AMI an extensible Petri nets interactive workshop

Jean-Marc BERNARD*, Jean-Luc MOUNIER*
Nicolas BELDICEANU**, Serge HADDAD*.

* CNRS / Laboratoire MASI et PAC C3
Universite Paris 6,
4 Place Jussieu, 75252 Paris Cedex 05

**I.N.T., 9 rue Charles Fourrier, 31011 Evry

Topics: Tools for Petri nets

Abstract

This paper presents an AMI extensible Petri nets interactive workshop which aim is to provide a base from tools developpement to their use. The workshop supplies with an homogeneous and user friendly interface distributed among workstations and hosts: It provides an interactive toolkit for creation, manipulation and analysis of typed graphs with attributes whose Petri nets are a particular case. A graphic editor MACAO has been elaborated for nets introductions and results display on a dedicated workstation which may be use either in connection with the workshop or alone. To speed up the introduction of results of the theory, the AMI workshop integrates a new inference motor NIMA which has been conceived in order to enable a fast trancription of theories into expert system rules. The AMI workshop provides the researcher and designer with future prospects for modelling, analysing and validating parallel systems.

1. Introduction

The Petri nets are a complete model for specification and validation of parallel systems. They are mostly used in industry and in research fields because of their suitability for modelling parallel systems, the legibility of their graphic representation and the scope of theoretlcal results obtained. Their use still depends on tools [Feldbrugge 87] availability. A first category of basic tools groups together construction and edition tools for Petri nets, whereas the second is made of more sophisticated simulation and analysis tools [Jensen 87].

The increasing complexity of the analysed systems and the requirements of the designers have led to hIgh level Petri nets. The theoritical results on those models are uncompleted but advance very fast. Users have at their disposal a wide range of analysis algorithms more and more powerful for ordinary nets [Starke 85...], coloured (Jensen 84, Colom 86...], with fifo [Behm 85, Memmi 85], predicated [Wheller 85, Vautherin 86, Treves 87...], stochastics [Florin 86, Chiola 87-1, Zenié 87...]. Graphic tools [Beaudouin 83, Montel 83, DESIGNTM 86, Chiola 87-2] and demonstration systems [Azema 85, Choppy 85, Queille 82, Fraisse 86] improve the modelisation surroundings. Despite this let of work, the tools stay fixed at the theoric level when they were programmed and are extended too slowly. This increasing gap between theories and their implementation leads to an outmoded heterogenous group of tools. A systematic integration of the forwarding theory asks for a tremendous work of analysis and programation because of the diversity and the complexity of the models. The rise of a new type of nets creates also problems of re-use and compatibility of the tools already achieved.

In order to speed up the fulfilment and plain integration of software tools we propose AMI an extensible Petri nets interactive workshop. Our aim is to provide a base underlying everything from tools development (programming, test, documentation and maintenance), to their use (conception and analysis of specific Petri nets).

It was important to find a common underlying model for Petri nets: the typed graphs with attributes. The AMI workshop enables to work on every kind of graphs. It groups together an interactive toolkit for creation, manipulation, transformation, analysis and checking the graphs properties. But it is basically designed in order to be extended to wider domains. A group of algorithm analysis for regular ARP nets [Haddad 86] we developped has been easily integrated. A graphic editor MACAO has been elaborated for nets introduction and results display on a dedicated workstation which may be used either in connection with the workshop or alone.

The introduction of new models and their referring theory would be slowed down by its cost and the programming or algorithms modification delays. A new inference motor NIMA has been concepted to enable a fast transcription of theories into expert system rules. It is specially suited for the difficult problems brought up by nets and graphs: multiple quantizers, efficiency of data representation choices adapted to those problems, cooperation between rules and algorithms [Beldiceanu 87,88]. This expert system is at the Integration stage in the AMI workshop.

Parts 2 and 3 show the AMI workshop motivations and its functional architecture, parts 4 and 5 detail the user and graphic Interfaces and part 6 and 7 a fulfilment of the workshop and a case of use.

2. Motivations

The AMI workshop satisfies a real need because at present there is no tool able to integrate new categories of Petri nets, their analysis tools and the graphical display of the analysis results. It is very difficult to share results between the tools because most of the present applications have no any common intermediary representation and there is no overall results management about an analysed net with different tools.

Four basical problems have led to the AMI workshop conception.

2.1 How to integrate new models?

For most tools the suitable net types are implicitly defined by the collection of available building and analyzing programs. Model extension generates inextricable modifications of structures implicating a rewritting of numerous programs.

To reach the aim, the AMI workshop manages models in an explicit way. The whole knowledge referring to each model defines a **domain** which may be described externally from the workshop and independently from future analysis tools.

A domain is composed by a description of its object (nodes and arcs), as well as their attributes. This description specifies different graphic kinds of nodes and arcs. For instance in the domain of an ordinary Petri net, the node *place* has got the attributes *name* and *marking*.

2.2 How to provide an aid of development and exploitation?

The implementation of operator interfaces is tiresome and comes again with each new application: data capture, applications functionalities (menus, diagnostics, helps), results picking up.

The interfaces normalisation improves the comfort of the user who gets accustomed to the similarity of uses and can consecrate time to use better the tool functionalities. This normalisation brings up to implement parametered interfaces, which are particularised by the programmer for each application through descriptors. The interface becomes by this way a data and no more a part of the code application.

In order to normalise the graphic capture and the results display, the workshop supplies a graphic interface MACAO. The MACAO tool for Modelling, Analysis, Computer Aided cOnception manipulates all kinds of graphs, particularly Petri Nets but also transition systems, PERT Networks ...For its integration, the graphic interface is configurated and guided by the workshop. It consists in particularising the interface, so the workshop enables the applications to have access to the graphic operations. For instance a *flow* computed by an application can be visualized graphically by highlighting nodes and arcs or by displaying a formula.

There remains software tools elaborated by researcher or trainees for particular pieces of theory. Despite their importance, these tools could be lost if they stay isolated because of the raw ergonomics, a succint use and results only exploitable through files interpretations. The AMI workshop decreases the unproductive work of the reprocessing of these tools.

2.3 How to assume sharing and communication between different tools?

When one has at his disposal different tools working on the same model, following problems appear:

- no external unified representation,
- no simultaneous possibility data entry,
- no sharing of results from different tools

Our workshop uses an internal communication language CAMI. CAMI allows to describe each graph and defines by this way a unique intermediary representation.

The workshop does the simultaneous capture and ensures the coherence between the graph manipulated by MACAO and its different internal representations for each tool.

Our workshop allows to insert the results of algorithms into a knowledge base which provides the place for communication between all those algorithms.

2.4 How to create new applications ?

In order to create a new application on high level model, it's interesting to be able to re-use algorithms working on underlying or similar models. Therefore, it is necessary to constitute an algorithms and data structure library and to have at disposal language which can put them in work easily. In adition, it's important to dispose of tools for interpreting the very often complex results of that new application.

The workshop integrates a specified expert system NIMA of graphs and nets analysis letting cooperating together algorithms and rules. The expert system approach allows a description of chosen analysis methods with rules and additionnally a description of the nets adaptated representations by facts. This approach enables also an automatic sequence of proving methods conducting by data, and an automatic results interpretation. It makes easy the integration of complex theorems on Petri nets with the help of rules very close to their mathematical definition.

Conclusion

In order to simplify tools integration, AMI offers a layered structure as well as domain and tool description. It enables an unified base for ordinary, coloured, predicate, temporised, stochastic nets... providing the user with different levels of net analysis and the programmer with opportunities of re-use and tools extension.

