.

4.3 DEGREE OF INFORMATION CONSISTENCY

4.3.1 Introduction

It is possible to introduce a measure that evaluates the extent to which the organizational decision-making process is consistent for the task at hand.

Given an interaction stage, t_{int} denotes the interactional transition that models this stage in the Petri Net representation, as shown in Figure 4.1. $p_1,...,p_r$ designate the memory and structural places of the preset of transition t_{int}. These places contain the information sent by the decision-makers who interact at this stage.

In particular, at each transition t_{int}, the decision-maker DM_h associates a class C^h to ieach input x_i. In this section, this class is denoted by $C^h(x_i, t_{int})$. As introduced in Chapter III, this class belongs to $\mathfrak{D}(p_h)$, partition of the alphabet X, that the designer defines a priori.

In order to achieve a higher consistency, the designer has to ensure that the r decision-makers who interact in the stage are provided with the same set of classes; therefore, it is assumed that:

$$\forall\,(i,j) \in \{1,...r\}\times\{1,...,r\}\ ,\ \ \mathfrak{D}(p_i) = \mathfrak{D}(p_j) \qquad (4.11)$$

4.3.2 Definitions

If m_1, ..., m_r designate the colors of the tokens in the preset of t_{int} that correspond to input x_i and that are fired by t_{int}, then $C^1(x_i,t_{int})$,..., $C^r(x_i, t_{int})$ denote their attribute C.

$V(x_i, t_{int})$ designates the vector $(C^1(x_i, t_{int}), ..., C^r(x_i, t_{int}))$, element of $[\Pi^*(\mathsf{X})]^r$.

$prob(C^1(x_i, t_{int}), ..., C^r(x_i, t_{int}))$ denotes the probability of having tokens with attribute $C^1(x_i, t_{int})$, ..., $C^r(x_i, t_{int})$ for the input x_i at the stage t_{int} in places p_1, ..., p_r. It will be written as $prob(V(x_i, t_{int}))$.

$z(V(x_i, t_{int}))$ is the number of subsets of two elements $\{C^a(x_i, t_{int}), C^b(x_i,t_{int})\}$ of

$\{C^1(x_i,t_{int}), ..., C^r(x_i, t_{int})\}$. Thus, we have:

$$z(V(x_i, t_{int})) = \binom{r}{2} = \frac{r!}{2!\,(r-2)!} \tag{4.12}$$

$n(V(x_i, t_{int}))$ is the number of subsets of two elements $\{C^a(x_i, t_{int}), C^b(x_i,t_{int})\}$ of $\{C^1(x_i,t_{int}),..., C^r(x_i, t_{int})\}$ such that $C^a(x_i, t_{int}) = C^b(x_i,t_{int})$.

We have then:

Definition 10: The degree of information consistency for stage t_{int} and input x_i is:

$$d(x_i, t_{int}) = \sum_{V(x_i, t_{int})} prob(V(x_i, t_{int})) \frac{n(V(x_i, t_{int}))}{z(V(x_i, t_{int}))} \tag{4.13}$$

For example, consider the case depicted in Figure 4.2:

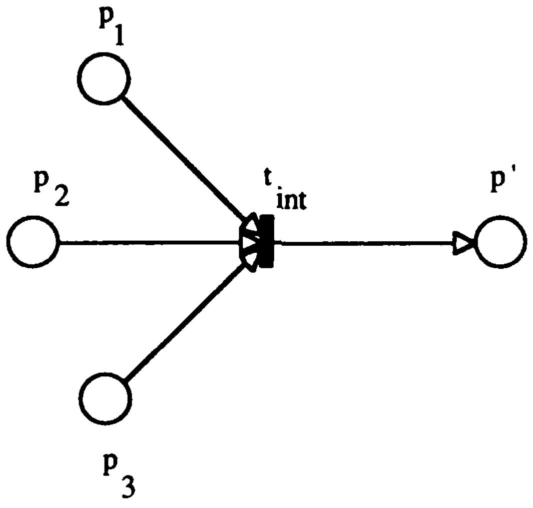


Fig. 4.2 Example of Degree of Information Consistency

We suppose that the set of classes is $\{C^1, C^2, C^3\}$ for all three places, i.e. :

- $\mathfrak{D}(p_1) = \{C^1, C^2, C^3\}$
- $\mathfrak{D}(p_2) = \{C^1, C^2, C^3\}$

- $\mathfrak{D}(p_1) = \{C^1, C^2, C^3\}$

Furthermore, consider the five colors, elements of X:

- $m_1 = (T_n, T_d^1, C^1)$
- $m_2 = (T_n, T_d^2, C^2)$
- $m_3 = (T_n, T_d^3, C^3)$
- $m_4 = (T_n, T_d^4, C^1)$
- $m_5 = (T_n, T_d^5, C^1)$

If M designates the marking, the three following cases are considered:

- case 1: $M(p_1)[m_1] = 1$; $M(p_2)[m_2] = 1$; $M(p_3)[m_3] = 1$.
- case 2: $M(p_1)[m_1] = 1$; $M(p_2)[m_4] = 1$; $M(p_3)[m_3] = 1$.
- case 3: $M(p_1)[m_1] = 1$; $M(p_2)[m_4] = 1$; $M(p_3)[m_5] = 1$.

Following relation (4.13), in case 1, the degree of information consistency is zero. In case 2, it is 1/3. In case 3, the degree of information consistency is 1.

In formula (4.13), the probability of having $C^1(x_i, t_{int}),...,C^r(x_i, t_{int})$, when not zero, is not always necessarily equal to 1. Indeed, a decision-maker can use different algorithms to process a same input x_i or can implement stochastic algorithms, and, therefore, can assign different classes to a same input.

By adding the degrees of information consistency $d(x_i, t_{int})$ for each organizational interaction t_{int} and each input x_i and weighing by the probability of having the input, one can measure the organizational degree of information consistency for the task at hand.

Definition 11: The organizational degree of information consistency, D, is defined by:

$$D = \sum_{x_i} prob(x_i) \sum_{t_{int}} d(x_i, t_{int}) \qquad (4.14)$$

This measure varies between 0 and 1, the value 1 corresponding to ideal information consistency of all interactions for the whole task. The next section introduces a measure of performance for synchronization.

4.4 A MEASURE OF PERFORMANCE IN SYNCHRONIZATION

4.4.1 Introduction

The interactions that take place during the decision-making process are determined by the organization's structure. When a decision-maker processes an item of information, the total processing time of this item for decision-maker DM_i consists of two distinct parts:

(i) the total time T_i^t during which the decision-maker actually operates on the information, i.e., the total time spent by the information in the decision-maker's algorithms.

(ii) the total time T_i^p spent by the information in memory without being processed.

The time T_i^p results from two factors:

- information can remain in the memory of the decision-maker until he decides to process it with the relevant algorithm. Since a particular algorithm cannot process two inputs at the same time, some inputs will have to remain unprocessed in memory for a certain amount of time until the relevant algorithm is available.
- information can also remain in memory because the decision-maker waits to receive data from another decision-maker.

As discussed in Chapter III, an organization is not well synchronized when the decision-makers have to wait for long periods before receiving the information that they need in order to continue their processing. On the contrary, the organization is well synchronized when these lags are small.

This section provides a measure of performance for synchronization, S_T, that determines quantitatively the extent to which the organization is synchronized, or, more accurately, the extent to which the organization is not synchronized.

4.4.2 Definitions

As explained in section 4.2.1, the stages of interest are interactional stages, i.e., Information Fusion and Command Interpretation. The transitions of interest are, therefore, interactional transitions.

The Petri Net model of such an interaction has been presented in Figure 4.1. As previously, places $p_1, ..., p_r$ are structural and memory places, p_k is the unique memory place of the preset of t_{int}, and $m_1, ..., m_r$ denote the colors of the tokens fired by t_{int}; $m_1 = (T_n, T_d^1, C^1), ..., m_r = (T_n, T_d^r, C^r)$. T_c is the current time, i.e., the time at which t_{int} fires.

Definition 12 : The **sojourn time** $T_s^h(x_i,t_{int})$ of the token m_h, representing the input x_i in the place p_h of the preset of transition t_{int}, measures the amount of time spent by the token in the place before it is fired:

$$T_s^h(x_i, t_{int}) = T_c - T_d^h \tag{4.15}$$

This quantity is zero when the firing occurs at the same time the token enters the place. Conversely, it differs from zerc when the firing cannot be initiated at the same time the token enters the place.

The following quantity can now be introduced:

$$S_L^{hj}(x_i, t_{int}) = T_s^h(x_i, t_{int}) - T_s^j(x_i, t_{int}) \tag{4.16}$$

$S_L^{hj}(x_i, t_{int})$ measures the difference between the sojourn times of the tokens representing x_i in p_h and p_j, i.e., the difference between the lengths of time that the information sent by DM_h and DM_j to DM_k remained inactive before being processed.

In Chapter III, it was stated that the degradation of synchronization was induced only by interactions where decision-makers have to wait for data. So, the quantity $S_L^{kj}(x_i, t_{int})$, where p_k represents the memory place, will be computed for each structural place p_j. If it is positive, it implies that the token m_k has spent more time in p_k than the token m_j in p_j. If it is negative, the converse is true. In the latter case, there is no degradation of synchronization, because DM_k is not ready to process the next task.

F(x) denotes the function defined on the set of rational numbers, **Q**, by:

$$\begin{aligned} \forall\, x \in \mathbf{Q},\ (x \geq 0) &\Rightarrow (\, F(x) = x\,) \\ (x < 0) &\Rightarrow (\, F(x) = 0\,) \end{aligned} \tag{4.17}$$

$INT(t_{int})$ denotes the set of indices h for the structural places p_h of $Pre(t_{int})$.

The total lag for the transition t_{int} in processing input x_i can now be defined as follows.

Definition 13: $S(x_i, t_{int})$, representing the total lag for transition t_{int} and the input x_i, is such that:

$$S(x_i, t_{int}) = \max_{h \in INT(t_{int})} \left(\, F[\, S_L^{kh}(x_i, t_{int})\,]\, \right) \tag{4.18}$$

or, from (4.16),

$$S(x_i, t_{int}) = \max_{h \in INT(t_{int})} \left(\, F\, [\, T_s^k(x_i, t_{int}) - T_s^h(x_i, t_{int})\,]\, \right) \tag{4.19}$$

Thus, $S(x_i, t_{int})$ measures the maximum of all the lags during which the decision-maker has to wait before having all the information he needs to continue his processing. The measure S does not take into consideration the items of information for which the decision-maker does not wait: this is consistent with the definition of synchronization given in Chapter III.

For instance, in the example illustrated in Figure 4.2, if p_2 denotes the memory place, we have:

- case 1: $S(x_i, t_{int}) = \max(F(T_d^2 - T_d^1),\ F(T_d^2 - T_d^3))$
- case 2: $S(x_i, t_{int}) = \max(F(T_d^4 - T_d^1),\ F(T_d^4 - T_d^3))$

- case 3: $S(x_i, t_{int}) = \max(F(T_d^4 - T_d^1), F(T_d^4 - T_d^5))$

On the basis of this definition, it is possible to define two measures. First, it is recalled that $\mathcal{A}$ denotes the set of all interactional transitions. Furthermore, $\mathcal{A}(k)$ denotes the set of all interactional transitions executed by decision-maker DM_k.

Definition 14: The measure of synchronization between decision-maker DM_k and the rest of the organization, S_k, is such that:

$$S_k = \sum_{x_i} prob(x_i) \sum_{t_{int} \in \mathcal{A}(k)} S(x_i, t_{int}) \tag{4.20}$$

It is the expected value of the sum of the maximum lags for the interaction stages executed by decision-maker DM_k for the inputs x_i.

Definition 15: The measure of synchronization for the organization, S_T, is such that:

$$S_T = \sum_{x_i} prob(x_i) \sum_{t_{int} \in \mathcal{A}} S(x_i, t_{int}) \tag{4.21}$$

It is the expected value of the sum of the maximum lags over the overall decision-making process for the inputs x_i.

On the one hand, the measures S_k, for each k, and S_T achieve their best values when they are zero. On the other hand, there is no bound on the values taken by these measures: it can be the case that they grow to infinity if a deadlock occurs. This type of situation will be investigated in Chapter V.

Since each interactional transition t_{int} belongs to one decision-maker, and one only, the following relation holds:

$$S_T = \sum_{k} S_k \tag{4.22}$$

Thus, one can compute the contribution of each individual decision-maker DM_k to the total synchronization measure S_T for the organization by taking the ratio S_k/S_T.

4.4.3 Example

Consider the following two-person hierarchical organization:

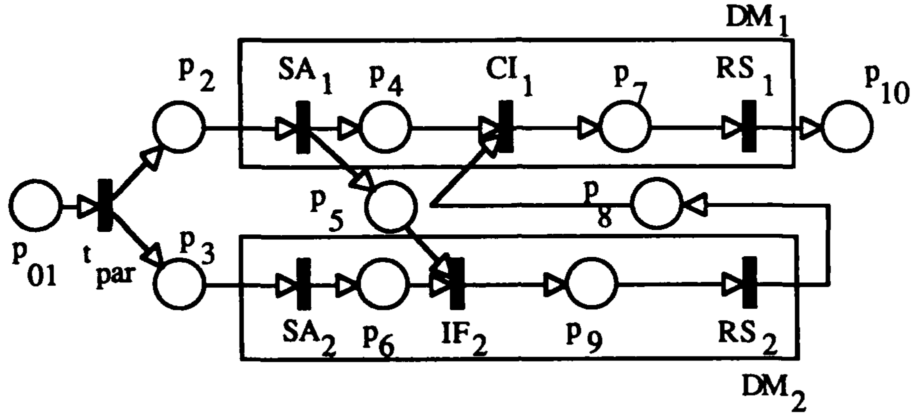


Fig. 4.3 Synchronization of Two-Person Organization

We assume that the processing times of the various stages are independent of the input x_i. Therefore, $S_L(x_i, t_{int})$ is written $S_L(t_{int})$.

T_s^4, T_s^5, T_s^6, T_s^8 denote the sojourn times of tokens respectively in places p_4, p_5, p_6 and p_8. IF_2 and CI_1 have only two input places. It follows that:

- $S_L(t_{IF2}) = F(T_s^6 - T_s^5)$
- $S_L(t_{CI1}) = F(T_s^4 - T_s^8)$

Then, the measures of synchronization S_T, S_1 and S_2 are such that:

- $S_1 = S_L(t_{CI1}) = F(T_s^4 - T_s^8)$
- $S_2 = S_L(t_{IF2}) = F(T_s^6 - T_s^5)$
- $S_T = S_L(t_{IF2}) + S_L(t_{CI1}) = F(T_s^6 - T_s^5) + F(T_s^4 - T_s^8)$

For all inputs, if f(t) denotes the firing time of transition t, we consider two cases:

- case 1: $f(t_{SA1}) = 2$, $f(t_{SA2}) = 1$, $f(t_{IF2}) = 2$, $f(t_{RS2}) = 2$, $f(t_{CI1}) = 2$, $f(t_{RS1}) = 2$

- case 2: $f(t_{SA1}) = 2$, $f(t_{SA2}) = 2$, $f(t_{IF2}) = 2$, $f(t_{RS2}) = 2$, $f(t_{CI1}) = 2$, $f(t_{RS1}) = 2$

The results for the measures S_1, S_2 and S_T are:

- in case 1, $S_T = 5$; $S_1 = 1$; $S_2 = 4$.
- in case 2, $S_T = 4$; $S_1 = 0$; $S_2 = 4$.
- in both cases, $T = 10$, where T denotes the total delay.

Therefore, when the processing times of SA_1 and SA_2 are not equal, the synchronization measure S_T is worse than when they have equal processing times. It should be noted that the processing delays in the two cases are equal. It follows that the fact that DM_2 takes more time to process an input improves the synchronization without affecting the total processing delay.

It must also be noted that, in such an example, the synchronization can be zero if, and only if, SA_1 and SA_2 have the same processing times, and IF_2 and RS_2 have zero processing times. The latter conditions are not realistic, which implies that the synchronization will never be zero.

The measure S_T will be used in Chapter VII to assess the impact of decision-support systems on the effectiveness of organizations and, in particular, on their coordination.

4.5 CONCLUSION

This chapter has introduced a methodology for evaluating the coordination in decision-making organizations. A Predicate Transition Net characterization of coordination has been presented. Two measures have also been derived: the degree of information consistency, D, and the measure of synchronization, S_T. The next chapter investigates the dynamics of decision-making processes and presents a simulation tool to perform the evaluation of such processes.

CHAPTER V

DYNAMICS OF COORDINATED DECISION-MAKING PROCESSES

This chapter investigates the dynamics of the decision-making processes that require coordination and assesses the impact of preprocessors on the cohesiveness of organizations. The relationship between the concepts of team of decision-makers and synchronization is emphasized. The simulation of Petri Nets is introduced as a tool for evaluating alternative organizational structures.

5.1 INTRODUCTION

In the remainder of the study, the consistency of the information exchanged by different decision-makers is assumed to be always perfect: thus, the concept of coordination of decision-making organizations will refer to the synchronization of the processes and to the concept of a team.

The model of coordination developed in Chapter III permits the analysis of interactions between different decision-makers who use protocols that constrain their activities.

When they interact, they must exchange information originating from the same state of the environment. This constraint implies that the synchronization of the various processes may be more difficult to achieve.

In particular, if the decision-makers have different perceptions of the task at hand, they may assign distinct priorities to the various inputs that originate from the environment; at the level of the interaction stages, it will result in important delays because the decision-makers receiving information from other members of the organization will have to wait until they receive the data that they need in order to carry on their own processing. This lack of coordination will cause a degradation both of the timeliness and of the synchronization of the organization that will result in a decrease of the effectiveness.

Thus, there are two types of reasons for which the synchronization of the organization will not be perfect:

- the inputs are not processed by the various decision-makers in the same order so that the information available at the Information Fusion and Command Interpretation stages cannot always be fused because it does not refer to the same input.
- the structure of the organization is such that, even if the decision-makers always process the inputs in the same order, there will be delays due to the different processing times of the various protocols.

This is illustrated in Figure 5.1 for the two-person hierarchical organization:

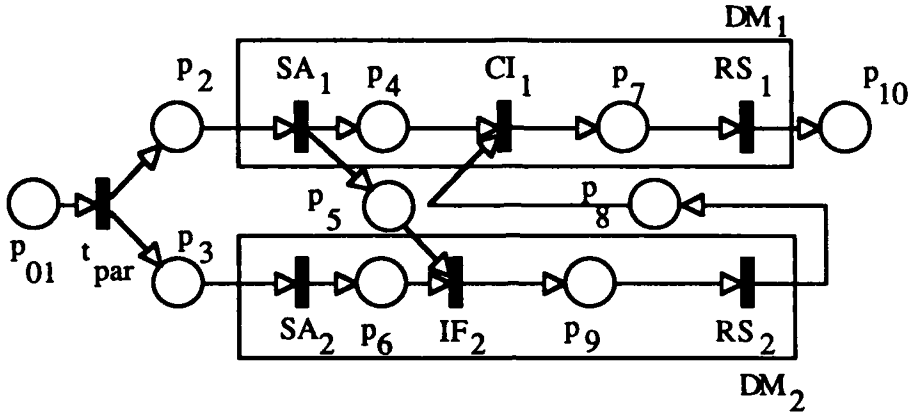


Fig. 5.1 Two-Person Hierarchical Organization

If SA_1 and SA_2 fire the tokens that are in p_2 and p_3 with different priorities, then the IF_2 will not always fire even though all its input places contain at least a token. Furthermore, even if SA_1 and SA_2 fire the tokens that are in p_2 and p_3 with the same priority, their firing times can be different so that the token in p_6 will have to wait for a certain amount of time before its corresponding token arrives in p_5.

These two categories of factors which degrade the synchronization are non-overlapping: An organization can have protocols such that no structural delay will affect the synchronization of the process and this independently from the schedules according to which the various inputs are handled. In the same way, the different decision-makers can have the same schedule but have protocols that lead to structural delays.

For an organizational structure and protocols with given execution times, the decision-making process is optimally synchronized when no delay occurs because of different orders in the processing of the various items of information, i.e., because of different schedules. It can be expressed with the Petri Net formalism as follows: each transition modeling an interaction stage can fire as soon as each of its input places contain a token. Conversely, the process is not optimally synchronized given the organizational structure when its dynamics leads to situations where such delays occur.

In particular, when the tempo of operations increases, the risks of poor coordination may increase too, since the decision-makers are confronted with multiple inputs that they can process in different order. It follows that, for a given organizational structure, the coordination of the process can be very sensitive to the dynamics of the activities that take place, even though the information always follows the same flow path.

The purpose of this chapter is to investigate the extent to which the coordination of an organization is affected by the dynamics of the decision-making process. That is, decision-making processes which are not optimally synchronized for a given organizational structure, are studied. In order to do so, the simulation of Petri Nets is presented as a tool for *the designer of decision-making organizations* to analyze and evaluate alternative configurations under different circumstances.

5.2 SIMULATION OF PETRI NETS

5.2.1 Basic Considerations

The performance evaluation of decision-making organizations using Timed Petri Nets has been developed in previous work (Hillion and Levis, 1987) for a specific class of Petri Nets, i.e., Event-graphs. The two assumptions that were used are:

- the decision-makers use the LFIFO rule in all their internal stages. Thus, the process is always optimally synchronized given the firing times of the various protocols.
- the organizational resources are always used as soon as they are available.

In Petri Net formalism, the first assumption means that all tokens are identical. Thus, if the tokens are not distinguishable, the transitions modeling interaction stages always fire as soon as each input place contains a token whatever its characteristics. Consequently, given the firing times of the transitions, the execution of the net is optimally synchronized. From an organizational standpoint, it expresses the fact that decision-makers, if they are ready to do so, always fuse the information as soon as they receive it, meaning that this information is what they need.

The second assumption corresponds to an infinite queue of identical tokens in the source place. It expresses the fact that the organization will process the information at its maximum throughput rate by using all its resources; indeed, no organizational resource can remain unused and the organization cannot process information at a higher rate since this would require that more resources be available at certain instants during the process.

This analysis has led to the definition of two measures of performance, i.e., maximum throughput rate and execution schedule. These measures provide upper bounds in the ideal case where all the decision-makers use the LFIFO rule in all their internal stages. In this context, the consideration of the scenario for which organizations achieve their maximum performance, i.e., the infinite queue of inputs, provides a means of comparing organizations for all scenarios: Indeed, if DMO_1 has a better maximum throughput rate than DMO_2, then DMO_1 will be able to handle without overload all the tasks that DMO_2 can handle without overload. These measures are therefore useful to evaluate alternative organizational structures under such an assumption. In the cases where the assumption no longer holds, it is necessary to introduce the Predicate Transition Net model of coordination presented in Chapter III.

