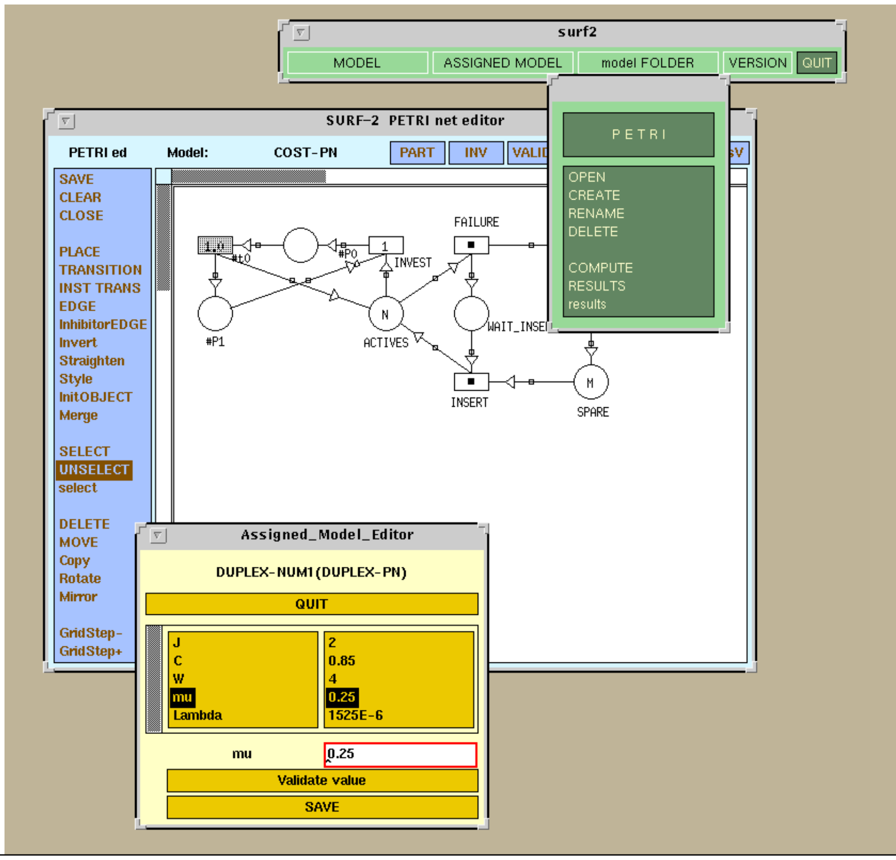
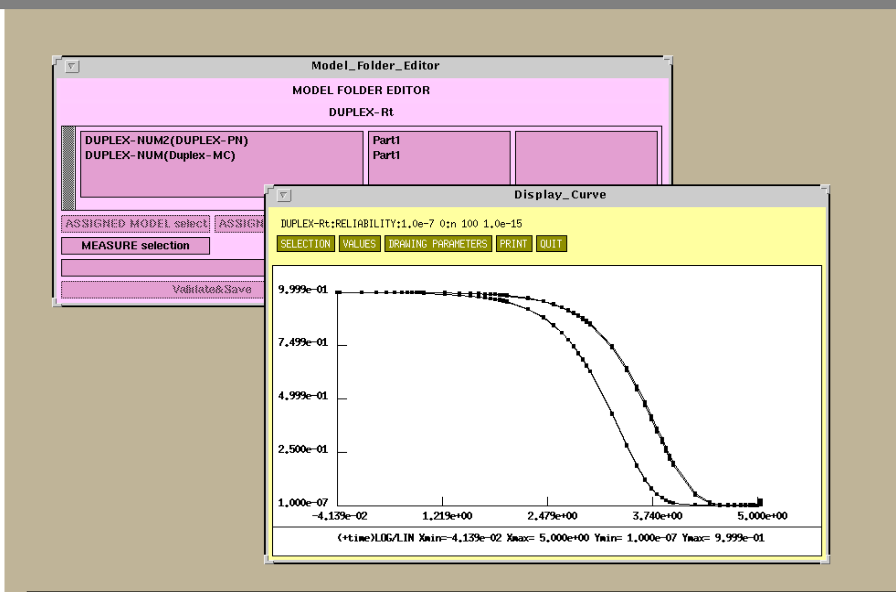


SURF 2 USER GUIDE



CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
LAAS-CNRS
7, Avenue du Colonel Roche
31400 TOULOUSE Cedex (France)

CONTENTS

CHAPTER 10

PETRI NET VERIFICATION RESULTS VIEWER

CHAPTER 11

MODEL FOLDER MENU

CHAPTER 12

MODEL FOLDER EDITOR

CHAPTER 13

MISCELLANEOUS MENU

CHAPTER 14

COMPUTATION

CHAPTER 15

RESULTS VIEWER

CHAPTER 16

PETRI NET INTERMEDIATE RESULTS VIEWER

APPENDIX 1

PARTITION SYNTAX

APPENDIX 2

TWO SIMPLE EXAMPLES

CHAPTER 1

SURF-2 OVERVIEW

1.1 INTRODUCTION

SURF-2 is a **software toolbox for evaluating system dependability**. System behavior may be modeled by either Markov chains or Generalized Stochastic Petri Nets (GSPN).

The specificity of SURF-2 lies in the following novel features:

- The association of the *modeling flexibility* of Petri nets and the *computational power* of Markov chains.
- A *highly interactive mode of dialog*, relying on a user-friendly graphic interface.
- A *wide range of available measures*. The possibility of superposing a reward structure onto a behavior model enables computation of combined dependability and reward measures such as the expected performance or cost.
- The concepts[1] of *model*, *assigned model* and *model folder* enable *sensitivity analyses* to be performed for a given system and *the comparison* of several competing architecture designs.

1.2 SURF-2 TOOLS BOX

SURF-2 consists of software tools acting on objects grouped in a database, see Fig 1.

1.2.1 Data Base Objects

The main objects of the database are the following:

- The **Model** object consists of a *graphical representation of the model* that describes the behavior of a system from the dependability viewpoint. A model is either a Markov chain or a GSPN. In addition to the topology of the behavior description, the *model* object contains *the specification of the initial probability vector* of the Markov chain (or the underlying Markov chain in the case of a GSPN description) and includes one or more *state partitions*. State partitions are needed to specify how the system dependability is to be assessed. Several state partitions can be defined in a given model. A state partition generally consists of two classes—called *proper* and *improper*—due to the principle of proper and improper service that underlies the definition of most dependability measures. For example, the pointwise availability of a system, *A(t)*, is defined as the probability for the system state of being in the proper class at time *t*. However, safety-type measures are the exception to the rule as they require the definition of an extra class referred to as —*catastrophic*—. Thus, a catastrophic class corresponds to the subset of states of the improper class that represents a catastrophic (as opposed to benign)

1. Definition of **model**, **assigned model** and **model folder** objects are given in the sequel.

system failure. By way of another example, the pointwise safety of a system, *S(t)*, is defined as the probability of the system state of not entering the catastrophic class before time *t*. In *model* objects, parameters are specified numerically or symbolically. A symbolic parameter is a local variable whose scope is limited to that of the model in which it has been defined. Symbolic parameters, together with the possibility of defining several state partitions, support the definition of highly generic, re-usable models that can be stored in the database.

- The **Assigned Model** object consists of a *mapping between the symbolic parameters of a model object and a set of numerical or symbolic values*. Symbolic values are employed to assign to some parameters, numerical values that are common to several assigned models associated with the same model or different models. Symbolic values are thus called global values. Global values are convenient for performing sensitivity analyses.

- The **Model Folder** object consists of *defining the dependability measure to be evaluated and specifying the computation environment*. The latter includes a set of assigned models and their inherent models involved in the computation, including the specification of a particular state partition. Furthermore, if the selected measure is either a pointwise measure or a reward measure (such as the expected performance or cost), the model folder then contains global values which are the time and the discount variable respectively. The user must assign numerical values to these two global values. In the same manner, he must assign numerical values to all the global values defined in the selected assigned models. Variables assignment is resumed in Fig 3.

  The diagrams shown in Fig 4 describe the two possible cases, that is the comparison of competing architecture designs and the sensitivity analysis. Of course, SURF-2 allows the two possibilities to be combined.

  Fig 2 illustrates the organization of Models, Assigned Models and Models Folder in Surf-2 data base.

- The **Results** are obtained by processing the dependability measure selected in the model folder and are stored in the database. These results can take the forms of tables of values, curves, asymptotic measures or mean sejourn times. Other results relative to Petri nets, like the reachability graph, the synthesis of structural properties, or the list of conflicts between transitions can also be stored in the database. All those results can be displayed or printed out.

- **Petri Net Verification Results** are obtained by verifying such Petri Net properties as boundedness, liveliness or pseudo-liveliness, as well as specific assertions on the markings.

- **Petri Net Intermediate Results** are obtained during Petri Net processing. They consist of the resulting symbolic Markov Chain, the Marquing Graph and the list of Conflicting Transition and Random Selector.

1.2.2 Menus

When SURF-2 is activated the **MAIN Menu** is displayed.

- The **Main Menu** (see chapter 3) gives access to the three following menus:
- The **Model Menu** (see chapter 4) allows to manage Models in data base and allows to call **Markov Chain Editor** or **Petri Net Editor**.
- The **Assigned Model Menu** (see chapter 7) allows to manage **Assigned Models** in data base and allow to call **Assigned Model Editor**, **Petri Net Verification** and **Petri Net Verification Viewer**.
- The **Model Folder Menu** (see chapter 11) allows to manage **Models Folder** in data base and allows to call **Computation**, **Result Viewer** and **Petri Net Intermediate Result Viewer**.

1.2.3 Tools

SURF2 provides the following tools:

- The **Markov Chain Editor** (see chapter 5) allows creation and modification of **Markov Chain Model**.
  *Access: Command **OPEN** or **CREATE** in **Model Menu**.*
- The **Petri Net Editor** (see chapter 6) allows creation and modification of **Petri Net Model**.
  *Access: Command **OPEN** or **CREATE** in **Model Menu**.*

- The **Assigned Model Editor** (see chapter 8) allows creation and modification of **Assigned Model**.
  *Access: Command **OPEN** or **CREATE** in **Assigned Model Menu**.*
- The **Petri Net Verification** (see chapter 9) runs a stuctural verification of a Petri Net and produce **Petri Net Verification Result**.
  *Access: Command **VERIFY MODEL** in **Assigned Model Menu**.*
- The **Petri Net Verification Results Viewer** (see chapter 10) displays or prints **Petri Net Verification Results**.
  *Access: Command **RESULT VERIFY** in **Assigned Model Menu**.*

- The **Model Folder Editor** (see chapter 12) allows creation and modification of **Model Folder**.
  *Access: Command **OPEN** or **CREATE** in **Model Folder Menu**.*
- The **Computation** (see chapter 14) produces **Result** and possibly **Petri Net Intermediate Result**.
  *Access: Command **COMPUTE** in **Model Folder Menu**.*
- The **Results Viewer** (see chapter 15) displays or prints **Results**.
  *Access: Command **RESULTS** in **Model Folder Menu**.*
- The **Petri Net Intermediate Results Viewer** (see chapter 16) displays or prints **Petri Net Intermediate Results**.
  *Access: Command **results** in **Model Folder Menu**.*

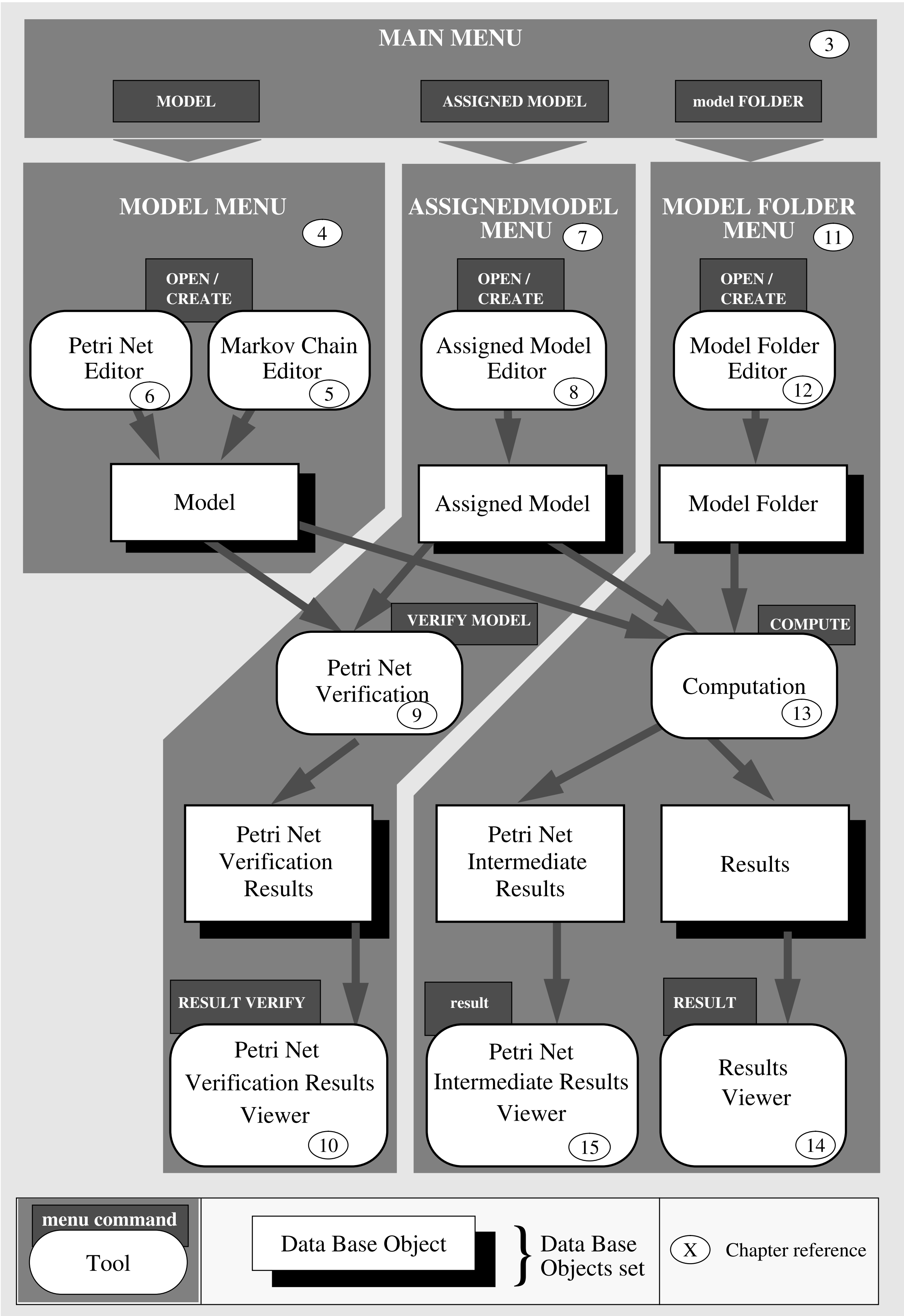


Fig 1 SURF 2 overview from user's viewpoint

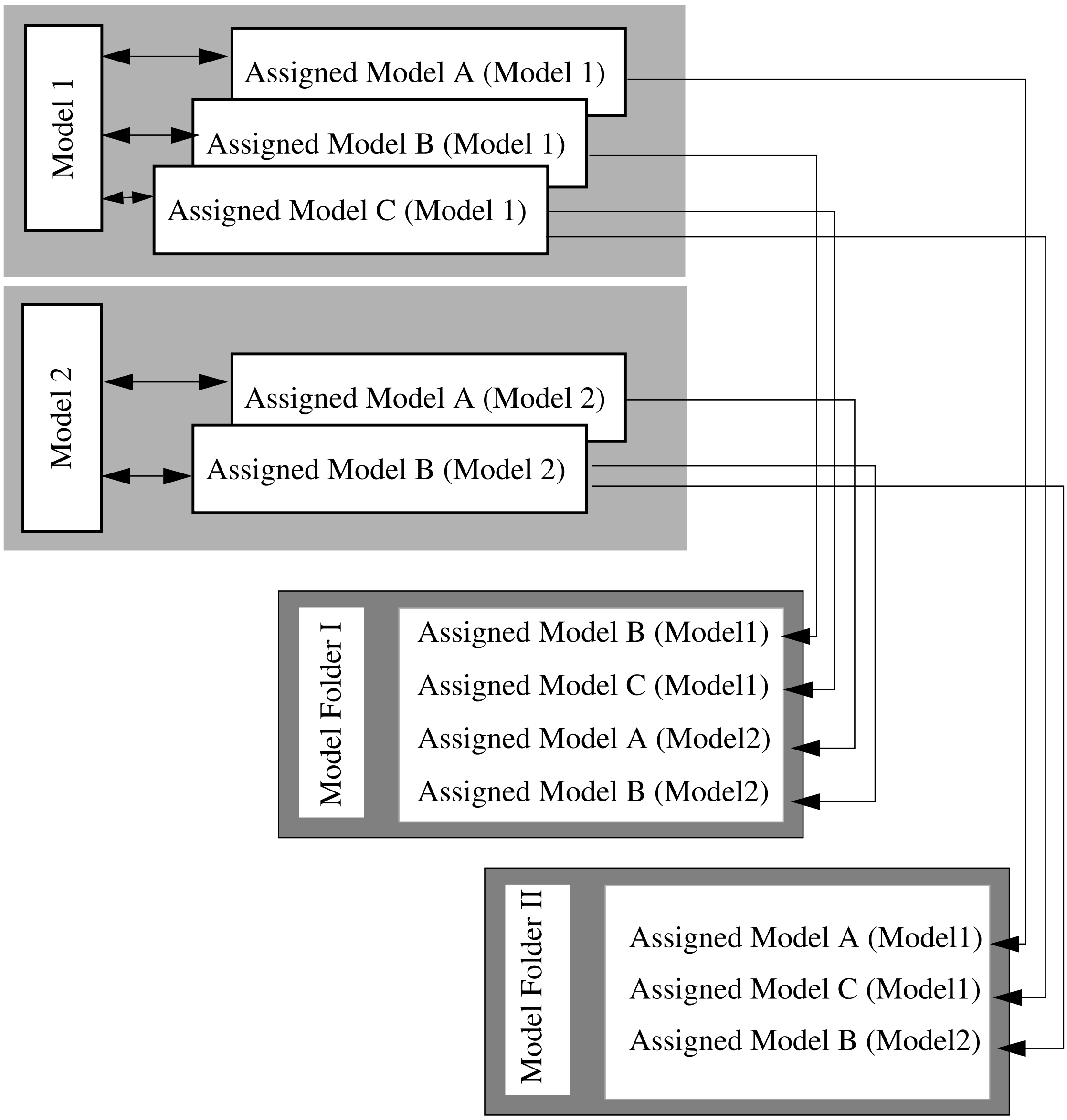


Fig 2 A data base example.

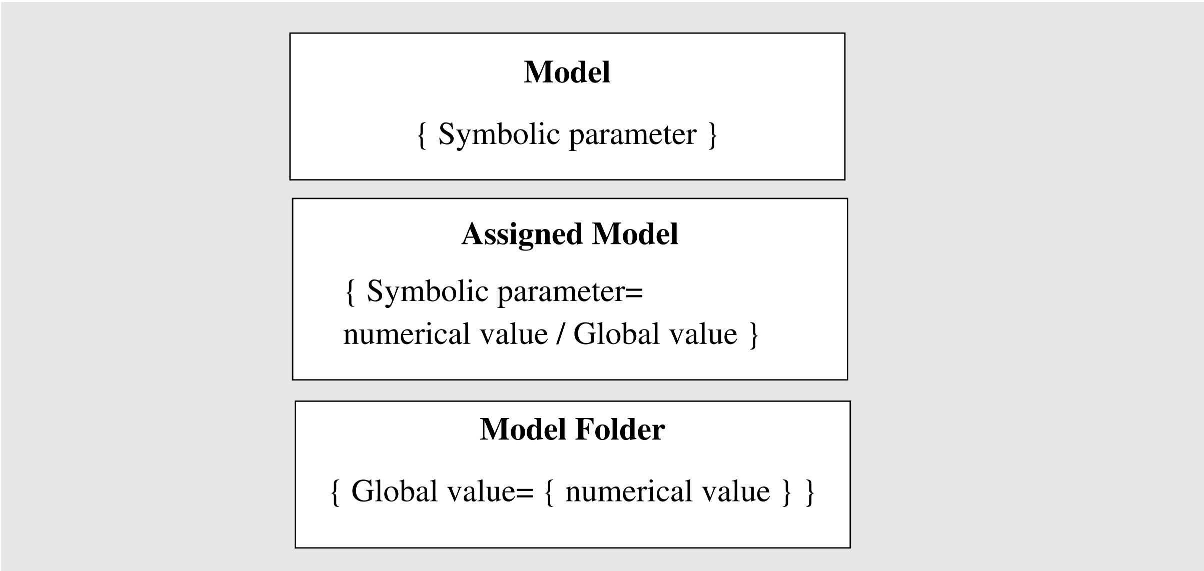


Fig 3 Symbolic parameters assignment.