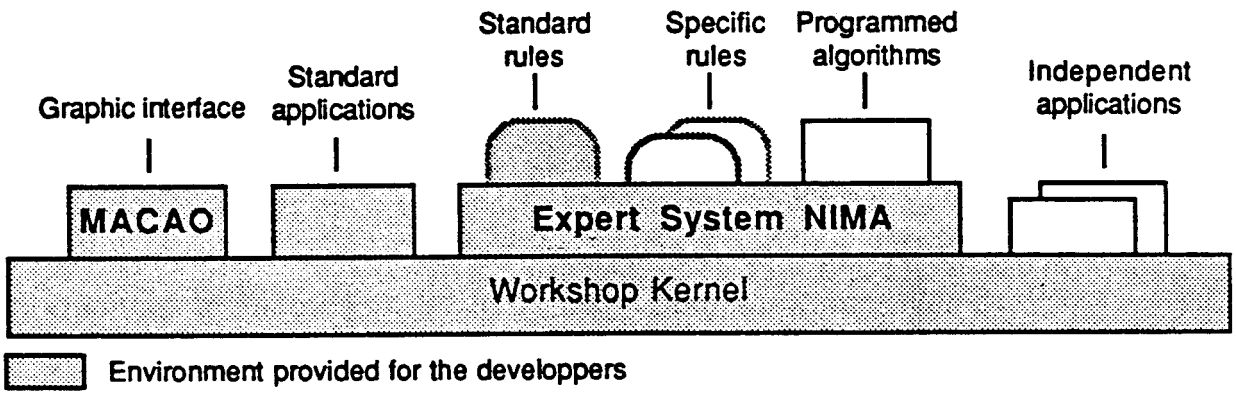

Fig 2.1: Structure of AMI workshop

The workshop offers to the programmer different ways for introducing new applications:

- new independent applications,
- programming with rules,
- mixing rules and classical programming.

3 Functional architecture

This workshop is a complex group. structured in layers allowing to break up the system and limit the number of interfaces that a primary element has to know.

3.1 Introduction to the layer structure

Each layer talks to its opposite one: one layer on the user side and one on the host side. The two sides manipulate the same data but the processings are different. The communication between layers on the same level are asymetric; because in order to avoid a conflict at the resources management level, master-slave kind of relations are established. The user side plays a role of master for each layer to leave initiative for the designer.

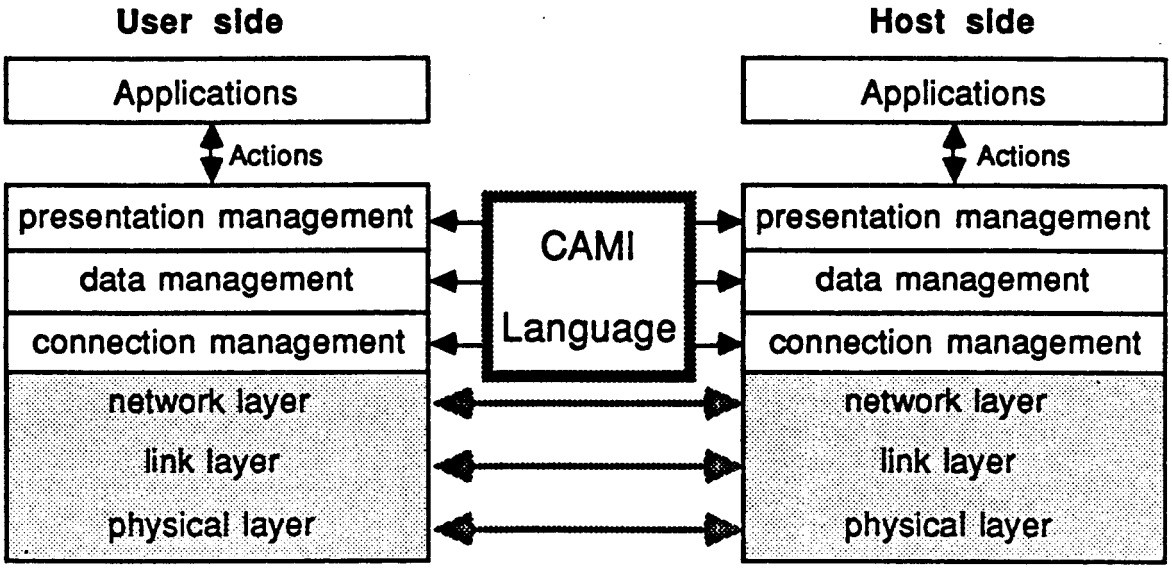

Fig 3.1: AMI Workshop layers

This paragraph introduces the AMI layers.

- The **connection management layer** provides the proceeding and functional ways to establish the data connection. It is responsible for the data exchange.
- The **data management layer** checks the user's name, his choice, his work domain and manages the data coherence.
- The **presentation management layer** is responsible for the presentation of the data exchanged by the application. It defines completely the manipulation of information and knows the functionality of those applications.
- The **application management layer** is the most external layer of the model. On the host side, the applications know the semantic of the work domain. The analysis tool ARP, the expert system are situated in this layer. The user side is the user interface and is composed of data capture and display applications: the graphic tool MACAO belongs to this layer.

3.2 The CAMI language

CAMI is an internal language for description and manipulation of objects. It's made of orders each referencing an action or a succession of specifications. It is divided into subsets associated to the communication between layers of same level.

In order to ensure the interapplications compatibility and an easy external translation, all the CAMI commands follow the same simple syntax: the label of the order followed by a list of parameters between brackets. This parameters, some optionals, specify the action generated and are only interpretated by the layers they are Intended to. They are principally of three types: characters strings, integers, enumerated types.

The CAMI langage while being readable is not an external language, the user wishing to specify a Petri net will do it graphically or in his Petri nets language [LRI87] which will be later translated into CAMIlanguage.

3.3. Connection management

The connection management is responsible for the exchanging (in full duplex and with no error) of data blocks called messages. The messages can be of variable size. The messages contain CAMI orders that will be interpretated by the superior layers.

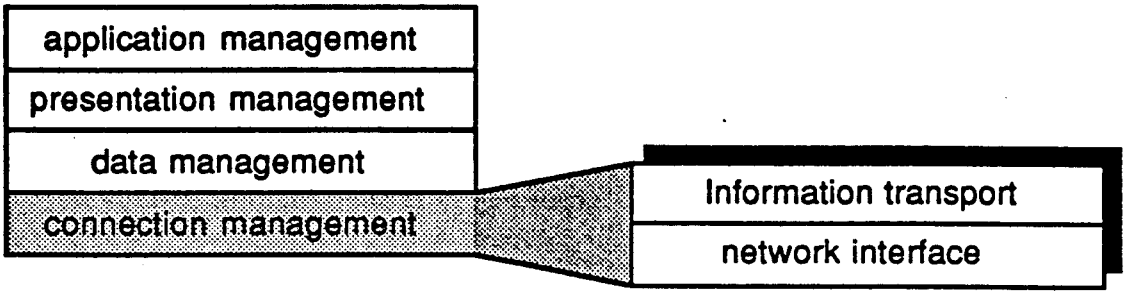

Fig 3.2: Details of the connection management layer

The **network Interface** ensures the transmission of message between the connection managment and network layers.

The **information transport** ensures the correct ordering of messages and an optimal use of the connection. It performs a stream control between transmitter and receiver. In case of irretrievable errors (no answer from the receiver ...) the superior layer is informed.

3.4 Data management

Data management is divided into four sublayers (Cf fig. 3.3): users, domains, attributes managers and data multiplexing sublayer.