This model is characterized by the following:

- each token representing an item of information is distinguished with respect to its time of entry in the net, its time of entry in the place where it stands, and the class of inputs associated with it.
- each place is associated with a rule of selection of tokens, e.g., FIFO or LIFO.
- the rule of enablement of transitions requires that tokens with the same time of entry in the net be in the input places.

The analysis of any Timed Predicate Transition Net is a complicated process that depends on the specific grammar used in executing the net. In such a context, the simulation of Petri Nets yields insight on the dynamics of the process from three standpoints:

(i) evaluation of measures such as the *throughput rate*, *response time* and *synchronization* for different scenarios and characteristics of the organization over the whole decision-making process. It may prove useful because, as it is shown in section 5.3.3, the fact that an organization performs well for a certain scenario does not imply necessarily that it will perform well for all conceivable scenarios. The quality of the performance of an organization is scenario-dependent: Two organizations can exhibit the same performance for certain scenarios and achieve very different levels for other scenarios.

(ii) evaluation of local measures of organizational performance. It is also possible to observe the dynamics of queues of items of information for different decision-makers and at different stages of the decision-making process.

(iii) evaluation of measures of variables over a limited period of time. It is possible to observe the process during periods of high or low activity.

Thus, the simulation of Petri Nets allows to overcome some of the analytical difficulties that are introduced by the model of coordination developed in Chapter III. In section 5.3, the implementation of the simulation as well as the results to which it can lead will be described for some specific examples.

5.2.2 Simulation Overview

The information needed to run the simulation software is presented in Appendix A. Furthermore, Appendix B contains a description of the architecture of the program as well as useful information concerning the data structures and the main functions.

The simulation of decision-making processes using Timed Petri Nets is implemented as a discrete-event simulation. In other words, the execution of a net is carried out through the activation of events that take place at discrete instants in time. This is consistent with the actual processes that take place in decision-making organizations: Decision-makers pass through different states, each of them corresponding to the activation or de-activation of one or several of their internal stages.

The events are inserted in an event-calendar. As far as the execution of Petri Nets is concerned, the event-calendar is formally defined as being an ordered list which indicates the sequence in which events are activated. For the simulation of decision-making processes, the events are ordered with respect to a time index.

Consequently, the simulation provides the organization's designer with a tool that allows him analyze, for a given strucuture, different protocols of operations and different scenarios.

5.3 APPLICATION

5.3.1 Introduction

This section provides specific examples of processes where the lack of coordination between the different decision-makers lead to a degradation of the organizational performance. This degradation can have various causes but they all relate to the concept of a team. In Chapter III, a team has been defined as being a group of persons who have a common goal, have the same interests and beliefs, and have activities that must be coordinated. The fact that different decision-makers have distinct interests or beliefs can lead them to adopt activities that are not coordinated with respect to the task at hand.

5.3.2 Example 1

The organization consists of two decision-makers who receive information for a common task. The commander DM_2 assesses the data that he receives from the environment by using always the same algorithm. In the same way, the subordinate DM_1 assesses the input from the environment with one algorithm. Then, he sends some information resulting from this assessment to his commander. The latter fuses his own result with this information and, on this basis, produces a command by using always the same algorithm. In turn, this command is sent to the subordinate DM_1. Eventually, DM_1 is responsible for producing a response on the basis of the command that he receives and of the results of his own assessment.

The Petri Net model of this organization has already been presented in Chapter II. It is shown in Figure 5.2 with the resource places present:

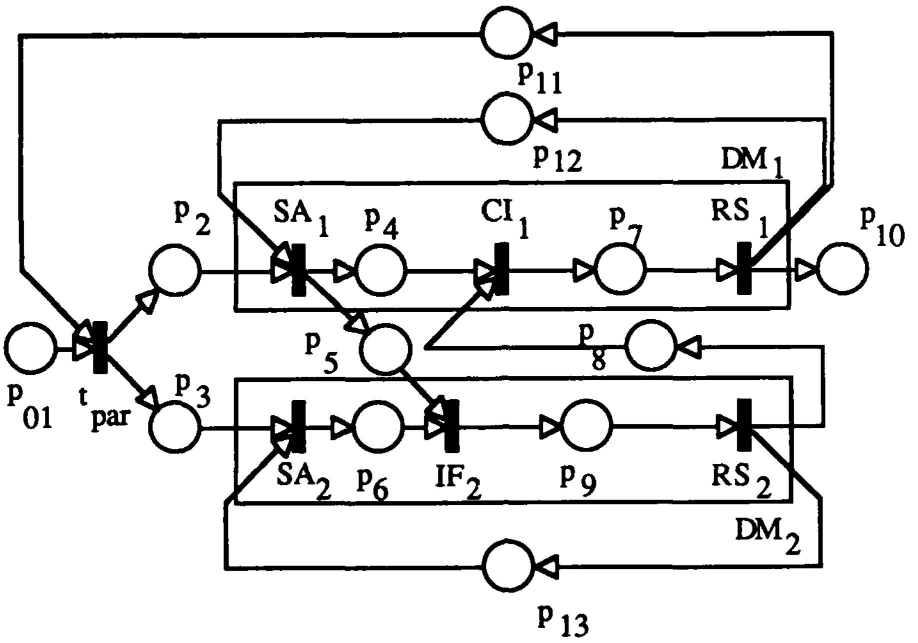


Fig. 5.2 Petri Net Model of Two-person Organization with Resource Places

Since both decision-makers have only one algorithm in each stage of their process, there is only one organizational strategy. Thus, the information can follow only one flow path. Each decision-maker needs the information sent by the other in order to complete his processing. In accordance with the model of coordination developed in this study, the Information Fusion and Command Interpretation stages require that the data fused originate from the same input. The organization will be perfectly synchronized if DM_1 and DM_2 never wait for the information from the other decision-maker in these stages.

It will be assumed that in all stages, but the SA stage, the decision-makers select the data with the LFIFO rule with priority given to the items of information that are in their memory places. Thus, for IF_2, RS_2, CI_1, RS_1, tokens are fired in the order with which they enter the memory place of their preset.

However, different conditions for the SA stages will be considered. Before having assessed any of the inputs that they have received and that they must process, the decision-makers may have to discriminate between them because they cannot perform their assessment on all of them at the same time: only one input can be assessed at a time.

Depending on the characteristics of the data that they receive from the environment, the decision-makers can use different rules to perform this selection: two cases will be investigated:

- the inputs have not been preprocessed by any decision-aid: in this case, the decision-makers have no information concerning the nature of the inputs and must use rules that are based on attributes independent from the characteristics of the inputs, e.g., time of entry in the organization.
- the inputs have been preprocessed by a decision-aid that aggregates them in classes, or zones of indifference (Chyen and Levis, 1985). In this case, the decision-makers have some information concerning the nature of the inputs and can use rules that assign priorities to these different classes.

The processing times, measured in some time unit, of the various stages are presented below; t_{par} denotes the partitioning stage, as introduced in Chapter III.

	t_{par}	SA_1	SA_2	CI_1	IF_2	RS_1	RS_2
Time:	1	10	10	10	10	10	10

Organization with no preprocessor

The scenario corresponds to the infinite queue of inputs, i.e., to the case where the organization always uses all its resources.

case 1: The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
Both SA_1 and SA_2 use the LFIFO rule.

case 2: The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
SA_1 uses the LFIFO rule whereas SA_2 uses the LLIFO rule.

case 3: The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
SA_1 uses the LLIFO rule; SA_2 uses the LFIFO rule.

case 4: The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
Both SA_1 and SA_2 use the LLIFO rule.

The time of entry of an input in the organization, T_i, is the time at which the sensors begin to process it, i.e., in Petri Net formalism the time at which the transition t_{par} fires. The time of leaving from the organization, T_o, is the time at which the organizational response is obtained, i.e., in Petri Net formalism the time at which a token appears in the sink place. The delay, T, is the difference T_o-T_i. As defined in Chapter IV, the synchronization S is measured for each token and the sum of the values of the measure for IF_2 and CI_1.

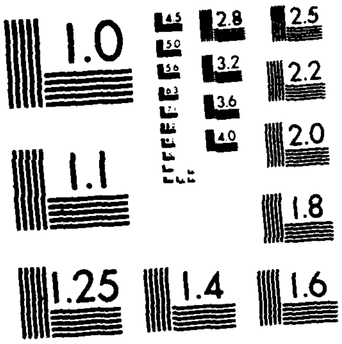
Table 5.1 provides the results for T_i , T_o, T as well as for the synchronization S for the first ten inputs which enter the net in each of these four cases.

TABLE 5.1 Synchronization and Delay - Cases 1 to 4

input #	case 1				case 2				case 3				case 4			
	T_i	T_o	T	S	T_i	T_o	T	S	T_i	T_o	T	S	T_i	T_o	T	S
1	0	51	51	20	0	51	51	20	0	51	51	20	0	51	51	20
2	1	61	60	20	1	-	-	-	1	-	-	-	1	-	-	-
3	2	101	99	40	2	101	99	40	2	101	99	20	2	101	99	40
4	3	111	108	40	3	151	148	110	3	131	128	90	3	61	58	20
5	51	151	100	40	51	201	150	90	51	161	110	30	61	151	90	40
6	61	161	100	40	101	251	150	90	101	191	90	30	101	201	100	40
7	101	201	100	40	151	301	150	90	131	221	90	30	151	251	100	40
8	111	211	100	40	201	351	150	90	161	251	90	30	201	301	100	40
9	151	251	100	40	251	401	150	90	191	281	90	30	251	351	100	40
10	161	261	100	40	301	451	150	90	221	311	90	30	301	401	100	40

The results obtained in case 1 are the same as in the case where the tokens have no identity. The steady-state of the process is K-periodic (Hillion and Levis, 1987) with a period of one. It is reached after the sixth input and is characterized by a constant delay and

AD-A191 950 DECISION AIDING AND COORDINATION IN DECISION-MAKING 2/2
ORGAINZATIONS(U) MASSACHUSETTS INST OF TECH CAMBRIDGE
LAB FOR INFORMATION AND D.. J M GREVET JAN 88
UNCLASSIFIED LIDS-TH-1737 N00014-84-K-0519 F/G 5/1 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

synchronization. The same conclusions can be drawn in case 2, case 3 and case 4: all three processes are K-periodic with a period of one. In case 2, the steady-state is reached after the fifth input whereas it is reached after the sixth input in case 3 and case 4.

Figure 5.3 shows some stages of the execution of the net in case 2.

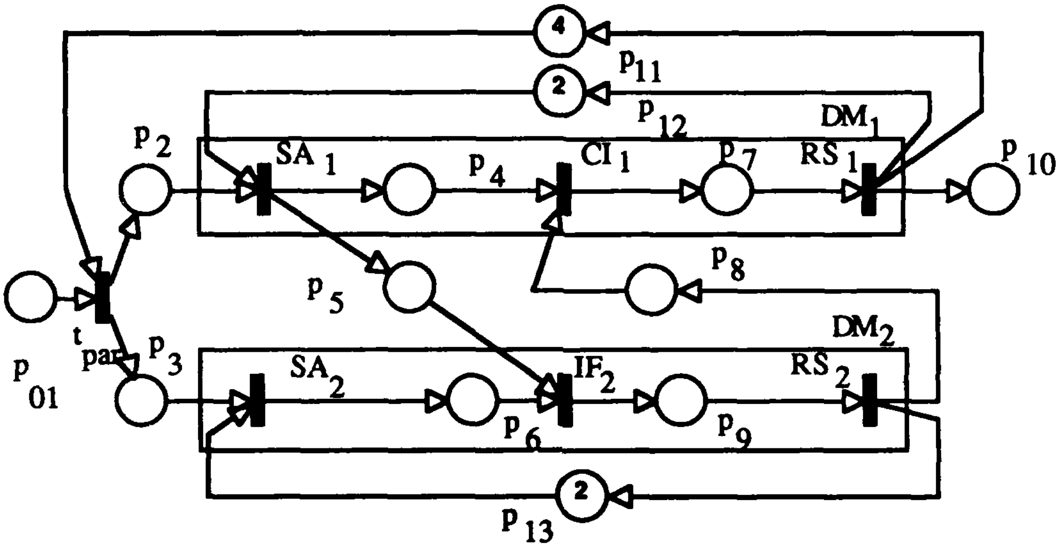


Fig.5.3 (a) Example1, Case 2 - Initial Marking

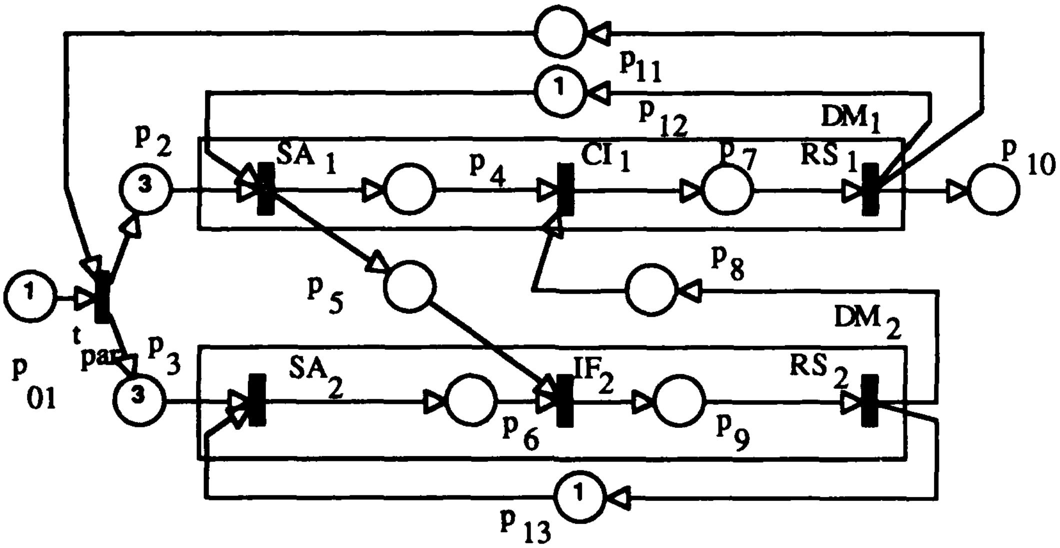


Fig.5.3 (b) Example1, C:ase 2 - Stage 1

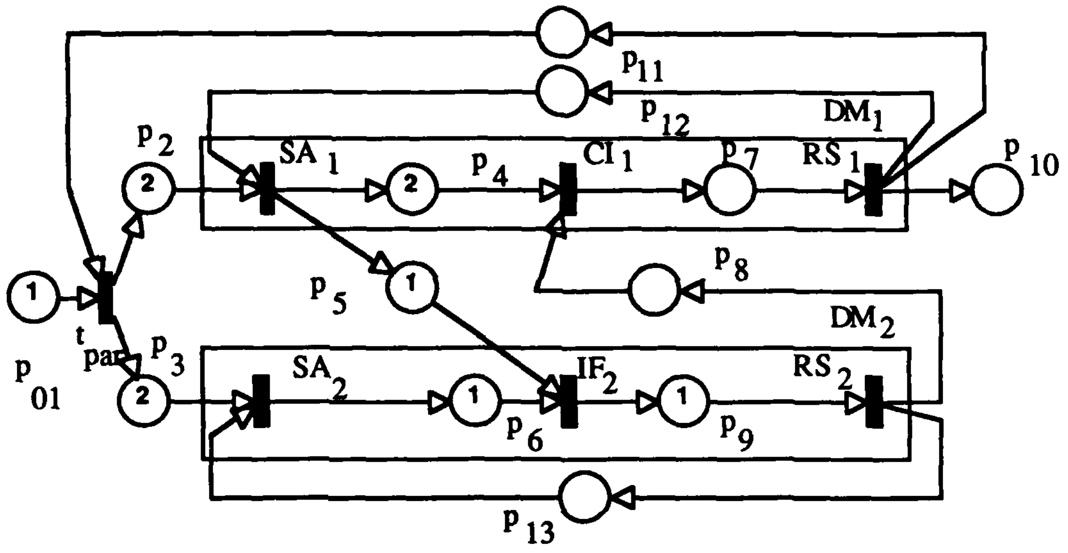


Fig. 5.3 (c) Example 1, Case 2 - Stage 2

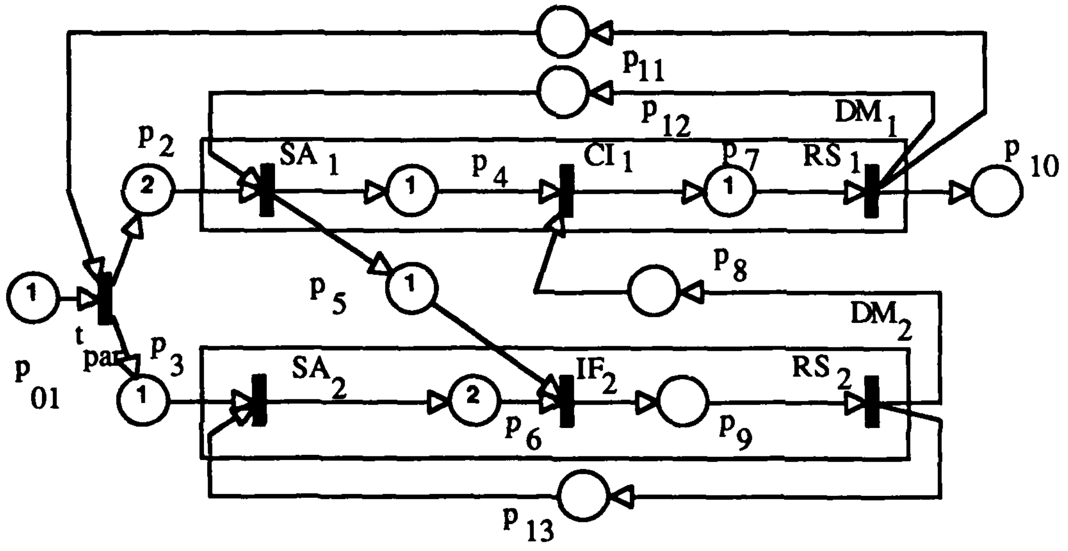


Fig.5.3 (d) Example 1, Case 2 - Stage 3

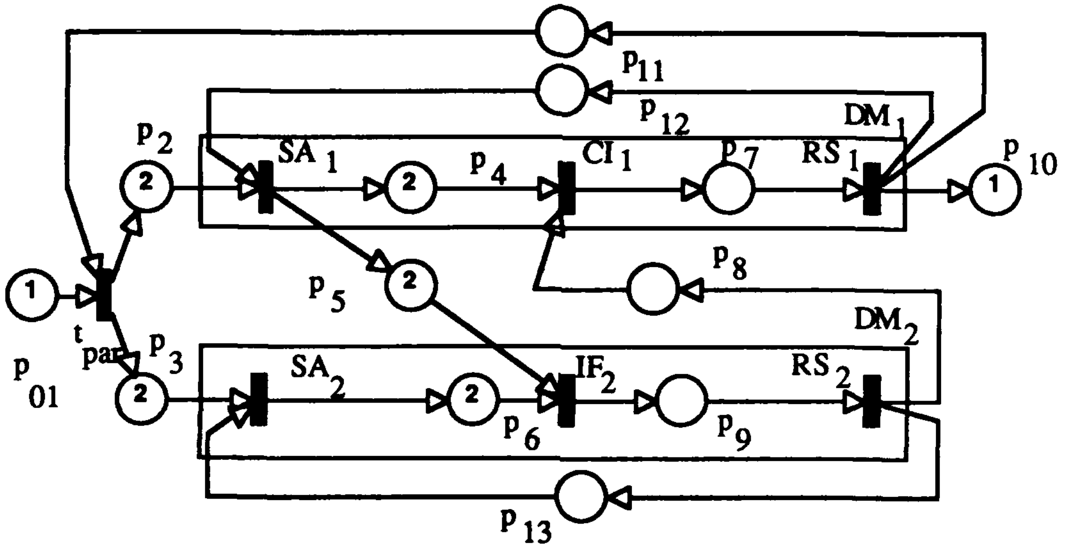


Fig.5.3 (e) Example 1, Case 2 - Stage 4

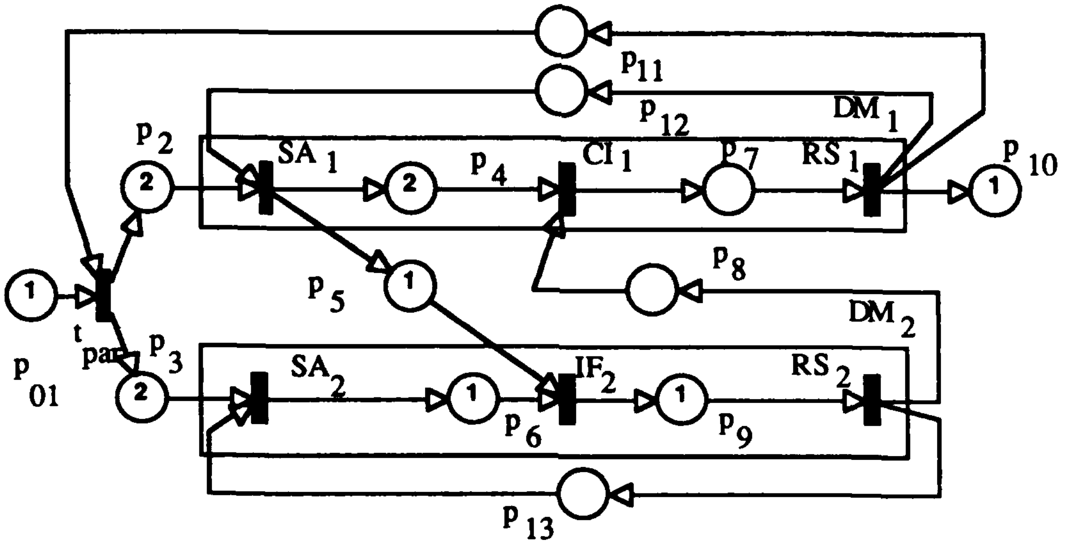


Fig.5.3 (f) Example 1, Case 2 - Stage 5

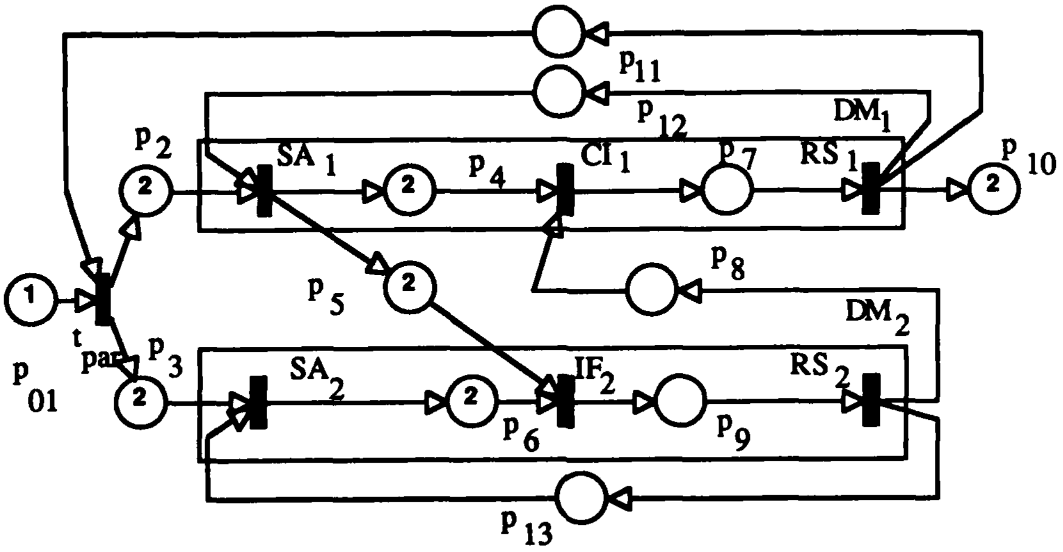


Fig.5.3 (g) Example 1, Case 2 - Stage 6

initial marking: Figure 5.3 (a) shows the representation of the net with the initial marking.

stage 1: In Figure 5.3 (b), three tokens are in the input places of SA_1 and SA_2 while these transitions are still executing for the first token. This results from the fact that their firing times is ten units of time whereas the firing time of t_{par} is only one unit of time. Then, since in case 2, SA_1 uses the LFIFO rule and SA_2 the LLIFO rule, they will process at the next firing tokens with different attributes T_n.

stage 2: In Figure 5.3 (c), IF_2 has fired the first token that entered the net, but cannot fire the next tokens. As explained in stage 1, the token in place p_5 entered the net at $T_c = 1$ whereas the token in p_6 entered the net at $T_c = 3$, and rule 2 of enablement is not satisfied.

stage 3: In Figure 5.3 (d), DM_2 has terminated the processing of the first input; a resource token has consequently appeared in p_{13} and SA_2 was enabled. The token fired by SA_2 entered the net at $T_c = 2$. Therefore, at this point, p_5 contains a token with attribute T_n equal to 1 whereas p_6 contains two tokens with attributes T_n equal to 2 and 3: IF_2 is still not enabled.

stage 4: In Figure 5.3 (e), DM_1 has terminated the processing of the first token.