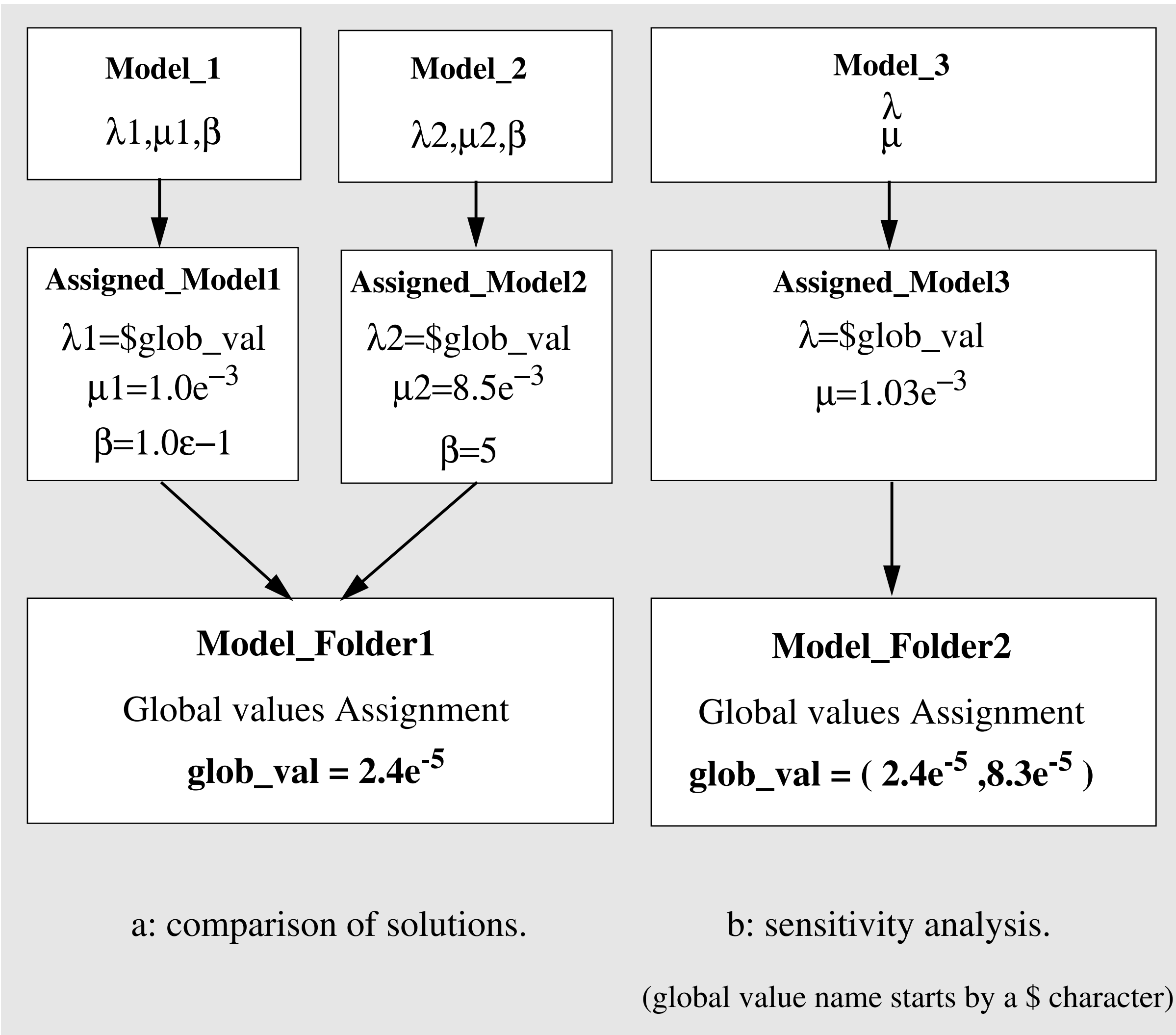
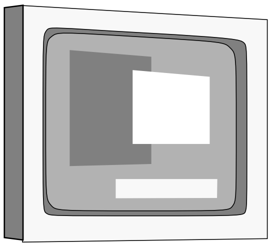


Fig 4 How to use global values.

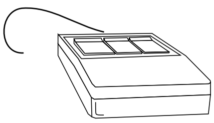
Chapter 2

BASIC CONCEPTS

2.1 TOOLS FOR SURF-2



SURF 2 is a multi-tasking toolbox that uses the **windows manager** of X Windows System. Several processes can be executed simultaneously. Each process runs in a separate window. The user can work in only one window at any given time. This window is called the active window. Windows can be moved using the mouse.

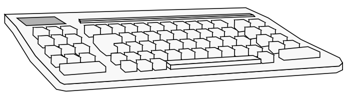


The mouse. Sliding the mouse results in a corresponding movement of the pointer on the screen. Location of the pointer indicates the active window.

Sun Workstations use a 3-button mouse as pointing device in SURF-2. Only the left and the center buttons are used:

The *left button* is used to select objects, user commands or make functions active.

The *center button* is used to move the scroll bars in the active window.



The keyboard is an input device solely used for texts (names of objects, comments,...) or numerical values.

2.2 DIALOG DESCRIPTION

User interacts with SURF 2 principally by means of dialog box.

SURF 2 dialog boxes are made up of six components:

- **Information**,
- **<Command>**,
- **Command**,
- **Selection**,
- **MultiSelection**
- **Text acquisition**,
- **Switch**.

2.2.1 Information.

Used to display an information, as shown in Fig 2.1. Information is not framed.

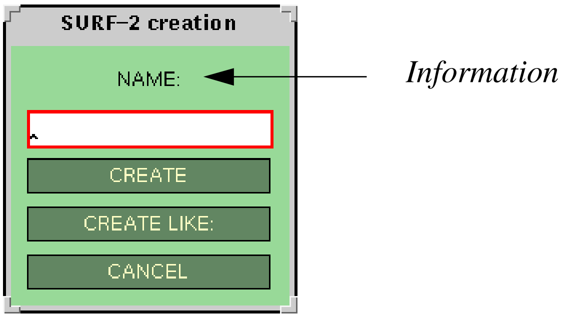


Fig 2.1 Dialog box Model creation.

2.2.2 <Command>.

When the pointer enters the **<Command>** area the corresponding menu is displayed as shown in Fig 2.2. **<Command>** frame is white.

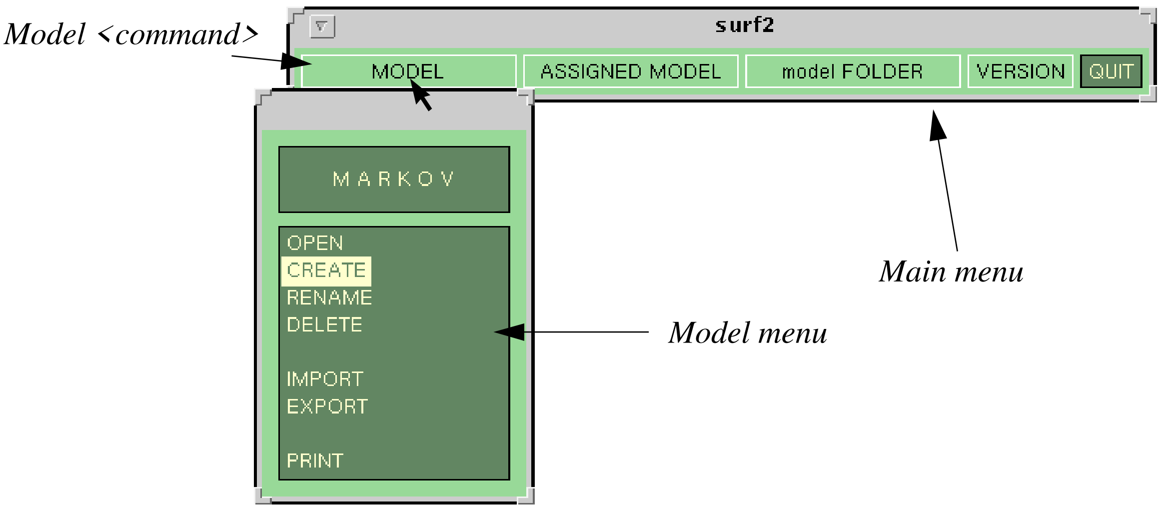


Fig 2.2 **Model <Command>** displays Model menu

<Commands> are just used in the main menu.

Each **<Command>** is identified by its name in the dialog box and the name of the dialog box. See below the description of **MODEL <Command>**.

The symbol "->" points at the action processed by SURF 2.

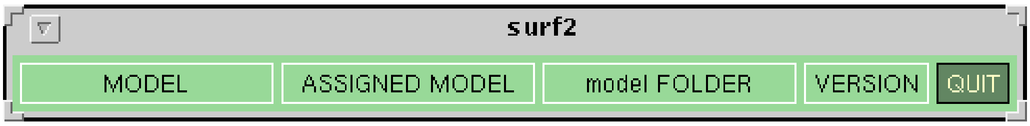


Fig 2.3 Main menu.

MODEL <Command>

- Main menu. (Fig 2.3)

-> Displays Model menu.

2.2.3 Command.

When the user clicks a command the corresponding action is activated. Fig 2.4 gives an example of dialog box with commands. **Command** frame is black.

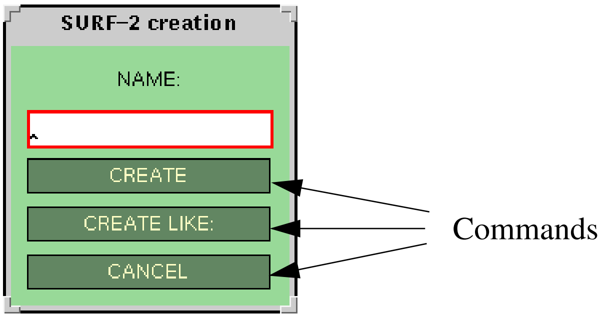


Fig 2.4 Dialog box Model creation.

Each **command** is identified by its name in the dialog box and the name of the dialog box. For example see below the description of command **CREATE**.

CREATE Command

- Dialog box Model creation. (Fig 2.4)

-> Creates a new model and displays Dialog box of Model Editor.

2.2.4 Selection.

A **Selection** allows to choice an element in the displayed list. The user click the chosen element, then the dialog box disappears.

Fig 2.5 gives an example of **Selection** in a dialog box. The scroll bar can be used to display the invisible part of the list inside the window.

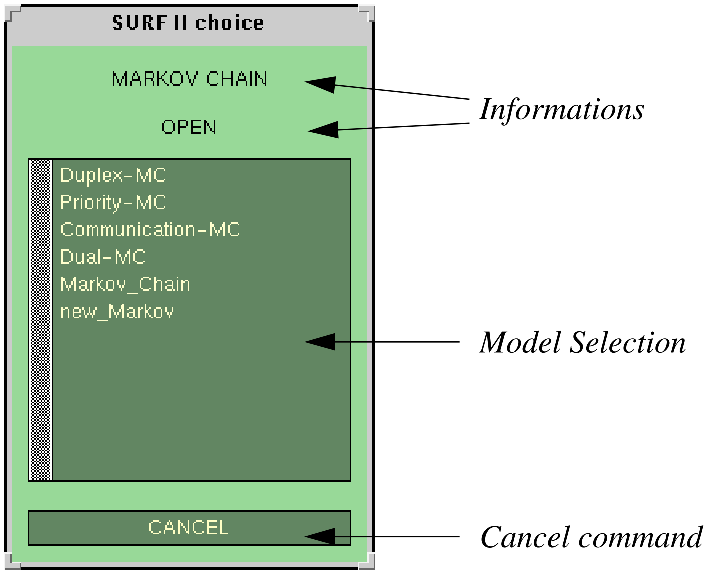


Fig 2.5 Dialog box to select the model to be opened

Each **Selection** is identified by the name of the dialog box and a label. See below the description of selection labeled: **Model Selection**. A short text explains the user choice and the action carried out by SURF 2.

Selection description example:

Model Selection

- *Dialog box to select the model to be opened (Fig 2.5)*

Choose a Markov chain (or a Petri net) in the list of available models.

-> The EDITOR window appears on the screen.

2.2.5 MultiSelection.

This dialog box component is just used in the tool: **Results Visualization** (Chapter 15).

A **MultiSelection** allows to choice a set of elements in the displayed list. When the user clicks an element its state (selected or unselected) is reversed. Selected element is displayed on a black background. Unselected element is displayed on a white background.

Fig 2.6 gives an example of **MultiSelection** in a dialog box. The scroll bar can be used to display the invisible part of the list inside the window.

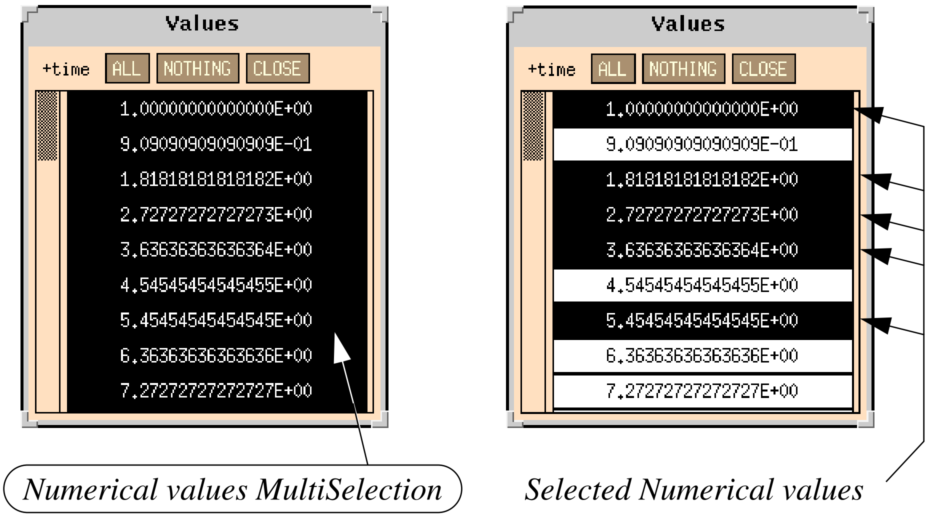


Fig 2.6 Dialog box to select numerical values of global values.

Each **MultiSelection** is identified by the name of the dialog box and a label. See below the description of Multiselection labeled: **Numerical values MultiSelection**. A short text explains the user choice.

Numerical values MultiSelection

- *Dialog box to select numerical values of global values. (Fig 2.6)*

Choose Numerical Values for the selected Global Value.

2.2.6 Text acquisition.

To enter a character string move the pointer in the enter area, the pointer shape becomes: I, and enter the character string with the keyboard. Fig 2.7 gives an example of **Text acquisition** in a dialog box. **Text acquisition** background is white.

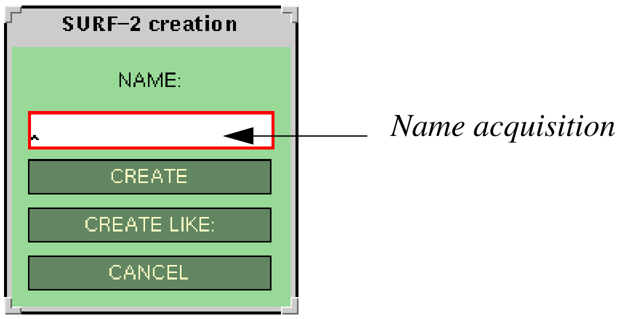


Fig 2.7 Dialog box Model creation

Each **Text acquisition** is identified by the name of the dialog box and a label. See below the description of **Name acquisition**, where text semantic is explained. SURF 2 does nothing else than text acquisition.

Name acquisition

- *Dialog box Model creation (Fig 2.7)*

Enter the name of the new model.

Text acquisition can be used to enter multilines text, it is fairly similar to the EMACS editor. The most common commands are listed below:

- Ctrl-a Beginning of the line
- Ctrl-b Previous character
- Ctrl-d Delete next character
- Ctrl-e End of the line
- Ctrl-f Next character
- Ctrl-h Delete previous character
- Ctrl-k Delete all characters up to the end of the line.

2.2.7 Switch.

Two texts are associated with a **switch**. Initially one of these texts is displayed, when the user clicks a text the other text takes its place, and so on. Fig 2.8 gives an example of dialog box with **switch**.

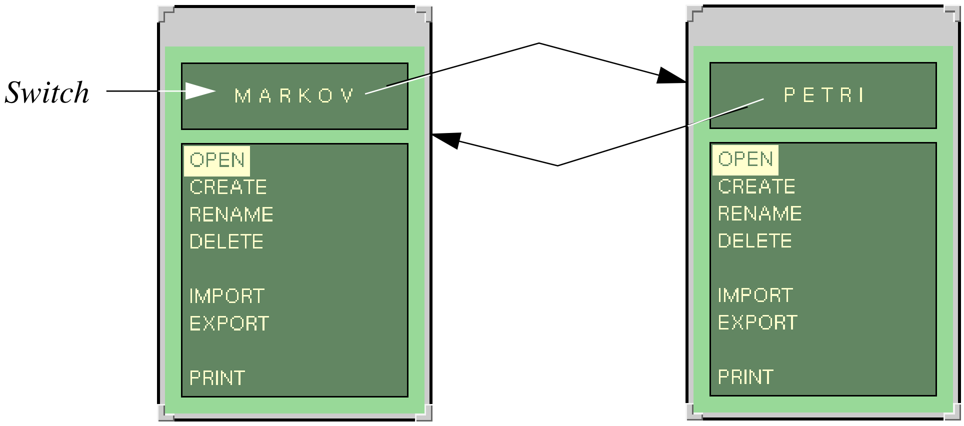


Fig 2.8 **Switch MARKOV / PETRI** in Dialog box **Model menu**

Each **switch** is identified by the two texts separated by a "/ " and the name of the dialog box. See below the description of **switch MARKOV / PETRI**:

Fig 2.9 **Model menu**

<u>MARKOV / PETRI</u> Switch

- *Model menu (Fig 2.9)*

Switch between MARKOV or PETRI modelization, the initial value is MARKOV.

2.3 CANCEL COMMAND

Generally in a dialog box there is a CANCEL COMMAND. When clicked, the current action is cancelled and SURF-2 comes back in the previous state:

CANCEL Command

- *Dialog Box* (where CANCEL appear)

-> The corresponding dialog box disappears.

Chapter 3

MAIN MENU

Color: Green

3.1 PURPOSE

MAIN MENU allows to display:

- MODEL MENU
- ASSIGNED MODEL MENU
- MODEL FOLDER MENU

These three menus give access to SURF-2 tools and permit to manage SURF-2 Data Bases.

Furthermore MAIN MENU allows to display the version number and to exit SURF-2 session.

3.2 ACCESS.

When a new SURF-2 session starts, it displays the Main menu as shown in Fig 3.1:

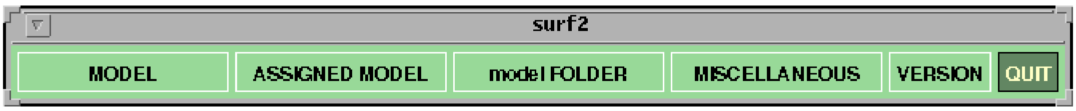


Fig 3.1 Main menu.

3.3 DESCRIPTION.

<u>MODEL</u> <Command>

- *Main menu. (Fig 3.1)*

-> Displays Model menu, see Fig 3.2.

Fig 3.2 Model menu

Refer chapter 4 for a detailed description of the Model menu.

<u>ASSIGNED MODEL</u> <Command>

- *Main menu. (Fig 3.1)*

-> Displays Assigned Model menu, see Fig 3.3.

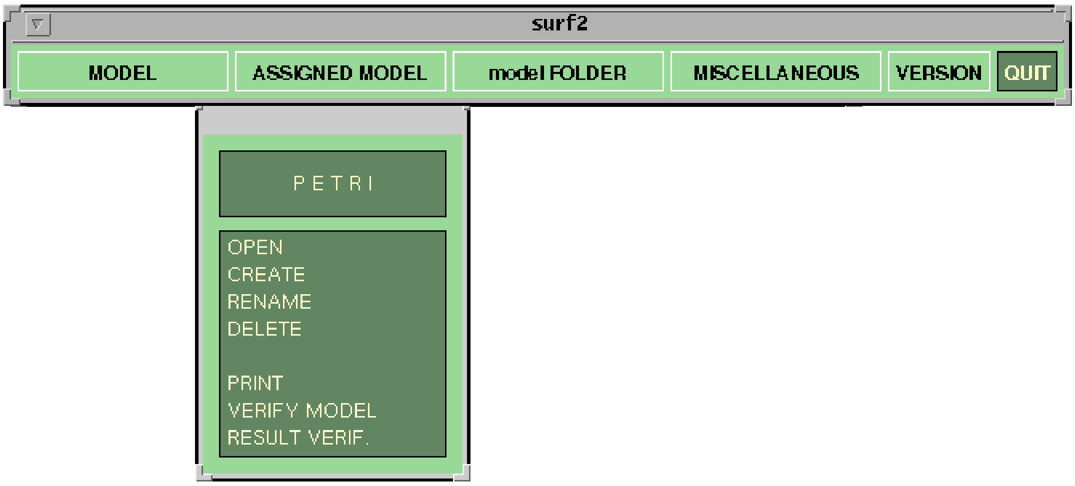


Fig 3.3 Assigned Model menu

Refer chapter 7 for a detailed description of the Assigned Model menu.

<u>MODEL FOLDER</u> <Command>

- *Main menu. (Fig 3.1)*

-> Displays Assigned Model menu, see Fig 3.4.

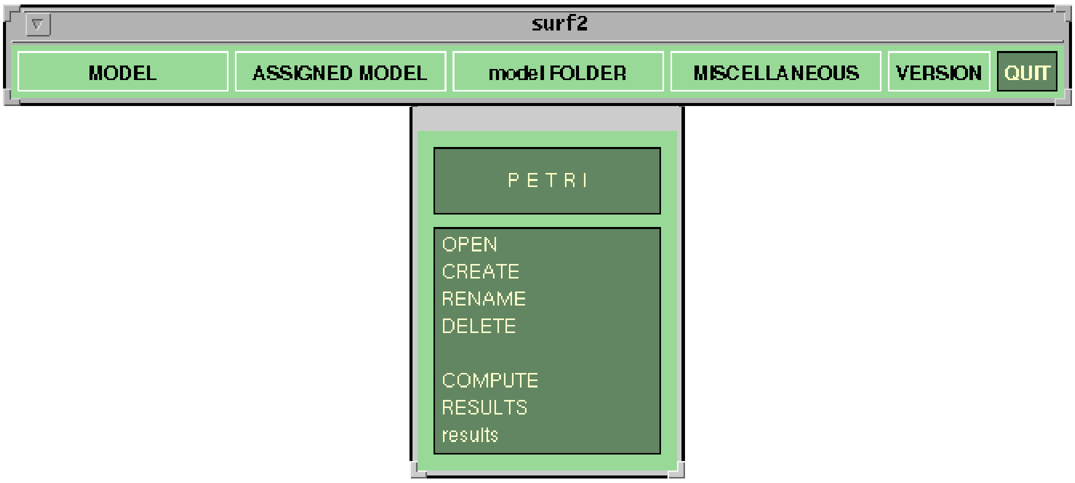


Fig 3.4 Model Folder menu

Refer chapter 11 for a detailed description of the Model FOLDER menu.

MISCELLANEOUS <Command>

- *Main menu. (Fig 3.1)*

-> Displays Miscellaneous menu, see Fig 3.4.