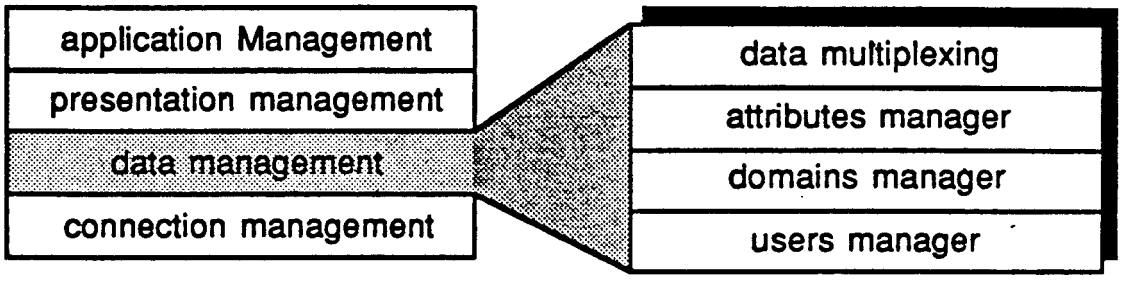

Fig 3.3: Details of the data management layer

The **users manager** keeps the information concerning the user to transmit them laterly to the upper layers. This sublayer adapts the communication to the version number corresponding to the tools in connection. On the user side, it identifies the user by his name and determines the idiom (English. French) used. On the host side, it checks with the name the access modes to the workshop, in particular the access authorization. CAMI orders at this level are open communication (OC) and close communication (FC).

The **domains manager** manipulates the list of the domain names known in AMI. On the host side, it has a synthetic view of all the domains (graphs, Petri nets...) on which a user may work. It transmits to the user side the list of the domains by means of CAMI orders: begin list (DD), the domains names(VD) and finally the end of list (FD). On the user side, it ensures that the designer works on a known workshop domain.

The attributes manager works on the attributes table of the user work domain. The domain table defines the syntax of the corresponding type of nets (cf Chap. 2.1). The graphs are made of two classes of objects (nodes, arcs) to which are associated litteral attributes of nomination and property. All the objects of the graph and all their attributes are referenced by a unique internal number.

Example of a table of the attributes of the domain "ordinary Petri nets":

For the ordinary Petri nets, the table of the domain contains nodes like *place* (round) and *transition* (rectangle) to which are associated the litteral attributes *name* and *marking* for the *place,* and *name* for the *transition.* There is only one kind of arc for this net: *the standard arc* having an attribute called *valuation.*

Node/Arc	Form	Type	Name/attribute	Nomination/Property	Unicity	default value
Node	round	*place*				
			name	Nomination	unique	
			marking	Property	unique	0
Node	rect	*transition*				
			name	Nomination	unique	
Arc	arrow	*standard arc*				
			valuation	Property	unique	1

The user having chosen his domain, the attributes manager on the host side, sends him the table of the domain (TD), followed by the list of the objects (OB), and the list of the attributes for each object (AT), and the end of the table (FT). Specific tools are availables for programmers to create new domains and their tables.

The **data multiplexing** sublayer allows the user to analyse at the same time several independent graphs: these analyses of a same graph make a session. On the user side, the attending of each analysis and the corresponding results will be displayed in different windows.

The CAMI orders to open (OS), suspend (SS), resume (AS), end a session (FS) are interpretated by the data multiplexing sublayer.

3.5 Presentation management

The presentation management is made of three sublayers : the coordinator offering the user a choice of applications, the dialog manager indicating him the functionalities of the application and the last one taking care of a transcoding between the CAMI language and the chosen applications.

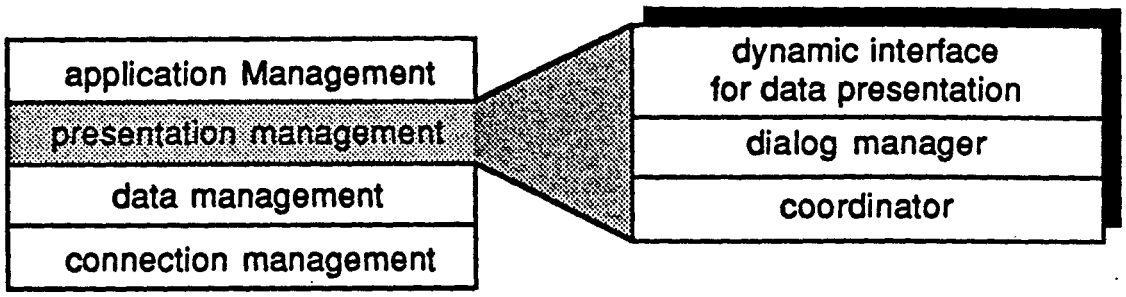

Fig 3.4: Details on the presentation management layer

During a session, the user may choose among several applications to analyse his graph. The **coordinator** deals with the link towards those applications, on the host side, according to this choice. The access conditions for a user to an application (casual user, beginner, expert, programmer or administrator) are also registred by the coordinator on the host side.

The **dialog manager** checks the application functionnalities. These functionnalities are expressed by a question tree (a group of questions in sequence) and transmitted to the host side. The dialog manager on the user side deals locally with the chaining of questions and transmits all the questions when the user has correctly asked for a coherent group. This language of questions (sublanguage of CAMI) allows not only to make one or several choices but also to transmit variable data like text or graphic objects.

The table of the attributes of the domain is not sufficient to express the results (answer) of a question, so the dialog manager asks the inferior sublayer (coordinator) to complete this table or to use a new domain to display the results. For instance in the case of ordinary Petri nets, the reachability graph is a new domain *transition systems:* only one kind of node with attribute *marking* and only one kind of arc with the attributes *transition* and *name.*

For the programmer, AMI gives a tool for creation and manipulation of questions trees. While defining questions, the programmer may bring together pre-defined functionalities such as asking for information on an application run, authorization to break off, stop the running analyses. The questions are managed by the dialog manager but their treatments are performed by the application.

Example of Questions tree defined on the host side

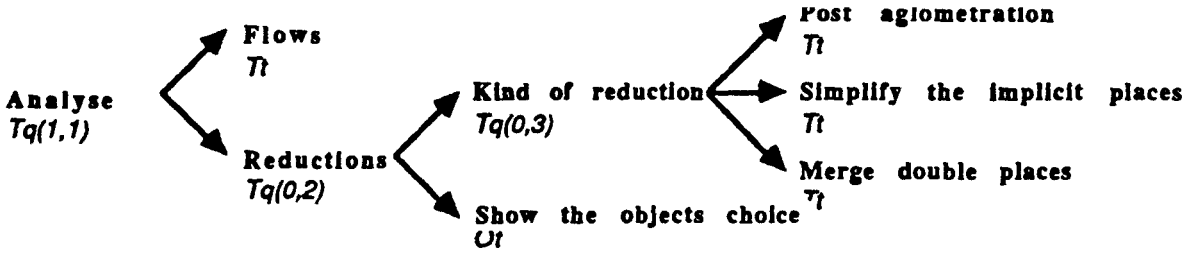

Fig 3.5: Questions tree

The figure 3.5 represents a questions tree. The Tq means that this question will be followed by another question, Ot means that the user have to select a set of object and Tt Is the leaf of the tree. The parameters between brackets are the minimum and the maximum simultaneous choices for the next question.