Consequently, a resource token has appeared in p_{11} and p_{12}. SA_1 was enabled and the token that entered the net at $T_c = 2$ was fired. Therefore, p_5 and p_6 contain now both a token with attribute $T_n = 2$; IF_2 is enabled and can fire. This leads to stage 5.

stage 5: In Figure 5.3 (f), IF_2 has fired. The situation is now the same as in Figure 5.3 (c), with a token present in the sink place p_{10}.

stage 6: In Figure 5.3 (g), the transition IF_2 is again enabled as in stage 4. The process now repeats itself. That is, it goes repeatedly through stages 2, 3, 4, 5 and 6.

However, one can see that the three tokens with attribute $T_n = 1$ are blocked in places p_3, p_4 and p_5, respectively. The processing of the corresponding input is **blocked** as shown in Table 5.1. This happens because there are always two tokens in the input places of SA_1 and SA_2 where the LFIFO and LLIFO rules are used.

In the steady-state, the delays for each input are identical in case 1 and case 4. In case 2, this delay increases by 50 percent. In case 3, the delay is reduced by 10 percent. However, since in the situations where a LLIFO rule is used the processing of one input is blocked, the delay for this input is infinite and the organization can use only three resources out of four for the other inputs. Thus, the thoughput rates decrease . Moreover, for the organizations under consideration in this study, a response must be given to each input in a timely manner. Thus, when the input represents a threat for which a response must be provided in a certain window of opportunity (Cothier and Levis, 1986), the LLIFO rule will degrade considerably the accuracy and timeliness of the organization.

In case 1, the synchronization of the organization is equal to 40 units of time in the steady-state. In case 2, it is equal to 90 and, so, degrades considerably. In case 3, the synchronization is equal to 30 units of time in the steady-state. It represents therefore an improvement with respect to case 1. In case 4, the synchronization in the steady-state is the same as in case 1.

Nevertheless, one must consider also the individual tokens that are blocked during the processing. In case 2 and case 3, the synchronization for the second token degrades considerably with respect to case 1. Indeed, if DM_1 uses the LFIFO rule and DM_2 the LLIFO rule, the item of information for which the process is blocked is in the input place of the SA stage of the latter, whereas it has been assessed by DM_1 and is in the memory place of his CI stage. Thus, the measure S for this input is infinite. The same situation occurs when DM_1 uses the LLIFO rule and DM_2 the LFIFO rule but, in this case, the degradation of the synchronization is due to the fact that DM_2 waits indefinitely in the IF stage for the data from DM_1 to arrive. In case 4, the second input is also blocked, but the two corresponding tokens remain in the input places of SA_1 and SA_2: it implies that none of the decision-makers will wait for the data from the other member for this input. From this standpoint, the synchronization of the activities for this input does not degrade.

The processing of the inputs in these four cases took place for a configuration in which there were four organizational resources and two resources for each decision-maker. The following cases examine a situation in which the organizational resources are increased by one unit. The scenario still corresponds to the infinite queue of inputs and the processing times of the protocols are not changed.

case 1': The initial marking of the resource places is:
$M^0(p_{11}) = 5; M^0(p_{12}) = 2; M^0(p_{13}) = 2.$
Both SA_1 and SA_2 use the LFIFO rule.

case 2': The initial marking of the resource places is:
$M^0(p_{11}) = 5; M^0(p_{12}) = 2; M^0(p_{13}) = 2.$
SA_1 uses the LFIFO rule whereas SA_2 uses the LLIFO rule.

case 3': The initial marking of the resource places is:
$M^0(p_{11}) = 5; M^0(p_{12}) = 2; M^0(p_{13}) = 2.$
SA_1 uses the LLIFO rule whereas SA_2 uses the LFIFO rule.

case 4': The initial marking of the resource places is:
$M^0(p_{11}) = 5; M^0(p_{12}) = 2; M^0(p_{13}) = 2.$
Both SA_1 and SA_2 use the LLIFO rule.

Table 5.2 provides the results for T_i, T_o and T as well as for the synchronization S for the first ten inputs which enter the net in each of these four cases.

In case 1', the process is K-periodic of period 2. In the steady-state, the thoughput rate and the synchronization are identical to case 1. In case 4', the process has a period equal to one. The synchronization and thoughput rate are identical to case 4. However, the second and third inputs remain blocked in the input places of the SA transitions. Thus, the organization can use only three out of its five resources to process the remaining inputs.

In case 2' and case 3', the performance of the organization is totally degraded by the fact that the whole process is blocked. The execution has reached a **deadlock**, i.e., no transition can fire. As it is shown in Table 5.2, five inputs remain in the organization which cannot produce a response for any of them.

Because DM_1 and DM_2 do not use the same strategies, the two items of information sent by DM_1 to DM_2 after the Situation Assessment stage do not correspond to the inputs that DM_2 is processing.

TABLE 5.2 Synchronization and Delay - Cases 1' to 4'

input #	case 1'				case 2'				case 3'				case 4'			
	T_i	T_o	T	S	T_i	T_o	T	S	T_i	T_o	T	S	T_i	T_o	T	S
1	0	51	51	20	0	51	51	20	0	51	51	20	0	51	51	20
2	1	61	60	20	1	-	-	-	1	-	-	-	1	-	-	-
3	2	101	99	40	2	-	-	-	2	-	-	-	2	-	-	-
4	3	111	108	40	3	-	-	-	3	-	-	-	3	101	98	20
5	4	151	147	40	4	-	-	-	4	-	-	-	4	61	57	40
6	51	161	110	40	51	-	-	-	51	-	-	-	61	151	90	40
7	61	201	140	40									101	201	100	40
8	101	211	110	40									151	251	100	40
9	111	251	140	40									201	301	100	40
10	151	261	110	40									251	351	100	40

Thus, since DM_2 has to wait for the information that he needs in order to proceed and since DM_1 has to wait for the commands from DM_2 to arrive, the activities of both decision-makers are blocked. This illustrates a situation where the lack of coordination leads to a severe degradation of the effectiveness of the organization.

Figure 5.4 shows the Petri Net representation of the state of the organization when the deadlock occurs. p_5 and p_6 contain tokens that do not have the same attribute T_n, and consequently, the rule 2 of enablement of transition IF_2 is not satisfied. Since the resource places p_{12} and p_{13} are empty, transitions SA_1 and SA_2 cannot fire and the tokens that have the same attribute T_n as the tokens in p_6 are blocked in p_2.

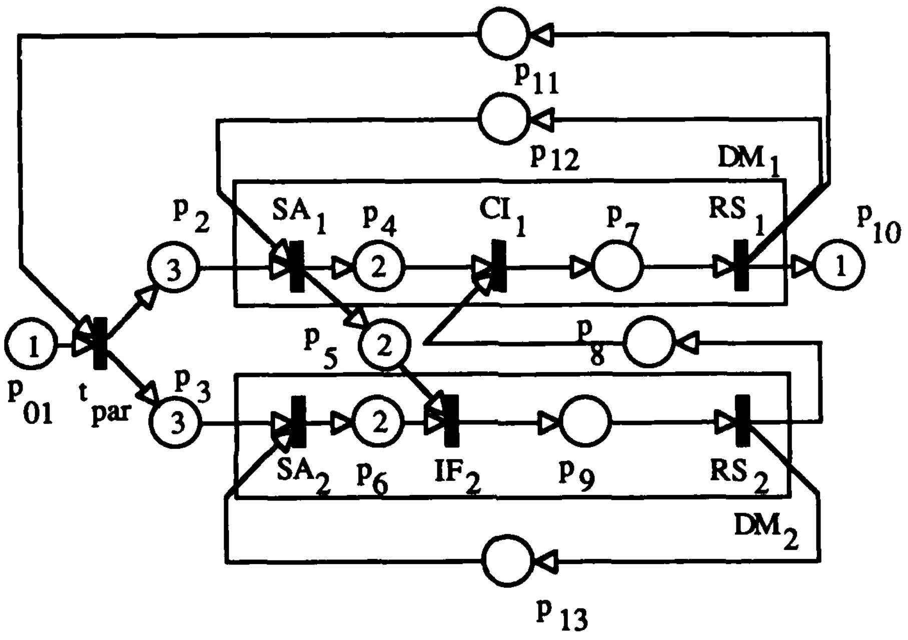


Fig. 5.4 Two-Person Hierarchical Organization with Deadlock

This type of situation would never occur if SA_1 and SA_2 used the LFIFO rule for the sequencing of the inputs: indeed, the interactional transitions would always fire as soon as the places of their preset contain a token since these tokens would necessarily have the same attribute T_n.

Organization with preprocessor

When a preprocessor that aggregates the inputs in zones of indifference is introduced between the source and the Situation Assessment stage of each decision-maker, it is possible for the latter to discriminate between the inputs by using a priority order between the zones.

The scenario used in this example consists of ten inputs that arrive with a constant interarrival time equal to five units of time. It is assumed, moreover, that the preprocessor aggregates these inputs in two classes denoted by C_1 and C_2. The sequence for the inputs is as follows:

$$C_1, C_1, C_1, C_1, C_2, C_2, C_1, C_1, C_1, C_1.$$

The processing time of the preprocessor is assumed to be null.

case 1": The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
Both SA_1 and SA_2 gives the priority to C_1.

case 2": The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
SA_1 gives the priority to C_1 and SA_2 to C_2.

case 3": The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
SA_1 gives the priority to C_2 and SA_2 to C_1.

case 4": The initial marking of the resource places is:
$M^0(p_{11}) = 4$; $M^0(p_{12}) = 2$; $M^0(p_{13}) = 2$.
Both SA_1 and SA_2 gives the prioriry to C_2.

Table 5.3 provides the results for T_i ,T_o and T as well as for the synchronization S for these four cases for the first ten inputs.

When both decision-makers give the same priority order to C_1 and C_2, the synchronization and the delay are degraded for one input : the sixth input in case 1" and the fourth input in case 4". The throughput rate is slighlty better in case 4" since the processing of the tenth input ends at $T_c = 281$, whereas it ends at $T_c = 291$ in case 1". In case 2", the performance of the organization is exactly the same as in case 1" even though the decision-makers do not use the same priority order. It shows that the lack of coordination in the ordering of the inputs does not always result in a degradation of performance: it depends on the scenario that the organization has to cope with. On the other hand, in case 3", the synchronization and the delay are degraded for most of the inputs. In the steady-state, when the inputs arrive in strings of ten inputs as above, the process repeats itself and these results hold.

TABLE 5.3 Synchronization and Delay - Cases 1" to 4"

input #	case 1"				case 2"				case 3"				case 4"			
	T_i	T_o	T	S	T_i	T_o	T	S	T_i	T_o	T	S	T_i	T_o	T	S
1	0	51	51	20	0	51	51	20	0	51	51	20	0	51	51	20
2	5	61	56	20	5	61	56	20	5	61	56	20	5	61	56	20
3	10	101	91	40	10	101	91	40	10	101	91	40	10	101	91	40
4	15	111	96	40	15	111	96	40	15	181	166	110	15	181	166	110
5	51	151	100	40	51	151	100	40	51	131	80	40	51	131	80	40
6	61	291	230	170	61	291	230	170	61	361	300	230	61	161	100	30
7	101	181	80	40	101	181	80	40	101	231	130	90	101	211	110	40
8	111	211	100	30	111	211	100	30	131	281	150	90	131	231	100	40
9	151	241	90	30	151	241	90	30	181	331	150	90	161	261	100	40
10	181	271	90	30	181	271	90	30	231	381	150	90	181	281	100	40

Conclusion

This example illustrates the fact that the dynamics of the decision-making process can lead to a severe degradation of the effectiveness of the organization when the decision-makers do not constitute an ideal team.

On the one hand, the deadlock in cases 2' and 3' shows that when a group of decision-makers do not use identical rules, they should not have to choose between too many inputs; they might be unable to perfom their task if they cannot get the information that they need from the other members of the organization. Thus, if the organizational resources are too numerous with respect to the capabilities of the decision-makers, i.e., their information processing rates, it will not be possible to produce a response for all inputs . For example, if two decision-makers observe a radar screen with multiple data on it, their interactions might be very poorly synchronized, if they do not use some common rule to coordinate their strategies for sequencing the inputs.

On the other hand, the introduction of a preprocessor in cases 1" to 4" shows that the coordination of the activities may be difficult to achieve when the decision-makers do not constitute a perfect team: indeed, if they have different interests or beliefs, the fact that they know the zones of indifference before the Situation Assessment stage can lead them to use different orders to process the inputs.

5.3.3 Example 2

The purpose of this example is to demonstrate that two organizations can perform equally well for a certain scenario and achieve different results for another scenario.

The organization consists of two decision-makers who perform their task in parallel. The processing of each input consists of a Situation Assessment stage and of a Response Selection stage for both of them. They do not interact during the execution of their activities. However, the production of the organizational response is obtained through the fusion of their responses in the Response Fusion stage. The inputs are aggregated in zones of indifference by a preprocessor that assists both decision-makers.

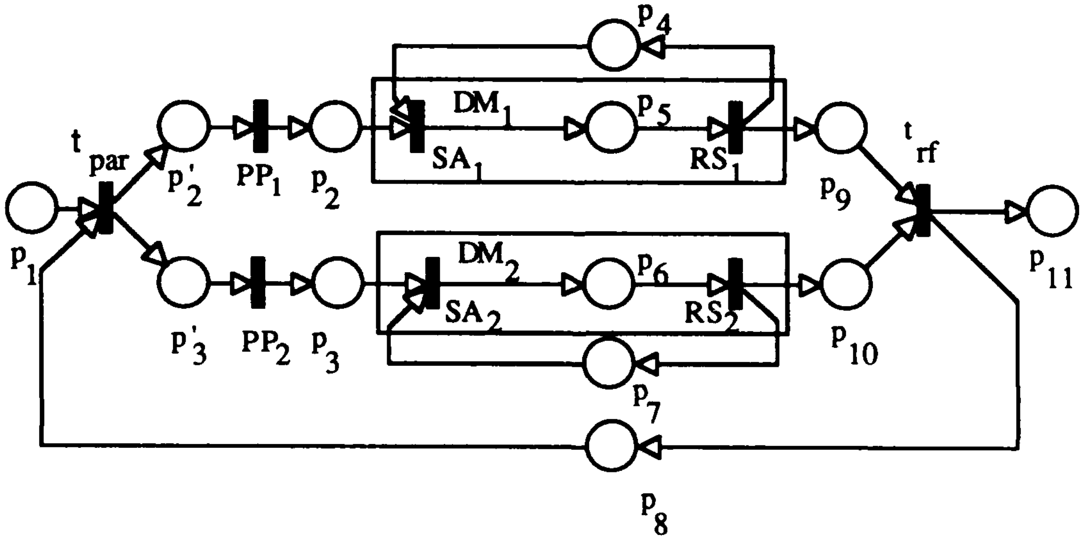


Fig. 5.5 Two-Person Parallel Organization with Preprocessor

In accordance with the definition of the synchronization in decision-making organizations, this organization will always be perfectly synchronized since the decision-makers never wait for information from the other decision-maker. For example, the fact that DM_1 produces his responses earlier than DM_2 will not introduce biases due to the degradation of the value of information for the former.

The processing times of the various stages are kept fixed for the different cases presented below:

	t_{par}	PP_1	PP_2	SA_1	SA_2	RS_1	RS_2	t_{rf}
Time:	0.5	0.5	0.5	2.5	2.5	2.1	2.1	0

The initial marking of the resource places is the following:

$$M^0(p_4) = 3; M^0(p_7) = 3; M^0(p_8) = 3.$$

The two scenarios SC_1 and SC_2 are such that the inputs are partitioned by the preprocessor in three classes C_1, C_2, C_3. In SC_1, the inputs are in an infinite queue and arrive according to the following sequence:

$$C_1, C_1, C_1, C_1, C_2, C_3, C_1, C_2, C_3, \dots .$$

In SC_2, the inputs arrive in the same sequence but by strings of three with an interval of eleven units of time between each string. The first string is:

$$C_1, C_1, C_1$$

The remaining strings are all:

$$C_1, C_2, C_3$$

Four cases are considered below:

case 1''': the scenario is SC_1 and both DM_1 and DM_2 uses the priority order C_1, C_2, C_3.

case 2''': the scenario is SC_1 and DM_1 uses the priority order C_1, C_2, C_3 whereas DM_2 uses the priority order C_3, C_2, C_1.

case 3''': the scenario is SC_2 and both DM_1 and DM_2 uses the priority order C_1, C_2, C_3.

case 4''': the scenario is SC_2 and DM_1 uses the priority order C_1, C_2, C_3 whereas DM_2 uses the priority order C_3, C_2, C_1.

Table 5.4 provides the results for T_i, T_o and T for these four cases for the first nine inputs.

One can see that in case 1''' and case 2''', the process is periodic after the fourth input. In case 3''', it is periodic after the first input. In case 4''', the process is periodic after the third input.

In case 1''' and case 2''', the organization performs equally well even though in the latter case the decision-makers do not adopt the same strategy for the sequencing of the inputs. Indeed, because the queues in the input places of the SA stages have never a length greater than one after the third input, the decision-makers are never confronted with a choice between inputs of different zones (the three first inputs belong to the same zone) .

TABLE 5.4 Delay - Cases 1''' to 4'''

input #	case 1'''			case 2'''			case 3'''			case 4'''		
	T_i	T_o	T	T_i	T_o	T	T_i	T_o	T	T_i	T_o	T
1	0	5.6	5.6	0	5.6	5.6	0	5.6	5.6	0	5.6	5.6
2	0.5	8.1	7.6	0.5	8.1	7.6	0.5	8.1	7.6	0.5	8.1	7.6
3	1	10.6	9.6	1	10.6	9.6	1	10.6	9.6	1	10.6	9.6
4	5.6	13.1	7.5	5.6	13.1	7.5	11	16.6	5.6	11	16.6	5.6
5	8.1	15.6	7.5	8.1	15.6	7.5	11.5	19.1	7.6	11.5	21.6	10.1
6	10.6	18.1	7.5	10.6	18.1	7.5	12	21.6	9.6	12	21.5	9.5
7	13.1	20.6	7.5	13.1	20.6	7.5	22	27.6	5.6	22	27.6	5.6
8	15.6	23.1	7.5	15.6	23.1	7.5	22.5	30.1	7.6	22.5	32.6	10.1
9	18.1	25.6	7.5	18.1	25.6	7.5	23	32.6	9.6	23	32.5	9.5

In case 3''' and case 4''', the scenario is such that for each string of inputs a queue of two information items builds in the input places of the SA stage for both decision-makers. Thus, when they do not adopt the same priority order and the inputs of the queue belong to different zones (after the third input), the delay slightly decreases for certain inputs, e.g., fourth and seventh input, but increases for others, e.g., fifth , sixth , eighth and ninth inputs.

This example shows that the organization, in the case of the infinite queue of inputs, performs equally well whether or not the two decision-makers adopt the same priority order for processing the inputs. However, for the other scenario, the fact that the organization members adopt different priority orders leads to a degradation of the organizational performance with respect to the case where these orders are identical.

5.4 CONCLUSION

This chapter has investigated the dynamics of decision-making processes in the context of the Predicate Transition Net model of coordination developed in Chapter III. Because of the distinguishability of tokens and of the rule of enablement of transitions, the analysis of the execution of such nets is complex. Thus, the simulation of Petri Nets has been introduced as a means of evaluating alternative organizational configurations for different scenarios.

It has been shown that when the decision-makers use different strategies to decide of the order in which they will assess the inputs, the performance can degrade because the interaction stages require from them that they exchange information pertaining to the same inputs.

This could occur, in particular, when decision-makers with different interests or beliefs are aided by a preprocessor that aggregates the inputs in zones of indifference. Therefore, when several decision-makers do not constitute a perfect team, the delay for producing an organizational response will increase for certain scenarios. Furthermore, since the synchronization of the process will degrade, it implies that the various decison-makers will wait for longer periods of time for the information that they need: thus, the value of the information will also degrade leading to the introduction of biases in the responses produced, in addition to the biases due to the algorithms used. Consequently, the responses will be less accurate.