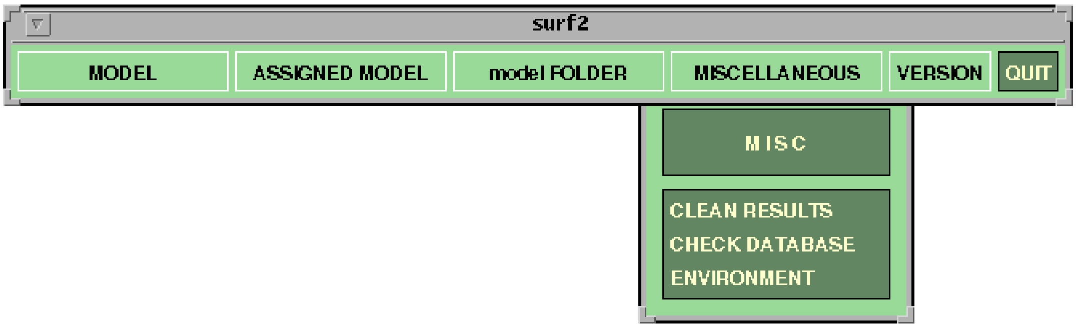


Fig 3.5 Miscellaneous menu

Refer chapter 13 for a detailed description of the MISCELLANEOUS menu.

VERSION <Command>

- *Main menu. (Fig 3.1)*

-> Displays the following message:

By clicking this window the message vanishes from the screen.

QUIT Command

- Main menu. (Fig 3.1)

->Terminates a session after a confirmation.

Report:

Message	**Error: can't quit - some actions are still in progress**

You must stop these actions[1] before exiting.

[1]. Since SURF-2 is a multi-tasking toolbox, several processes can be executed simultaneously. Thus, to avoid conflicts between competiting processes sharing common data, consistency controls and conflicting situations analyses are continuously performed by a supervisory task.

Chapter 4

MODEL MENU

Color: Green

4.1 PURPOSE

MODEL MENU allows to manage Models in the SURF2 data base and gives access to Models Editors: MARKOV Chain Editor and PETRI Net Editor.

4.2 ACCESS

MODEL *Command* of MAIN MENU provides access to MODEL MENU.

Fig 4.1 shows both aspects of model menu for MARKOV model or PETRI model.

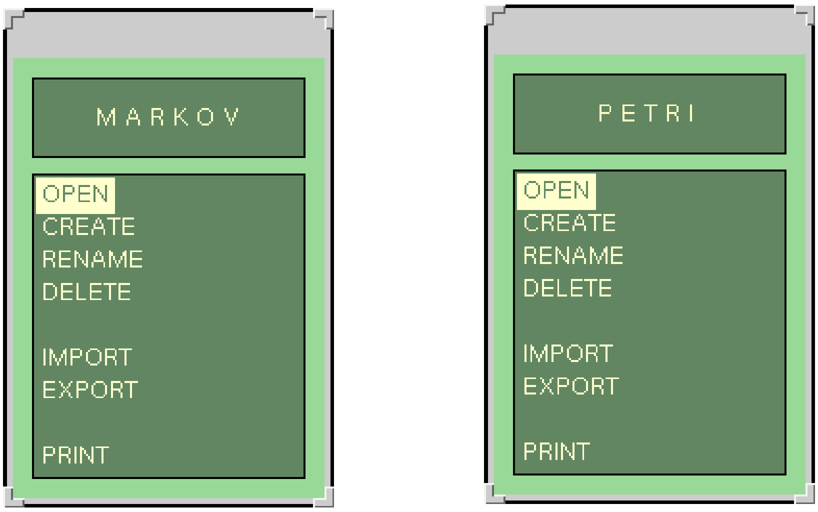


Fig 4.1 **Model menu**.

4.3 DESCRIPTION

<u>MARKOV / PETRI</u> Switch

- *Model menu. (Fig 4.1)*

-> Switches between MARKOV or PETRI modelization, the initial value is MARKOV.

<u>OPEN</u> Command

- *Model menu. (Fig 4.1)*

-> Displays Dialog box to select the model to be opened: Fig 4.2. The model type depends on the switch state: MARKOV or PETRI.

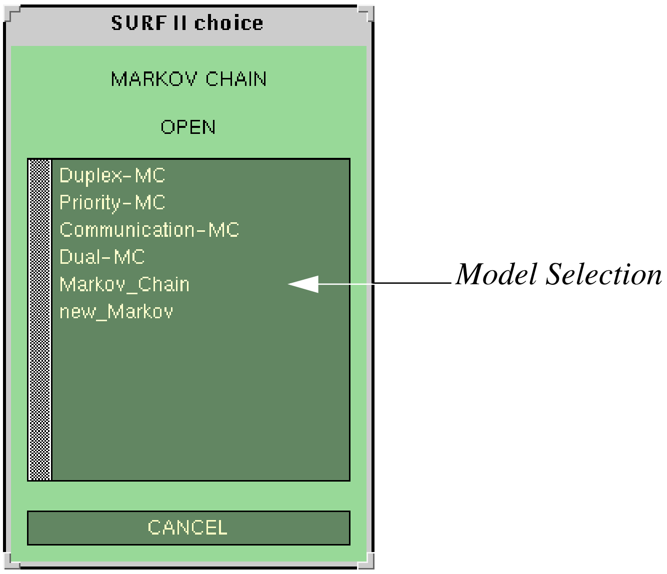


Fig 4.2 Dialog box to select the model to be opened.

Model Selection

- Dialog box to select the model to be opened. (Fig 4.2)

Choose a Markov chain (or a Petri net) in the list of available models.

-> The EDITOR window appears on the screen: Fig 4.3.

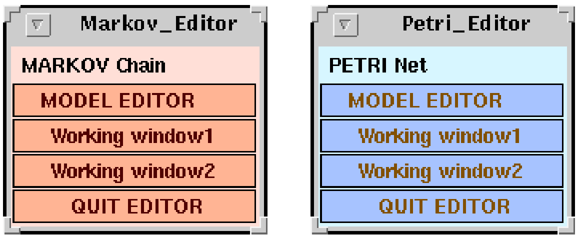


Fig 4.3 Dialog box of Model Editors.

See chapter 5 to use Markov Editor and chapter 6 to use Petri Editor.

CREATE Command

- Model menu. (Fig 4.1)

-> Displays Dialog box to Model creation: Fig 4.4.

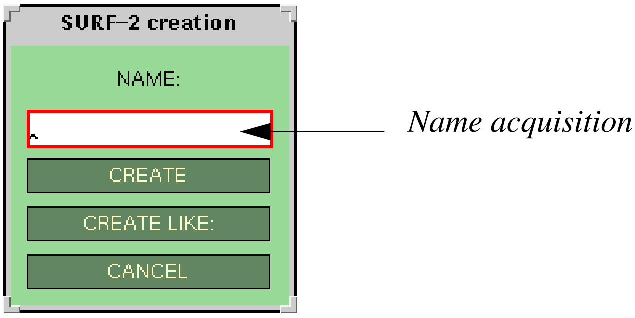


Fig 4.4 Dialog box to Model creation.

Name Acquisition

- *Dialog box to Model creation. (Fig 4.4)*

Enter the name of the new model.

CREATE Command

- *Dialog box to Model creation. (Fig 4.4)*

-> Creates a new model and displays Dialog box of Model Editors. (Fig 4.3).

CREATE LIKE Command

- *Dialog box to Model creation. (Fig 4.4)*

-> Displays dialog box to select the model to be duplicated: Fig 4.5.

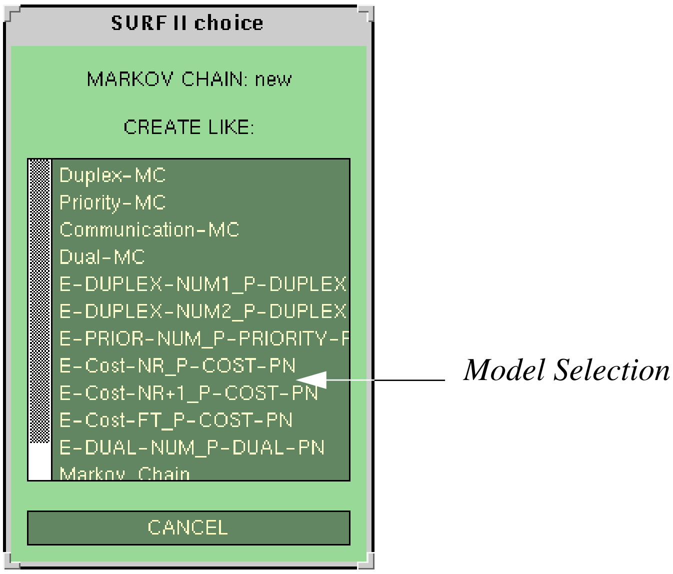


Fig 4.5 Dialog box to Select the model to be duplicated.

Model Selection

- *Dialog box to Select the model to be duplicated. (Fig 4.5)*

Choose the name of the model that will be replicated in the list of existing models.

-> Displays dialog box of Model Editor: Fig 4.3.

The new model thus created will be a true-copy of the model previously chosen by the user from which he will make all the modifications necessary to build up the new model.

> Markov graphs derived from Petri nets are directly reachable with this function.

Report (CREATE, CREATE LIKE).

Message |	**GIVE THE NAME TO THE NEW MODEL**
Message |	**Error: name already exists**
Message |	**NAMES WITH A'/' ARE NOT ALLOWED**

RENAME Command

- Model menu. (Fig 4.1)

-> Displays dialog box to select the model to be renamed: Fig 4.6.

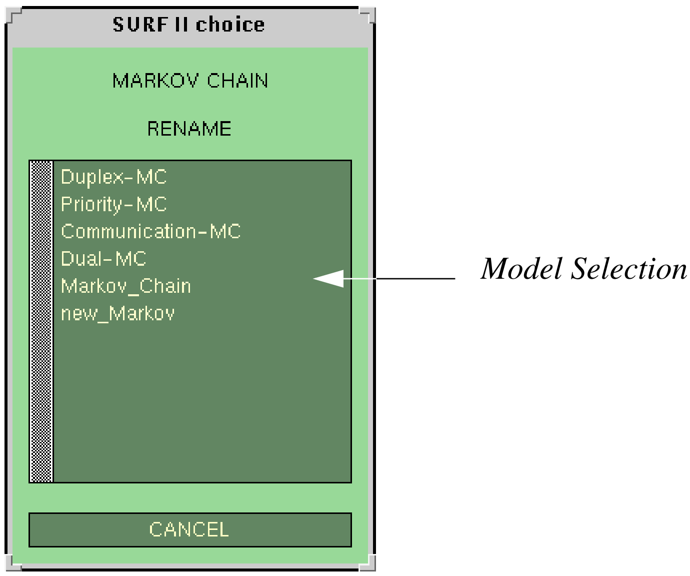


Fig 4.6 Dialog box to select the model to be renamed.

Model Selection

- Dialog box to select the model to be renamed. (Fig 4.6)

Choose a Markov chain (or a Petri net) in the list of available models.

-> Displays dialog box to rename: Fig 4.7.

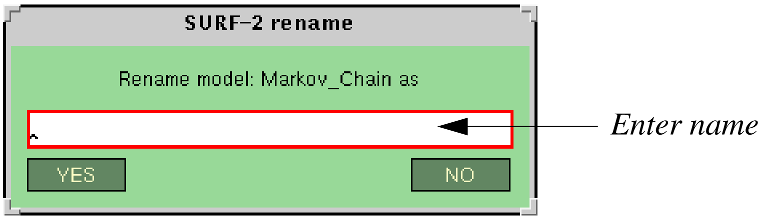


Fig 4.7 Dialog box to Rename.

Name Acquisition

- *Dialog box to Rename. (Fig 4.7)*

Enter the new name of the model.

YES Command

- *Dialog box to Rename. (Fig 4.7)*

-> Renames the model.

NO Command

- *Dialog box to Rename. (Fig 4.7)*

-> Dialog box to rename disappears. The model is not renamed.

DELETE Command

- *Dialog box to select the model to be deleted. (Fig 4.8)*

-> Displays dialog box to select the model to be deleted: Fig 4.8.

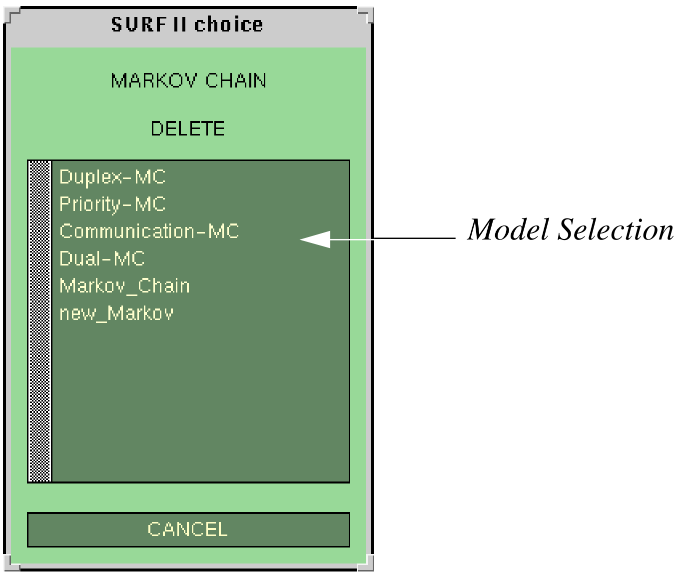


Fig 4.8 Dialog box to select the model to be deleted.

Model Selection

- *Dialog box to select the model to be deleted. (Fig 4.8)*

Select the name of model to be deleted.

-> Displays dialog box to confirm model deletion: Fig 4.9.

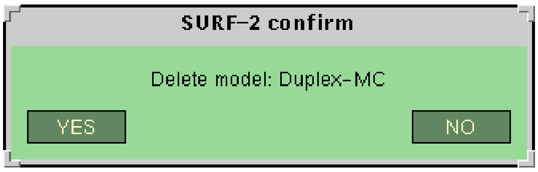


Fig 4.9 Dialog box to confirm model deletion

YES Command

- *Dialog box to confirm model deletion (Fig 4.9)*

-> Deletes model. Dialog box to confirm model deletion disappears.

Report

After model deletion the following message is displayed:

Message	**DELETING model:. . . TERMINATED**

NO Command

- *Dialog box to confirm model deletion (Fig 4.9)*

-> Dialog box to confirm model deletion disappears. The model is not deleted.

EXPORT Command

- *Model menu. (Fig 4.1)*

-

> Before using this command the user must create a directory. The full path name of the directory is defined by the environment variable SURF2_EXPORT. Model is exported in this directory.
>
> - *A Markov model consists of two files: Graph and Part and a Petri model consists of three files: Invariant, Net and Part.*
> - *A new export overwrites the previous export.*
> - *ASSIGNED MODEL and MODEL FOLDER cannot be exported or imported.*

-> Displays dialog box to select the model to be exported: Fig 4.10.

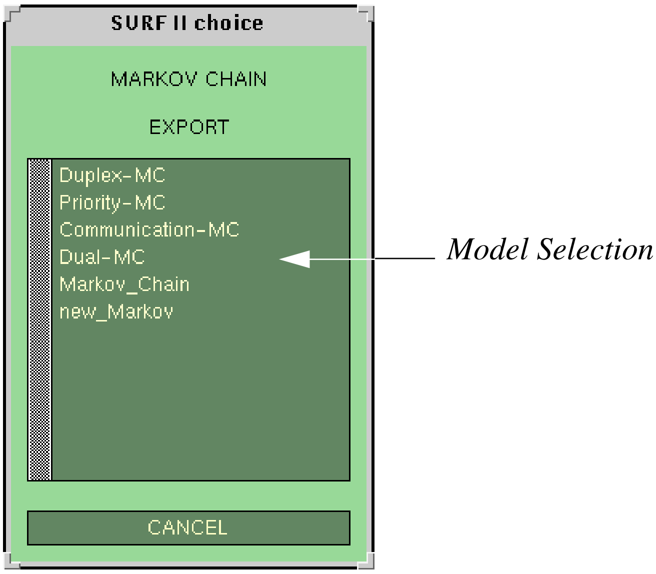


Fig 4.10 Dialog box to select the model to be exported

Model Selection

- Dialog box to select the model to be exported (Fig 4.10)

Select the name of the model to be exported.

-> Displays dialog box to confirm model exportation: Fig 4.11.

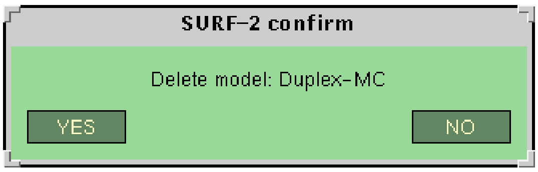


Fig 4.11 Dialog box to confirm model exportation.

YES Command

- Dialog box to confirm model exportation. (Fig 4.11)

-> Exports the model in the directory specified by the environment variable SURF2_EXPORT. Dialog box to confirm model exportation disappears.

Report

- After model exportation the following message is displayed:

Message |	**EXPORTING model: . . . in directory: . . . TERMINATED**

- if SURF2_EXPORT is not well defined:

Message |	**Environment SURF2_PORT can't be found**

NO Command

- Dialog box to confirm model exportation. (Fig 4.11)

-> Dialog box to confirm model exportation disappears. The model is not exported.

IMPORT Command

- Model menu. (Fig 4.1)

> Before using this command the user must export a Markov model (Graph and Part file) or a Petri model (Invariant, Net and Part files) in the directory defined by the environment variable SURF2_EXPORT.

-> Displays dialog box to imported model: Fig 4.12.

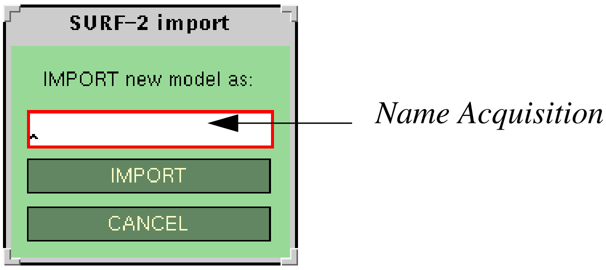


Fig 4.12 Dialog box Import

Name Acquisition

- Dialog box Import (Fig 4.12)

Enter the name of the new model that will be created by import from the directory specified by the environment variable SURF2_EXPORT.

IMPORT Command

- Dialog box Import (Fig 4.12)

-> Imports the model in the SURF 2 data base.

Report

- If importation is correctly done and model valid:

Message	**IMPORTING TERMINATED. Model: model_name >> VALID <<**

- If the model imported is invalid:

Message	**IMPORTING TERMINATED. Model: ... , Error in ...**

See error file in the directory defined by the environment variable SURF2_EXPORT.

- if SURF2_EXPORT is not well defined:

Message	**IMPORTATION FAIL. can't read the environment variable <SURF2_PORT>**

Message	**Environment SURF2_PORT cannot be found**

PRINT Command

- Model menu. (Fig 4.1)

The environment variable SURF2_IMP specifies the printer to use. If "SURF2_IMP" is not defined a PostScript file is generated in the home directory. Its name is automatically generated as follows: "MARKOV_name_of_model.ps"

-> Display Dialog box to select the model to be printed: Fig 4.13.

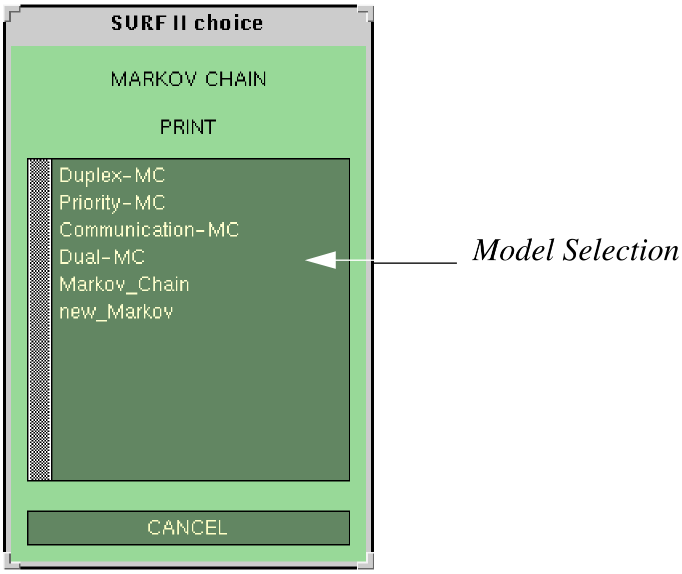


Fig 4.13 Dialog box to Select the model to be printed out.

Model Selection

- Dialog box to Select the model to be printed out. (Fig 4.13)

Select the name of the model to be printed out.

-> Displays dialog box to print. This dialog box may be different according to the type of model selected.

a) Markov chain

Fig 4.14 shows the printout option menu for Markov chains:

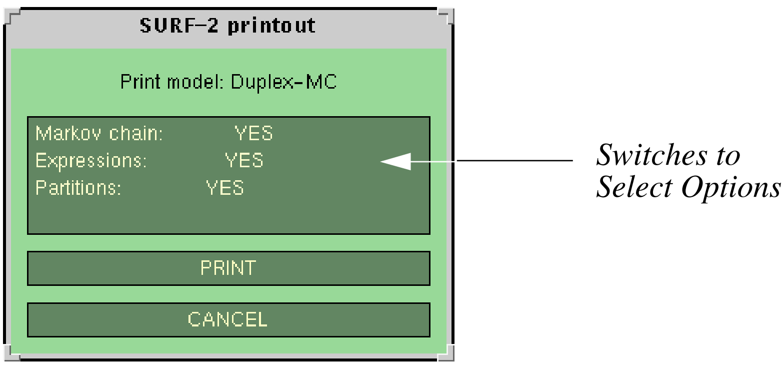


Fig 4.14 Dialog box to print Markov chains.

Each option can be either selected (YES) or unselected (NO). If the initial selections are not suitable, the user can change them.

(Markov chain) YES / NO Switch

- Dialog box to print Markov chains. (Fig 4.14)

Choose YES if you want to print Markov Chain.

(Expressions) YES / NO Switch

- Dialog box to print Markov chains. (Fig 4.14)

Choose YES if you want to print Expressions.

(Partitions) YES / NO Switch

- Dialog box to print Markov chains. (Fig 4.14)

Choose YES if you want to print Partitions.

PRINT

- Dialog box to print Markov chains. (Fig 4.14)

-> Prints model.