Inteprretation of the Questions tree on the user side

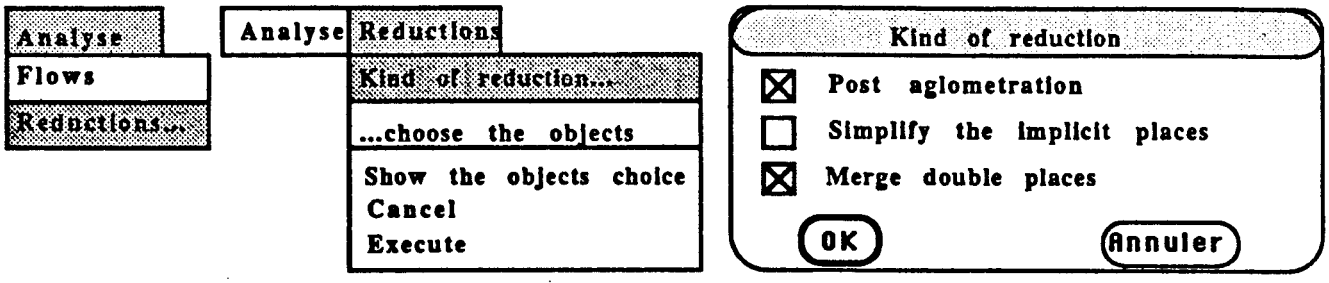

Fig 3.6: associated menus

This example (Cf Fig 3.6) represents a questions tree and Its display on the user side. The questions *Analysis* and *reduction* are displayed by menus and the others by dialog windows. The "Terminal object" in the Item "...Choice of objects" shows that the user can elect a group of objects in his graph before asking for this menu. This choice of objects may be a selection of graphics objects controlled by the upper layer.

The dialog manager, at the user side, performs an interpretation of the question tree. In the Interpretation above it adds the Item *Visualize the choice* In order to show the user choice, the item *cancel* allows at each step to go back and the Item *execute* shows that he has finished to answer the questions. It uses a dialog window for the *Kind of Reduction* because all the items are terminal texts,.

The **dynamic Interface for data presentation** (Cf Fig 3.4) simplifies the integration of applications not conceived to accept the litteral description of graphs in CAMI language. It transcodes, by rewritting rules the subgroup of CAMI orders of this layer into actions, specific to each application. At this step, the graphs are described by CAMI orders listing all the nodes (CN) followed by all the arcs (CA). Each one of these objects is followed by the list of its attributes (CT) stemming from the table of the domain.
Making clear this three commands: .

CreatNode (kind of node, node number)
CreatText (kind of text, text nb, object number, string)
CreatArc (kind of arc, arc number, beginning node, ending node)

Transcoding table for NIMA for ordinary Petri net domain.

commande	%1	%n	Actions
CN	Place		RelAjoute(placetransition, %2.Place)
CN	Transition		RelAjoute(placetransition, %2, Transition)
CA	Arc		RelAjoute(DebArc,%2,%3) RelAjoute(FinArc,%2.%4)
CT	Valuation		RelAjoute(ValArc,%2,%3)
CT	Marking		MatAffecte(Marking, M0,%3,%4)

The programmer defines the actions for a domain and his application only once.

Draw of an ordinary Petri net with MACAO, the CAMI commands which descibe this net, and the actions generated for the expert system NIMA:

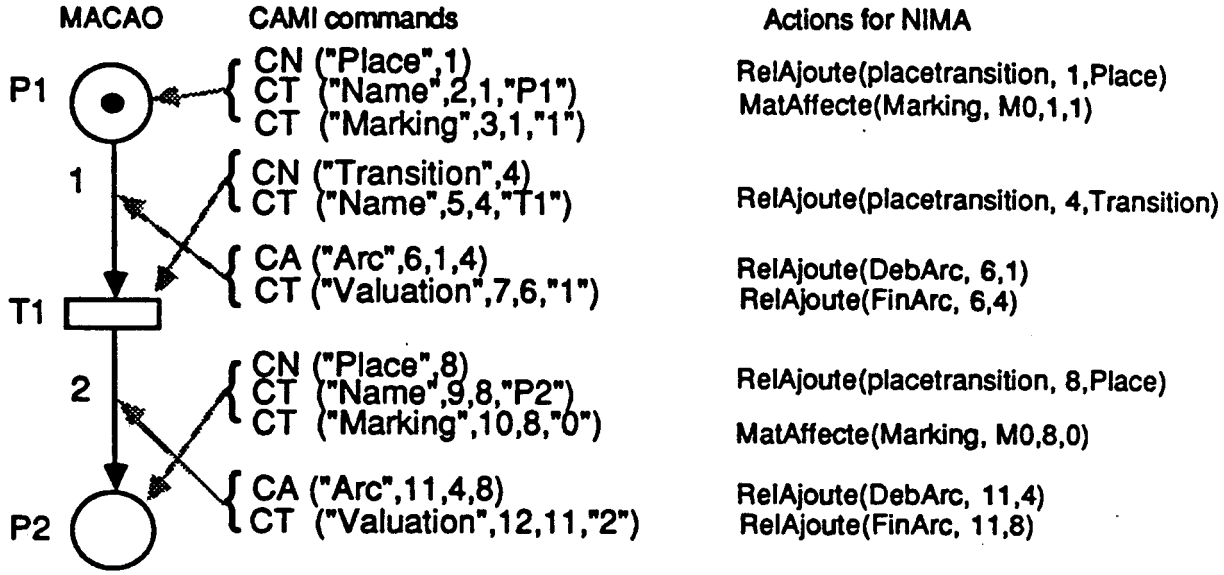

Fig 3.7: Different representations of the Petri net

3.6 Applications management

Adding application in the workshop goes through embedding it by a group of standard interfaces. They allow to control the course of the application centralize the requests for "input output file" and maintain the integrity of the results. Every application owns one graph description. .

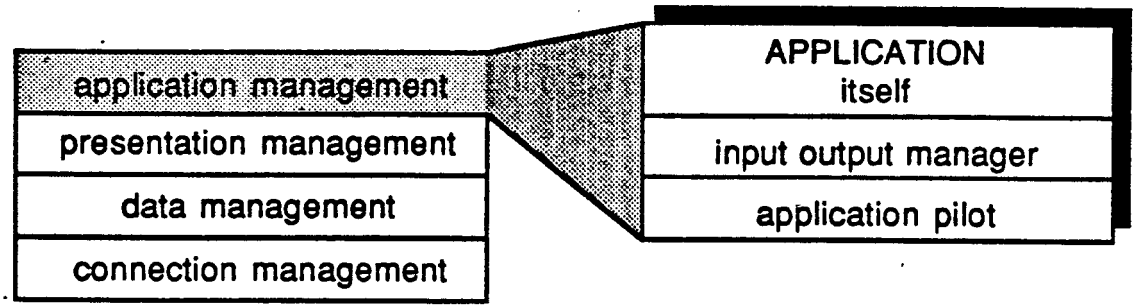

Fig 3.8 Details on the application layer

The **application pilot** offers to the application a set of predefined functions which translate the parameters of function into actions for the presentation management layer. So, an application, on the host, can reach graphical primitives of the user interface. Conversely, the user interacts on the course of the application by actions on the user interface transmitted to the pilot (on the host side) of the application.

The **input output manager** centralizes the opening and closing of files. On the host side, it hides the exact position of the used files. This allows to distribute the applications on sites and to check the coherence of the data. By the record date of the graph on the user side and the name of the user, the Input-output manager establishes the presence of the graph description on the host for the application. It keeps the link between this description and the results files: this allows to avoid useless transfers of descriptions.

The applications are divided into two classes: applications for graphs Introduction and applications for analysis. This two classes correspond respectively to the application on the user side and on the host side. For each class, applications have been implemented: a graphic application MACAO described below and an expert system NIMA. Applications for litteral introduction and interrogation of accessible markings graphs are being developped.