In the remainder of this study, it is assumed that the decision-making process is such that the lack of synchronization is due only to the structural delays that occur because of the processing times of the various protocols. The underlying hypothesis will be that all the decision-makers process the inputs in LFIFO order in all their internal stages.

CHAPTER VI

A MODEL OF A DECISION SUPPORT SYSTEM

This chapter addresses decision support systems (DSS) as well as their relationship to coordination in decision-making organizations. A generic model of a decision-maker interacting with a DSS is presented. The focus is on the architecture of the system and on the different system components that the decision-maker can access.

6.1 INTRODUCTION

The rapid changes in computer technology have led to the development of decision support systems which are integrated in decision-making organizations in order to improve their effectiveness. DSS's have become an increasingly important part of the military Command, Control and Communications (C^3) systems (Waltz and Buede, 1986). In this context, the DSS's, also called battle management systems, automate the fusion of data concerning the tactical situation and the quantitative evaluation of alternative courses of action.

Decision aids are defined as any technique or procedure that restructures the methods by which problems are analyzed, alternatives developed and decisions taken. Keen and Scott Morton (1978) emphasize that decision support systems, a particular form of decision aids, have specific advantages:

"(i) the impact is on decisions in which there is sufficient structure for computer and analytic aids to be of value, but where decision-makers' judgment is essential.

(ii) the payoff is in extending the range and capability of decision-makers' decision processes to help them improve their effectiveness.

(iii) the relevance for decision-makers is the creation of a supportive tool under their own control, which does not attempt to automate the decision process, predefine objectives, or impose solutions".

Thus, DSS's do not automate the decision-making process, but must facilitate it. When confronted with a particular task, the decision-maker keeps the choice of performing it by himself or requesting information from the DSS. This selection depends on the reliability of the DSS or, more exactly, on the extent to which the organization members rely on it.

The evaluation of the effectiveness of a decision-making organization consisting of human decision-makers aided by a DSS is a complex issue: many interrelated factors affect the effectiveness of the overall system, e.g., the limited information processing capacities of the decision-makers, the hardware and software characteristics of the DSS, or the extent to which the organization members use and rely on the decision aid. One important question is to know whether or not the overall organization, when aided by the DSS, is more effective in fulfilling its mission.

Earlier work has assessed the impact of preprocessors (Chyen and Levis, 1985; Weingaertner and Levis, 1987) and databases (Bejjani and Levis, 1985) on the workload of the decision-makers. However, it seems necessary to measure the extent to which the DSS can affect the coordination of the various decision-makers who use it. Indeed, the introduction of a DSS in an organization is a process that can lead either to an improvement or to a degradation of its cohesiveness, depending on the functionality and capabilities of the DSS, as well as on the perception of and access to the DSS that the decision-makers have.

6.2 COORDINATION OF AIDED DECISION-MAKERS

In the case of C^3 organizations, the amount of data that must be handled for a typical mission is very large. For example, the antisubmarine warfare (ASW) mission requires the surveillance of a vast area where multiple sensors gather information on the environment. The typical information requirements in this tactical area (Waltz and Buede, 1986) are the following:

- the surveillance area covers 2000×2000 km.
- the sensor systems consist of 4 surveillance aircrafts, 12 ASW ships, and 2 ASW submarines.
- the number of targets in track can be as high as 200.
- the number of reports per minute ranges from 1000 to 5000.

In this context, there is clear need for a computerized decision-aiding system for the coordination of the activities of the various decision-makers who participate in this process. Such a decision support system can modify the activities of a decision-maker because the latter has to consider the possibility of querying the system (Weingaertner and Levis, 1987). For each input and each stage of his internal decision-making process, the decision-maker must make **meta-decisions** concerning the use of the DSS . These meta-decisions are of three types:

- the decision-maker does not query the DSS and performs all processing by himself.
- the decision-maker sends a query to some component of the system and relies totally on the response.
- the decision-maker sends a query to some component of the system, but compares its response with his own assessment.

When several decision-makers use a DSS for a common task, this system can increase or decrease the coordination of the group:

(i) The DSS is not likely to change the goal of the decision-makers and, therefore, will not alter the commonality of this goal. In the same way, their activities will still require coordination in order to achieve a higher performance. Nevertheless, it is quite possible that the DSS modify the beliefs that the decision-makers have about the environment . For example, if decision-maker DM_1 accesses a database that contain data on the state of the environment, he can get a perception of it which is more realistic than that of decision-maker DM_2 who does not access the same data.

(ii) The consistency of information can be improved, if the decision-makers access a database that provide the same information to all of them. For example, if two decision-makers query in parallel a centralized database in order to assess the same input from the environment, it is likely that they will obtain assessments of the situation that are more consistent than if they perform this assessment alone. On the other hand, if they query the database at different instants, this can lead to a degradation of the consistency of their assessments because the information in the database may have been modified between the two accesses. In the same way, if they query decentralized databases that are not upgraded in a coordinated manner, degradation in consistency will result.

(iii) The synchronization of the activities that take place during the decision-making process can be increased when, for example, the DSS allows the decision-makers who have protocols with long processing times to perform them in a more timely manner so that they are able to communicate with other members of the organization in good synchronization. Conversely, the DSS can lead to a degradation of synchronization when it increases the delays that occur at the interaction stages. Furthermore, if a decision-maker gets enough information by accessing the DSS, he can modify the priority order according to which he processes the inputs. Thus, the DSS is likely to alter the synchronization of the organization both by extending the number of information flow paths with different processing times and by changing the order in which the inputs are processed by the decision-makers.

Consequently, a DSS can have diverse effects on the coordination of an organization. The quantitative evaluation of the effectiveness of an organization aided by a decision support system will be carried out in Chapter VII for an example, under the assumption that the decision-makers are not overloaded.The Measures of Performance that will be investigated are the measure of accuracy, J, the total delay, T, and the measure of synchronization, S_T.

The next section introduces the model of decision support system that will be used.

6.3 MODELING OF A DECISION SUPPORT SYSTEM

6.3.1 Characteristics of the System

It is not possible to define a generic type of decision support system because DSS's are, in general, application-oriented and, therefore, quite specific to the organizations which use them and to the task that must be performed.

The following model takes into account several capabilities and characteristics which are common to most of the real systems. In particular, it takes into consideration the fact that most real DSS's are hybrid systems that contain both elements of centralization and decentralization, i.e., that have facilities shared by several users and facilities accessed individually. Its architecture is depicted in Figure 6.1.

From a physical standpoint, the DSS consists of a mainframe shared by the organization and which is accessed by the decision-makers through remote intelligent terminals and a communication network. The terminals are called "intelligent" to the extent that they provide the users with the opportunity to do local processing without querying the central system.

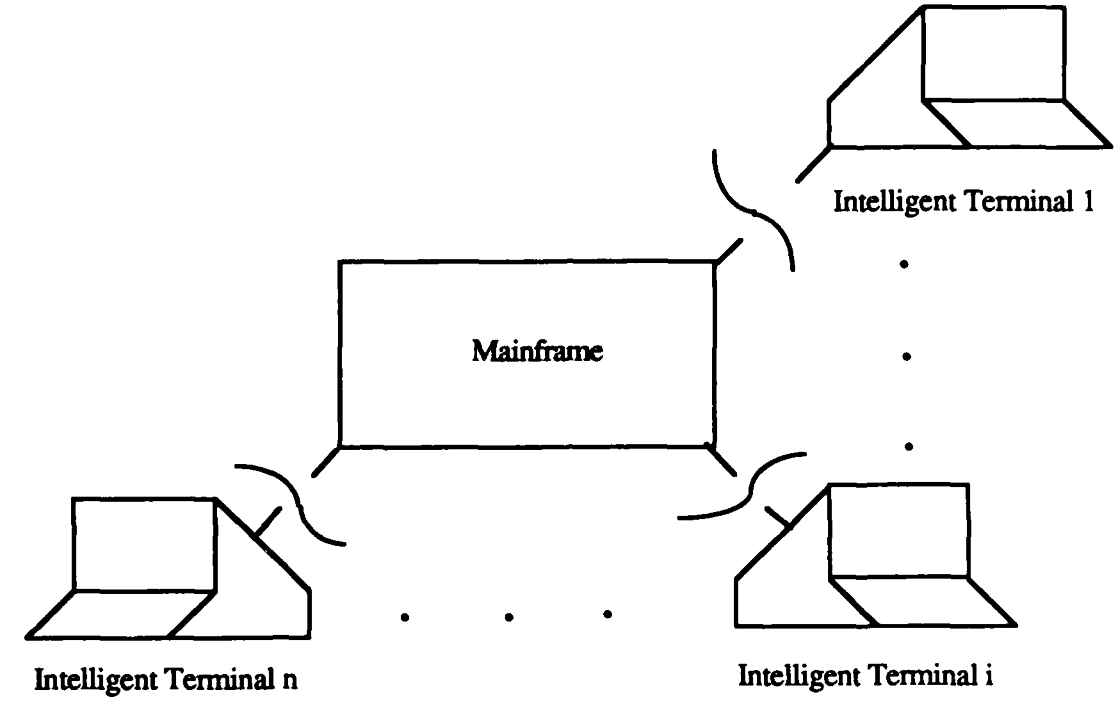


Fig. 6.1 Architecture of the Decision Support System

The DSS provides a multiple-access capability to the decision-makers who can query it in parallel. Several databases are stored in the mainframe so that a decision-maker can get information concerning the state of the environment as well as concerning the possible responses that he can give to any input: it implies that the decision-maker can query the database both in his Situation Asessment stage and in his Response Selection stage.

The applications implemented on the system do not embody any heuristic and do not develop alternative solutions. They implement models and doctrines well-known to the decision-makers. Consequently, the processing of any particular information by a decision-maker DM_i involves some or all of the four essential components described in Figure 6.2: the decision-maker DM_i, the intelligent terminal i that he uses, the communication network, and the mainframe.

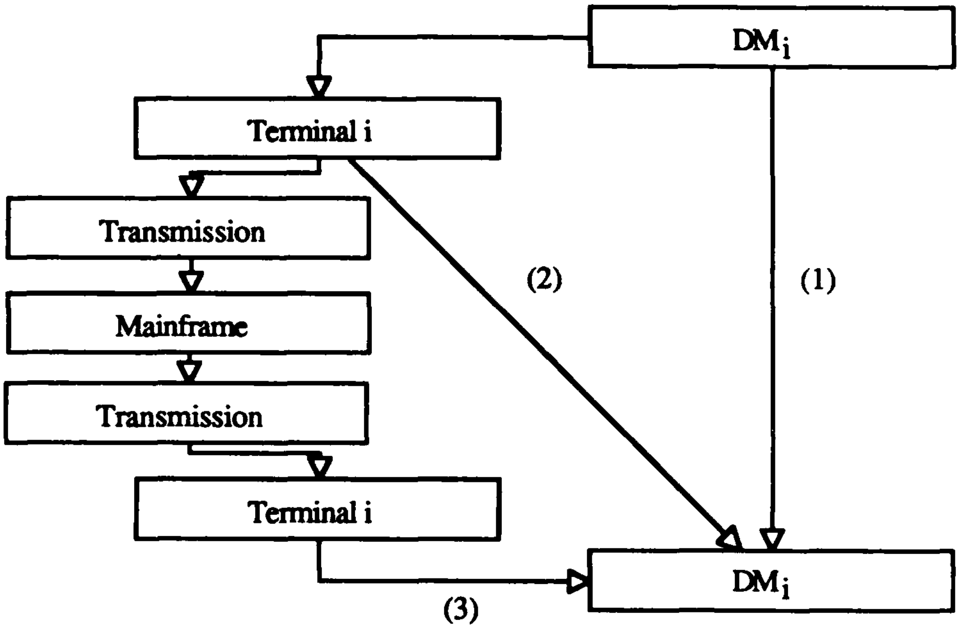


Fig. 6.2 DM_i Interacting with the DSS

For each of the three paths illustrated above, the amount of time that it takes to process the input for each internal stage of DM_i depends on several factors:

(i) in path 1, the decision-maker processes the information by himself; this takes an amount of time equal to the processing time of the corresponding protocol.

(ii) in path 2, the decision-maker uses only the intelligent terminal. The total amount of time taken by this operation corresponds to the sum of the following delays:

- time spent by the decision-maker to query the terminal;
- time spent by the terminal to process and display the information;
- time spent by the decision-maker to assess the response.

(iii) in path 3, the decision-maker uses the terminal as a dumb terminal to query the mainframe. The total delay of this operation is the sum of the following delays:

- time spent by the decision-maker to query the mainframe;
- time spent by the terminal to access the network;
- time of transmission to the mainframe;
- time spent by the mainframe to recognize the query and initiate the processing;

- time spent by the mainframe to process the information;
- time spent by the mainframe to access the network;
- time of transmission to the terminal;
- time spent by the terminal to display the information;
- time spent by the decision-maker to assess the response.

The use of the mainframe involves the execution of operations that can take an amount of time which depends to a large extent on the physical configuration of the system. In particular, the delay of transmission through the communication network can vary over a wide range according to the specific route use which depends, in turn, on the origin and the destination.

Furthermore, a query to the mainframe may be much more subject to errors due to noise and the distortion in the transmission than a query to the intelligent terminal.

On the other hand, the mainframe contains databases with information on the state of the environment and, thus, accessing it may be quite useful in providing responses that are more accurate.

6.3.2 Petri Net Model of Decision-Maker Aided by a DSS

The Petri Net model of a decision-maker DM aided by a DSS is given in Figure 6.3. This model represents the different information flow paths that exist when a decision-maker interacts with the DSS for any internal stage of his decision-making process.

Figure 6.3 illustrates the information flow paths for the case where the DM uses only one algorithm f for performing his task. The symbols in the figure represent the following:

- u is the decision variable for choosing between the five alternatives:

(1) DM performs the stage by himself.

(2) DM queries the mainframe, performs his own processing, and compares the two results.

(3) DM queries the intelligent terminal, performs his own processing, and compares the two results.

(4) DM queries the mainframe and relies on its response.

(5) DM queries the intelligent terminal and relies on its response.

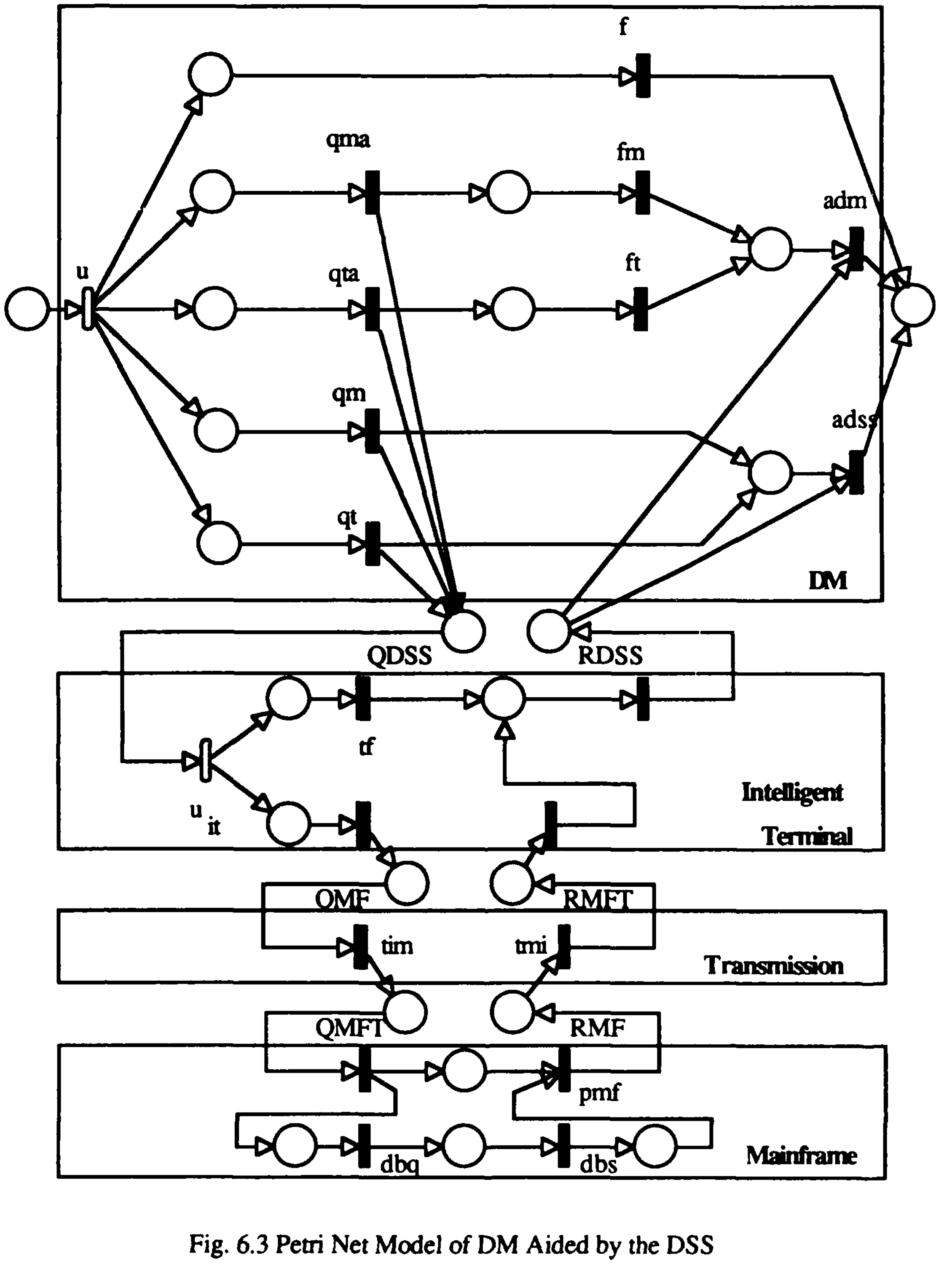


Fig. 6.3 Petri Net Model of DM Aided by the DSS

- qma is the algorithm used by DM to query the mainframe in alternative 2.
- qta is the algorithm used by DM to query the intelligent terminal in alternative 3.
- qm is the algorithm used by DM to query the mainframe in alternative 4.
- qt is the algorithm used by DM to query the mainframe in alternative 5.
- fm is the algorithm that DM executes when he has queried the mainframe in alternative 2.
- ft is the algorithm that DM executes when he has queried the intelligent terminal in alternative 3.
- adm is the algorithm used by DM to assess the response of the DSS and to compare it with the result of his own processing in alternatives 2 and 3.
- adss is the algorithm used by DM to assess the response of the DSS in alternatives 4 and 5.
- QDSS is the query sent by DM to the DSS.
- RDSS is the response sent by the DSS to DM.
- u_{it} is the decision variable which determines whether the intelligent terminal or the mainframe must process the query.
- tf is the algorithm performed by the intelligent terminal to process the query.
- QMF is the query sent by the intelligent terminal to the mainframe.
- RMFT is the response from the mainframe transmitted by the network to the intelligent terminal.
- tim is the protocol of transmission from the intelligent terminal to the mainframe.
- tmi is the protocol of transmission from the mainframe to the intelligent terminal.
- QMFT is the query from the intelligent terminal transmitted by the network to the mainframe.
- RMF is the response from the mainframe.
- pmf is the algorithm performed by the mainframe for processing the query.
- dbq is the algorithm that queries the database.
- dbs is the algorithm that performs the search in the database.

The decision-maker has five alternatives for performing the processing of the input. In the Situation Assessment stage, the probability of having any of these alternatives can depend on the nature of the input x if there is some preprocessor between the source and the decision-maker, i.e., prob(u|x). It can also be indenpendent of the input, i.e., prob(u). In any other stage, the choice can be conditioned by the information that the decision-maker knows before executing the stage.

This model shows that the decision-maker interacts with the DSS by fusing the information that the latter produces.Therefore, it is possible to evaluate the synchronization between DM and the DSS to know whether or not the decision-maker waits for long delays before he receives the responses to his queries.

The places labelled QDSS and RDSS represent the structural places that contain the information exchanged by DM and the DSS. In accordance with the Predicate Transition Net model developed in Chapter III, the transitions adm and adss are the only interactional transitions. These transitions will fire only if the tokens in their input places have the same attribute T_n, i.e., they correspond to the same input from the environment. The measure of the synchronization between DM and the DSS would evaluate, for each input and each stage, the sojourn time of the item of information in the memory place of the preset of adm or adss.

It has been assumed here that the DSS provides a response for each query: therefore, the decision-maker is certain that he will always obtain a response to the query that he has made. Furthermore, since this model assumes that a decision-maker can process only one input at a time in any of his internal stages, the number of items of information present at the same time in the net illustrated above is equal to zero or one. It implies that the DM waits for the response to the query before processing any other information in the corresponding stage. Consequently, the lack of synchronization between the DM and the DSS should not alter to a large extent the effectiveness of the organization since the responses from the DSS always present the same value to DM. In the remainder of the study, it is assumed that the synchronization between DM and DSS is perfect.

6.4 CONCLUSION

This chapter has presented a model of a decision support system in decision-making organizations. A generic Petri Net model of a decision support system has been described that differentiates between the different information flow paths that an input can follow. These depend on the meta-decision that the decision-maker makes concerning the use of the decision support system, i.e., whether he accesses the intelligent terminal, the mainframe or performs the processing by himself. The next chapter presents an application of this model.

CHAPTER VII

APPLICATION

This chapter presents the evaluation of the coordination of a decision-making organization aided by a decision support system. Three measures of performance are computed, i.e., the accuracy, J, the expected delay, T, and the synchronization S_T. The performance loci are constructed for different cases: on this basis, the relation between accuracy, expected delay and synchronization is investigated.

7.1 INTRODUCTION

The impact of a decision support system, DSS, on the coordination of a two-person organization is the key question addressed in this chapter. As stated in Chapter V, the degradation of the synchronization of a decision-making organization can result from two types of factors:

- the dynamics of the activities which lead the decision-makers to process various inputs with different priority orders.
- the information flow paths that each decision-maker uses to perform his task.

In Chapter V, the impact of the first category of factors on the decision-making process was investigated and, in order to do so, a simulation program for executing Petri Nets was introduced. This chapter assesses the second type of factors, i.e., it measures the extent to which the existence of information flow paths with different processing times can degrade the synchronization of the decision-making activities.

This type of situation arises when the decision-makers are provided with a DSS which allows them to access different local or remote computer facilities. As introduced in Chapter VI, DSS's can alter significantly the coordination of the activities, depending on the configuration of the system with respect to the organization.

The objective of this chapter is to assess quantitatively for a specific example the impact of the DSS on the accuracy, timeliness and synchronization of the decision-making process.