Report

Message	**PRINTOUT OF MODEL ... TERMINATED**

b) Petri net

Fig 4.15 shows the printout option menu for Petri nets:

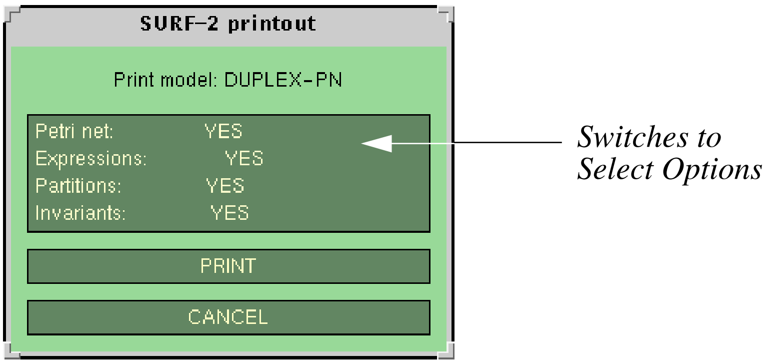


Fig 4.15 Dialog box to print Petri nets.

Each option can be either selected (YES) or unselected (NO). If the initial selections are not suitable, the user can change them.

(Petri Net) YES / NO Switch

- Dialog box to print Petri nets. (Fig 4.15)

Choose YES if you want to print Petri Net.

(Expressions) YES / NO Switch

- Dialog box to print Petri nets. (Fig 4.15)

Choose YES if you want to print Expressions.

(Partitions) YES / NO Switch

- *Dialog box to print Petri nets. (Fig 4.15)*

Choose YES if you want to print Partitions.

(Invariants) YES / NO Switch

- *Dialog box to print Petri nets. (Fig 4.15)*

Choose YES if you want to print Invariants.

PRINT

- *Dialog box to print Petri nets. (Fig 4.15)*

Prints the model.

Report

Message	PRINTOUT OF MODEL ... TERMINATED

Note

- Initial markings, rates, probabilities and weights of edges will be printed on the graph itself.
- Comments will be printed out in a separate page with the name of the associated object in front of each one. An example of Petri net printout is given below:
  - list of places:
    <name> <comment> <initial marking>
  - list of immediate transitions:
    <name> <comment> <probability>
  - list of timed transitions:
    <name> <comment> <rate>
  - list of edges:
    <origin edge> <destination edge> <weight>

Chapter 5

MARKOV CHAIN EDITOR

Color: Salmon

5.1 PURPOSE

Markov Chain Editor allows to define a model composed of:

- a markov chain
- a list of partitions

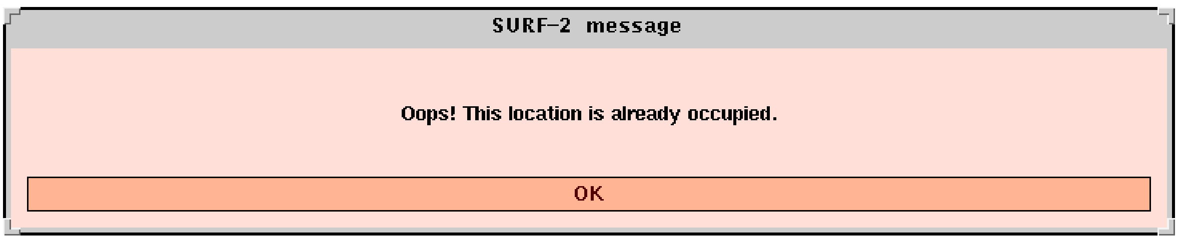
5.2 ACCESS

To access the Markov chain editor, first select one of the following two commands in the MODEL MENU:

- OPEN Command in the case of an existing Markov chain, see Chapter 4 page 1
- CREATE Command if the user wants to create a new Markov chain, see Chapter 4 page 2.

5.3 MESSAGES

Editor message are displayed in a dialog box:



When the user is notified of the message, he must click OK to go on.

5.4 MARKOV EDITOR DIALOG BOX

OPEN or CREATE commands of MODEL MENU gives access to the window shown above:

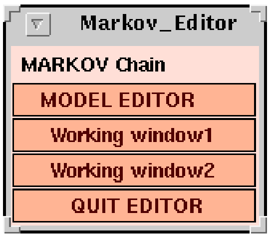


Fig 5.1 Markov Editor dialog box.

MODEL EDITOR Command

- *Markov Editor dialog box. (Fig 5.1)*

-> Displays the Markov chain editing window, see Fig 5.2.

Working window1 Command

Working window2 Command

- *Markov Editor dialog box. (Fig 5.1)*

-> Opens working window (refer page 21 for a detailed description).

QUIT EDITOR Command

- *Markov Editor dialog box. (Fig 5.1)*

-> (if nothing is modified since last save) quit the Markov Chain EDITOR.

-> (if the model has been modified but not saved) one of the two following warnings is displayed:

Message | WARNING: graphics modifications! confirm CLOSE by re-clicking.

this message indicates that graphics modifications have been introduced.

Message | WARNING: model semantics modified! confirm CLOSE by re - clicking.

this message indicates that semantics modifications have been introduced.

Save your work or Re-click to quit without save.

5.5 INTRODUCTION TO MODEL EDITING WINDOW

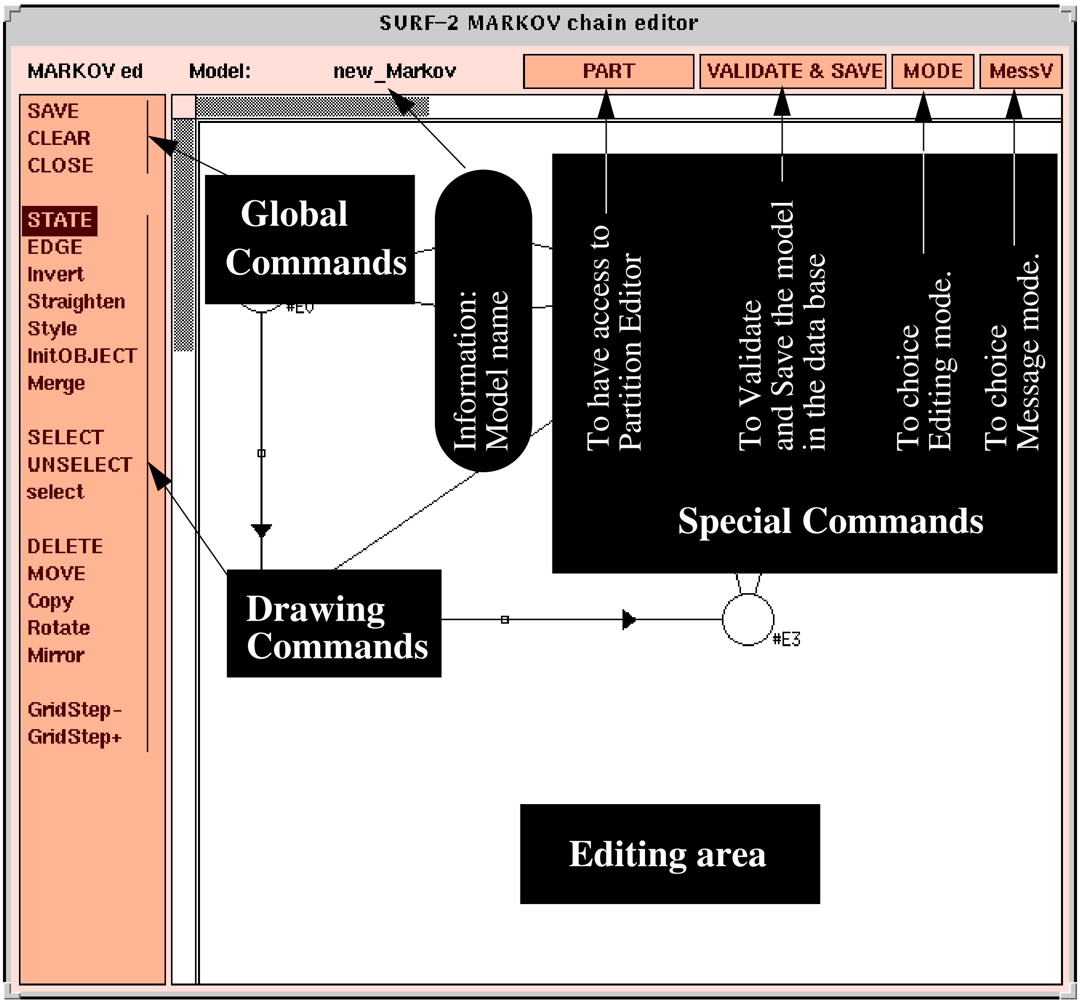


Fig 5.2 Markov chain editing window.

An overview of Markov chain editing window is shown on Fig 5.2.

The editing window is made of four areas:

- The Editing area where the Markov chain can be displayed and modified,
- A Global Command menu to manage the model (page 3),
- A Drawing Command menu, for drawing the chain (page 4),
- and few others Commands (page 16)
  - commands to access partitions and invariants editors.
  - convenient commands

5.6 GLOBAL COMMANDS

SAVE Command

- *Markov chain editing window. (Fig 5.2)*

-> Saves the model drawing on disk without any validation.

CLEAR Command

- Markov chain editing window. (Fig 5.2)

-> (At first click) the following message is displayed:

Message | Confirm CLEAR by re - clicking

-> (After confirmation by re-clicking) Clears the editing window.

CLOSE Command

- Markov chain editing window. (Fig 5.2)

-> Closes the editing window. The model is first temporarily saved on disk.

Report

When clicking this function without having saved the graphics modifications, the following warning is displayed:

Message | WARNING: graphics modifications! confirm CLOSE by re - clicking.

When clicking this function without having saved the semantics modifications, the following warning is displayed:

Message | WARNING: model semantics modified! confirm CLOSE by re - clicking.

5.7 DRAWING COMMANDS

5.7.1 Markov Chain representation

A markov Chain is composed of **states** and **edges**: Fig 5.4.

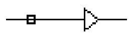
Objects		Attributes
STATE		Name Initial probability Comment
Edge		Rate Comment

Fig 5.3 Markov Chain components.

A Markov Chain is displayed as show Fig 5.4:

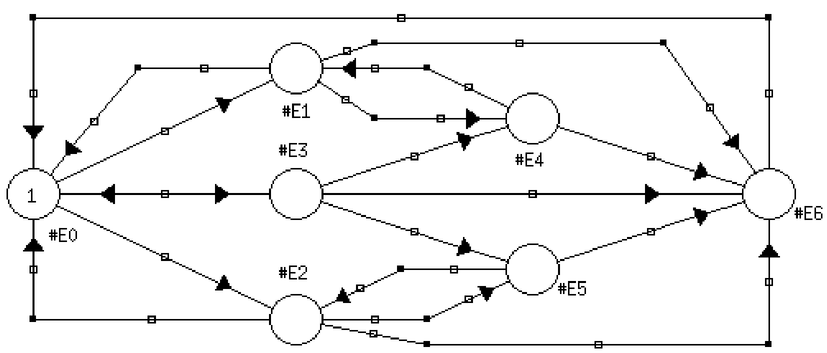


Fig 5.4 A Markov Chain.

An edge is composed of **arrow**, **vertex** and **handgrip** show Fig 5.5:

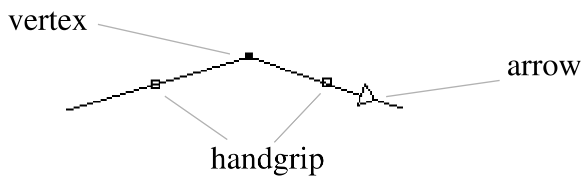


Fig 5.5 Edge components.

> ***Pending edges are not allowed:***
> - *Origin and destination state must be created before corresponding edge.*
> - *When a state is deleted all edges associated with this state are deleted.*

Selection:

States and edges may be grouped in a current **selection** that can be manipulated as a whole. Only one selection can be done at a time.

States are explicitly selected. An edge is selected if its origin state and its destination state are selected. Selected states are shaded. The selection area (Fig 5.6) is defined by a rectangular box.

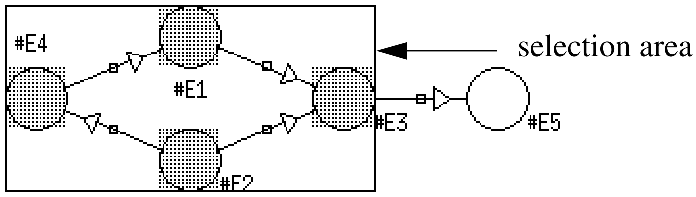


Fig 5.6 The current selection.

5.7.2 Description of basic actions for Drawing Functions

Some editor functions need complementary information(s) to be done, for example **state location** for a **state creation**.

These informations are defined by means of few basic **actions** (the pointer shape indicate the action expected by the editor). Use the left button of the mouse.

Locate

Pointer shape: +

Used to enter a location.

Move the cursor at the wished position and click.

Pick

Pointer shape:

Used to select a single object or selection.

- If you want to pick a state move the pointer near the middle of this state and click.
- If you want to pick an edge move the pointer on one of these edge element: (handgrip,

arrow, vertex see Fig 5.5) and click.

- If you want to pick the selection move the pointer inside the selection area and click.

Pick down

Pointer shape:

Allows to select an object when the function needs more information than this simple object selection

Move the pointer near the middle of the object to select (state, edge handgrip, edge arrow or edge vertex) then press button. The button must be pressed while moving.

Pick up

Pointer shape:

Allows to select a second object after a first selection by a Pick down.

Used after a **Pick down** therefore the button is pressed. Move the pointer near the middle of the object to select (state, edge handgrip, edge arrow or edge vertex) then release the button.

A rubber band is drawn from pick down location to current pointer position.

Locate up

Pointer shape:

Allows to enter a new location for an object previously selected by a Pick down.

Used after un **Pick down** therefore the button is pressed. Move the pointer to the wished free position and then release the button.

Define area

Pointer shape:

Allows to define an area.

Move the cursor at the top left vertex of the area and press button, the pointer takes the shape. Hold the button down and move the pointer to bottom right vertex of the area, then release the button.

Adjust copy

Pointer shape with copy outline:

Allows to locate the selection copy.

When the button is pressed the outline of the selection copy is displayed. Hold down the button and choose the copy location, then release the button.

5.7.3 Reports

Pick, Pick down, Pick up:

When an object has not been correctly designated, the following message is displayed:

Message	**Designate the object to be "***command"***.**

Where "*command"* refers to the current command.

Locate, Locate up:

If the user pick an object when a free location is expected the following message is displayed:

Message	**Oops! This location is already occupied**

5.7.4 Drawing Commands definition

For each drawing command we define:

- The name of the command.
- The dialog box where the command appears.
- The list of basic actions needed by the command.
- -> The command effect.

> *The Command must be selected before designating the object to which it has to be applied. The command remains active as long as a new one has not been selected.*
>
> *By example, after a STATE Command, use repeat locate action to create a set of state.*

STATE Command

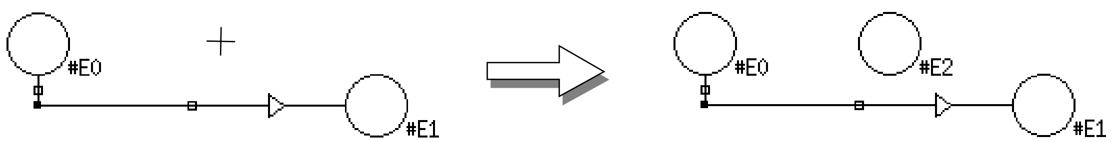
- *Markov chain editing window. (Fig 5.2)*

Locate Position of the new state

-> State creation.

A state is graphically depicted by a circle and a name is automatically assigned to each state as follows: #En (where # is a reserved character used for automatically labelling a state; 'n' being the state number in the graph).

0 is assigned to initial probability of the state.



EDGE Command

- *Markov chain editing window. (Fig 5.2)*

pick down the origin of the new edge.

pick up the extremity of the new edge.

-> Creation of an oriented edge graphically depicted by a segment with a handgrip and an arrow.

Rate associated to the edge is not defined.

Report

If you do not click center of the state, following messages are displayed:

- if the origin node has not been correctly designated:

Message	**Designate the origin of edge.**

- if the destination node has not been correctly designated:

Message	**Oops! Give the origin and the destination of the edge again.**

- if the origin node and the destination node which have been designated merge into one:

Message	**Oops! The origin and the destination overlap Give the origin again**

- if you try to draw an existing edge:

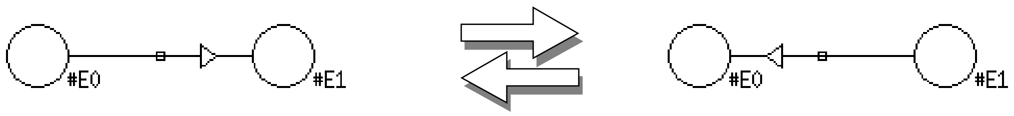
Message	**Oops! This edge already exist.**

Invert Command

- *Markov chain editing window. (Fig 5.2)*

Pick Edge to invert.

-> invert the direction of the selected edge.

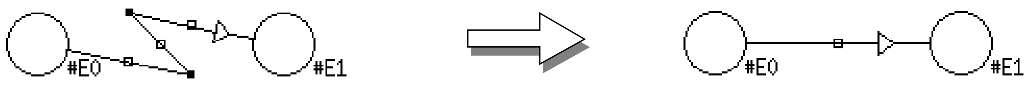


Straighten Command

- *Markov chain editing window. (Fig 5.2)*

Pick Edge to straighten.

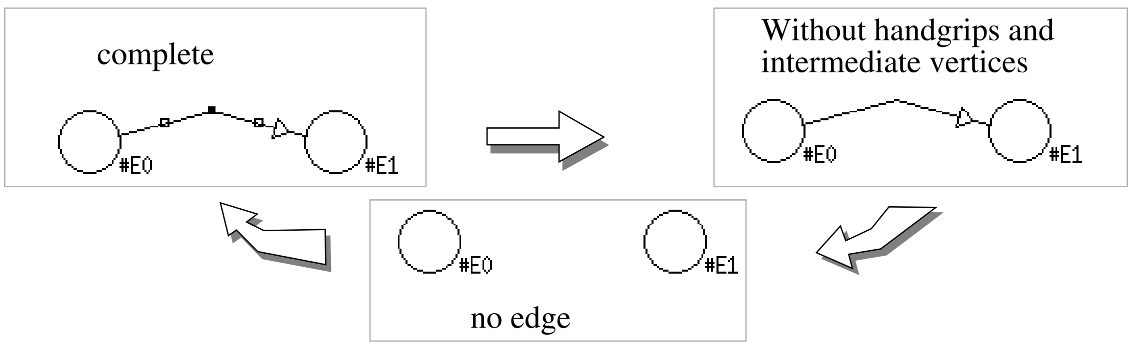
-> straighten selected edge.



Style Command

- *Markov chain editing window. (Fig 5.2)*

-> change edges representation in the graph:



<u>InitOBJECT</u> Command (for a state)

- Markov chain editing window. (Fig 5.2)

<u>**Pick**</u> State to initialize.

-> Displays dialog box for state attributes (Fig 5.7).

The selected state is marked with a cross.

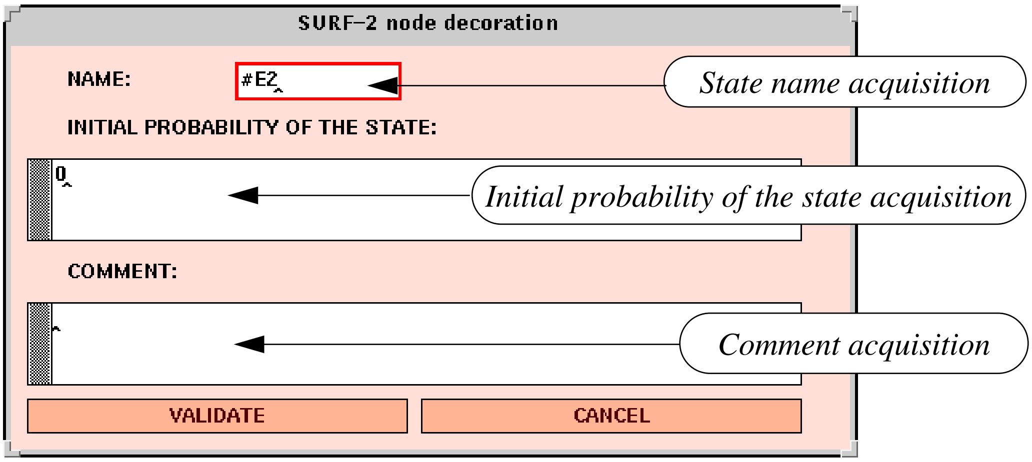


Fig 5.7 Dialog box for state attributes.

<u>State name</u> Acquisition

- Dialog box for state attributes. (Fig 5.7)

Enter a new name if you wish to replace the one that has been automatically assigned. First erase the old name using <BACKSPACE> on your keyboard.

Report

The user cannot give a name beginning with #. A warning message indicates this error:

Message	**Names preceded by # are reserved system words.**

<u>Initial probability of the state</u> Acquisition

- Dialog box for state attributes. (Fig 5.7)

Enter the initial probability of the state.

The syntax is defined in Expressions defining objects page 3 of appendix 1.

Syntax control is done by **VALIDATE&SAVE command**, see page 18.

<u>Comment</u> Acquisition

- Dialog box for state attributes. (Fig 5.7)

A comment can be added. A vertical scroll bar can be used to display the portion of text that is not visible inside the text window.

<u>VALIDATE</u> Command

- Dialog box for state attributes. (Fig 5.7)

-> Updates state without syntax control, see Fig 5.8.

The Dialog box for state attributes disappears.

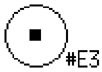
State representation	initial probability
	0.0
	short expression. The value is displayed
	long expression. The value is not displayed use InitObject to read value

Fig 5.8 Visualization of initial probability

<u>CANCEL</u> Command

- *Dialog box for state attributes. (Fig 5.7)*

-> The Dialog box for state attributes disappears. The state is not modified.

<u>InitOBJECT</u> Command (for an edge)

<u>Pick</u> Edge to initialize.

-> Displays dialog box for edge attributes.

The selected edge is marked with a cross.