.

The analysis applications are themselves divided into two classes; those which build dynamically the net created by the user (incremental applications) and those who only accept a whole net (atomic applications). An incremental application has to be able to check when asked the semantic and validity of the partial net.

4 User Interface properties in AMI .

To obtain a correct interactive system, highspeed answers are needed to allow the user to immediatly see the results of his directives. In MACAO, to improve execution speed of the commands when an action needs a long computing time, the user is immediatly warned that the action has been taken in account (immediat informative feedback), and he remains informed about the evolution of his command.

Most of the software tools take the lead on the user: selection of possible questions at a qiven time is small and known at the conception of the sofware; others can really be led by the user like expert system or software conceived with an object oriented approach. The first approach suits very well unskilled people or unfamiliar to the domain (novices), while the other is better to expert users. The NIMA expert system is the best that can be adapted to the user's level and that can offer him high level questions. The high level questions need algorithms chainings or rules directed by the data.

The AMI workshop directs the programmer towards the conception of a more friendly interface by using the dialing manager (Cf Chap 3.5).

5. Functionnalities of the AMI graphic application: MACAO

MACAO allows to build and handle simultaneously several graphs. This graphs are made of typed objects. MACAO like all the applications in the workshop distinguishes two classes of objects (nodes and arcs) to which are associated litteral attributes.

For efficency purposes MACAO uses the CAMI language and all the user actions are directly expressed as CAMI commands. The graphs are recorded also in this format.

In the Graph drawing, we distinguish the esthetic, syntactic and semantic parts.

- The **esthetic** part concerns the graphical form of the objects like node size, line kind, colours or positions. The colour or the gray levels may be used to highlight specific objects and visualize some results.

- The **syntactic** part characterizes the different kinds of objects. It reflects the domains table contain (Cf Chap. 3.4). For example, a node of the *place* kind, the definition of attributes of the *marking* type for the objects of the *place* kind.

- The **semantic** part only concerns the kinds of litteral objects: it garantees the values by default of some literal attributes or their unicity. By unicity we mean, the attribut unicity and not the content unicity. We guarantee that an object will have only one name but not that all the objetcs instances will have different names.

The esthetic and syntactic parts are dealt with, by MACAO while the semantic part of the model is analysed by applications on the host side. Thus, MACAO allows to deal alone with the esthetic or syntactic graphs without any link with the workshop.

6. Fulfillment

The AMI workshop meets the goals of Interactivity and conviviality which is an immediate and informative feedback. The analysis tools for Petri nets as well as the graphic Interface need a big computing power, particularly for Industrial nets. Putting tools and Interface on the same machine would increase the answering times therefore the workshop is spread between a dedicated workstation (microcomputer) and servers. That way, this fastidious graphic work which is conception of nets becomes autonomous as well as the exploitation of the analysis results. Moreover microcomputers are easier to carry and allow homeworking.

The chosen architecture is based on the layers structure and makes clear the position of the workstation as the **frontal** of the workshop. The applications being the **dorsal** and the server **leading** the whole (Cf Fig 6.1). The connection between the workstation (Macintosh) and the host (Sun series3) uses serial lines. Thus the Macintosh becomes an asynchronous terminal of the Sun. Then it is enough to be able to connect oneself as a terminal for the Sun to have the Macintosh being the workshop frontal. This connection is therefore possible either directly on a serial port of the Sun or by a bridge on Ethernet.

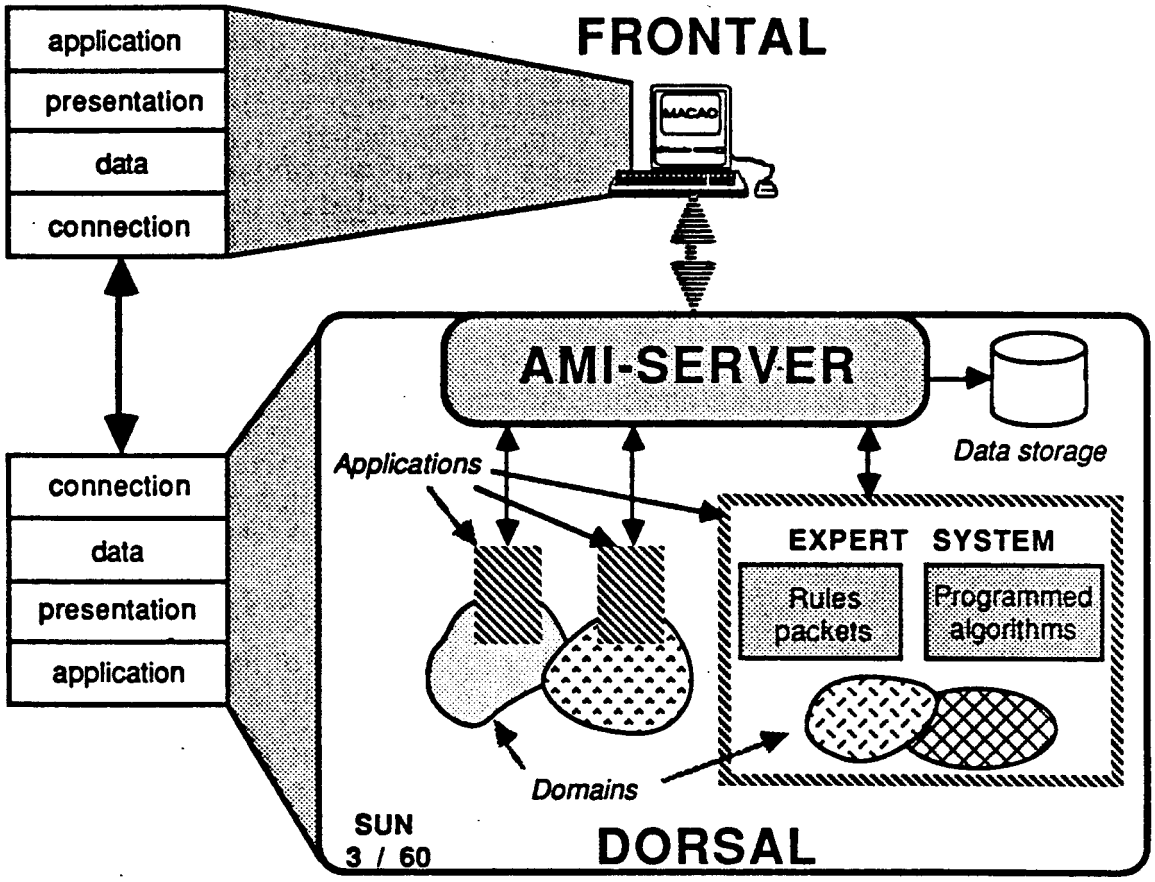

Fig 6.1: architecture of the workshop AMI

6.1 The Frontal

It concerns everything related to the user. The frontal is composed of two parts. The first one is composed with all the layers (user side) from connection to presentation. The second one is made of MACAO application. MACAO allows to build and manipulate simultaneously several graphs with all the well-known opportunities given by the Macintosh environment: user friendly interface, pull down menu, cut / paste towards all kind of applications (ex: text processing), cooperative multitask ...

In the following the user and graphic interfaces will be called MACAO to ease the explanation. MACAO exploits at maximum level the workstation: possible use of several kind of monitors (big, colours), adaptability to the speed of processor and co-processor...

6.2 The AMI-Server

The server is resident on a Sun under Unix Bsd 4.2. It is made of the **connection manager**, the **virtual system**, the **communicator**, and the **presenter**.