7.2 THE ORGANIZATION AND THE TASK

7.2.1 Introduction

The large amount of data (cf. Chapter VI) that have to be handled by antisubmarine-warfare systems shows the importance of coordinating the activities of the various participants in the process. The example presented in this section aims at modeling the organizational structure and decision-making activities in such a context for a simple case, i.e., a two-person organization. The task models a mission of surveillance that consists of listening to detect enemy submarines. In such an environment, the use of decision support systems to process the signals and discriminate between them is necessary.

7.2.2 The Organization

The organization consists of a submarine and a surface ship which are in charge of tracking enemy submarines. It is a hierarchical organization where the submarine is the subordinate and the surface ship the commander. This example has been studied from another standpoint by Papastavrou (1986). The Petri Net model of such an organization is presented in Figure 7.1.

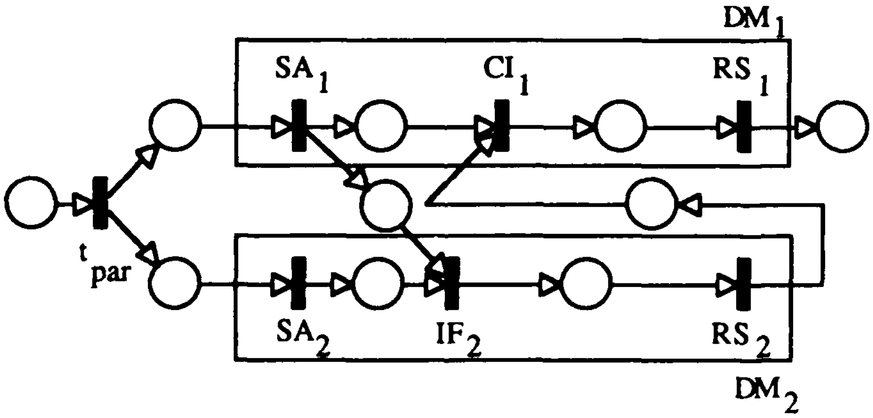


Fig. 7.1 Petri Net Model of Subordinate (DM_1) and Commander (DM_2)

The decision-making process of the commander and the subordinate have three stages each. In the Situation Assessment stages, they assess the signals that they receive from the environment. The subordinate sends the result of his own assessment to the commander, who fuses in the Information Fusion stage this information with his own assessment. On the basis of the result of this interaction, the commander identifies the signal and produces an order which is the sent to the subordinate. The latter interprets the order in the Command Interpretation stage and produces the organizational response.

7.2.3 The Task

The task is modeled as the alphabet X and the probability distribution prob(x) such that:

- $X = \{x_i = a_i b_i c_i d_i e_i f_i \,/\, (a_i, b_i, c_i, d_i, e_i, f_i) \in \{0,1\}^6\}$
- $\forall\ (x_i, x_j) \in X \times X,\ \text{prob}(x_i) = \text{prob}(x_j)$

Therefore, each input consists of an ordered string of six bits. There are 64 possible inputs. These inputs represent the signals that must be identified by the organization in order to produce the response. It is assumed, furthermore, that these inputs are equiprobable, so that the probability distribution prob(x) is defined by:

$$\forall\, x_i \in X,\ \text{prob}(x = x_i) = \frac{1}{64} \tag{7.1}$$

The organization can produce four responses, labelled R_1, R_2, R_3 and R_4:

- if the bits a_i and d_i are both equal to 0, the signal does not come from an enemy submarine, and therefore, the submarine DM_1 should not do anything. This response is R_1. The probability of having such an input is 1/4.

- if b_i and e_i are both equal to 0, the signal comes from an enemy submarine which is trying to test the capabilities of submarine DM_1. This one should deceive it by underreacting. This response is R_2. The probability of having such an input is 3/16.

- if the bits c_i and d_i are both equal to 0, the signal comes from an enemy submarine which is moderatly threatening submarine DM_1. The latter should overreact to this threat to deter the enemy submarine. This response is R_3. The probability of having such an input is 9/64.

- otherwise, the signal comes from an enemy submarine which is threatening submarine DM_1. In this case, DM_1 should also overreact but at a higher level than previously. This response is R_4. The probability of having such an input is 27/64.

The following table summarizes these possibilities:

TABLE 7.1 Organizational Responses

input	response
$(a_i, d_i) = (0, 0)$	R_1
$(a_i, d_i) \neq (0, 0)$ $(b_i, e_i) = (0, 0)$	R_2
$(a_i, d_i) \neq (0, 0)$ $(b_i, e_i) \neq (0, 0)$ $(c_i, f_i) = (0, 0)$	R_3
$(a_i, d_i) \neq (0, 0)$ $(b_i, e_i) \neq (0, 0)$ $(c_i, f_i) \neq (0, 0)$	R_4

The partitioning of the input is done according to the following rule:

- the submarine, DM_1, receives the first three bits $a_ib_ic_i$.
- the surface ship, DM_2, receives the last three bits $d_ie_if_i$.

The decision-making process takes place on the basis of this partitioning. The following table presents the cost matrix used in this example. It gives the costs associated with the discrepancies between the ideal responses and the actual responses provided by the organization.

TABLE 7.2 Cost Matrix

actual R ideal R	R_1	R_2	R_3	R_4
R_1	0	2	3	4
R_2	4	0	1	2
R_3	6	4	0	2
R_4	8	4	3	0

The computation of the accuracy, J, of the organization is done in accordance with the formula introduced in Chapter II.

It is assumed that, when DM_1 and DM_2 assess the input by themselves, without querying the DSS, they produce correctly the first two bits of the strings of three bits. That is, for an input $x_i = a_ib_ic_id_ie_if_i$, the result of the SA of DM_1 is $a_ib_iu_i$ where u_i is the value of the third bit that DM_1 produces : it is assumed that this value is equal to c_i with probability 1/2. In the same way, the result of the SA of DM_2 is $d_ie_iv_i$ where v_i is the value of the sixth bit that DM_2 produces: this value is equal to f_i with probability 1/2.

7.2.4 Organization Aided by the DSS

The model of the decision-maker aided by the DSS was presented in Chapter VI. It is assumed that the decision-makers query the DSS only during their Situation Assessment stages. Figures 7.2 and 7.3 provide the models of the organization aided by the DSS: in Figure 7.2, the model of DSS is aggregated; in Figure 7.3, the whole model is shown.

In the model of the decision-maker DM_i aided by a DSS introduced in Chapter VI, DM_i has five alternatives to perform his processing in any internal stage where he can use the DSS. The Petri Net representation of the organization aided by the DSS introduced in the previous figures show that only three of these five alternatives are considered:

(i) DM_i does not access the DSS but processes the information alone.

(ii) DM_i queries the intelligent terminal and relies on its response.

(iii) DM_i queries the DSS and compares its response to his own assessment.

In the remainder of this section, the following notation will hold:

- SA_i represents alternative (i).
- IT_i represents alternative (ii).
- MF_i represents alternative (iii).

This model shows that multiple flow paths can be used to process the information. Each decision-maker has three alternatives with respect to the use of the DSS, and, therefore, there are nine pure organizational strategies:

- (SA_1, SA_2)
- (SA_1, IT_2)
- (SA_1, MF_2)
- (IT_1, SA_2)
- (IT_1, IT_2)
- (IT_1, MF_2)
- (MF_1, SA_2)
- (MF_1, IT_2)
- (MF_1, MF_2)

A mixed strategy $\delta_i(p_i^1, p_i^2, p_i^3)$ for decision-maker DM_i corresponds to a certain combination of his three pure strategies SA_i, IT_i and MF_i weighted by the probabilities p_i^1, p_i^2, p_i^3.

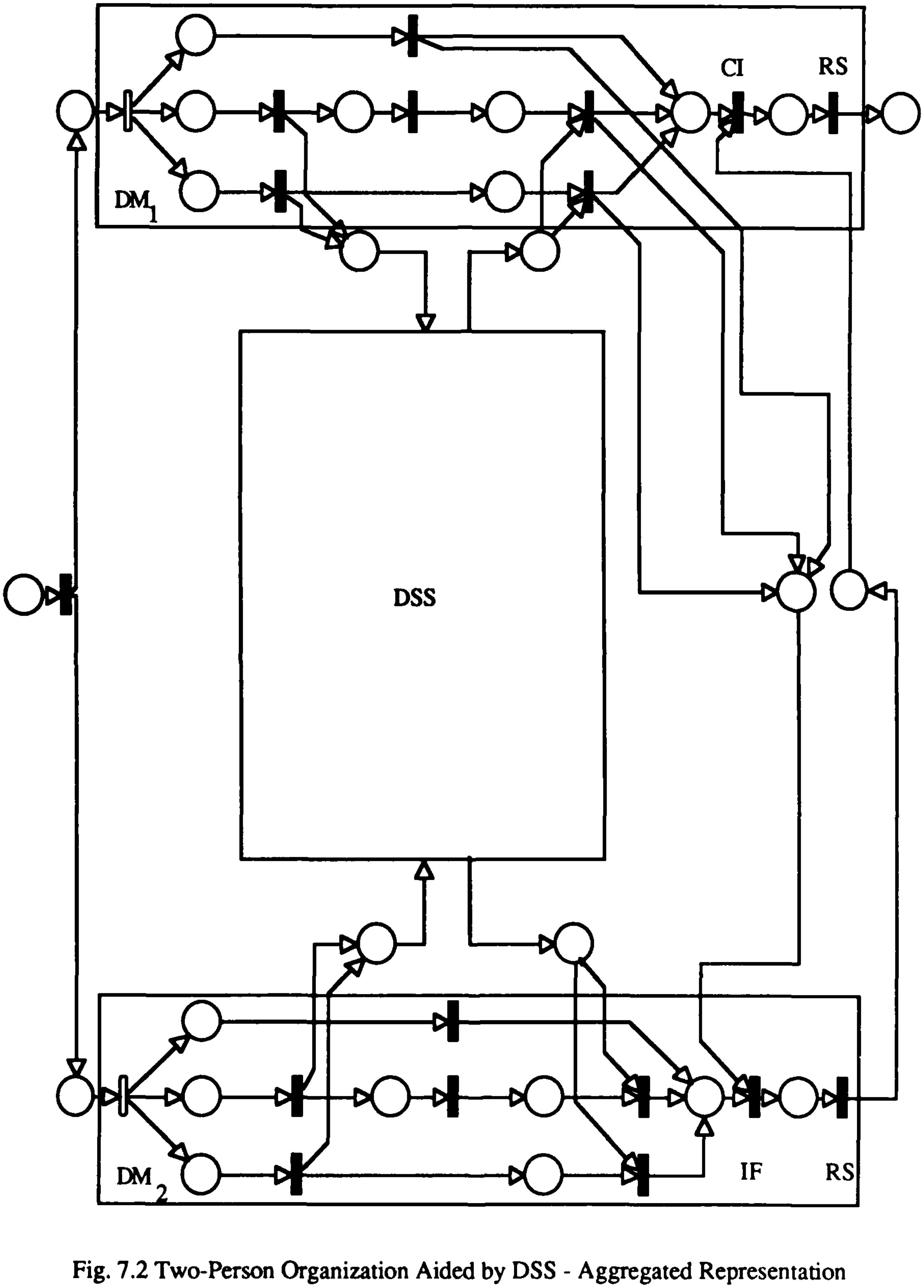


Fig. 7.2 Two-Person Organization Aided by DSS - Aggregated Representation

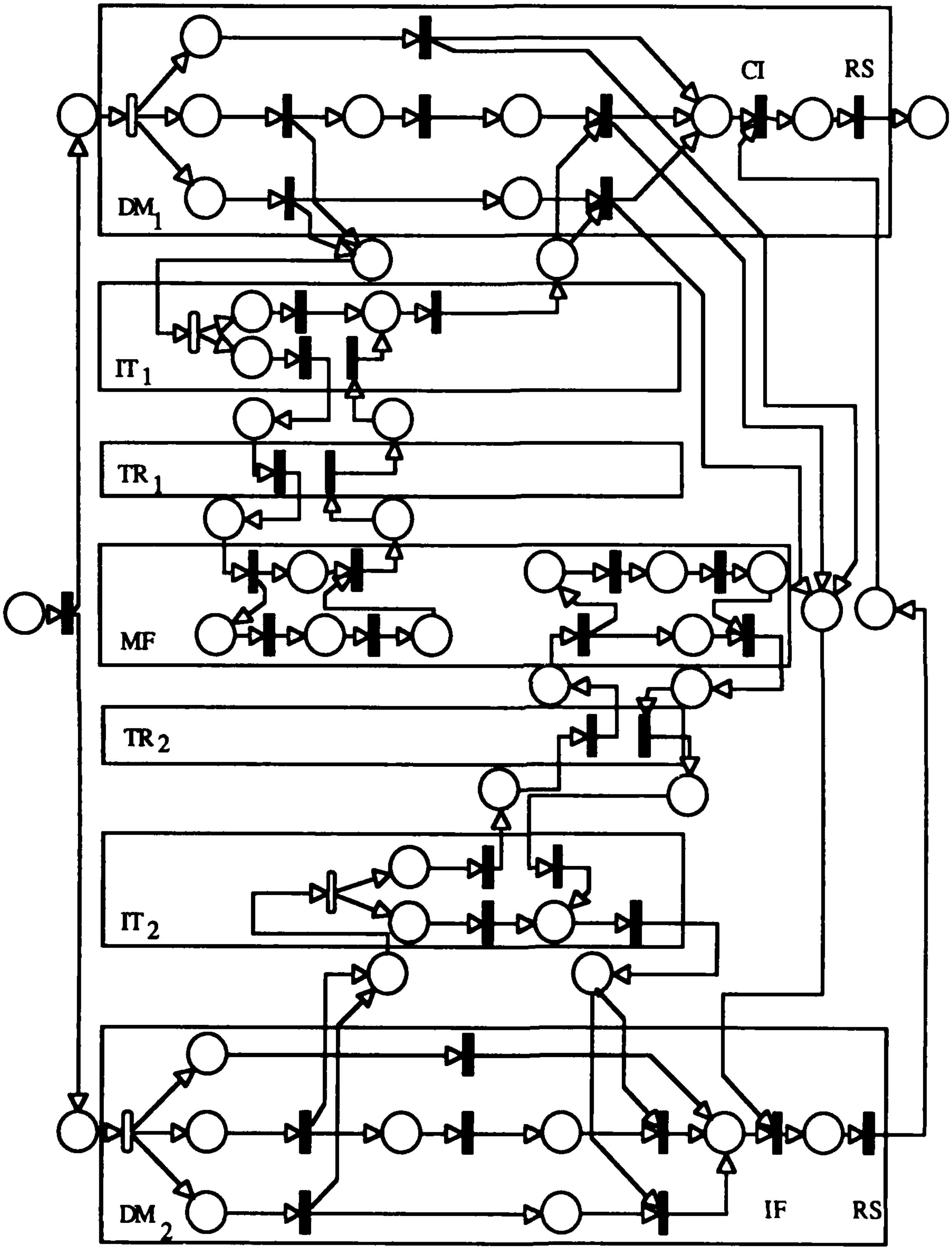


Fig. 7.3 Two-person Organization Aided by DSS

An organizational behavioral strategy is the combination of the mixed strategies of DM_1 and DM_2. Therefore, it corresponds to $(\delta_1(p_1{}^1, p_1{}^2, p_1{}^3), \delta_2(p_2{}^1, p_2{}^2, p_2{}^3))$.

It is assumed that the processing of information through the use of the DSS provides different results depending on whether the intelligent terminal or the mainframe is queried. When the intelligent terminal is accessed, the decision-makers can produce correctly the first bit of the strings of three bits. That is, for an input $x_i = a_ib_ic_id_ie_if_i$, the result of the SA of DM_1 for the alternative IT_1 is $a_iu_iy_i$ where u_i and y_i are the values of the second and third bits that DM_1 produces: each of these two values is equal to the actual value with probability 1/2. In the same way, the result of the SA of DM_2 for the alternative IT_2 is $d_iv_iz_i$ where v_i and z_i are the values of the fifth and sixth bits that DM_2 produces: each of them corresponds to the actual value with probability 1/2.

When the mainframe is accessed, the decision-makers are able to produce correctly all three bits of their respective strings. That is, for an input $x_i = a_ib_ic_id_ie_if_i$, the result of the SA stage of DM_1 for the alternative MF_1 is $a_ib_ic_i$. The result of the SA stage of DM_2 for the alternative MF_2 is $d_ie_if_i$.

This means that, when the organizational strategy is (MF_1, MF_2), the organization will be able to produce the correct response for all inputs. For all other strategies, the responses provided may differ from the ideal response.

The access to the intelligent terminal provides, however, a means of improving the timeliness of the decision-making process. Indeed, it will be assumed that the amount of time necessary to process the information is lower when the decision-maker uses his intelligent terminal than when he queries the mainframe or performs his processing alone.

In the remainder of this chapter, the amounts of time taken by the decision-makers to perform the different algorithms are all equal to one unit of time, except for the Situation Assessment algorithms. The different cases that will be investigated concerning the processing times of these algorithms are presented in the next section.

These considerations account for what occurs in most situations in Command and Control systems. The different decision-makers have access to different facilities which do not have the same response time or the same accuracy. On the one hand, an intelligent terminal is likely to provide faster responses because it is co-located with the decision-maker. However, it has no centralized database which can aggregate data from multiple sensors to get a global picture of the situation and, therefore, the responses it can provide are necessarily less accurate. On the other hand, the access to the mainframe may require the communication of data from and to remote locations through a network: the response time can be quite long.

The next section presents the results obtained for different access times to the mainframe. In each case, the performance loci have been constructed for the three measures, J, T, S_T.

7.3 RESULTS AND INTERPRETATION

7.3.1 Results

The results on the accuracy of the responses produced by the organization for the nine pure organizational strategies, are listed in Table 7.3. We notice that the accuracy is maximal when both decision-makers query the mainframe. The accuracy reaches its worst level when they both query their intelligent terminal.

TABLE 7.3 Accuracy of the Organization

strategy	SA_1	SA_1	SA_1	IT_1	IT_1	IT_1	MF_1	MF_1	MF_1
	SA_2	IT_2	MF_2	SA_2	IT_2	MF_2	SA_2	IT_2	MF_2
J	0.42	0.76	0.21	0.76	1.01	0.69	0.21	0.69	0.00

Two cases have been investigated as far as the processing times of the Situation Assessment stages are concerned:

(i) case 1: SA_1 and SA_2 take 10 units of time. IT_1 and IT_2 take 5 units of time. MF_1 and MF_2 take 15 units of time. This case corresponds to the situation where the processing times of both decision-makers are equal when they use the same strategy with respect to the use of the DSS.

(ii) case 2: SA_1 and SA_2 take 10 units of time. IT_1 and IT_2 take 5 units of time. MF_1 takes 15 units of time but MF_2 takes 10 units of time. This corresponds to the situation where the commander has a faster access to the mainframe than the subordinate because of a better transmission time.

One can notice that, in both cases, when the two-decision makers perform their situation assessment by themselves, they take the same amount of time to do it. It implies that at the first interaction, i.e., the Information Fusion stage of DM_2, this one will not have to wait for the information from DM_1: they are perfectly synchronized for this interaction.

Tables 7.4 and 7.5 show the results for the expected delay, T, and the synchronization, S_T, in case 1 and case 2, for the nine pure organizational strategies.

TABLE 7.4 Delay and Synchronization in Case 1

strategy	SA_1 SA_2	SA_1 IT_2	SA_1 MF_2	IT_1 SA_2	IT_1 IT_2	IT_1 MF_2	MF_1 SA_2	MF_1 IT_2	MF_1 MF_2
T	15	15	20	15	10	20	20	20	20
S_T	2	7	7	7	2	12	7	12	2

In case 1, the maximum delay is obtained when at least one of the decision-makers accesses the mainframe. The minimum delay is reached when both decision-makers use their intelligent terminal. It can be noticed that, when the maximum delay is reached, the synchronization can have very different values depending on the coordination of the strategies of the decision-makers. When they both access the mainframe, the synchronization is optimal for the delay of 20 units of time. For this same delay, this synchronization can degrade considerably, if one of them accesses his intelligent terminal as the other queries the mainframe.

TABLE 7.5 Delay and Synchronization in Case 2

strategy	SA_1 SA_2	SA_1 IT_2	SA_1 MF_2	IT_1 SA_2	IT_1 IT_2	IT_1 MF_2	MF_1 SA_2	MF_1 IT_2	MF_1 MF_2
T	15	15	15	15	10	15	20	20	20
S_T	2	7	2	7	2	7	7	12	7

In case 2, the maximum delay is reached when DM_1 accesses the mainframe. Nevertheless, when DM_2 queries the mainframe alone, the delay does not increase to this level. It can also be noticed that the optimal synchronization can no longer be obtained when the delay is maximal. Furthermore, the worst value for the synchronization is reached only for one pure organizational strategy, i.e., (MF_1, IT_2). This value was reached for two pure organizational strategies in case 1, i.e., (MF_1, IT_2) and (IT_1, MF_2).

The interpretation of the impact of the DSS on the three Measures of Performance is carried out in the next section.

7.3.2 Interpretation

The interpretation of the results obtained in the previous section can be done through the consideration of the performance loci for the measures J, T and S_T.

The performance loci for the two cases presented in the previous section are shown in Figures 7.4 and 7.5. They represent the values of J, T and S_T reached for each organizational strategy, pure or behavioral.

These figures show the relations between the various measures of performance. It is recalled that:

- the lower the value of T, the better the delay.

- the lower the value of S_T, the better the synchronization.
- the lower the value of J, the better the accuracy.

Before synthesizing the results, some facts concerning these loci must be first noted:

(i) One can see that, in case 2, the part of the locus where J is the lowest, i.e., where the accuracy is the best, corresponds to higher values of S_T than in case 1. It shows that there exists a trade-off between accuracy and synchronization when the DSS does not have the same response times for the two decision-makers.

(ii) In both cases, when the expected delay, T, is minimal, the synchronization, S_T, is also minimal. This is due to the fact that the intelligent terminals provide the fastest way of performing Situation Assessment, and that the delay will be minimal only if both decision-makers query their terminal. The assumption that these terminals give the responses to both decision-makers in the same amount of time is realistic because there is no delay due to transmission and the algorithms that they use are similar.

(iii) Conversely, the fact that the synchronization is minimal does not imply that the delay will be minimal. In case 1, the synchronization reaches its lowest value for all possible values of the delay. It corresponds to the fact that, for any delay, the decision-makers can find some way to be as well synchronized as possible.

(iv) If a constraint is imposed on the delay, the synchronization of the organization does not degrade. One can notice that the more stringent the constraint on T, the more likely the synchronization will reach a good value. In case 2, the synchronization does not reach its lowest value for all values of T : as in case 1, the best values of S_T are obtained for the lowest delays.