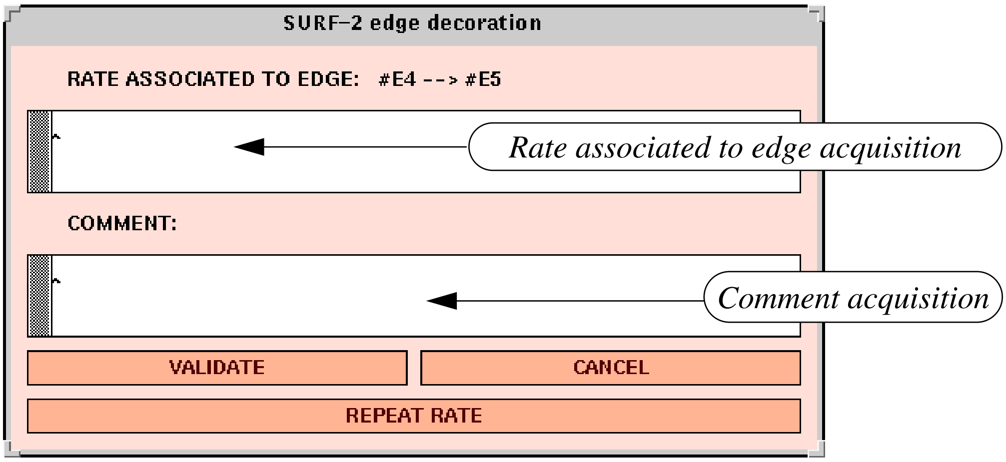


Fig 5.9 Dialog box for edge attributes.

<u>Rate associated to edge</u> Acquisition

- *Dialog box for edge attributes. (Fig 5.9)*

Enter rate associated to edge.

The syntax is defined in Expressions defining objects page 3 of appendix 1.

Syntax control is done by **VALIDATE&SAVE command**, see page 18.

<u>Comment</u> Acquisition

- *Dialog box for edge attributes. (Fig 5.9)*

A comment can be added. A vertical scroll bar can be used to display the portion of text that may not be visible inside the text window.

REPEAT RATE Command

- Dialog box for edge attributes. (Fig 5.9)

-> Assigns the same value as the one already assigned to the previous selected rate.

VALIDATE Command

- Dialog box for edge attributes. (Fig 5.9)

-> Updates edge without syntax control

The Dialog box for state attributes disappears.

A filled arrow (▶) indicates that the edge has been assigned.

CANCEL Command

-> The Dialog box for edge attributes disappears. The edge is not modified.

As long as dialog box is displayed, the user cannot use other functions of the model editor.

Merge Command

- Markov chain editing window. (Fig 5.2)

Pick down	Select a state (node that will be deleted)
Pick up	Select a state (node to be kept).

-> Merges first state on the second state, see Fig 5.10.

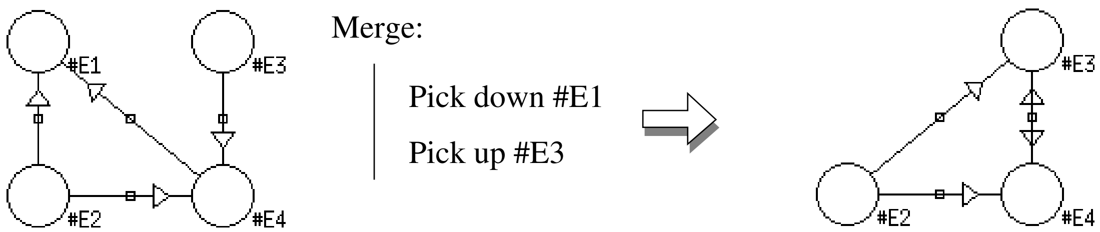


Fig 5.10 Merging of two states

Report

- if the first node has not been correctly designated:

Message	**Designate the first node.**

- if the second node has not been correctly designated:

Message	**Node to kept: Indicate again the node to delete and the node to keep.**

- in case of unauthorized merging:

Fig 5.11 (a):

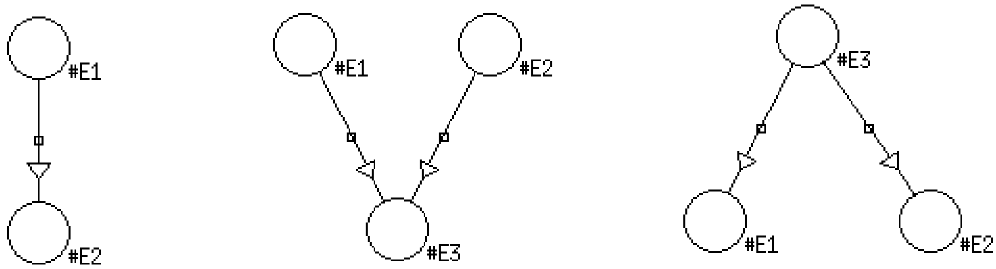
Message	**MERGING REFUSED:**
	an edge is already connected to the states.

Fig 5.11 (b):

Message	**MERGING REFUSED:**
	The merge would create two identical edges with the same origin.

Fig 5.11 (c):

Message	**MERGING REFUSED:**
	The merge would create two identical edges with the same destination.



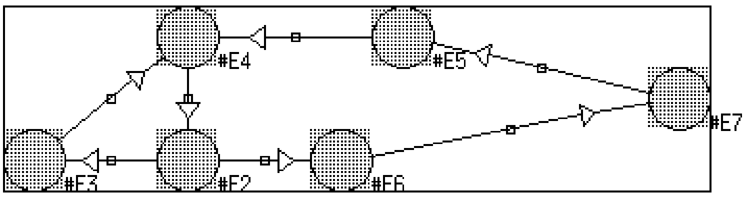
(a) (b) (c)

Fig 5.11 Three examples of unauthorized merging (#E1 with #E2).

SELECT Command

- Markov chain editing window. (Fig 5.2)

-> Selects all the objects in the editing window.



UNSELECT Command

- Markov chain editing window. (Fig 5.2)

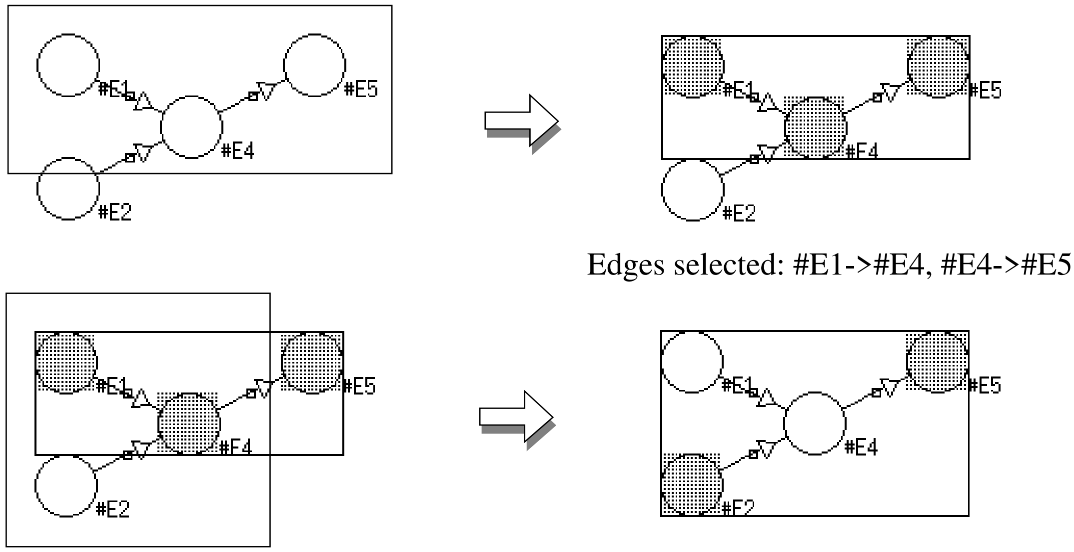
-> Unselects all the objects in the editing window.

select Command

- Markov chain editing window. (Fig 5.2)

Define area the selection area.

-> Reverses the selection attribute (select / unselect) of each state located in the selection area. Selected states are shaded, see Fig 5.12.



Edges selected: #E1->#E4, #E4->#E5

No Edge selected

Fig 5.12 Examples of states and edges selection.

DELETE Command (state or edge)

- Markov chain editing window. (Fig 5.2)

pick state or edge.

-> Deletes state or edge.

DELETE Command (Current selection)

- Markov chain editing window. (Fig 5.2)

pick selection.

-> (At first click) the following message is displayed:

Message	**Confirm the delete operation by a second click in the selection.**

-> (After confirmation by re-clicking) deletes current selection.

Pending edges are not allowed:

- When a state is deleted all edges associated with this state are deleted.

MOVE Command

- Markov chain editing window. (Fig 5.2)

Pick down The state or edge element or selection to move.

Locate up The new location.

This function is used to move either a state, an intermediate vertex, a handgrip or a selection.

-> Whenever a handgrip is moved, a new vertex is created as well as two additional handgrips.

When several segments of edges are aligned following the shift of an intermediate vertex, the edge is completely redrawn (unnecessary vertices and handgrips are automatically deleted).

Whenever a state is moved, edges connected to this state are automatically redrawn.

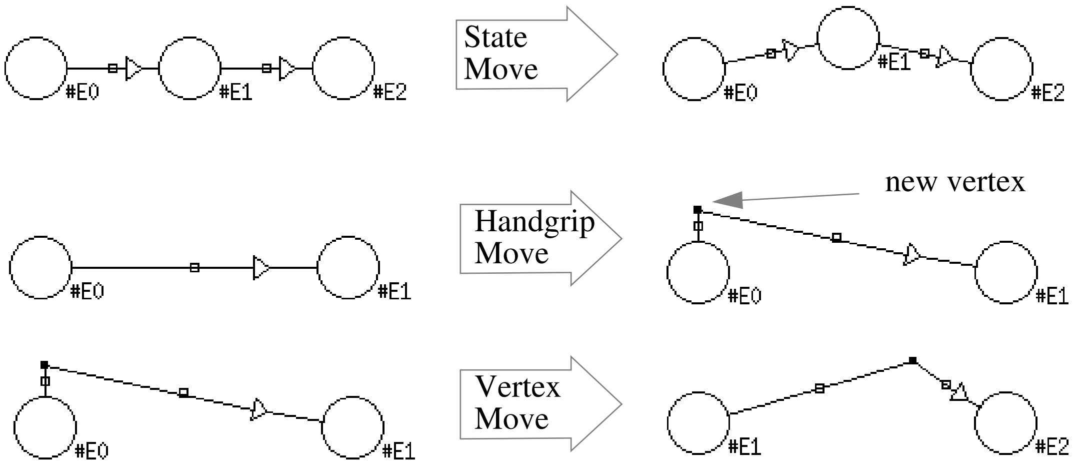


Fig 5.13 Moving examples.

Copy Command

- Markov chain editing window. (Fig 5.2)

adjust copy locate the selection copy.

-> Copy the current selection from either the editing window or a working window.

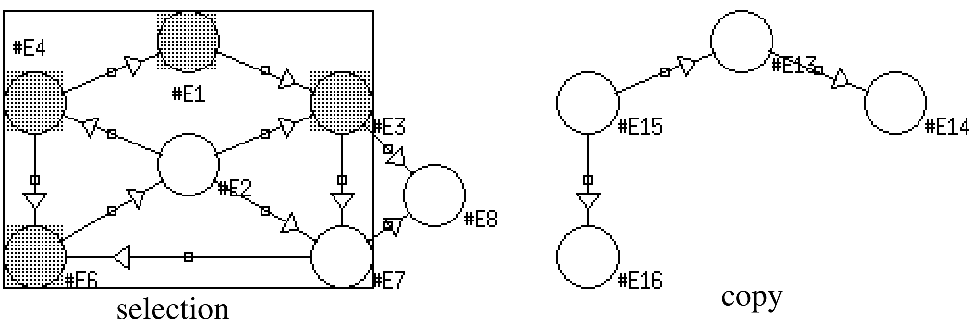


Fig 5.14 Selection copy

Report

- If there is no selection, the following message appears:

Message | No selection.

Warning: SURF 2 (selection, copy) is often named (copy, past).

Rotate Command

- *Markov chain editing window. (Fig 5.2)*

-> A $\frac{\pi}{2}$ rotation is performed on selection:

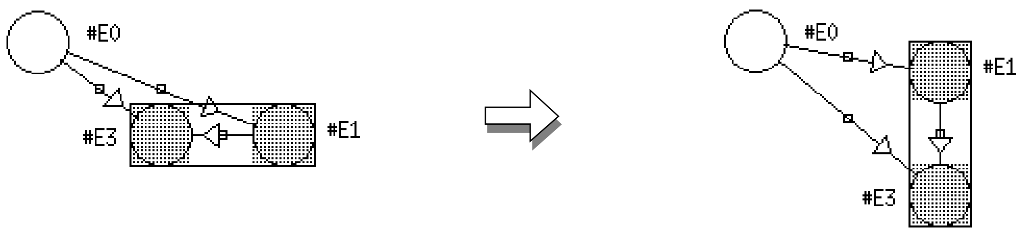


Fig 5.15 Rotate facility.

Report

If a rotation that would result in overlapping objects cannot be carried out, and a warning message is displayed:

Message	**You can’t do that, the nodes would be too close.**

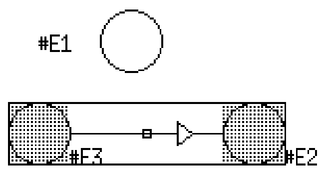


Fig 5.16 Unauthorized rotation.

Mirror Command

- *Markov chain editing window. (Fig 5.2)*

-> Flip the selected object(s) vertically.

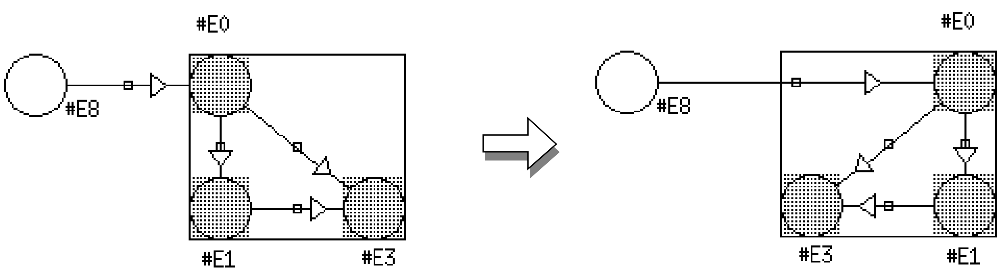


Fig 5.17 Mirror operation.

By clicking a second time the former configuration is displayed.

Report

Flipping selected objects that would result in overlapping states is not authorized. In this case, a message warns the user that such an operation cannot be performed, as shown in Fig 5.18.

Message | You can't do that, the nodes would overlaps.

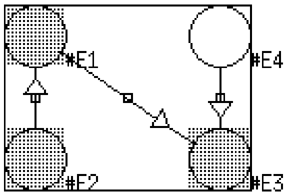


Fig 5.18 Unauthorized mirror operation.

GridStep + Command

GridStep - Command

- Markov chain editing window. (Fig 5.2)

-> increases (+) or reduces (-) the size of graph.

Initially a full-scale graph is displayed on the editing window. However, the user may wish to reduce the size of the graph, especially if it is too wide to be entirely visible on the screen (though the editing window has vertical and horizontal scroll bars that make it possible to display invisible portions of the graph on the screen).

The size of the scroll bars changes according to the scale of the grid.

When GridStep - has been consecutively clicked several times, labels assigned to the objects (names of states and edges, initial probabilities) are no longer displayed on the screen to avoid overloading the graph with texts.

Report

repetitive clicks enable us to decrease (*resp. increase*) the graph up to a threshold. Warning messages are displayed to indicate that the lower limit (*resp. upper limit*) has been reached.

Message | WARNING: The size of the objects can't be decreased any further.

Message | WARNING: the size of objects can't be increased any further.

5.8 OTHERS EDITOR COMMANDS

Part Command

- Markov chain editing window. (Fig 5.2)

-> Displays Editor of state classes or event classes, partitions and reward structures.

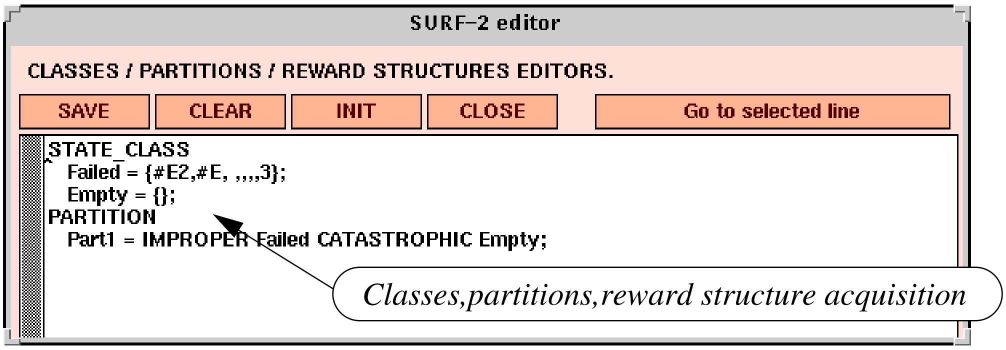


Fig 5.19 Classes / partitions / reward structures Editors

Classes partitions reward structures Acquisition

- *Classes / partitions / reward structures Editors (Fig 5.19)*

Enter the state- and event-classes, the partitions, and the reward structures.

The syntax used to define these different objects is detailed in appendix 1 page 4.

SAVE Command

- *Classes / partitions / reward structures Editors (Fig 5.19)*

-> Saves the expressions of classes (state or event classes), partitions and reward structures on disk, before returning to the EDITOR model.

Report

If no modification has been performed since the last save, the following message is then displayed:

Message	WARNING: partition has not been modified.

CLEAR Command

- *Classes / partitions / reward structures Editors (Fig 5.19)*

-> (At first click) the following message is displayed:

Message	WARNING: re-click CLEAR to confirm

-> (After confirmation by Re-click) deletes the editing area, erasing all the expressions typed by the users.

INIT Command

- *Classes / partitions / reward structures Editors (Fig 5.19)*

-> Writes, in the editing area, the key-words used for defining the classes, partitions and reward structures:

STATE_CLASS

EVENT_CLASS

PARTITION

IMPROPER

CATASTROPHIC

REWARDS

YIELD_RATE

BONUS

CLOSE Command

- *Classes / partitions / reward structures Editors (Fig 5.19)*

-> (if nothing is modified) Closes classes partitions reward structures editors.

-> (If modifications have been performed since the last saving operation) the following message is displayed:

Message	**WARNING: PARTITION MODIFIED.**
	Re-click CLOSE to confirm

Re-click to close the editor. Modifications that have not been saved are not lost as long as you remain in Markov Chain Editor.

Go to selected line Command

- *Classes / partitions / reward structures Editors (Fig 5.19)*

-> If a number is selected the text pointer goes to the corresponding line.

If there is no selected number the following message is displayed:

Message	**Select line number.**

This command is used after a **VALIDATE & SAVE Command** to access erroneous line.

VALIDATE & SAVE Command

- *Markov chain editing window. (Fig 5.2)*

-> Saves the model on disk and then runs the validation module for graphics and semantics verification purposes.

Report

- If the editing window is empty or has been cleared, the following message appears:

Message	**ERROR: empty file.**

- Once the model has been validated, and no error has been found, the following message appears:

Message	**Valid Model.**

- If errors have been detected, a debugging window is displayed (Fig 5.20):

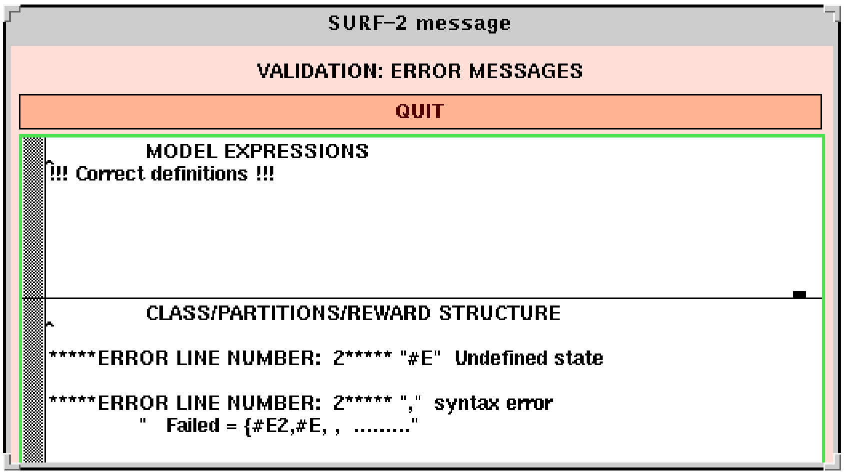


Fig 5.20 Debugging window for Markov chains.

The debugging window consists of two areas displaying error messages:

- The upper area showing:

  either error messages relative to the expressions associated with the objects,

  or the message: !!! Correct definitions !!! if no error has been detected.

- The lower area displaying:

  either error messages relative to definition of classes, partitions and reward structures,

  or the message: !!! Correct definitions !!! if no error has been detected.

<u>MODE / mode</u> switch

- Markov chain editing window. (Fig 5.2)

Choose **MODE** if you want to use MARKOV CHAIN EDITOR in read write mode.

Choose **mode** if you want to use MARKOV CHAIN EDITOR in read-only access mode.

The read-only access mode inhibits all the functions which might semantically modify the model, namely: **CLEAR**, **STATE**, **EDGE**, **Invert**, **DELETE**, **Copy**.

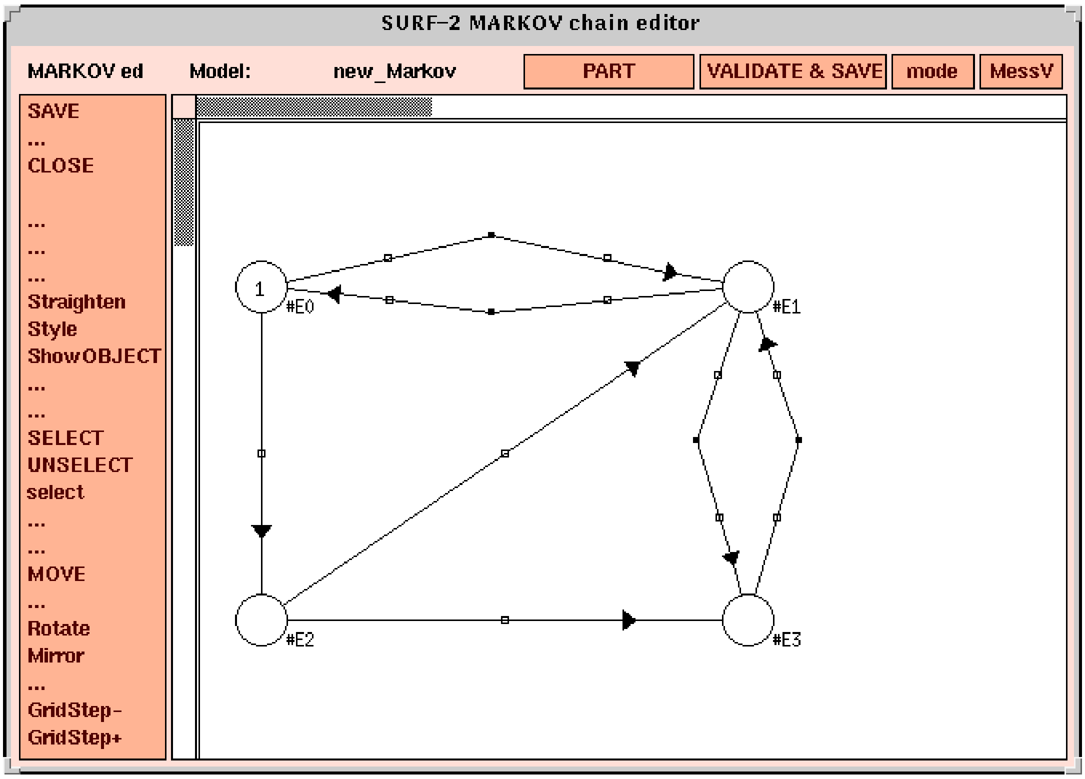


Fig 5.21 Markov chain EDITOR in read-only access mode.