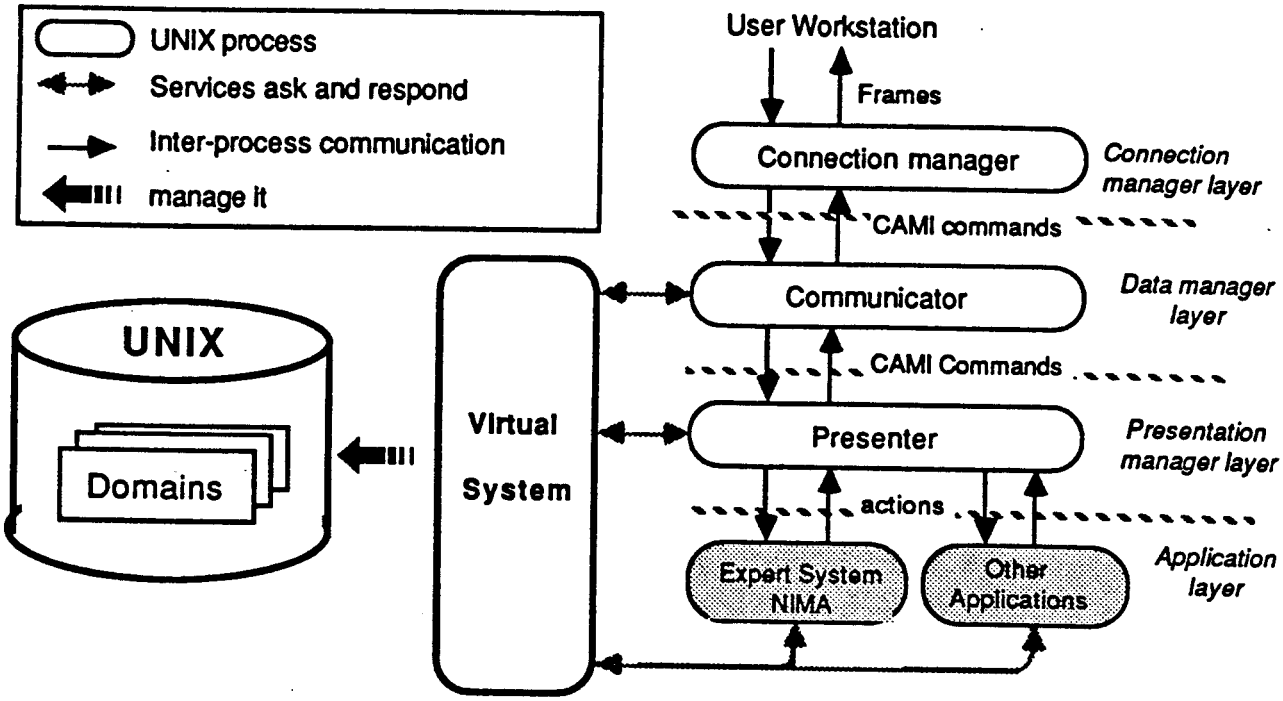

Fig 6.2 Structure of AMI-server

6.2.1 The connection manager

The connection manager manages the physical connection between the Macintosh and the Sun, it uses the protocol developped for the workshop (Pomme-Soleil) based on the HDLC adapted to an asynchronous transmission problem. Its principal characteristics are:

-use of an emission window,
-variable length frames,
-correct ordering of messages,
-line rupture algorithm dectection,
-acknowledgment of frames with delay.

The Sun allows multitasking of the emission and the transmission processes and the Macintosh allows only at a time one task.

The implementation led to a specification of the problems due to the protocols variables exchanges between emission and reception. This implementation in C language has been made in order to make the source programs of the protocol identical for the Macintosh and as well as for the Sun.

6.2.2 The virtual system

The virtual system groups together all the services allowing all the layers of AMI-server and the dorsal input-output manager, to access to the UNIX system, especially to the files.

Several maintenance tools call for the virtual system in order to manage the applications characteristics such as their state of integration (debugging), their input, output data files.

The informations concerning the domains and the users data are structured into two tree structure files: the AMI-server tree structure (Cf Fig 6.3) and the user tree structure. So only the virtual system knows this structure and all the applications use high level primitives in order to manipulate or reach these tree structures.

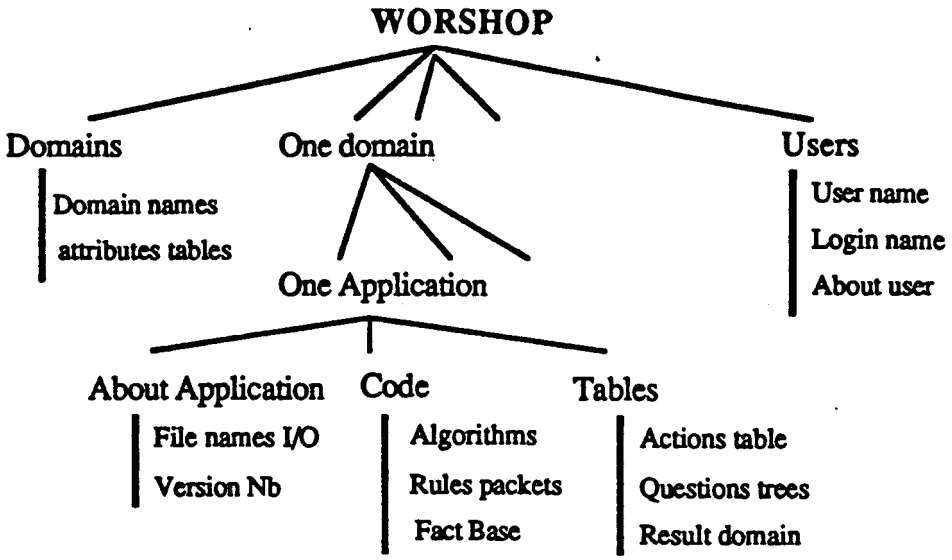

Fig 6.3 Tree structure of the server-AMI

The "workshop administrator" must be able to change this two tree structures without having to make a link of the applications. Thus we used the "client-server" model from the Unix 4.2 system [Sun87] where the server-UNIX manages the workshop's knowledge and the Clients-UNIX make requests of services. The part Server-UNIX being a cyclical process on the Sun (Cf fig 6.4) and the Client part is added to the applications processes and on the other processes of the AMI-server (Cf fig 6.2). The client-UNIX is a library BibSV (Cf fig 6.4). By this way, the virtual System makes the accesses to the services independent from their execution. This mode will also allow to distribute the information from the domains to other sites.