(v) In case 2, the more the timeliness of the organization degrades, the more the synchronization will degrade too. It derives from the fact that DM_1 is not provided with the same access quality as DM_2: the amount of time taken by his queries to be transmitted to the mainframe is substantially more important than for DM_2. When he uses the mainframe, there is no way for the organization to be well synchronized. DM_2 will have to wait for long amounts of time before receiving the data that he needs in his information fusion stage.

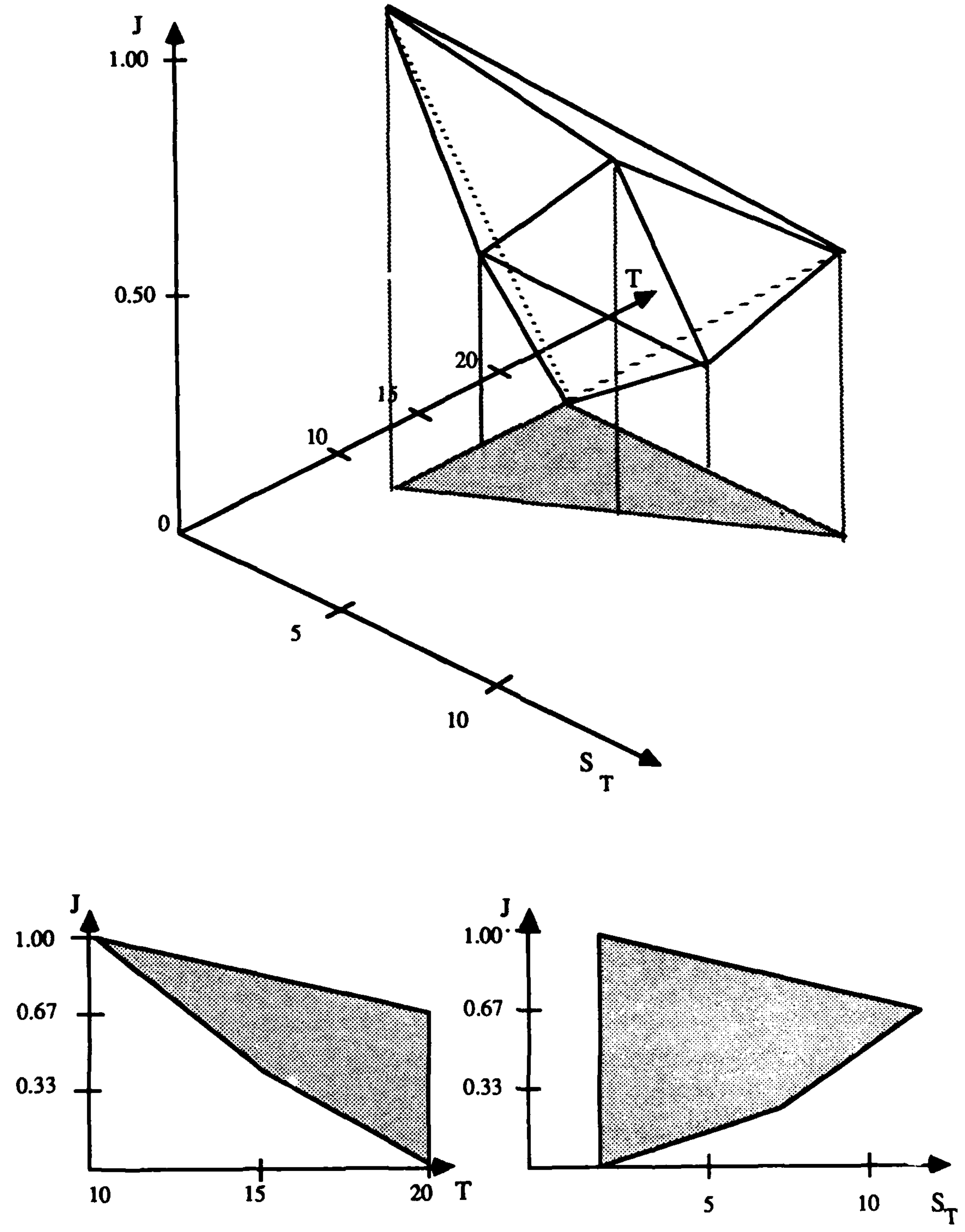


Fig. 7.4 Performance Locus in Case 1

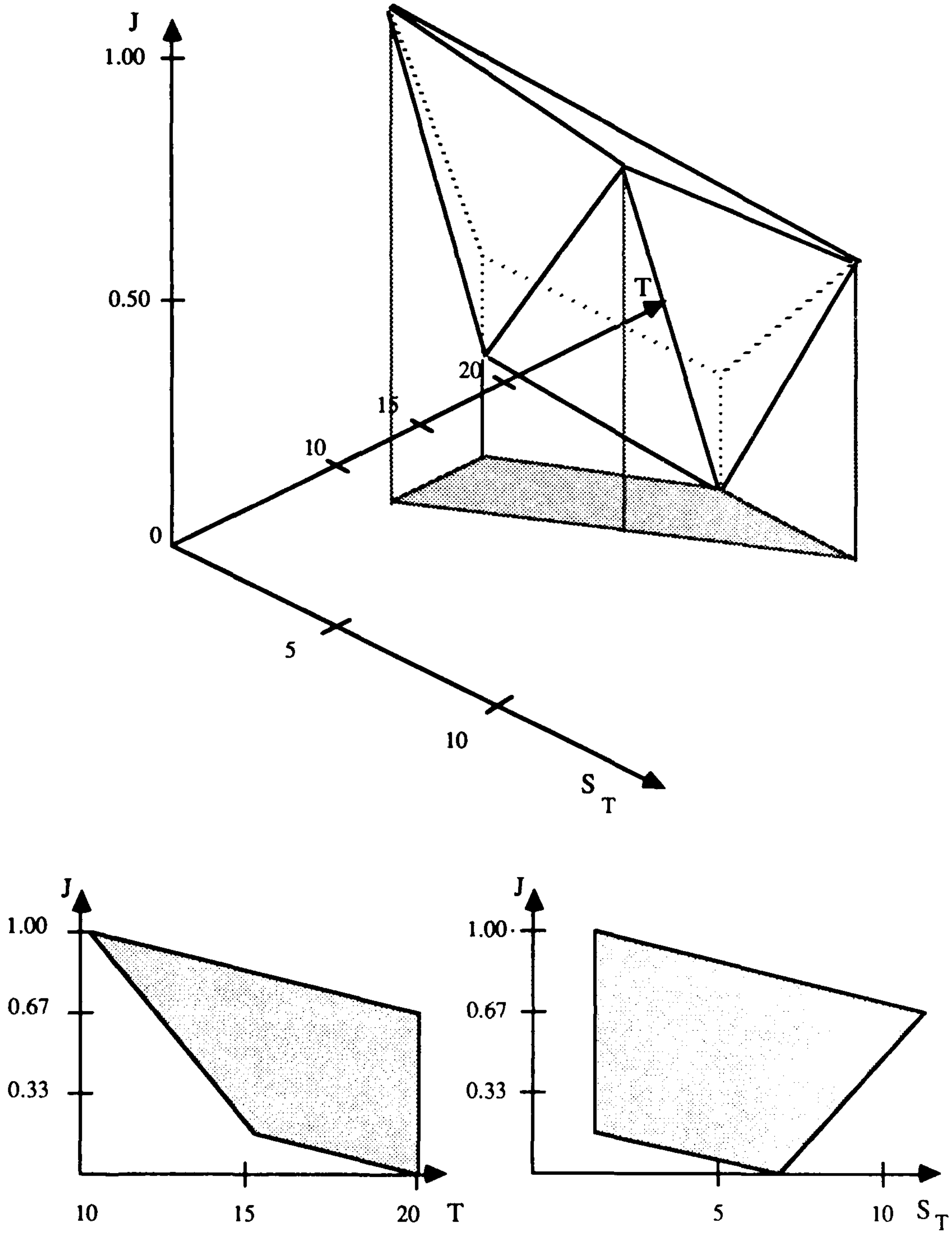


Fig. 7.5 Performance Locus in Case 2

These facts show that the introduction of a decision support system in an organization can have different effects on the coordination of the activities.

If the organization members are well coordinated when they do not use the DSS, the latter can degrade this coordination because of two factors:

- the decision-makers can use a larger number of facilities; it is difficult to coordinate their access.
- the quality of the responses provided by the DSS, i.e., the response time and the information produced, can be very different from one decision-maker to another.

Therefore, on the one hand, *the decision-makers have many more alternatives that they can use to perform their task; the coordination of these activities is consequently more difficult to achieve*. On the other hand, in coordinating their activities, the organization members must take into account the fact that the *DSS does not perform equally well for all of them*.

This latter consideration is illustrated by case 2 of the example: there is a trade-off between accuracy and timeliness which is coupled with a trade-off between accuracy and synchronization. In order to achieve a good accuracy, the organization members must use strategies which lead to a degradation in timeliness and synchronization. Conversely, if the decision-makers wants to be well synchronized, the accuracy will degrade because they cannot access the mainframe together.

Therefore, the decision support system, depending on its characteristics, leads to mixed effects on the effectiveness of the organization. As in case 2, it can lead to an improvement both in accuracy and timeliness of the organization, but the coordination then degrades. Conversely, in case 1, it cannot produce an improvement both in accuracy and timeliness; but coordination is always highest when accuracy or timeliness are optimal.

7.4 CONCLUSION

This chapter has assessed quantitatively the impact of a decision support system on the activities of a decision-making organization. As far as the concept of coordination is concerned, it can be concluded that the synchronization of an organization can be degraded when the number of information flow paths is increased. This degradation can be important if it takes different amounts of time for the information to flow through these various paths.

Therefore, the introduction of a decision support system can alter considerably the synchronization of the various activities for two reasons:

- the decision-makers can use different strategies with respect to the use of the system. It can result from the fact that some of them rely more than others on the DSS.
- the capabilities offered to the various decision-makers by the system may differ. For example, a certain decision-maker may have faster access to the central database than another one, because of different transmission times.

However, the fact that some decision-makers are provided with better capabilities can allow the organization to improve both the timeliness and the accuracy of the process.

CHAPTER VIII

CONCLUSIONS AND DIRECTIONS FOR FURTHER RESEARCH

8.1 CONCLUSIONS

The purpose of this study was to investigate the concept of coordination in information processing and decision-making organizations and to assess the extent to which decision aids alter their cohesiveness. The framework of the research carried out is the quantitative methodology to evaluate alternative organizational structures.

Five stages constitute the bases on which the argument is founded:

1/ analysis of the concept of coordination in decision-making organizations.

2/ modeling of decision-making processes that require coordination.

3/ development of a set of tools to evaluate the coordination of decision-makers.

4/ modeling of decision support systems in decision-making organizations.

5/ investigation of specific examples.

1/ The concept of coordination is defined as relating to :

- the extent to which the decision-makers constitute a team.
- the consistency of the information exchanged by the different organization members.
- the synchronization of the various activities.

The latter issue bears on the dynamics of the decision-making process. A decision-making organization is perfectly synchronized for the task at hand if none of its members waits for the information that he needs at any stage of the process. If it is not the case, the value of information when it is actually processed may have decreased, leading to a degradation of the organizational effectiveness.

It is indicated that the synchronization can be modified by two types of factors:

- the dynamics of the decision-making process can lead various organization members to process data with different priority orders.
- the number of information flow paths in the organizational structure with different

processing times can be increased by the introduction of new components.

The consistency of information shows the extent to which different pieces of information can be fused without contradiction.

2/ The modeling of processes that require coordination is developed using the basic model of the single interacting decision-maker refined through the use of the Predicate Transition Net formalism. In particular, tokens representing symbolic information carriers have been differentiated on the basis of three attributes which account for characteristics that decision-makers can use to discriminate between various data.

The protocols of interactions between organization members model the fact that they must refer to the same input when they fuse data. Different strategies for selecting the information to process have been introduced, e.g., FIFO or priority order between classes of data.

3/ The evaluation of the coordination is based on a characterization of the firing of interactional transitions in the Predicate Transition Net model developed. Furthermore, two measures are introduced in order to perform a quantitative evaluation of the coordination of decision-making processes, i.e., the degree of information consistency and the measure of synchronization.

A simulation system for Petri Nets has been developed. Ordinary Petri Nets, Timed Petri Nets and Predicate Transition Nets, using the grammar of execution introduced in this work, can be simulated. The dynamics of different decision-making processes can be investigated by simulating the execution of the corresponding Petri Net models. Quantitative results concerning the processing delays and the synchronization of the activities can be obtained.

4/ A model of a decision support system is presented. It accounts for the fact that most real systems contain facilities accessed individually by the users and databases and mainframes shared by all organization members. It is also consistent with the fact that a decision-maker has to make meta-decisions with respect to the use of the decision support system.

5/ The examples show that a preprocessor can degrade the synchronization of the activities if the decision-makers process data in different order on the basis of the information they obtain by using the decision aid. Furthermore, when the priority order implemented by organization members differ, the fact that multiple data are displayed at the same time to them can degrade considerably the organizational effectiveness, because of the likelihood that organization members do not refer to the same inputs when interaction increases.

The introduction of a decision support system can also degrade the coordination of the activities because the decision-makers have to access various facilities which do not have the same response times; they depend on the configuration of the system.

8.2 DIRECTIONS FOR FURTHER RESEARCH

Research can be carried on in several directions to integrate the different concepts, models and evaluation tools presented in this work in a comprehensive study of coordination in decision-making organizations. Suggestions for future work are presented in the remainder of this section.

First, the Predicate Transition Net framework can be used to investigate different situations by developing new rules for selecting tokens and for the enablement of transitions. For example, one can investigate the performance of organizations where the protocols of interactions allow mutual adjustment between the organization members: for instance, if DM_1 recognizes that he always wait for the information from DM_2 to arrive, he can decide to take more time to perform his processing or to query a database. In this context, for given protocols of interaction, the synchronization and accuracy can be improved. Therefore, the design of organizational structures in which decision-makers can decide to alter their activities to adjust to the processes performed by other components (human or technological) can be analyzed.

Second, it seems necessary to incorporate the modeling of algorithms performed by the decision-makers when they choose the items of information that they process. In so doing, it would be possible to assess the impact of such decisions on the workload of each member. Furthermore, the relationship between the bounded rationality constraint and the concept of coordination needs to be investigated. Indeed, when a decision-maker is overloaded, it may be the case that he sends information inconsistent with what another member knows.

Third, the impact of databases on the consistency of information must be assessed. The fact that different decision-makers share a common database can increase the consistency of the information that they exchange. Conversely, if they access decentralized databases which are not upgraded in the same manner, the quality of their interactions could degrade. In the same way, two decision-makers accessing a common database but at different instants can get different information on the state of the environment.

Fourth, the impact of decision aids which provide alternatives to decision-makers can be assessed. Indeed, if a decision aid produces different responses to a query, the decision-maker who queried the system may improve the accuracy of his processing but the timeliness of the process can be degraded, if a substantial amount of time is needed to assess the various responses given by the decision aid.

Finally, the development of the simulation package to integrate various tools for evaluating organizational structures should be continued so that the investigation of the dynamics of decision-making processes in various circumstances can be pursued.

REFERENCES

Andreadakis, S.K., and A.H. Levis, 1987, "Accuracy and Timeliness in Decision-Making Organizations," *Proc. 10th IFAC World Congress*, Pergamon Press, New York.

Bejjani, G.J., and A.H. Levis, 1985, "Information Storage and Access in Decision-Making Organizations," *Proc. 8th MIT/ONR Workshop on C^3 Systems*, LIDS-R-1519, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Boettcher, K.L., and A.H. Levis, 1982, "Modeling the Interacting Decision-Maker with Bounded Rationality," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-12, No.3, pp. 334-344.

Boettcher, K.L., and A.H. Levis, 1983, "Modeling and Analysis of Teams of Interacting Decision-Makers with Bounded Rationality," *Automatica*, Vol.19, No.6, pp.703-709.

Bouthonnier, V., and A.H. Levis, 1984, "Effectiveness Analysis of C^3 Systems," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-14.

Brams, G.W., 1983, *Réseaux de Petri: Théorie et Pratique, T.2, Modélisation et Application*, Masson, Paris.

Chyen, G. H-L., and A.H. Levis, 1985, "Analysis of Preprocessors and Decision Aids in Organizations," *Proc. IFAC/IFIP/IFORS/IEA Conference on Analysis, Design and Evaluation of Man-Machine Systems*, Varese, Italy.

Conant, R.C., 1976, "Laws of Information which Govern Systems," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-6, No.4, pp.240-255.

Cothier, P.H., and A.H. Levis, 1986, "Timeliness and Measures of Effectiveness in Command and Control," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-16, No. 6.

Dersin, P., and A.H. Levis, 1981, "Large Scale System Effectiveness Analysis," LIDS-FR-1072, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Eddy, W.B., 1985, *The Manager and The Working Group*, Praeger, New-York.

Genrich, H.J., and K. Lautenbach, 1981, "Systems Modeling with High-Level Petri Nets," *Theoretical Computer Science*, No.13, pp.109-136.

Hall, S.A., and A.H. Levis, 1984, "Information-Theoretic Models of Memory in Human Decision-Making Models," *Proc. 9th IFAC World Congress*, Pergamon Press, New York.

Hillion, H.P., and A.H. Levis, 1987, "Timed Event-Graph and Performance Evaluation of Systems," *Proc. 8th European Workshop on Applications and Theory of Petri Nets*, Zaragoza, Spain.

Jin, V.Y., A.H. Levis, and P.A. Remy, 1986, "Delays in Acyclical Distributed Decision-Making Organizations," *Proc. IFAC Symposium on Large Scale Systems: Theory and Applications*, Zurich, Switzerland.

Karam, J.G., and A.H. Levis, 1985, "Effectiveness Analysis of Evolving Systems," *Proc. 8th MIT/ONR Workshop on C^3 Systems*, LIDS-R-1519, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Keen, P.G.W. and M.S. Scott Morton, 1978, *Decision Support Systems: an Organizational Perspective*, Addison-Wesley, Reading, MA.

Levis, A.H., 1984, "Information Processing and Decision-Making Organizations: a Mathematical Description," *Large Scale Systems*, No. 7, pp. 151-163.

Marschak, J. and R. Radner, 1972, *The Economic Theory of Teams*, Yale University Press, New Haven, CT.

Martin, P.J-F., and A.H. Levis, 1987, "Measures of Effectiveness and C^3 Testbed Experiments," *Proc. 1987 Symposium on C^2 Research*, National Defense University, Fort McNair, Washington DC.

Papastavrou, J.D., 1987, "Distributed Detection with Selective Communications," MS Thesis, LIDS-TH-1563, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Perdu, D., 1987, "Modeling and Evaluation of Expert Systems in Decision-Making Organizations," MS Thesis, Laboratory for Information and Decision Systems, MIT, Cambridge, MA.

Peterson, J.L., 1981, *Petri Net Theory and Modeling of Systems*, Prentice-Hall, Englewood Cliffs, N.J.

Reisig, W., 1985, *Petri Nets: an Introduction*, Springer Verlag, Berlin.

Remy, P.A., and A.H. Levis, 1987, "On the Generation of Organizational Architectures using Petri Nets," *Proc. 8th European Workshop on Applications and Theory of Petri Nets*, Zaragoza, Spain.

Riedel, S.L., and G.F. Pitz, 1986, "Utilization-Oriented Evaluation of Decision Support Systems," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-16, No. 6, pp.980-995.

Shannon, C.E. and W. Weaver, 1949, *The Mathematical Theory of Communication*, University of Illinois Press.

Tabak, D., and A.H. Levis, 1985, "Petri Net Representation of Decision Models," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-15, No. 6.

Waltz, E.L., and D.M. Buede, 1986, "Data Fusion and Decision Support for Command and Control," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-16, No. 6, pp.865-879.

Weingaertner, S.T., and A.H. Levis, 1987, "Evaluation of Decision Aiding in Submarine Emergency Decision-Making," *Proc. 1987 Symposium on C^2 Research*, National Defense University, Fort McNair, Washington DC.

APPENDIX A

SIMULATION: USER'S MANUAL

A.1 INTRODUCTION

The MIT Petri Net Simulation System (MIT/SIM) makes use of the Design graphics package, version 2.1, developed by Meta Software Corporation and runs on Macintosh with at least one megabyte in RAM. One can draw Petri Nets and then execute them according to grammar rules that the user can define by himself.

This appendix is an introduction to the use of the MIT/SIM System from a user's standpoint. Information concerning the architecture and development of the program can be found in Appendix B.

The MIT/SIM System allows to simulate different levels of Petri Nets: it allows the simulation of Timed and non-Timed ordinary Petri Nets; furthermore, it embodies basic Predicate Transition Net features that relate to the model of coordination developed in this thesis. The user is responsible for defining the type of simulation that he wants to implement.

The present appendix shows the procedures to set up the system before running a simulation. It consists of the following parts:

- Additional notions on the execution of Petri Nets
- Starting the simulation
- Simulation menus
- Node information
- Sample session

A.2 ADDITIONAL NOTIONS ON THE EXECUTION OF PETRI NETS

This section provides additional information concerning the notions of *depth-first*, *breadth-first*, *pull-out* and *step*. The governing rule for the execution of a Petri Net is that a transition *may* fire when it is enabled.

When implementing the execution of a Petri Net on a digital computer, the automation of the firing process raises issues concerning the fact that a computer executes instructions sequentially. If two transitions may be fired concurrently according to the net formalism, it is necessary to order their firing to simulate the same step on the computer. In some cases, the execution of the net may be very sensitive to this ordering.

In non-Timed Petri Nets, the main problem arises from the fact that time cannot be used as an index to order the different events that must occur: transitions fire instantaneously; rules must be defined in order to automate the execution of the net without requiring the user to make the choice of what event should take place next.

Depth-first execution is accomplished by considering first for the next firing the set S of transitions enabled by the tokens produced by the last firing. However, if the firing of these transitions requires the resolution of conflicts involving transitions that do not belong to S, these conflicts must be resolved.

Breadth-first execution requires the firing of all enabled transitions at a given marking M before considering new enablements. If there exists a conflict for these transitions, it must be resolved; otherwise, any conflict involving a transition not enabled for the marking M must not be resolved and the transition enabled for the marking M will fire.

In the example of Figure A.1, we assume that p_1 and p_2 both contain a token and that all transitions have zero firing times. Furthermore, we assume that *when concurrency occurs the place with the smallest index is considered first*. Then, the sequences of transition firings will be the following:

- depth-first: t_1, t_{11}, t'_{11}, t_{12}, t'_{12}, t_2, t_{21}, t'_{21}, t_{22}, t'_{22}.
- breadth-first: t_1, t_2, t_{11}, t_{12}, t_{21}, t_{22}, t'_{11}, t'_{12}, t'_{21}, t'_{22}.