Names of inhibited Commands are replaced with **...** in the list of Commands. Moreover, the **ShowOBJECT** Commands is substituted for the **InitOBJECT** Commands.

The Classes partitions reward structures editors past in read only access mode:

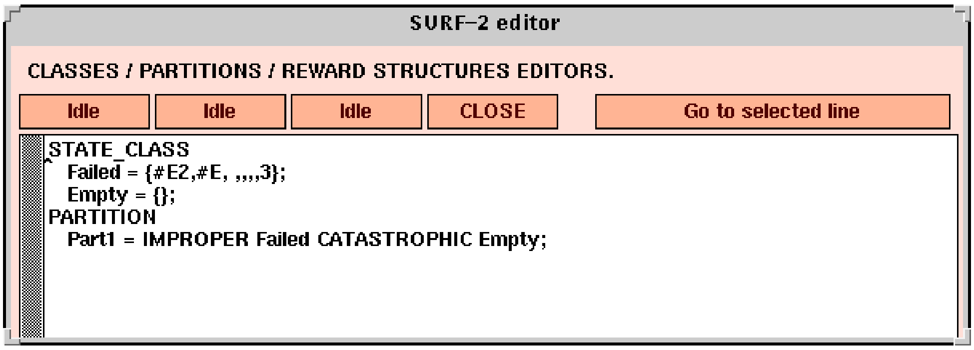


Fig 5.22 EDITOR classes / partitions / reward structures (read-only access mode).

MESS V / MESS C Switch

- *Markov chain editing window. (Fig 5.2)*

If you choose MESS V (mode Verbose) all messages are displayed.

If you choose MESS C (mode Concise) only important messages are displayed.

5.9 WORKING WINDOWS

The user can open two working windows.

Working windows look like editing windows, except the **VALIDATE & SAVE command** which is replaced with **READmodel command**.

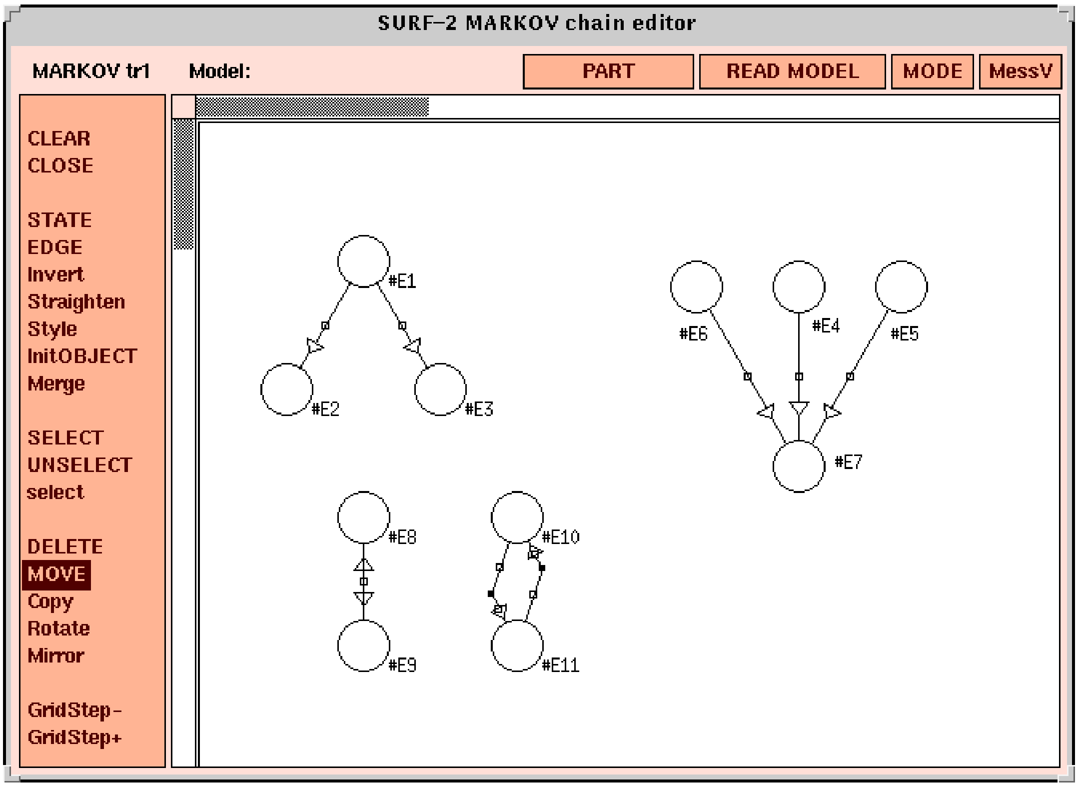


Fig 5.23 Markov chain working window.

The type of model (MARKOV) and the name of the working window (tr1 or tr2) are indicated in the uppermost part of the window.

Edited model (whole or part) can be saved temporarily in a working window.

A selected model can be read in a working window, this model (whole or part) can be copied in the editing window while drawing a new model. It is a way to import predefined patterns.

Markov chains derived from Petri nets can directly be reached through this function (see also the CREATE LIKE function).

READmodel Command

-> (At first click) the following message is displayed:

Message | Confirm the Model read-access by second click on READmodel.

-> (After confirmation by Re-click) displays Select model dialog box Dialog box to Select the model to be read. (Fig 5.24).

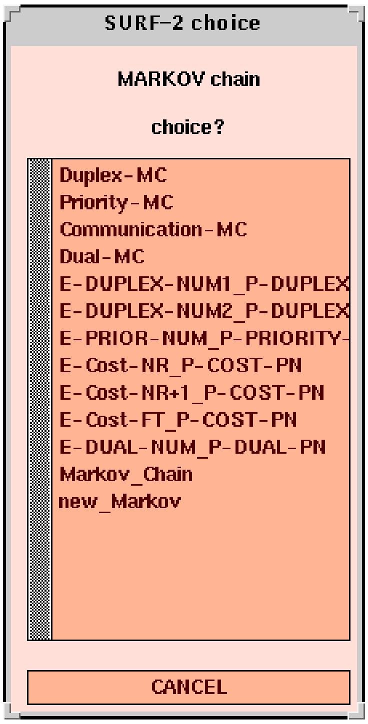


Fig 5.24   Dialog box to Select the model to be read.

Model Selection

- Dialog box to Select the model to be read. (Fig 5.24)

Select the name of the model to be read.

-> Reads the model.

Clicking CLOSE closes the working window but no modifications of the graph will be saved on disk since the EDITOR is in read-only access mode.

Chapter 6

PETRI NET EDITOR

Color: Blue

6.1 PURPOSE

Petri Net Editor allows to define a model composed of:

- a Petri Net
- a list of Classes, Partitions and Rewards.
- a list of Invariants.

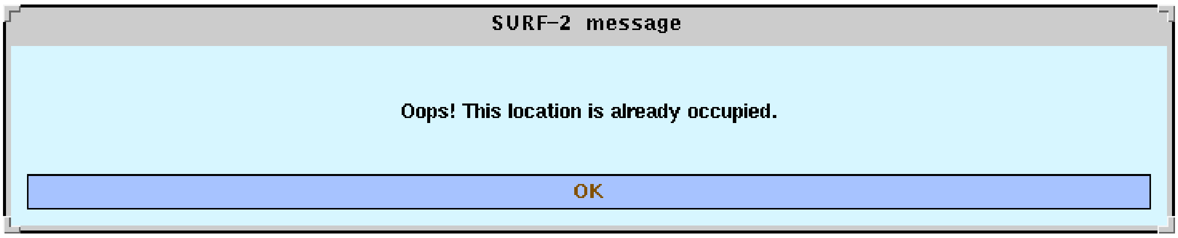
6.2 ACCESS

To access the Petri net editor, first select one of the following two commands in the MODEL MENU:

- OPEN Command in the case of an existing Petri net, see chapter 4 page 1,
- CREATE Command if the user wishes to create a new one, see chapter 4 page 2.

6.3 MESSAGES

Editor message are displayed in a dialog box:



When the user is notified of the message, he must click OK to go on.

6.4 PETRI EDITOR DIALOG BOX

OPEN or CREATE commands of MODEL MENU give access to the window shown below:

Fig 6.1 Petri Editor dialog box.

MODEL EDITOR Command

- *Petri Editor dialog box. (Fig 6.1)*

-> Displays the petri net editing window (see Fig 6.2).

Working window1 Command

Working window2 Command

- *Petri Editor dialog box. (Fig 6.1)*

These commands can be used to open two different working windows (refer page 27 for a detailed description).

QUIT EDITOR Command

- *Petri Editor dialog box. (Fig 6.1)*

-> (if nothing is modified since last save) quits the Petri Net EDITOR.

-> (if the model has been modified but not saved) one of the two following warnings is displayed:

Message | WARNING: graphics modifications! confirm CLOSE by re-clicking.

this message indicates that graphics modifications have been introduced.

Message | WARNING: model semantics modified! confirm CLOSE by re - clicking.

this message indicates that semantics modifications have been introduced.

Save your work or Re-click to quit without save.

6.5 INTRODUCTION TO MODEL EDITING WINDOW

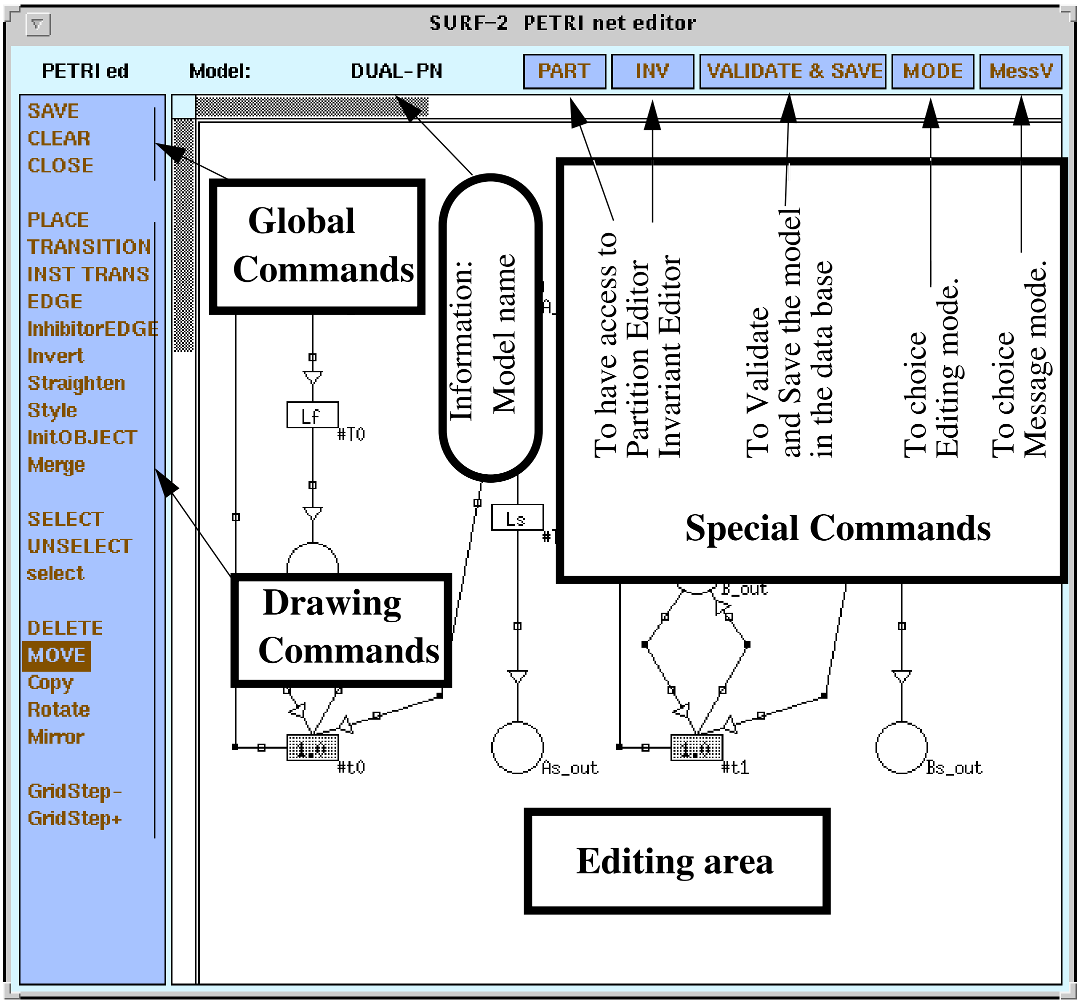


Fig 6.2 Petri net editing window.

An overview of Petri net editing window is done by Fig 6.2.

The editing window is made of four areas:

- The Editing area where the Petri net can be displayed and modified,
- A Global Command menu to manage the model (page 3),
- A Drawing Command menu, for drawing the graph (page 4),
- and few others Commands (page 21):
  - convenient commands
  - commands to access partitions and invariants editors.

6.6 GLOBAL COMMANDS

SAVE Command

- Petri net editing window. (Fig 6.2)

-> Saves the model drawing on disk without any validation.

CLEAR Command

- Petri net editing window. (Fig 6.2)

-> (at first click) the following message is displayed:

Message | Confirm CLEAR by re - clicking

-> (After confirmation by re-clicking) Clears the editing window.

CLOSE Command

- Petri net editing window. (Fig 6.2)

-> Closes the editing window. The model is first temporarily saved on disk.

Report

When clicking this function without having saved the graphics modifications, the following warning is displayed:

Message | WARNING: graphics modifications! confirm CLOSE by re - clicking.

When clicking this function without having saved the semantics modifications, the following warning is displayed:

Message | WARNING: model semantics modified! confirm CLOSE by re - clicking.

6.7 DRAWING COMMANDS

6.7.1 Petri net representation

A Petri net is composed of **places**, **timed transitions** and **immediate transitions** (Fig 6.3)

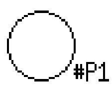
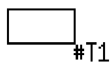
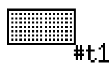
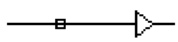
	Objects		Attributes
Nodes	PLACES		Name Marking Comment
Nodes	Timed Transition		Name Rate Comment
Nodes	Immediate Transitions		Name Initial Probability Comment
	Edge		Weight Comment

Fig 6.3 Petri net components.

A Petri net is displayed as show Fig 6.4:

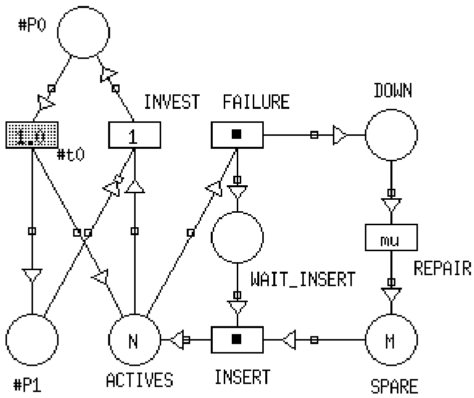


Fig 6.4 Petri net.

Edge components:

An edge is composed of **arrow**, **vertex** and **handgrip** show :

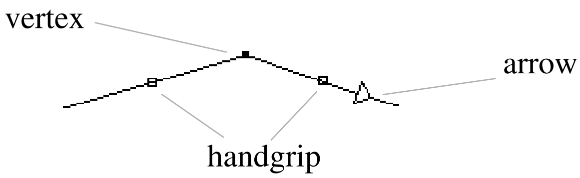


Fig 6.5 Edge components

.Pending edges are not allowed:

- *Origin and destination state must be created before corresponding edge.*
- *When a state is deleted all edges associated with this state are deleted.*

Selection:

Places transitions and edges can be grouped in a current **selection** that can be manipulated as a whole. Only one selection can be done at a time.

Places and transitions are explicitly selected. An edge is selected if its origin and its destination are selected. Selected places and transitions are shaded. The selection area (Fig 6.6) is defined by a rectangular box.

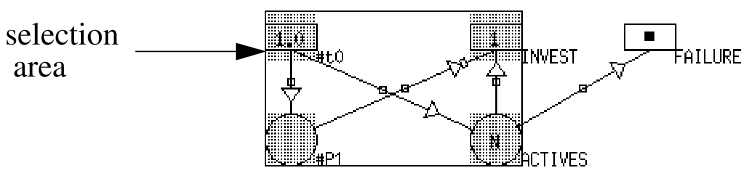


Fig 6.6 The current selection.

6.7.2 Description of basic actions for Drawing Functions

Some editor functions need complementary information(s) to be done, for example **Place location** for a **Place creation**.

These informations are defined by means of few basic **actions** (the pointer shape indicates the action expected by the editor). Use the left button of the mouse.

Locate

Pointer shape: +

Used to enter a location.

Move the cursor at the wished position and click.

Pick

Pointer shape:

Used to select a single object or selection.

- If you want to pick a node move the pointer near the middle of this node and click.
- If you want to pick an edge move the pointer on one of these edge element: (handgrip, arrow, vertex, see) and click.
- If you want to pick the selection move the pointer inside the selection area and click.

Pick down

Pointer shape:

Allows to select an object when the function need more information than this simple object selection.

Move the pointer near the middle of the object to select (state, edge handgrip, edge arrow or edge vertex) then press button. the button must remain pressed.

Pick up

Pointer shape:

Allows to select a second object after a first selection by a Pick down.

Used after a **Pick down** therefore the button is pressed. Move the pointer near the middle of the object to select (state, edge handgrip, edge arrow or edge vertex) then release the button.

A rubber band is draw from pick down location to current pointer position.

Locate up

Pointer shape:

Allows to enter a new location for an object previously selected by a Pick down.

Used after un **Pick down** therefore the button is pressed. Move the pointer to the wished free position and then release the button.

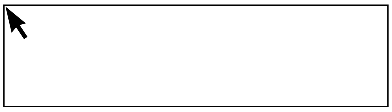
Define area

Pointer shape:

Allows to define an area.

Move the cursor at the top left vertex of the area and press button; the pointer takes the ⌟ shape. With the button pressed move the pointer to bottom right vertex of the area and then release the button.

Adjust copy

Pointer shape with copy outline: 

Allows to locate the selection copy.

When the button is pressed the outline of the selection copy is diplayed. Hold down the button and choose the copy location, then release the button.

6.7.3 Reports

Pick, Pick down, Pick up:

When an object has not been correctly designated, the following message is displayed:

Message	**Designate the object to be "***command***".**

Where "*command*" refers to the current command.

Locate, Locate up:

If the user picks an object when a free location is expected the following message is diplayed:

Message	**Oops! This location is already occupied>**

6.7.4 Drawing Commands definition

For each drawing command we define:

- The name of the command.
- The dialog box where the command appear.
- The list of basic actions need by the command.
- -> The command effect.

> *.The Command must first be selected before designating the object to which it has to be applied. The command remains active as long as a new one has not been selected.*
> *By example, after a* ***PLACE Command****, user repeats* ***locate action*** *to create a set of places.*

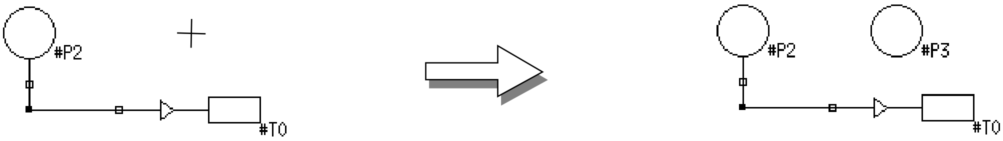
PLACE Command

- Petri net editing window. (Fig 6.2)

Locate Position of the new place

-> Place creation.

A place is graphically depicted by a circle and a name is automatically assigned to each place as follows: #Pn, where # is a reserved character used for automatically labelling a place and 'n' being the state number in the graph.



A place has an initial marking by default equal to zero.

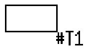
TRANSITION Command

- *Petri net editing window. (Fig 6.2)*

Locate Position of the new timed transition

-> Timed transition creation.

A timed transition is graphically depicted by a rectangle and a name (#Tn).



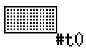
INST TRANS Command

- *Petri net editing window. (Fig 6.2)*

Locate Position of the new immediate transition

->Immediate transition creation.

An immediate transition is graphically depicted by a rectangle and a name (#tn).



EDGE Command

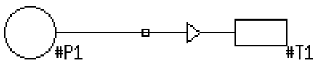
- *Petri net editing window. (Fig 6.2)*

pick down the origin of the new edge.

pick up the extremity of the new edge.

-> Creation of an oriented edge graphically depicted by a segment with a handgrip and an arrow.

Integer positive (or zero) values can be assigned to the edges. Edge weights are initially set to one. (see also the InitOBJECT function).



Report

Failure to accurately click the center of the state will result in the following messages being displayed:

- if the origin node has not been correctly designated.

Message	**Designate the origin of edge.**

- if the destination node has not been correctly designated.

Message	**Oops! Give the origin and the destination of the edge again.**

- if the origin node and the destination node which have been designated merge into one.

Message	**Oops! The origin and the destination overlap Give the origin again**

- if you try to draw an existing edge.

Message	**Oops! This edge already exist.**

- An edge necessarily joins a place and a transition. If the user tries to connect nodes of same type, the following message will appear:

Message	**Oops! Edges between two nodes of same type are not allowed.**

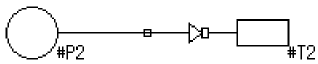
InhibitorEDGE Command

- Petri net editing window. (Fig 6.2)

pick down the place origin of the new edge.

pick up the transition extremity of the new edge.