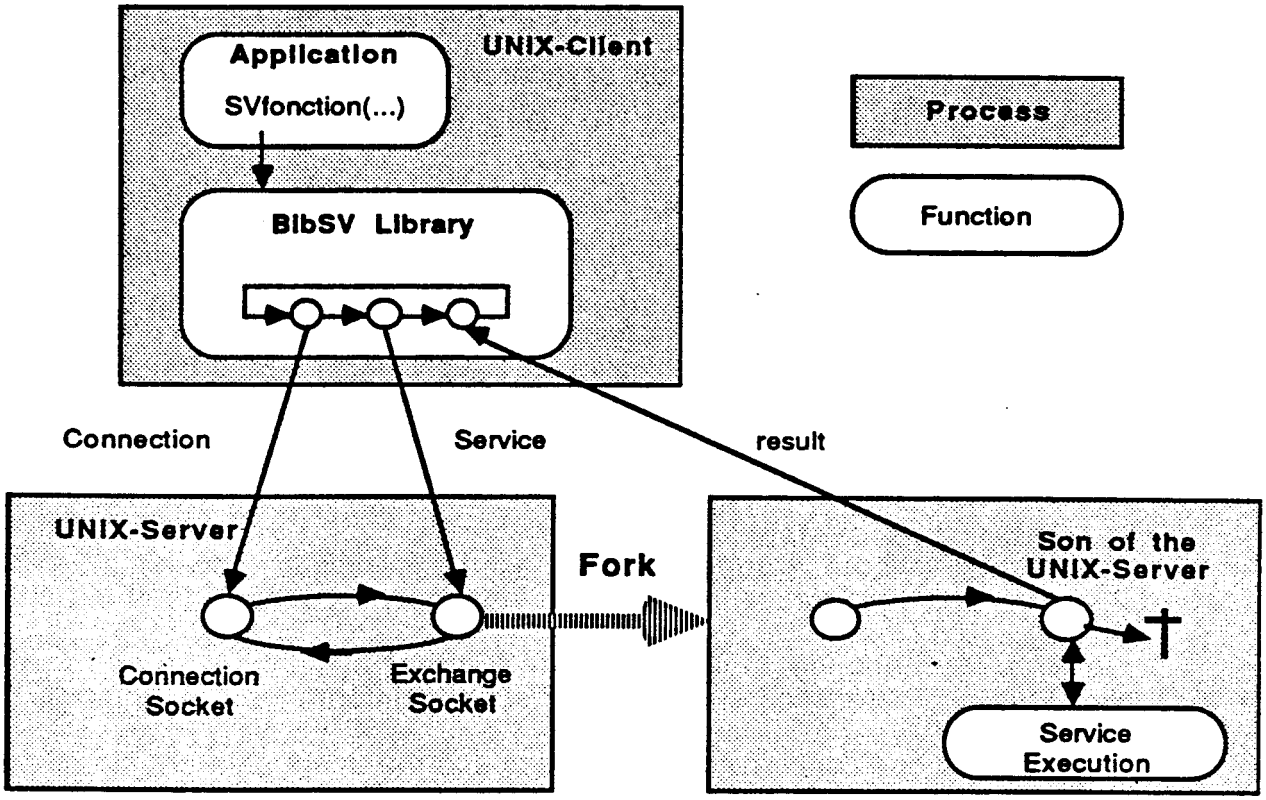

Fig 6.4 The "client-server" UNIX model In AMI

To access to the virtual system, the application calls a function (SVfunction) from the BibSV library (Cf Fig 6.4). This function prepares the service call and becomes the client-UNIX. The server-UNIX is blocked until a client requets service (socket connection). With the connection in hand, the server-UNIX, then forks (duplicates itself) a child process. This child process handles the service and gives the result back to the application. This "fork" is necessary because in case of a big failure in the execution that would kill the child, the father would still be alive.

6.2.3 The communicator

The **communicator** is the process dealing with all the functionalities of the data management layer. It checks the version number of the frontal and allows the simultaneous opening and control of several graphs on the workstation and links the user's name on the Macintosh and his login name on the Sun. It recognizes the user (access security) knows by calls to the virtual system where are situated the user data (management of the user tree structure in his home directory).

The communicator uses the virtual system in order to know all the domains available on the Sun. It transmits the list of domain names to the frontal. It sends the attributes table of the domain chosen on the frontal.

6.2.4 The presenter

The **presenter** is the process dealing with all the functionalities of the presentation management layer. It compares the users acces levels to the different applications in order to suggest some of them. Among the elected applications the user will have to favour one (the primary application). This one will offer him an interactive dialog and the visualization of the results. The other applications (secondaries) share the same graph description as the primary.

The presenter knows the global state of the questions tree of the primary application. It tells the user on his Macintosh about the state of the running questions on the host machine. It ensures the desynchronization between the frontal and the applications, leaves the running of the applications in the background when the user disconnects himself from the workshop.

6.3 The Dorsal

The dorsal groups together the applications working on the users data: simulation, validation, reduction, accesibility graphs and so on, of these nets flows.

The dorsal pilot functions (Cf Chap 3.6) are provided to the application by use of a library (BibPL). The dorsal input-output manager uses the virtual system to ensure integrity between the graph descriptions in the user tree and the graph description on the Macintosh. To integrate an application, the programmer must link with two libraries BibSV (Cf Chap 6.2.2) and BibPL. Thus, the AMI workshop provides a set of object-oriented libraries that automatically implement the user-friendly interface, thus simplifying and speeding up the process 0 software development. It also offers by using the NIMA expert system an algorlthms base and a package of rules re-usable on different domains (at present graphs and Petri nets).

The ARP a software for regular nets analysis based on an extension of the Petri nets developed in our laboratory has been included in the AMI workshop. ARP brings the designer the definition of a litteral description language (LDR) and validation tool. An ARP pilot has been realized, it changes in one way the actions stemming lrom CAMI language into an LDR description and on the other way the visualization of the results. This integration has allowed the user to build graphically the nets, to hide the existence of a compilation phase and visualize graphically the generated flows. In brief, the use of ARP has been radically improved. The regular nets theory has otherwise evolved [HAD88] , ARP and the next tools on this nets will be developed around the expert system.

6.4 Case of use

The frontal (Macintosh) records the user name and emulates an asynchronous terminal on the Sun and opens the communication (Cf Chap3.4). A dialog in CAMI language (invisible to the user) is established between the frontal and the server: the layer managing the users of the Sun checks the access autorisation to the superior layers of AMI.

After reception of the domains list, the user creates a new graph and chooses a domain, or resumes his work on a prior one. The AMI-server transmits to the user the applications available on the domain. The user chooses an application and the server transmits him a questions tree (Cf Chap. 3.5) containing informations about the questions.

We consider the use of the ARP software. ARP adds a questions tree on the workstation. So the *UnixTM* and *A.R.P* menus appears (Cf fig 6.5).

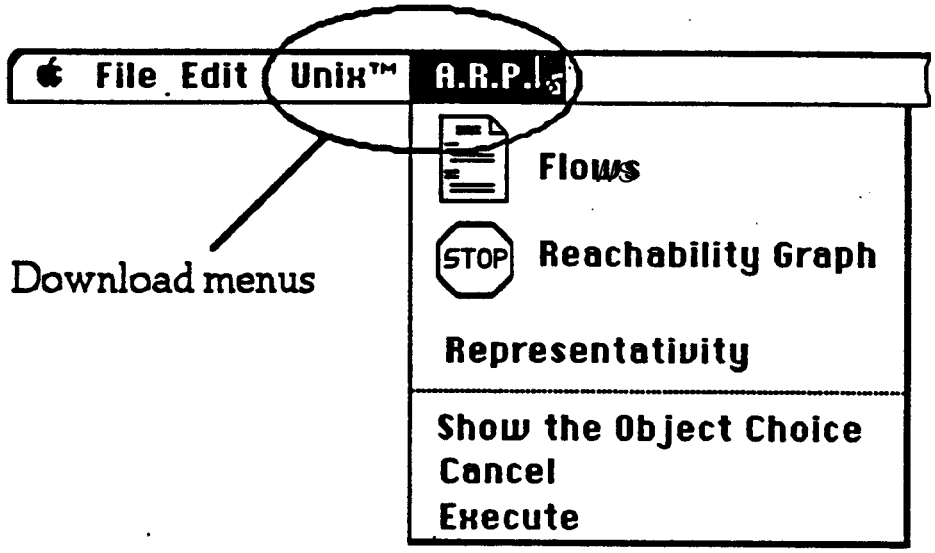

Fig 6.5: Downloaded menus

The figure 6.5 shows the state of the questions: the result of *flow computing* is available, the *Reachability graph* is not complete because It has been stopped and the *Representativity* has not been asked yet.