In this context, a **step** in depth-first corresponds to the firings of all transitions for a branch, e.g., t_1, t_{11}, t'_{11} in the example above. A step in breadth-first corresponds to the firing of all transitions at a particular instant, e.g., t_1, t_2. In Timed Petri Nets, a step corresponds to the firing of all the transitions that must occur at the same time.

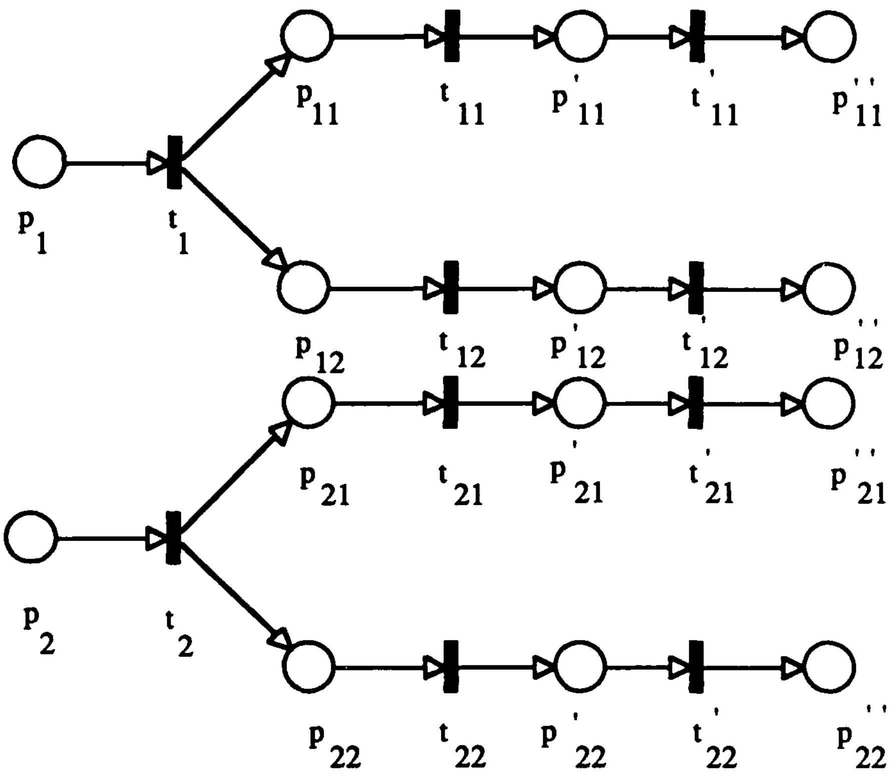


Fig. A.1 Depth-first vs. Breadth-first

The **pull-out** strategy refers to situations where confusion occurs as illustrated in Figure A.2.

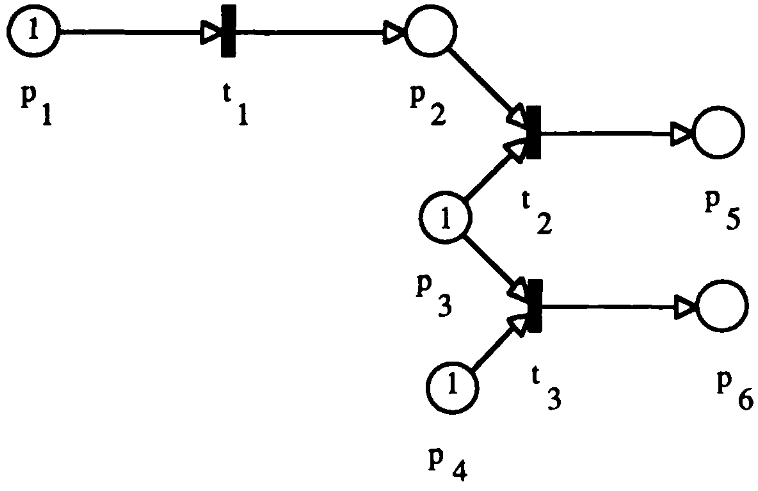


Fig. A.2 Confusion Situation

If t_1 fires first, t_2 and t_3 are both enabled. When the stragtegy adopted is breadth-first, only places p_3 and p_4 are considered for the next firing: then, no conflict between t_2 and t_3 needs to be resolved and t_3 fires. If the strategy is depth-first, the token produced by t_1 is in place p_2; it enables t_2 which is considered first for the next firing: if the pull-out process is implemented, transition t_2 fires without resolution of the conflict; otherwise, the resolution of the conflict must take place in accordance with the definition of the depth-first strategy.

Therefore, when executing a Petri Net, it is important to recognize whether it is an ordinary Petri Net or a Timed Petri Net or a partial Timed Petri Net, i.e., a Timed Petri Net with some transitions having zero firing times . For Timed Petri Nets, the automation is done with respect to time. In the other cases, the user must decide of some strategy to resolve concurrency issues that might occur because of instantaneous firings.

A.3 STARTING THE SIMULATION

(i) Before starting the simulation program, the user must have created a Petri Net diagram using the MIT/PN graphics package. The **Mit-pn** menu of this application must be used in order to create the diagram of interest. If the user does not want to provide immediately the nodes with data, he can use the **automatic labelling** option that will assign names to the nodes: p_1, p_2, etc... for places; t_1, t_2, etc... for transitions; s_1, s_2, etc... for switches; c_1,c_2, etc... for connectors. By using the **turn on/off labels** option, the user can have the labels appear on or disappear from the screen. The user should avoid situations where switches have more than one input place. Once, the user has created the net, he must save it and quit from the MIT/PN program.

(ii) The MIT/SIM is a separate application program. In order to run the simulation, the user must double-click on the icon named MIT/SIM. The environment of the MIT/SIM System is automatically brought on the screen. The Petri Net that the user created in step (i) can be loaded by using the **Open** option in the **File** menu. At this stage, the user is provided with the environment of the MIT/SIM System and with the Petri Net that he wants to execute. If the user wants to load another net that already exists, he can **Close** the file in the **File** menu and **Open** the file of interest.

At this stage, the user must define what type of simulation will be run. The following paragraphs present the different capabilities of the MIT/SIM system.

A.4 SIMULATION MENUS

Each of the following paragraphs corresponds to one of the five menus specific to the MIT/SIM System and which are illustrated on Figure A.3.

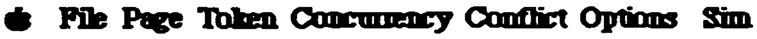


Fig. A.3 Simulation Menus

This figure shows the top of the screen once the user has opened a file containing the Petri Net that he wants to simulate.The five menus that are explained in this section are: **Token, Concurrency, Conflict, Options** and **Sim.** These menus contain all the commands required to initialize and run the simulation as well as to get information on the status of the Petri Net during the execution. The **File** and **Page** menus are the customary ones for Design applications.

A.4.1 Token Menu

Figure A.4 shows the different commands that can be accessed from this menu.

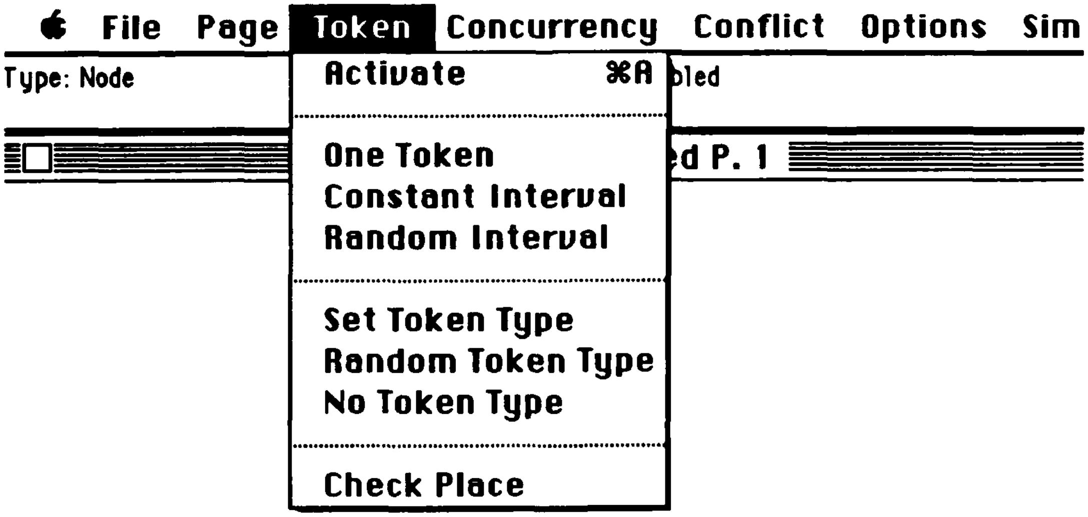


Fig. A.4 Token Menu

Activate:

function: Used to insert one or several tokens in a place before running a simulation. The time of arrival of the tokens and their classes are set according to the choices made in the **Token** menu

use: Once the type of tokens - i.e., number, arrival times and classes - have been selected with the **Token** menu, click on the place of interest and call "Activate". If a mistake has been made, reinitialize the net with "Initialize".

One Token:

function: Used to specify that "Activate" will create at current time one token in the selected place.

use: Call "One Token". As long as the check mark is present, "Activate" willcreate one token at a time unless specified otherwise (cf. "Infinite" in section A.4.5).

Constant Interval:

function: Used to specify that a finite string of tokens with constant interarrival time starting at current time will be activated in the selected place. The number of tokens and the interarrival time are specified in the dialog window that is brought on the screen when "Activate" is called.

use: Call "Constant Interval". As long as the check mark is present, this rule of activation is implemented but the dialog window appears for each place selected.

Random Interval:

function: Used to specify that a finite string of tokens with random interarrival time starting at current time will be activated in the selected place. The number of tokens and the maximum interarrival time are specified in the dialog window that is brought on the screen when "Activate" is called.

use: Call "Random Interval". Implemented as long as the check mark is present but the dialog window appears for each place selected.

Set Token Type:

function: Allows the user to specify the class of each token that is activated. A class is an integer number.

use: Call "Set Token Type". Implemented as long as the check mark is present. A dialog window is brought on the screen asking for the class of the token.

Random Token Type:

function: Used to specify that the classes of the tokens to be activated are chosen at random. Interesting when a string of several tokens is generated.

use: Call "Random Token Type". A dialog window is brought on the screen asking for the number of classes. The classes' numbers start from 0. Implemented as long as the check mark is present.

No Token Type:

function: Used to specify that the tokens have no class (class 0).

use: Call "No Token Type". Implemented as long as the check mark is present.

Check Place:

function: Al'¬ws the user to check the list of tokens that stand in any place. Tokens have four attributes: time of arrival in net, time of arrival in place, status (usable = 1, not usable = 0), class (integer number).

use: Click on the place of interest. Call "Check Place". The list of tokens is saved in a file called "myfile1" that you can access from the MIT/SIM environment using the desk accessories **Redwriter** or **Mockwrite** in the menu.

A.4.2 Concurrency Menu

Figure A.5 shows the different commands that can be accessed from this menu.

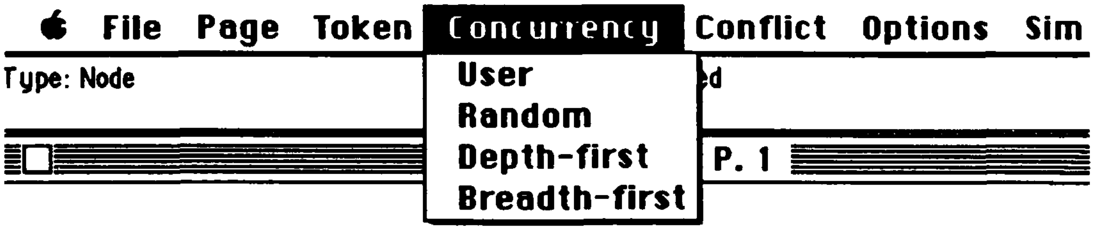


Fig.A.5 Concurrency Menu

User:

function: Used to have the user execute the Petri Net. At each stage, the user is asked to select among the possible places the one that should be considered for the next firing of a token.

use: Call "User". The possible places are surrounded by boxes and the user must click on the place that he has chosen. As long as the check mark is present, the user is asked the execute the net.

Random:

function: Used to automate the process of selecting the next place to be considered. The choice is made at random .

use: Call "Random". As long as the check mark is present, the choice is made automatically at random.

Depth-first:

function: Used to automate the firing of concurrent transitions which fire instantaneously, i.e., that have a zero firing time, according to the depth-first strategy. (cf. section A.2).

use: Call "Depth-first". As long as the check mark is present, the depth-first strategy is implemented when applicable.

Breadth-first:

function: Used to automate the firing of concurrent transitions which fire instantaneously, i.e., that have a zero firing time, according to the breadth-first strategy. (cf. section A.2).

use: Call "Breadth-first". As long as the check mark is present, the breadth-first strategy is implemented when applicable.

A.4.3 Conflict Menu

Figure A.6 shows the different commands that can be accessed from this menu.

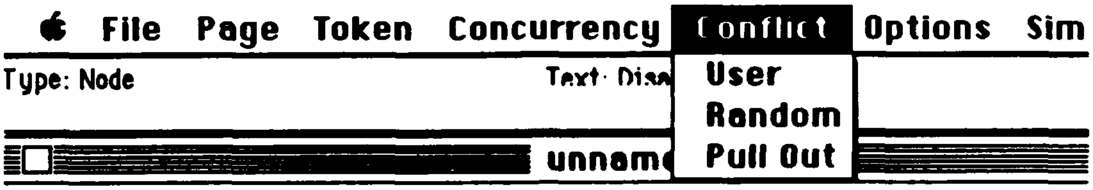


Fig. A.6 Conflict Menu

User:

function: Used to ask the user to resolve the situations of conflict between different transitions.

use: Call "User". The transitions in conflict are surrounded by boxes and the user must click on the transition that he has chosen. As long as the check mark is present, the user will be asked to resolve the conflict situations.

Random:

function: Used to resolve automatically the situations of conflict between different transitions. The transition that will fire is selected at random.

use: Call "Random". As long as the check mark is present, the resolution of conflict situations will be automatically done at random.

Pull-out:

function: Used to implement the Pull-out strategy when a confusion situation occurs (cf. section A.2).

use: Call "Pull-out". As long as the check mark is present, the pull-out strategy will be implemented for confusion situations.

A.4.4 Options Menu

Figure A.7 shows the different commands that can be accessed from this menu:

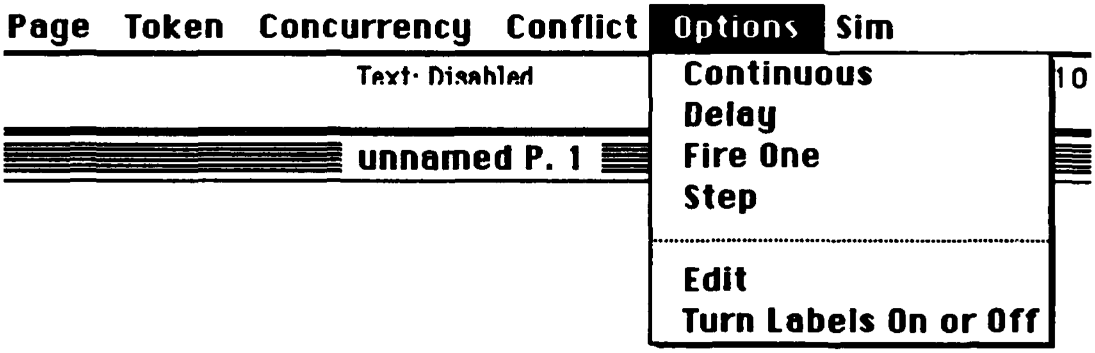


Fig. A.7 Options Menu

Continuous:

function: Used to specify that the execution takes place continuously unless the user interrupts it.

use: Call "Continuous". As long as the check mark is present, the continuous execution will be implemented.

Delay:

function: Used to implement a continuous execution as above but the firing process is slowed down so that it becomes observable.

use: Call "Delay". As long as the check mark is present, the delayed execution will take place.

Fire One:

function: Used to specify that the execution is implemented one firing at a time. The user must call "Run" in the **Sim** menu in order to proceed.

use: Call "Fire One". Implemented as long as the check mark is present.

Step:

function: Used to specify that the execution is implemented step by step (cf. section A.2). The user must call "Run" in **Sim** in order to proceed.

use: Call "Step". Implemented as long as the check mark is present.

Edit:

function: Used to insert the data corresponding to the nodes (cf. section A.5). Cannot be used to activate tokens.

use: Click on the node of interest. Call "Edit". A window appears on the screen that can be filled as explained in section A.5. In order to keep the new data, press "return" or click on "change".

Turn labels on/off:

function: Turn on and off labels of the nodes that are on the screen.

use: Call "Turn labels on/off".

A.4.5 Simulation Menu

Figure A.8 shows the different commands that can be accessed from this menu:

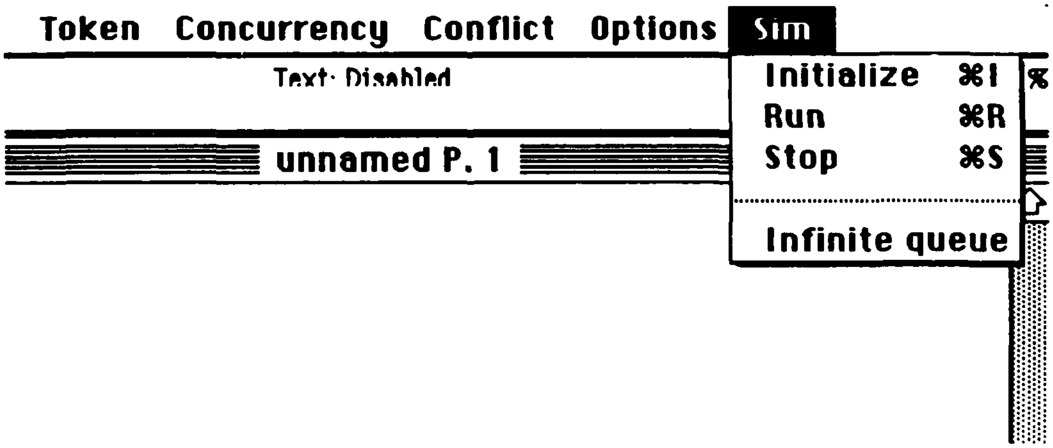


Fig.A.8 Simulation Menu

Initialize:

function: Used to initialize the simulation process before it can be run.

use: Call "Initialize" before activating any token in the net . Must be used every time a simulation is to be run; in particular, when a diagram has just been brought on the screen or when re-running a simulation for a diagram that has just been simulated.

Run:

function: Used to run the simulation once the process of initialization is over.

use: Call "Run".

Stop:

function: Used to interrupt a simulation that is being executed.

use: Call "Use" whenever it is necessary even if the simulation is still running.

Infinite queue:

function: Used to specify that an infinite string of tokens will be activated starting at current time for each place selected with a "comment" record equal to " i ". (cf. section A.5). Infinite string means that as soon as a token leaves the place, a new token is activated in it.

use: Call "Infinite". Implemented as long as the check mark is present. Concerns only places with "comment" record equal to " i ". (cf. section A.5).

A.5 NODE INFORMATION

A.5.1 Places

Figure A.9 shows the window used to enter the information concerning places.

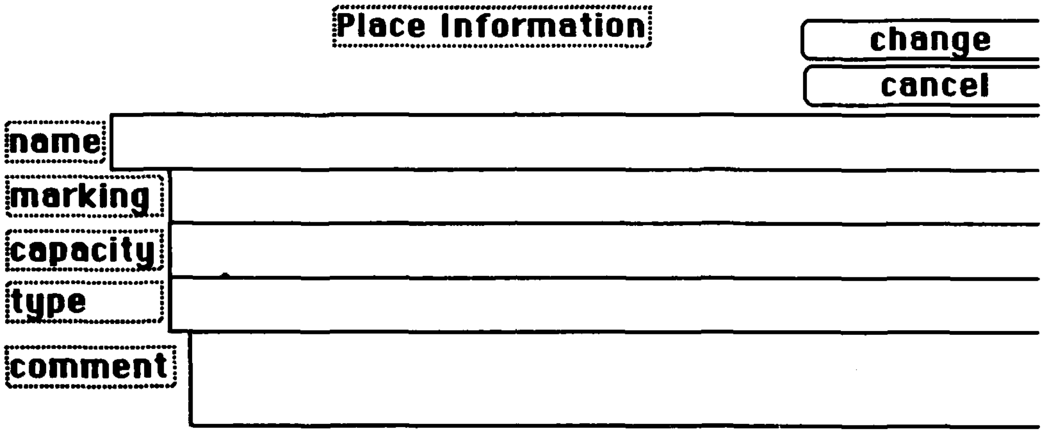


Fig.A.9 Place Information

name: contains the name of the place.

marking: contains the current marking of the place.

capacity: contains the capacity of the place. ***Zero means that the capacity is infinite***. In the current version of MIT/SIM, capacities should be left infinite.

type:

blank or " r ": the place carries tokens with no class. Used in ordinary Petri Nets when tokens have no attributes or to represent resource places in the model of decision-making organizations.

" f ": the selection of tokens in this place is done according to the rule LFIFO, i.e., local FIFO: the token that arrived first in the place and that can be fired is fired first.

" l ": the selection of tokens in this place is done according to the rule LLIFO, i.e., local LIFO: the token that arrived last in this place and that can be fired is fired first.

" s ": same as "r" but tokens have different types. Used for certain types of transitions (cf. section A.5.2).

" w ": tokens with the lowest class number are selected, using the LFIFO for tokens that correspond to the same lowest class number.

" h ": tokens with class 0 are selected first and otherwise with the highest class number. LFIFO is applied if several tokens apply.

comment:

" i ": used to specify that an infinite string of tokens can be built in this place.