-> Creation of an oriented edge graphically depicted by a segment with a handgrip and an arrow.Weight is set to zero.



-

Report

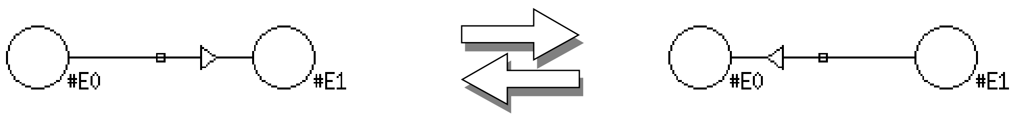
Message	**Inhibitor edge from an output transition to a place is not allowed.**

Invert Command

- Petri net editing window. (Fig 6.2)

Pick Edge to invert.

-> Invert the direction of the selected edge.



Report

An inhibitor edge can't be inverted:

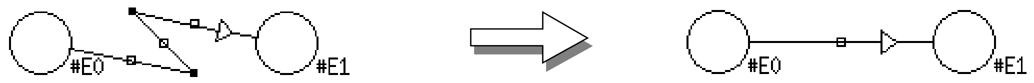
Message	**Inhibitor edge from an output transition to a place is not allowed.**

Straighten Command

- Petri net editing window. (Fig 6.2)

Pick Edge to straighten.

-> straightens selected edge.

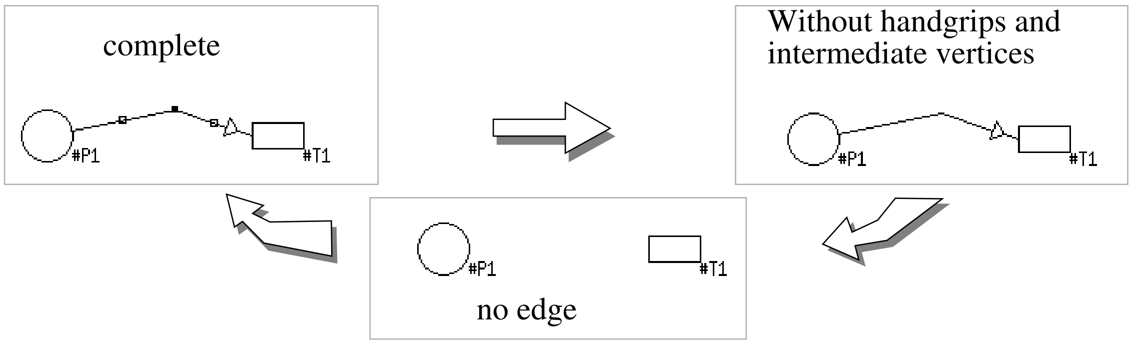


Style Command

- *Petri net editing window. (Fig 6.2)*

No action

At each invocation its function changes edge representation for all edges in the graph.



InitOBJECT Command (for a place)

- *Petri net editing window. (Fig 6.2)*

Pick Place to initialize.

-> Displays dialog box for place attributes (Fig 6.7)

The selected place is marked with a cross.

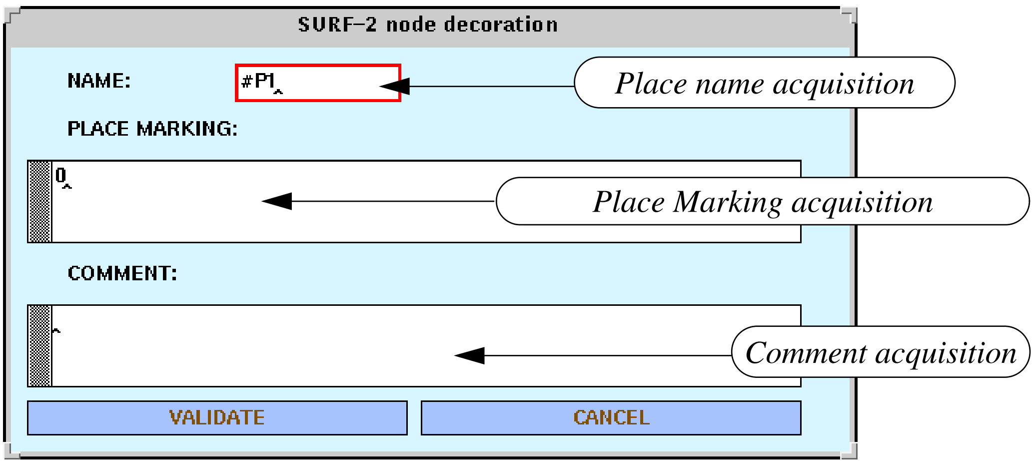


Fig 6.7 Dialog box for place attributes.

Place name Acquisition

- *Dialog box for place attributes. (Fig 6.7)*

Enter a name if you wish to replace the one that has been automatically assigned. First erase the old name using <BACKSPACE> on your keyboard.

Report

The user cannot give a name beginning with #. A warning message indicates this error:

Message |	**Names preceded by # are reserved system words.**

Place Marking Acquisition

- Dialog box for place attributes. (Fig 6.7)

Enter the expression of place marking.

The syntax is defined in Expressions defining objects page: 3 of appendix 1.

Syntax control is done by **VALIDATE&SAVE command**, see page 27.

Comment Acquisition

- Dialog box for place attributes. (Fig 6.7)

A comment can be added. A vertical scroll bar can be used to display the portion of text that may not be visible inside the text window.

VALIDATE Command

- Dialog box for place attributes. (Fig 6.7)

-> Up to date place without syntax control, see Fig 6.8.

The Dialog box for place attribute disappears.

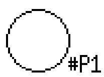
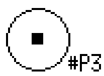
Place:	marking:
	0
	short expression. The value is displayed
	long expression. The value is not displayed use InitObject to read value

Fig 6.8 Visualization of place marking

CANCEL Command

-> The Dialog box for place attributes disappears. The place is not modified.

InitOBJECT Command (for a timed transition)

- Petri net editing window. (Fig 6.2)

Pick Timed transition to initialize.

-> Displays dialog box for timed transition attributes.

The selected transition is marked with a cross.

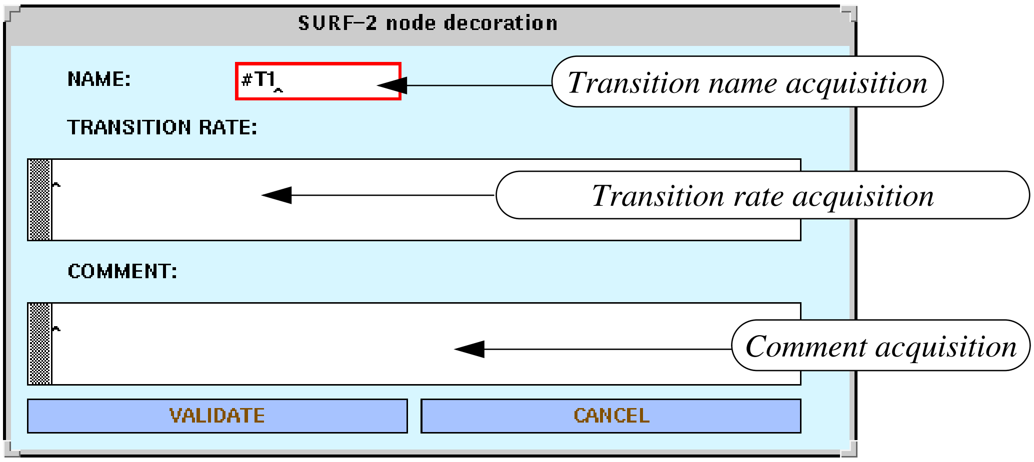


Fig 6.9 Dialog box for timed transition attributes.

Transition name Acquisition

- Dialog box for place attributes. (Fig 6.7)

Enter a name if you wish to replace the one that has been automatically assigned. First erase the old name using <BACKSPACE> on your keyboard.

Report

The user cannot give a name beginning with #. A warning message indicates this error:

Message	**Names preceded by # are reserved system words.**

Transition rate Acquisition

- Dialog box for timed transition attributes. (Fig 6.9)

Enter the rate of timed transition.

The syntax is defined in Expressions defining objects page: 3 of appendix 1.

Syntax control is done by **VALIDATE&SAVE command**, see page 27.

Comment Acquisition

- Dialog box for timed transition attributes. (Fig 6.9)

A comment can be added. A vertical scroll bar can be used to display the portion of text that may not be visible inside the text window.

VALIDATE Command

- Dialog box for timed transition attributes. (Fig 6.9)

-> Updates transition without syntax control, see Fig 6.10.

The Dialog box for place attribute disappears.

Timed transition:	rate:
#T1	not defined
0.5 #T2	short expression. The value is displayed
■ #T3	long expression. The value is not displayed use InitObject to read value

Fig 6.10 Visualization of timed transition rate

CANCEL Command

-> The Dialog box for timed transition attributes disappears. The transition is not modified.

InitOBJECT Command (for an immediate transition)

- Petri net editing window. (Fig 6.2)

Pick Immediate transition to initialize.

-> Displays dialog box for immediate transition attributes. The selected transition is marked with a cross.

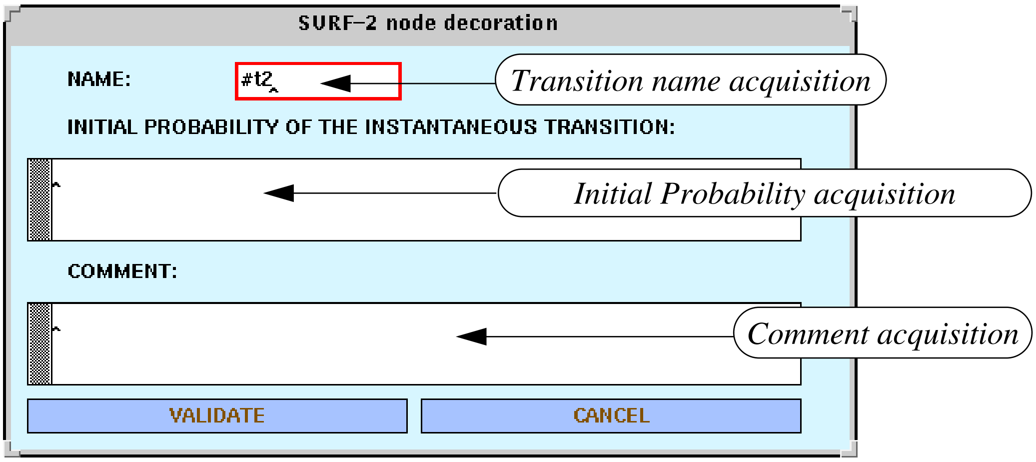


Fig 6.11 Dialog box for immediate transition attributes.

Transition name Acquisition

- Dialog box for immediate transition attributes. (Fig 6.11)

Enter a name if you wish to replace the one that has been automatically assigned. First erase the old name using <BACKSPACE> on your keyboard.

Report

The user cannot give a name beginning with #. A warning message indicates this error:

Message	Names preceded by # are reserved system words.

Initial Probability Acquisition

- Dialog box for immediate transition attributes. (Fig 6.11)

Enter the initial probability of immediate transition.

The syntax is defined in Expressions defining objects page: 3 of appendix 1.

Syntax control is done by **VALIDATE&SAVE command**, see page 27.

Comment Acquisition

- Dialog box for immediate transition attributes. (Fig 6.11)

A comment can be added. A vertical scroll bar can be used to display the portion of text that may not be visible inside the text window.

VALIDATE Command

- Dialog box for immediate transition attributes. (Fig 6.11)

-> Up to date transition without syntax control, see Fig 6.12.

The Dialog box for immediate transition attribute disappears.

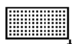
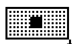
Immediate transition:	initial probability:
 #t1	not defined
 #t2	short expression. The value is displayed
 #t3	long expression. The value is not displayed use InitObject to read value

Fig 6.12 Visualization of immediate transition rate

CANCEL Command

-> The Dialog box for immediate transition attributes disappears. The transition is not modified.

InitOBJECT Command (for an edge)

- Petri net editing window. (Fig 6.2)

Pick Edge to initialize.

-> Displays dialog box for edge attributes.

The selected edge is marked with a cross.

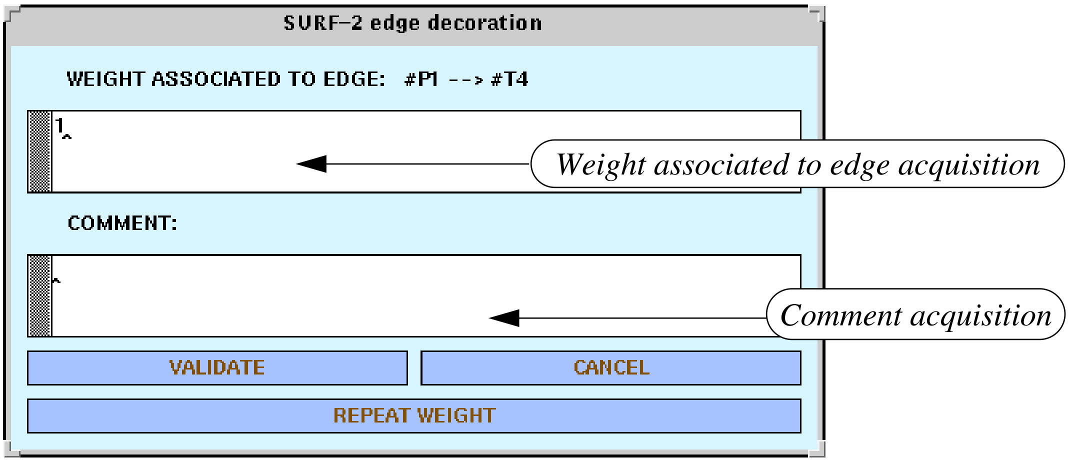


Fig 6.13 Dialog box for edge attributes.

Weight associated to edge Acquisition

- Dialog box for edge attributes. (Fig 6.13)

Enter the weight of edge.

The syntax is defined in Expressions defining objects page: 3 of appendix 1.

Syntax control is done by **VALIDATE&SAVE command**, see page 27.

Comment Acquisition

- Dialog box for place attributes. (Fig 6.7)

A comment can be added. A vertical scroll bar can be used to display the portion of text that may not be visible inside the text window.

REPEAT Weight Command

- Dialog box for edge attributes. (Fig 6.13)

-> Assigns the same value as the one already assigned to the previous selected weight.

VALIDATE Command

- Dialog box for edge attributes. (Fig 6.13)

-> Updates transition without syntax control, see Fig 6.14.

The Dialog box for edge attribute disappears.

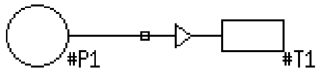
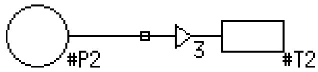
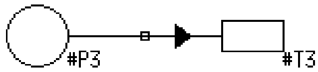
Edge:	weight:
	not defined
	short expression. The value is displayed
	long expression. The value is not displayed use InitObject to read value

Fig 6.14 Visualization of weight edge

<ins>CANCEL</ins> Command

-> The Dialog box for edge attributes disappears. The edge is not modified.

> *.As long as InitOBJECT dialog box is displayed, the user cannot use other functions of the model editor.*

<ins>Merge</ins> Command

- *Petri net editing window. (Fig 6.2)*

<ins>Pick down</ins> Select a place (resp. a transition), this node will be deleted.

<ins>Pick up</ins> Select a place (resp. a transition), this node will be kept.

-> Merge first object with on the second, see Fig 6.15

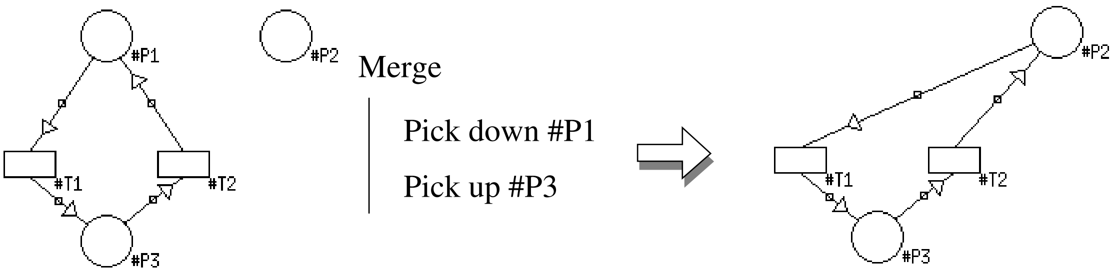


Fig 6.15 Merge of two places

Report

- if the first node has not been correctly designated:

Message	**Designate the first node.**

- if the second node has not been correctly designated:

Message	**Node to kept: Indicate again the node to delete and the node to keep.**

- in case of unauthorized merging:

Fig 6.16 (a):

Message	**MERGING REFUSED:**
	The merge would create two identical edges with the same origin.

Fig 6.16 (b):

Message	**MERGING REFUSED:**
	The merge would create two identical edges with the same destination.

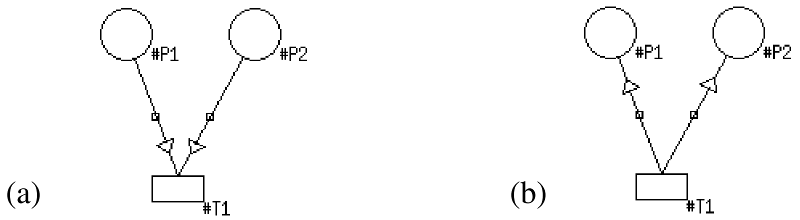


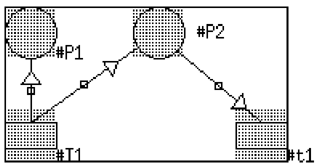
Fig 6.16 Examples of unauthorized merging (#P1 with #P2).

SELECT Command

- *Petri net editing window. (Fig 6.2)*

No action

-> Selects all the objects in the editing window.



UNSELECT Command

- *Petri net editing window. (Fig 6.2)*

No action

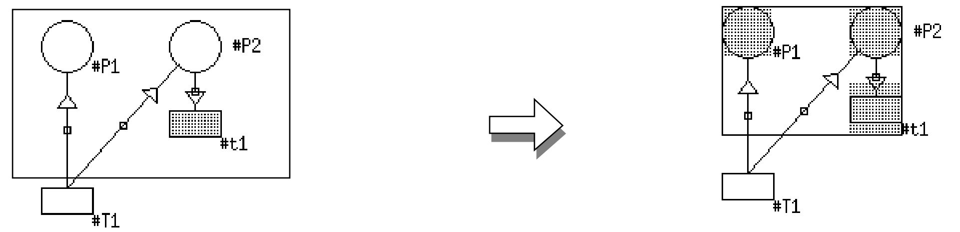
-> Unselects all the objects in the editing window.

select Command

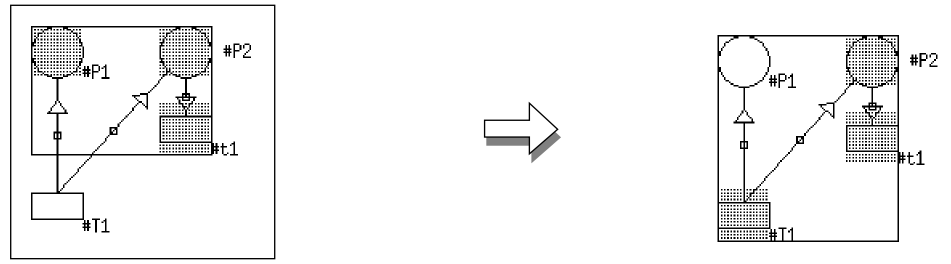
- *Petri net editing window. (Fig 6.2)*

Define area the selection area.

-> Reverses the selection attribute (select / unselect) of each state located in the selection area. Selected states are shaded.



Edges selected: #P2->#t1,



Edge selected: #T1->#P2, #P2->#t1

Fig 6.17 Example of nodes and edges selections.

<u>DELETE</u> Command (place, transition or edge)

- *Petri net editing window. (Fig 6.2)*

<u>pick</u> place, transition or edge.

-> Deletes place, transition or edge.

<u>DELETE</u> Command (Current selection)

- *Petri net editing window. (Fig 6.2)*

<u>pick</u> selection.

-> (At first click) the following message is diplayed:

Message	Confirm the delete operation by a second click in the selection.

-> (After confirmation by re-clicking) deletes current selection.

Pending edges are not allowed:

- *When a state is deleted all edges associated with this state are deleted.*

<u>MOVE</u> Command

- *Petri net editing window. (Fig 6.2)*

<u>Pick down</u> The place, transition, edge element or selection to move.

<u>Locate up</u> The new location.

This function is used to move either a place, a transition, an intermediate vertex, a handgrip or a selection.

-> Whenever a handgrip is moved, a new vertex is created as well as two additional handgrips.

When several segments of edges are aligned following the shift of an intermediate vertex, the edge is completely redrawn (unnecessary vertices and handgrips are automatically deleted).

Whenever a state is moved, edges connected to this state are automatically redrawn.

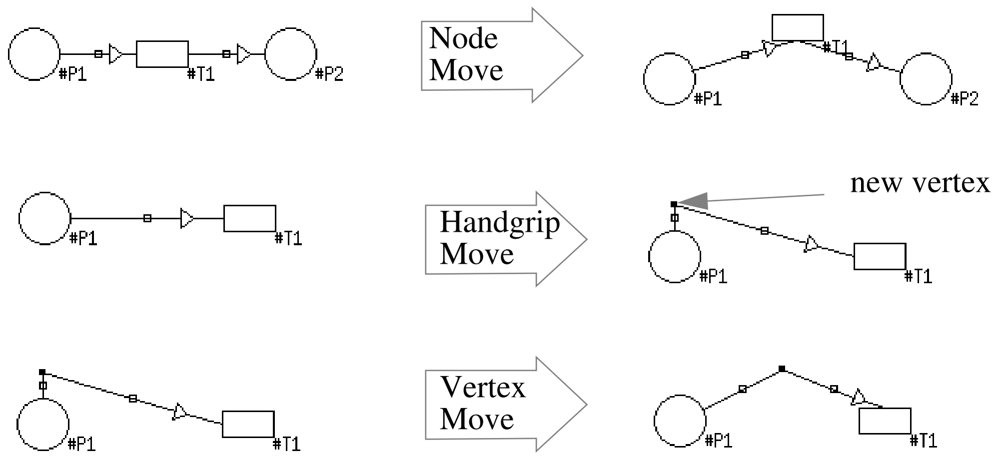


Fig 6.18 Moving examples.

Copy Command

- Petri net editing window. (Fig 6.2)

adjust copy locate the selection copy.