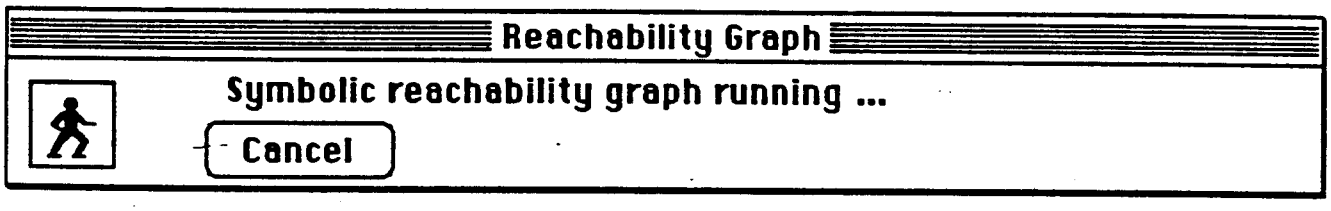

Fig 6.6: Information window

After the choice of a Downloaded menu item, the frontal shows temporarily an information window on the course of the running operation on the server. An icon represents the state of the question while the text line as well as the *stop* button are controled by the application pilot. This window is automatically closed at the end of the question computing. If the user stops the process by using the *stop* button, the state of the process is recalled in the menu in the form of a STOP icon and it will possible to resume this work.

7. Conclusion and perspectives

AMI gives an outlet with new modelling, validation, proving tools offering new working methods to the users.

Now analysis tools can involve at a similar speed in comparison with theories because the workshop provides to the user working conditions which permit him to consecrate entirely his time to the algorithm development while using tools elaborated by other researchers.

Algorithms, their data structure, extracts from analysis applications from temporised and coloured stochastic Petri nets are at the moment being integrated in the workshop above the expert system.

In the next AMI version, the AMI-server, which presents a synthetic view in domain and their respective applications, will use the NIMA expert system. by that way, the server will manage with more subtlety the workshop knowledges and offer to researchers and designers information tools about the whole algorithms and rules available in the workshop.

8. References

P .Azema, G.Papapanagiotakis: "Protocol analysis by using predicate nets", Fifth international workshop on Protocol Specification, Testing and Verification, Toulouse-Moissac, June 1985, M.Diaz ed., North Holland, 1986

P .Behm: "RAFAEL: A tool for analysing parallel systems In the L environnement", Sixth european workshop on applications and theory of Petri nets, Espoo -FInland, June 26 -28,1985.

M.Beaudouin-Lafon: "Petripote, a graphic system for Petri net design and simulation", 4th europ. workshop on applications and theory of Petri nets, Toulouse France, June 1983, pp 20-30.

N.Beldiceanu: "Un langage de règles de production basé sur des contraintes et des actions", 7ieme journées Internationales sur les systemes experts et leurs applications, Avignon, mai 1987.

N.Beldiceanu:"Langage de regles et moteur d'inférence basé sur des contraintes et des actions. Application aux réseaux de Petri". Thèse de docteur de l'université de Paris VI, 1988.

G.Berthelot, J. Vautherin, G. Vidal-Naquet: "A syntax for the description of Petri Nets" Rapport LRI Université Paris Sud 1987.

G.Chiola: "Structural Analysis for generalized stochastic Petri nets: some results and prospects", eighth european workshop on applications and theory of Petri nets, Zaragoza -Spain, June, 1987.

G.Chiola:"A Graphical Petri Net Tool for Performance Analysis" in International workshop on modelling techniques and performance evaluation, PARIS, March 87, pp 297-307.

C.Choppy, C.Johnen: "Petrirève: Petri net transformations and proofs with rewriting systems" Sixth european workshop on applications and theory of Petri nets, Espoo -FInland, June 26 -28, 1985.

J.M.Colom, J.Martinez, M.Silva: "Packages for validating discrete production systems modelled with Petri nets", Proceedings of the IMACS-IFAC Symposium, June 1986.

DESIGNTM. User guide. MetaSoftware Corporation, Massachusett 86.

F.Feldbrugge, K.Jensen: "Petri Net Tool Overview 1986", in Petri Nets: Applications and relationship to other models of concurrency. W.Brauer, W.Reisig, G.Rozenberg editors, L.N.C.S. n°255, Springer-Verlag 1987, pp 20-62

G.Florin and S.Natkin: "RDPS: a software package for the validation and the evaluation of dependable computer systems", Proceedings IFAC SAFECOMP workshop, Sarlat, France, October 1986.

P.Fraisse, C.Johnen, N.Trèves: "SERPE, an extensible structure for analysis of Petri nets", RR L.R.I. Université d'Orsay. September 1986.

K.Jensen: "The design of a program package for an introductory Petri net course", in Advances in Petri nets 1984, G.Rosenberg editor, Springer-Verlag L.N.C.S Vol 188, 1985, pp 259-266.

K.Jensen: "Computer tools for construction, modification and analysis of Petri nets", In Petri nets Applications and relationships to other models of concurrency, W .Brauer, W.Reisig, G.Rozenberg editors, L.N.C.S. n°255, Springer Verlag 1987, pp 4-19.

S.Haddad, J-M .Bernard: "ARP un outil de spécification et de validation des systèmes répartis", Troisième Colloque de Génie logiciel AFCET, Versailles, mai 1986.

S.Haddad: "Une catégorie régulière de réseaux de Petri de haut niveau: définition, propriétés et réductions. Application à la validation de systèmes distribués", Thèse de l'université PARIS VI,1987

S.Haddad: "Towards a general and powerful computation of flows for parametrized coloured nets", Rapport de Recherche MASI, à paraître Janvier 88.

G.Memmi, A.Finkel: "An Introduction to FIFO-nets, monugeneous nets: a subclass of FIFO-nets", Theoretical Computer Science 35-1985, pp 191-214.

B.Montel and al: "Ovide a sofware package for the validation of systems represented by Petri nets based models", Proceedings of the 4th International Workshop on Protocol Specification, Testing and Verification, Toulouse-Moissac,june 1983, pp 292-308.

J-P.Queille, J.Sifakis: "Iterative methods for the analysis of Petri nets", Second European Workshop on Application and Theory of Petri nets, Strasbourg , September 1980", In Applications and Theory of Petri nets *87,* Informatik-Fachberichte, C.Glrault and W.Reisig ed., Springer Verlag, 1982, pp 161-167

P.H.Stark: "Petri netz Maschine -a software tool for analysis and validation of Petri Nets", in Systems Analysis and Simulation, Proceedings of the 2nd International Symposium, Berlin 1985, Oxford:Pergamon 1985,pp 474-475.

SUN Workstation: Networking on the Sun Workstation: Inter-Process Communication Primer Chapter 4

N. Trèves: "COMBAG: a tool for the computation of a basis and a set of generators of semi-flows for Pr/T systems", Intern.conf. on Parallel Processing and Applications, L'Aquila, Italy, September 1987, pp.181-192.

J. Vautherin: "Parallel systems specification with coloured Petri nets and algebraic abstract data type", 7th European workshop on Application and Theory of Petri Nets, Oxford, June 86.

G.R.Wheeler, M.C.Wilbur-Ham, J. Billington, J.A.Gilmour: "Protocol analysis using numerical Petri nets", Fith International workshop on Protocol Specification, Testing and Verification, Toulouse-Moissac, June 1985, M.Diaz ed., North Holland, 1986.

A.Zénié, J-P. Luguern: "Using CS-PN sofware for protocol specification, validation and evaluation", 4th International workshop on software specification and design, Monterey, Califomia, April 1987.