A.5.2 Transitions

Figure A.10 shows the window used to enter the information concerning the transitions:

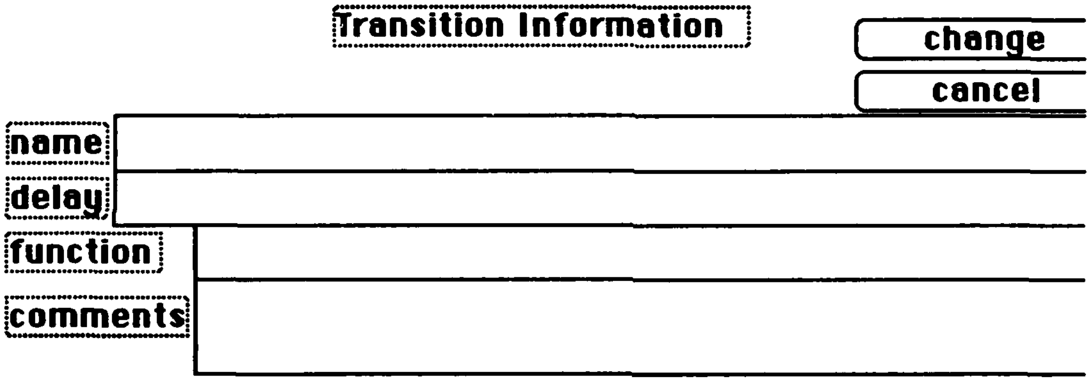


Fig.A.10 Transition Information

name: name of transition.

delay: time delay for the firing of the transition.

function:

blank or " n ": transition is enabled whenever there is a token in each of its input places, regardless of its class. When it fires, it puts a token in each output place with no class attached to it.

" f ": the tokens which are in the input places of type different from blank or " r " must have the same time of entry in the net and same class for the enablement. Input places of type " r " must have a token. When it fires, it puts in each ouput place a token with the same attributes (time of entry in net and class).Should not have any place of type " s " in input.

" jn ": behaves like transitions of type " f " for the enablement and firing, but must also have a place of type " s " in input that contains a token of class " n " (n = integer between 0 and 9). Should not have any place of type " s " in output.

" in ": behaves like transitions of type " f " for the enablement. It can have output places of type " s " in which tokens of class n will be inserted (n = integer between 0 and 9). For other type of places, it behaves like transitions of type " f ". Should not have any place of type " s " in input.

comments: any comment can be inserted here.

A.5.3 Switches

Figure A.11 shows the window used to enter the information concerning the switches:

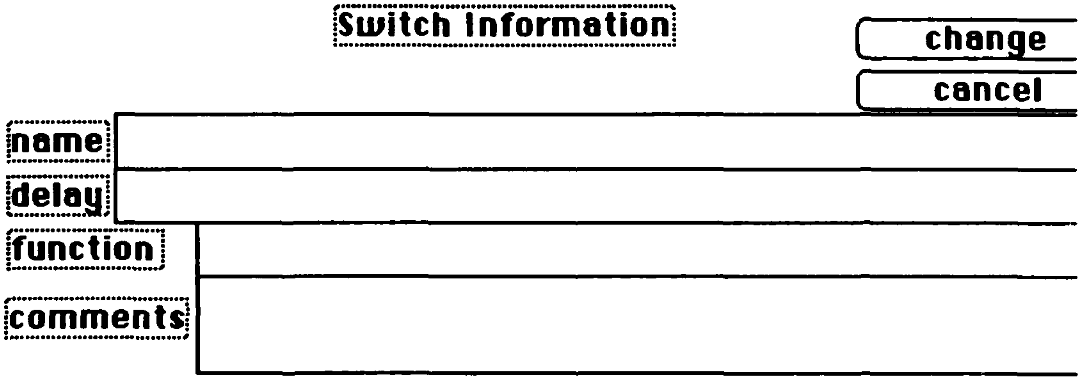


Fig.A.11 Switch Information

name: name of switch.

delay: time delay for the firing of the switch.

function: enabled when a token is in input place. Should have only one input place.

blank or " u ": user is asked to select the place to which the token will be routed.

" r ": one of the output connectors is chosen at random and the token is moved through this conector.Attributes of the token are kept.

" o ": the token is routed through the connector with the highest priority if it can carry the token (output place must not be full). Since capacities are not fully implemented, token will always be routed through first connector in the current version. Atributes of the token are kept.

" p ": the token is routed through a connector selected on the basis of the probability distribution defined in the output connectors' record called "probability that arc is open". Attributes of the token are kept.

" s ": the token is routed through the connector that has a priority equal to its class. Attributes of the token are kept.

comments: any comment can be inserted here.

A.5.4 Connectors

Figure A.12 shows the window used to enter the information cocerning the connectors:

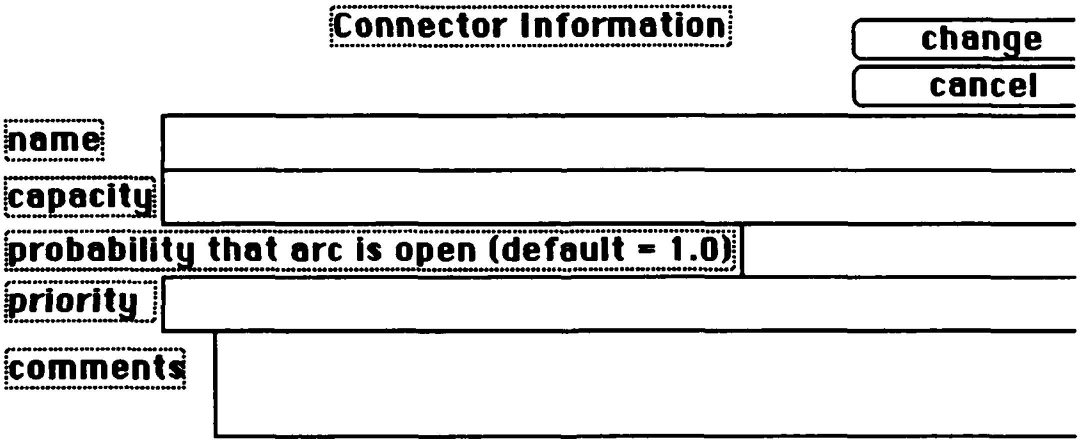


Fig.A.12 Connector Information

name: name of the connector.

capacity: anything can be inserted here. Not used in the current version.

probability: probability for switches of type " p ". The sum does not need to be equal to one (normalization occurs during execution).

priority: an integer. Used for switches of type " o " or " s ". In the latter case, the priorities should correspond to the classes of the tokens activated in the net.

comments: anything can be inserted here.

A.6 SAMPLE SESSION

This section shows how to run a simulation. Even though the guidelines presented here are very general, they are common to all sessions that execute Petri Nets with the MIT/SIM System.

If the Petri Net is already on the screen with the MIT/SIM environment, skip steps 1-2.

1> Double-click on the icon named MIT/SIM.

2> Open the file containing the Petri Net by using **Open** in **File** menu.

3> Fill, if necessary, the information for the nodes as explained in section A.4

4> Initialize the system by calling **Initialize** in **Sim** menu.

5> Select the type of tokens you want to activate with **Token** menu.

6> Click on the place where you want to activate these tokens.

7> Call **Activate** in **Token** menu.

8> Repeat, as necessary, steps 5-7 for all places where tokens have to beactivated.

9> Set the option for the run by using **Concurrency, Conflict** and **Options** menus.

10> Call **Run** in **Sim** menu.

The execution of the Petri Net starts and can be interrupted in two different ways:

- call **Stop** in **Sim** menu. The execution is suspended until you call **Run** in **Sim.**
- take any action with the mouse or the keyboard. The execution is suspended until the process called with the mouse is over. The simulation starts over again by itself.

It is not recommended, during an interrupt, to change the structure and setting of the Petri Net. It has, indeed, no meaning as far as the execution of the Petri Net is concerned and can cause unexpected results. However, the user can change the options used to run the simulation, i.e., the options in **Concurrency, Conflict** and **Options** menus. The user can take any other action that does not modify at all the execution currently processed.

APPENDIX B

SIMULATION : DEVELOPER'S MANUAL

B.1 INTRODUCTION

The MIT/SIM System has been developed on the Macintosh Plus using the Design Open Architecture Development System (Design OADS), version 2.1, developed by Meta Software Corporation, and the Lightspeed C environment, version 1, developed by Think Technologies. The references of the corresponding manuals are provided at the end of this chapter.The Design OADS provided the meta-language of Design functions that allow to access at a high-level the graphical tools of the Design environment. The Lightspeed C compiler supplied the development tool to write the program in C language in a convenient manner.

This appendix has two objectives:

- to provide specific information concerning the methods used to simulate Petri Nets.
- to be a support for the future maintenance and enhancements of the MIT/SIM System.

It is therefore oriented toward technical issues that relate to the programming of the Petri Net simulation software. However, the ideas developed here do not refer to the coding itself but rather to the data structure and architecture of the program. More detailed information concerning the coding of the functions implemented can be found in the code itself.

The program is a discrete-event simulation: it generates and handles events that modify the system in a discrete manner. It is consistent with the execution of Petri Nets which is a discrete process, i.e., firing of transitions that are enabled and insertion of tokens in places. Thus, the simulation of Petri Nets is controlled by the appearance in a calendar of elementary events and by the sequential processing of them as they are encountered. The event calendar is defined as being an ordered list which indicates the sequence in which events are activated.

This section consists of the following parts:

- elementary events

- event calendars
- data structure for the nodes
- architecture of the program

B.2 ELEMENTARY EVENTS

The program makes use of the management of calendars in order to determine at each instant what is the next event to implement. The elementary events are the events that appear on the calendars and that are recognized by the event-manager which decides what particular actions to take on the basis of the event encountered.

There are five elementary events described in the subsequent paragraphs. These events are characterized by three essential parameters: the index that is used to order them in the calendars: time, the type of event, e.g., an integer, and the node of the Petri Net where the event takes place.

B.2.1 Activation of a Token in a Place

The type of this event is zero. It refers only to places. When this event is encountered, the following actions must be taken:

- Check if the place is not empty (token already removed)
- Find all the output transitions of the place that are enabled.
- If several transitions are enabled, resolve the conflict (user's choice or at random)
- Recognize if the selected transition is a switch or a transition.
- Check if the transition (or switch) is not already executing.
- If it is, compute the time of end of execution and reactivate the token in the place at this time.
- If it is not, select the output place in the case of a switch.
- Insert in the calendar the event "transition enabled"

In the remainder of this appendix, this event is denoted by **ev-activate**.

B.2.2 Transition Enabled

The type of this event is one. It takes place only for transitions and switches. When this event is encountered, the following actions must be taken:

- Check if the transition is still enabled.
- Check if the transition is free to fire.
- Recognize the function of the transition ("function" in transition's record - cf. A.5)
- Depending on the function of the transition, determine the set of tokens that can be fired by the transition.
- For each input place, depending on the function of the transition and the type of the place, select the token that must be removed and insert the corresponding event in the calendar.
- Compute the firing time of the transition
- If it is not zero, insert in the calendar the event that will reset the fill pattern of the transition on the diagram when the firing is over.
- For each output place, create the event "activate a token" at the time corresponding to the end of the firing. Depending on this time and the execution strategy implemented, this event is inserted in different parts of the calendar.

In the remainder of this appendix, this event is denoted by **ev-enabled**.

B.2.3 Remove Token

The type of this event is three. It concerns only places. When this event is encountered on the calendar, the following actions are taken:

- The token is removed from the list of tokens of the place.
- If an infinite string of tokens was generated for the place (cf. A.5), a token is inserted in the place and the event "activate token" for the place is inserted in the calendar.
- If the place was full, find the input transitions or switches of the place that are enabled, resolve the possible conflicts and insert on the calendar "transition enabled".

In the remainder of this appendix, this event is denoted by **ev-remove**.

B.2.4 Token Usable

The type of this event is five. It concerns only places. When this event is encountered in the calendar, the following actions are taken:

- The token is fetched in the list of tokens of the place
- Its status is set to usable.

In the remainder of this appendix, this event is denoted by **ev-usable**.

B.2.5 Reset Transition

The type of this event is six. It concerns only transitions and switches. When this event is encountered, the fill pattern of the corresponding transition is reset to the inactive mode.In the remainder of this appendix, this event is denoted by **ev-reset**.

B.2.6 Chaining of Events

It is possible to define a causality in the generation of the events described in this section.

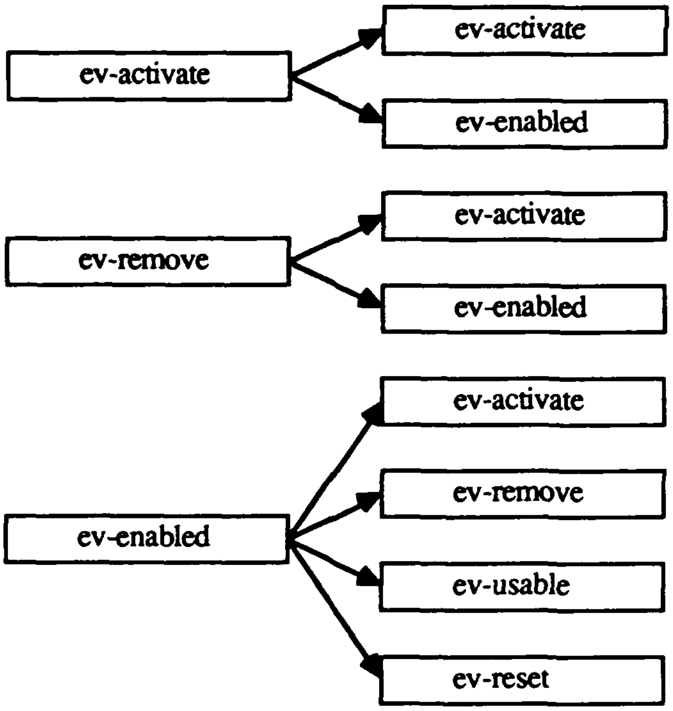


Fig. B.1 Chaining of Events

The arrows on this figure mean "... generate...". When the user activates token in the net, the event **ev-activate** is created and inserted in the calendar. Thus, at the end of the initialization process, the calendar consists only of events **ev-activate**. Then, the event-manager reads these events in the calendar and generates events in accordance with the chaining described above. The simulation is terminated when no event remains in the calendar.

B.3 EVENTS CALENDARS

Three different events calendars are used to manage the elementary events described above. They are denoted by:

- **Head**
- **Head-Temp**
- **Head-Trans**

The main event calendar is **Head**. When an event, that must take place at a later stage of the execution is created, i.e., an event which should not be considered in the step that is currently executed (cf. Appendix A), *it is inserted in this calendar*. The only events that can be generated for steps posterior to the current one are: **ev-activate, ev-reset, ev-usable.** Indeed, the events **ev-remove** and **ev-enabled** must be activated as soon as they have been created: it results from the fact that as soon as a transition is recognized by an event **ev-activate** as being enabled, it must fire (**ev-enabled**) and remove tokens from its input places (**ev-remove**) before anything can happen in the Petri Net. Consequently, the calendar **Head** contains only the events **ev-activate, ev-reset, ev-usable**. When the user activates tokens in the net, the corresponding events **ev-activate** are inserted in this calendar. Then, when the execution is started or at the end of each step, all the events of the calendar **Head** that have the same index are extracted from it and are considered for the execution of the next step.

This set of events is inserted in the calendar **Head-Temp** that contains all the events that must occur in the current step. Thus, as long as **Head-Temp** is not empty, the step is not over. Events can be inserted in all three calendars, but as long as **Head-Temp** is not empty, no event will be extracted from **Head**.When a new step is begun, all the events **ev-reset** are first implemented so that all the corresponding transitions are reset before anything else can happen.Then, all the events **ev-activate** are recognized and the corresponding places receive a token. The occurrence of several **ev-activate** in **Head-Temp** implies that several tokens can be considered for the next firing and, therefore, the resolution of this concurrency is done by the user or automatically. When a place has been selected, the event **ev-activate** is implemented for it, and the event **ev-enabled** for the transition enabled - selected in the case of conflicts - is inserted in the list **Head-Trans**.

The calendar **Head-Trans** contains then all the events **ev-enabled** and **ev-remove** that are generated by this event. As long as tokens must be removed from input places or that new transitions can fire when tokens are removed from these places - transitions become free to fire - this calendar contains events. All these events arer executed before any other event in **Head-Temp** is considered.

Consequently, the event-manager, which fetches the next event to execute, works as follows:

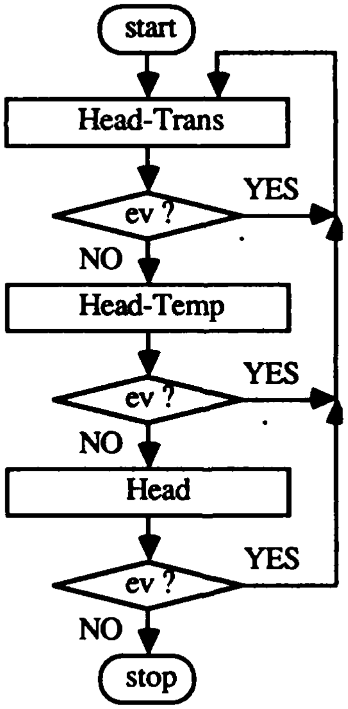


Fig.B.2 Event Manager

Each time the event-manager is accessed, the system checks first if the Macintosh does not require that other events independent from the simulation be executed first. So, in general the Macintosh will do first all the actions that do not concern the execution of the net and, then, when no such event is encountered, it will execute the simulation events - null events for the Macintosh operating system - in accordance to the rules defined above. This technique allows to manage interrupts from the user in a convenient manner.

The following example provides a simple execution with the evolution of the different calendars; we assume here that t has a non-zero firing time equal to d units of time:

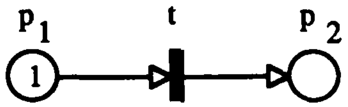


Fig.B.3 Transition Firing - Stage 1

stage1:

- **Head** contains "activate p_1".
- event "activate p_1" extracted from **Head** and inserted in **Head-Temp.**
- event "activate p_1" generates event "t enabled" which is inserted in **Head-Trans** at the current time.
- event "t enabled" generates event "remove p_1" inserted in **Head-Trans**, event "activate p_2" and event "reset t" inserted in **Head** d units of time later.

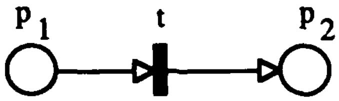


Fig. B.4 Transition Firing - Stage 2

stage 2:

- transition t is active

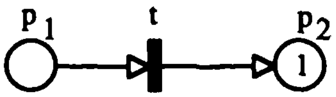


Fig. B.5 Transition Firing - Stage 3

stage3 :

- events "reset t" and "activate p_2" are extracted from **Head** and inserted in **Head-Temp.**
- "reset t" is implemented and transition t is reset.
- "activate p_2" is implemented and a since it has no output transition, all calendars are empty. The execution is terminated.

When the system is initialized with the option **initialize**, these lists are set to null. The creation of events and their insertion on the calendars are done by different modules (cf. code). In particular, any event can be either inserted in the middle of a calendar or pushed on top of the calendar. This latter feature is particularly useful for implementing the depth-first strategy which corresponds to pushing the event **ev-activate** for the last place activated on top of the calendar **Head-Temp.** In the case of the breadth-first strategy, the event **ev-activate** is inserted in the list **Head** so that all the events of **Head-Temp** will be terminated first.

B.4 DATA STRUCTURE FOR THE NODES

The data structures of the nodes have been set in accordance with the information that each node contains in the model developed in the present study. It is possible to develop these data structures in order to simulate and get results on other types of Petri Nets that the MIT/SIM System. Some suggestions of future enhancements are presented at the end of this paragraph.

The data structures presented here do not correspond to the data structures embodied in the dialog windows that appear on the screen when the user request information from a node and that essentially contain constant information - except for the marking of places. They rather concern the information of the nodes that vary during the execution of the net.

The places contain the following information:

- the number of tokens currently usable.
- the list of tokens that are contained in the place: each token consists of four attributes, i.e., its class, its time of entry in the net, its time of entry in the place where it currently stands, the status of the token (usable or not usable) .
- the total number of tokens in the place.

The transitions and switches contain the following information:

- the time of end of execution .
- the status of execution - active or inactive.

Connectors have no data structure assigned to them.

If the model is refined to allow the transitions to handle several tokens at once, it would be sufficient to include in the transitions' data structure the current number of tokens in the transition and, if necessary, the list of these tokens. It is therefore important to recognize that this system provides a framework that can be enhanced in a consistent manner.

B.5 ARCHITECTURE OF THE PROGRAM

This section presents the global architecture of the program. More detailed information is embodied in the comments included in the code of the program.

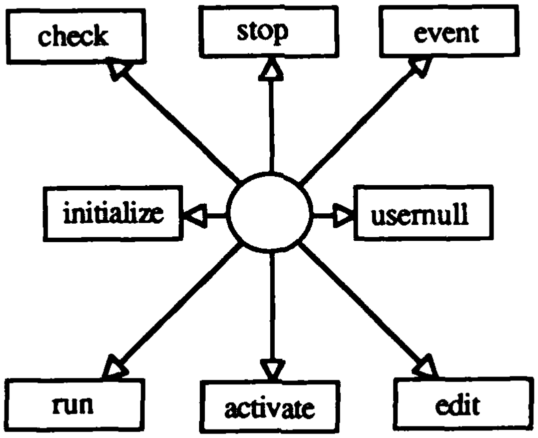


Fig.B.6 Architecture of MIT/SIM System

The modules described on the figure above do not embody explicitly all the functions that can be executed: the main ones are shown, the **event** module encompassing all the remaining functions that can be accessed either through the menus of the MIT/SIM System itself or through the access of usual Macintosh facilities.

The program recognizes:

- **initialize** which resets to null pointers all the calendars and generates the initial values that are inserted in the data structures of each node.

- **activate** which creates events of the type **ev-activate** on the **Head** calendar.
- **run** which sets the global boolean variable RUN to TRUE.
- **stop** which sets the global boolean variable RUN to FALSE.
- **check** which creates a file called "myfile1" in which the data concerning the selected place are inserted. It allows the user to be able to control at any instant the list of tokens that stand in any place of the net.
- **edit** which allows the user to check and modify the characteristics, e.g., delay, of any particular node.
- **event** that embodies all the other actions that the user can take either by accessing a menu of the MIT/SIM System, for example in order to change the run options, or by accessing for instance the scroll bars of the diagram to move it or the desk accessories of the system to perform certain operations.
- **usernull** which recognizes if RUN is TRUE and, if so, implements the events that may be inscribed on the different calendars of the simulation.

The architecture of the MIT/SIM System is flexible and allows for the development of new functions in order to simulate more general Petri Nets. In particular, it is possible to implement different enablement and firing rules by inserting new functions in the framework that already exists: the structure of the calendars and the way they are scanned should remain intact.

The references for the Design and Lightspeed C manuals are the following:

- Design Open Architecture Development System, version 2.1, Reference Manual, Meta Software Corporation, 1987, Cambridge, MA.
- Lightspeed C User's Manual, version 1, First Edition, Think Technologies, Lexington, MA.

END
DATE
FILMED
5-88
DTIC