-> Copy the current selection from either the editing window or a working window.

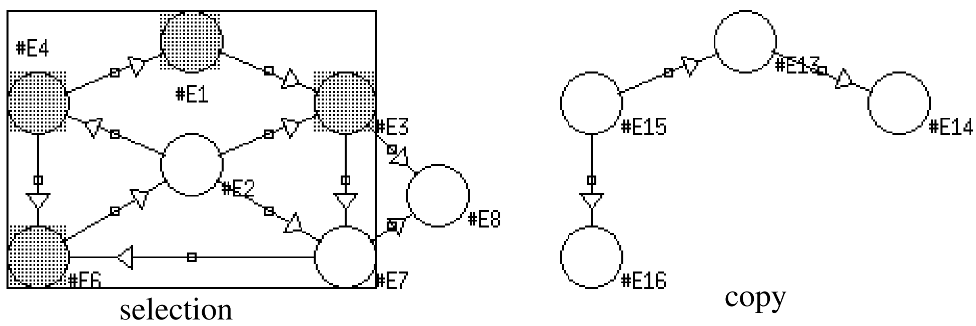


Fig 6.19 Selection copy

Report

If there is no selection, the following message is displayed:

Message | No selection.

Warning: SURF 2 (selection, copy) is often named (copy, paste).

- *Objects can be selected either from the editing window or a working window.*

Rotate Command

- Petri net editing window. (Fig 6.2)

-> A $\frac{\pi}{2}$ rotation is performed on selection:

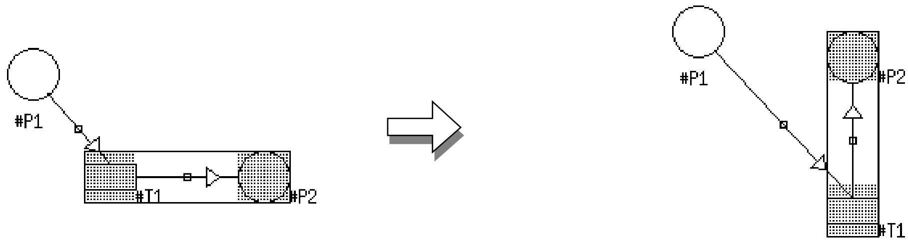


Fig 6.20 Rotate facility.

Report

A rotation that would result in overlapping objects cannot be carried out and will give rise to the display of a warning message:

Message | You can't do that, the nodes would be too close.

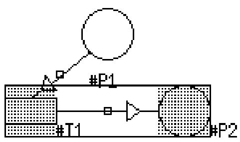


Fig 6.21 Unauthorized rotation.

Mirror Command

- Petri net editing window. (Fig 6.2)

-> Flips the selection vertically.

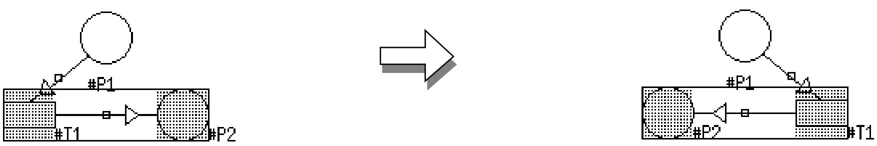


Fig 6.22 Mirror operation.

By clicking a second time the former configuration is displayed.

Report

Flipping selected objects that would result in overlapping states is not authorized.

In this case, a message warns the user that such an operation cannot be performed, as shown in Fig 6.23.

Message | You can't do that, the nodes would overlaps.

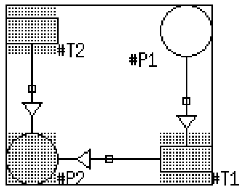


Fig 6.23 Unauthorized mirror operation.

GridStep + Command

GridStep - Command

- Petri net editing window. (Fig 6.2)

-> increases (+) or reduces (-) the size of graph.

Initially a full-scale graph is displayed on the editing window. However, the user may wish to reduce the size of the graph, especially if it is too wide to be entirely visible on the screen (though the editing window has vertical and horizontal scroll bars that make it possible to display invisible portions of the graph on the screen). To reduce the size of a graph, the user must click ***GridStep -*** *.Conversely, it is possible to increase the size of a graph previously reduced by clicking* ***GridStep +****.*
Repetitive clicks enable us to decrease (resp. increase) the graph up to a threshold. Warning messages are displayed to indicate that the lower limit (resp. upper limit) has been reached.
When GridStep - has been consecutively clicked several times, labels assigned to the objects (names of states and edges, initial probabilities) are no longer displayed on the screen to avoid overloading the graph with texts.

Report

repetitive clicks enable us to decrease (*resp. increase*) the graph up to a threshold. Warning messages are displayed to indicate that the lower limit (*resp. upper limit*) has been reached.

Message | WARNING: The size of the objects can't be decreased any further.

Message | WARNING: the size of objects can't be increased any further.

6.8 OTHERS EDITOR COMMANDS

PART Command

- Petri net editing window. (Fig 6.2)

-> Displays Editor of state classes or event classes, partitions and reward structures.

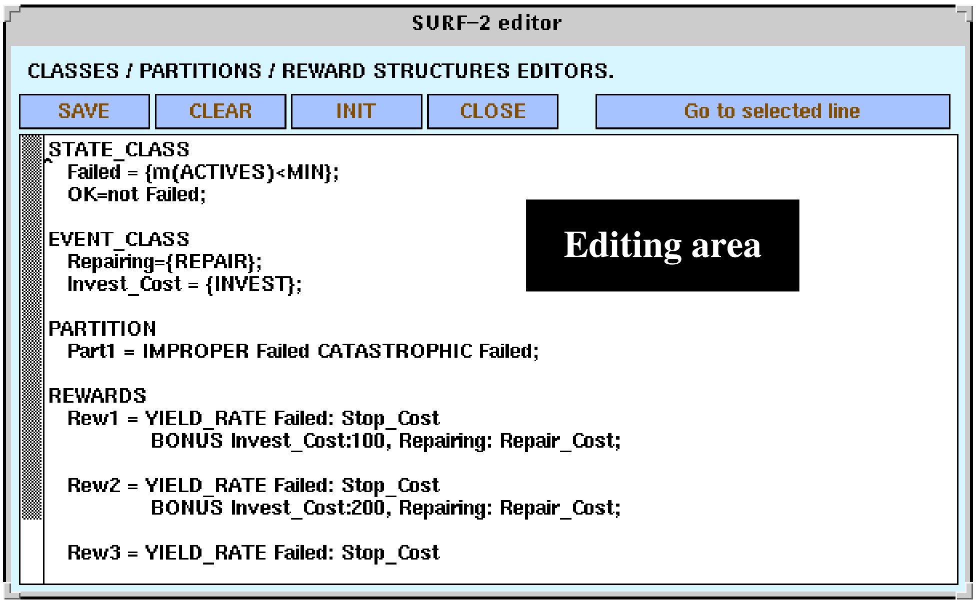


Fig 6.24 Classes / partitions / reward structures Editors

Classes partitions reward structures Acquisition

- *Classes / partitions / reward structures Editors (Fig 6.24)*

Enter the state- and event-classes, the partitions, and the reward structures.

The syntax used to define these different objects is detailed in appendix 1 page 4.

SAVE Command

- *Classes / partitions / reward structures Editors (Fig 6.24)*

-> Saves the expressions of classes (state or event classes), partitions and reward structures on disk, before returning to the EDITOR model.

Report

If no modification has been performed since the last save, the following message is then displayed:

Message	**WARNING: partition has not been modified.**

CLEAR Command

- *Classes / partitions / reward structures Editors (Fig 6.24)*

-> (At first click) the following message is displayed:

Message	**WARNING: re-click CLEAR to confirm**

-> (After confirmation by Re-click) deletes the editing area, erasing all the expressions typed by the users.

INIT Command

- *Classes / partitions / reward structures Editors (Fig 6.24)*

-> Writes, in the editing area, the key-words used for defining the classes, partitions and reward structures:

STATE_CLASS

EVENT_CLASS

PARTITION

IMPROPER

CATASTROPHIC

REWARDS

YIELD_RATE

BONUS

CLOSE Command

\- *Classes / partitions / reward structures Editors (Fig 6.24)*

-> (if nothing is modified) Close Classes partitions reward structures editors.

-> (If modifications have been performed since the last saving operation) the following message will be displayed:

Message	**WARNING: PARTITION MODIFIED.**
	Re-click CLOSE to confirm

Re-click to closes the editor. Modifications that have not been saved are not lost as long as you remain in Markov Chain Editor.

Go to selected line Command

\- *Classes / partitions / reward structures Editors (Fig 6.24)*

-> If a number is selected, the text pointer goes to the corresponding line.

If there is no selected number the following message is displayed:

Message	**Select line number.**

This command is used with **VALIDATE & SAVE Command** to access erroneous line.

INV Command

\- *Petri net editing window. (Fig 6.2)*

-> Displays Editor of state classes or event classes, partitions and reward structures.

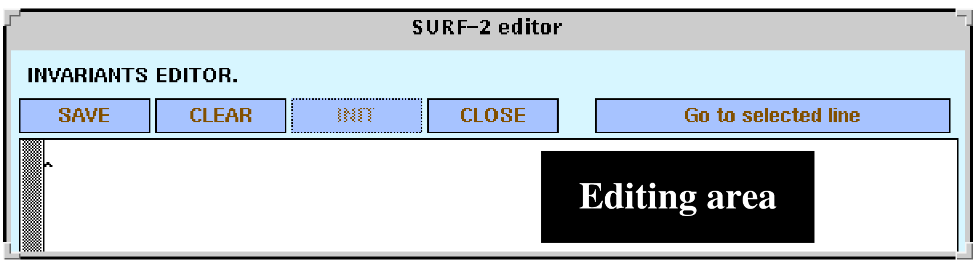


Fig 6.25 INVARIANTS Editors

The Invariant editor is similar to the classes partitions reward structures editors (see page 21)

VALIDATE & SAVE Command

- Petri net editing window. (Fig 6.2)

-> Saves the model on disk and then runs the validation module for graphics and semantics verification purposes.

Report

- If the editing window is empty or has been cleared, the following message appears:

Message	**ERRORR: empty file.**

- Once the model has been validated, and no error has been found, the following message appears:

Message	**Valid Model.**

- If errors have been detected, a debugging window is displayed:

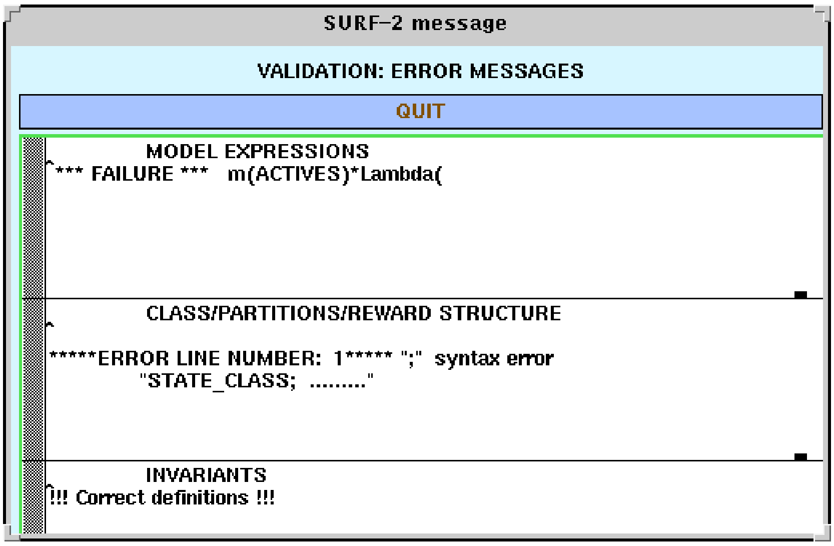


Fig 6.26 Debugging window for Petri Net.

The debugging window consists of three areas displaying error messages:

- The upper area showing:

  either error messages relative to the expressions associated with the objects,

  or the message: !!! Correct definitions !!! if no error has been detected.

- The middle area displaying:

  either error messages relative to definition of classes, partitions and reward structures,

  or the message: !!! Correct definitions !!! if no error has been detected.

- The lower area displaying:

  either error messages relative to definition of invariants,

  or the message: !!! Correct definitions !!! if no error has been detected.

MODE / mode switch

- *Petri net editing window. (Fig 6.2)*

Choose **MODE** if you want to use MARKOV CHAIN EDITOR in read write mode.

Choose **mode** if you want to use MARKOV CHAIN EDITOR in read-only access mode.

The read-only access mode inhibits all the functions which might semantically modify the model, namely: **CLEAR**, **STATE**, **EDGE**, **Invert**, **DELETE**, **Copy**.

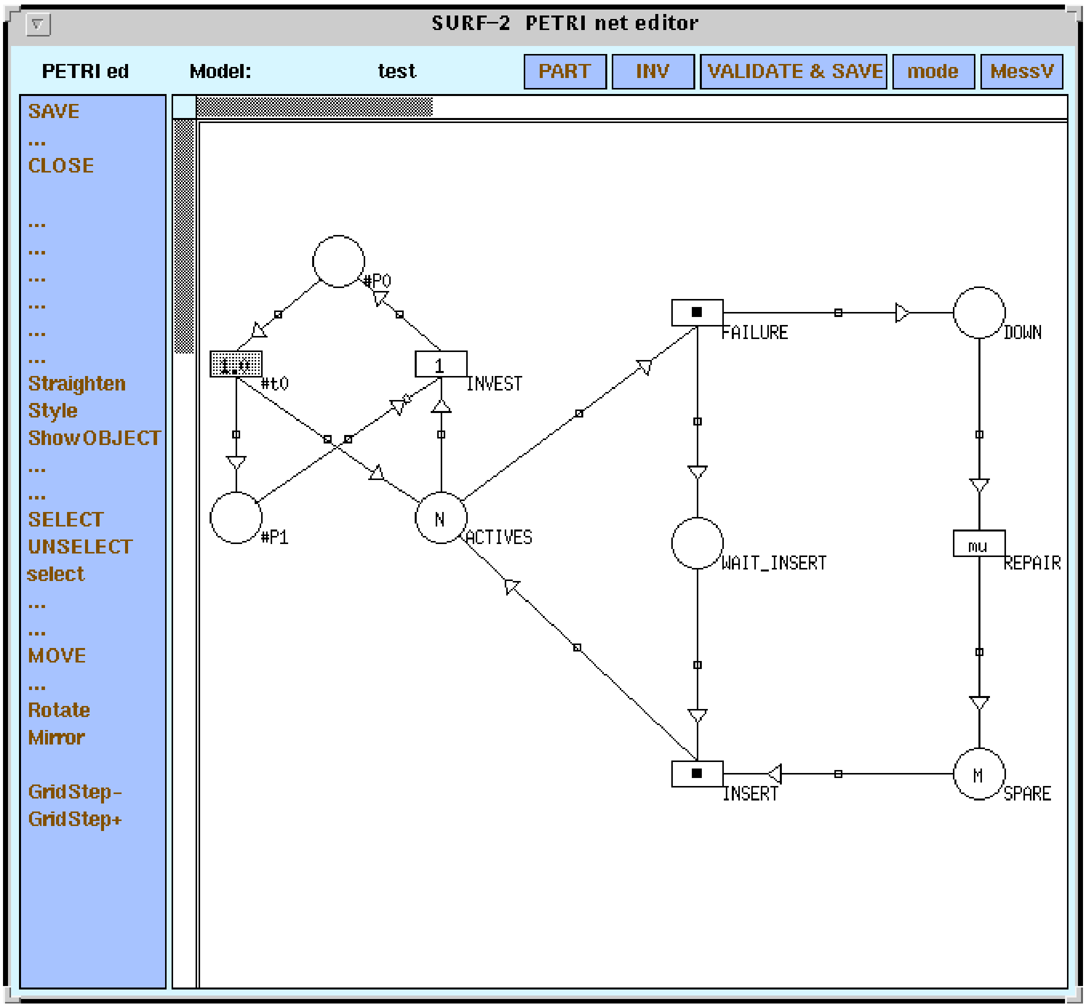


Fig 6.27 Markov chain EDITOR in read-only access mode.

Names of inhibited Commands are replaced with **...** in the list of Commands. Moreover, the **ShowOBJECT** Command is substituted for the **InitOBJECT** Command.

The Classes partitions reward structures editors switch to read only access mode:

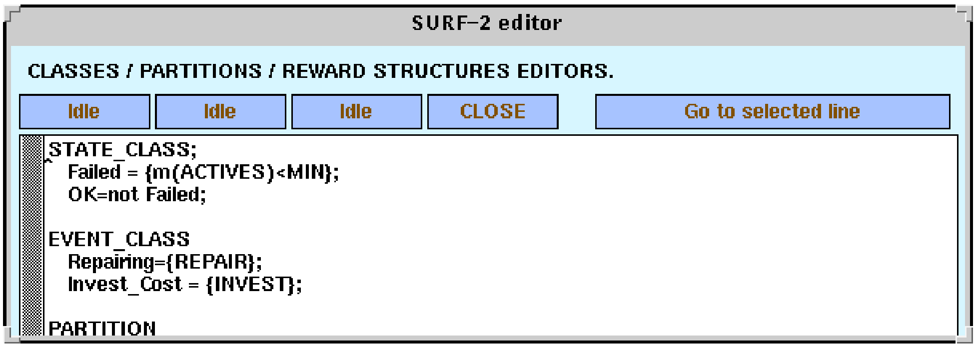


Fig 6.28 classes / partitions / reward structures EDITOR (read-only access mode).

The Invariants editors switch to read only access mode:

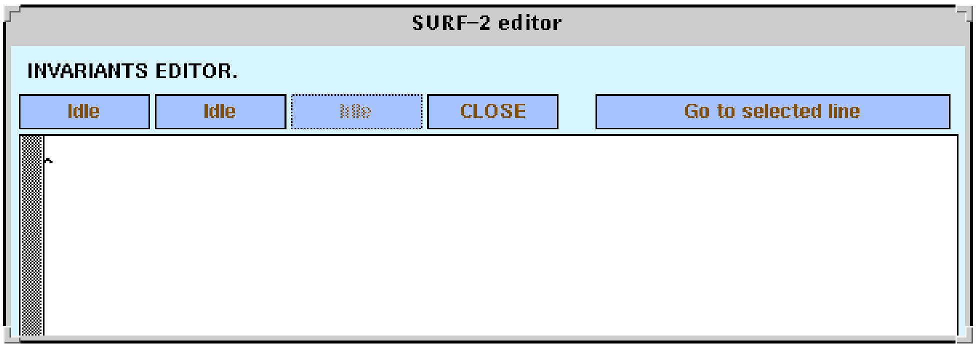


Fig 6.29 Invariants EDITOR (read-only access mode).

MESS V / MESS C Switch

- Petri net editing window. (Fig 6.2)

If you choose MESS V (mode Verbose) all messages are displayed.

If you choose MESS C (mode Concise) only important messages are displayed.

6.9 WORKING WINDOWS

The user can open two working windows.

Working windows are like editing windows, except for the **VALIDATE & SAVE** command which is replaced by **READmodel** command.

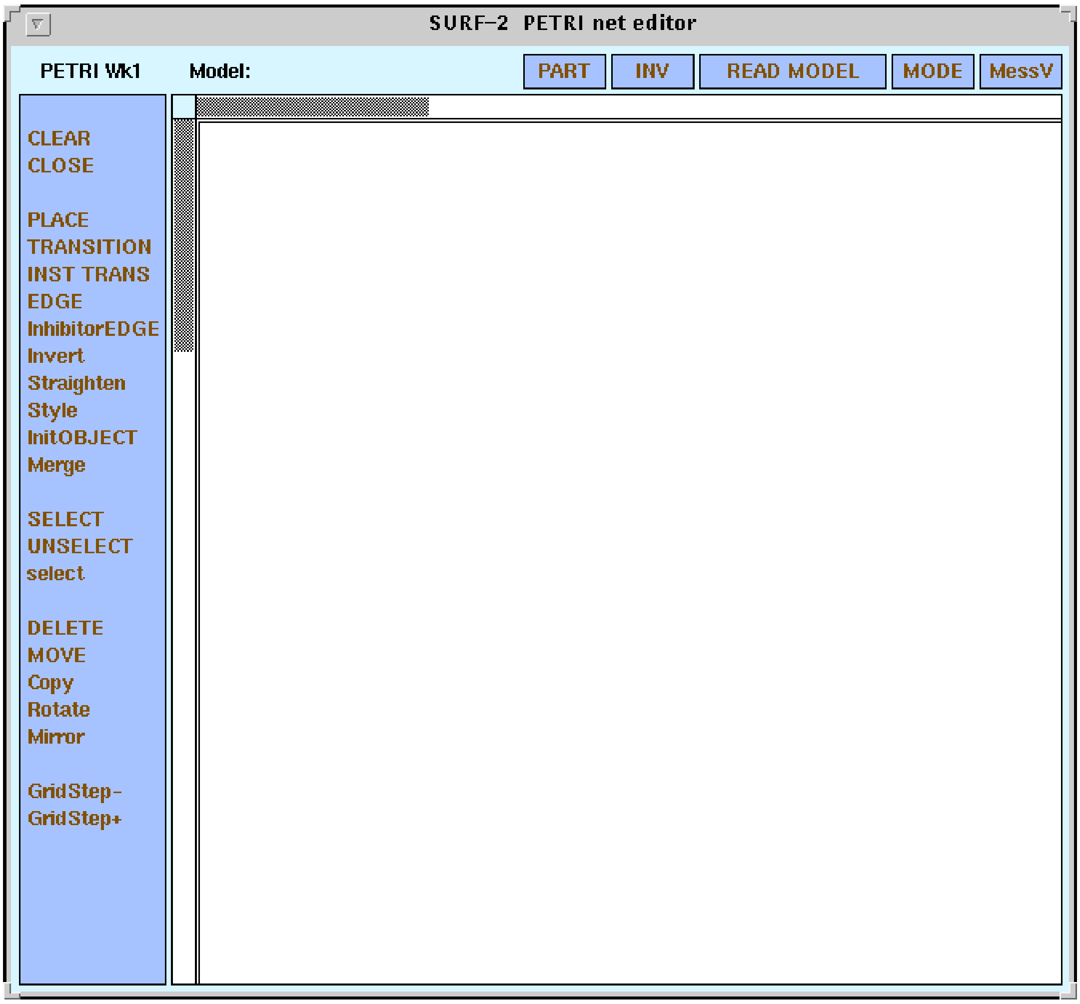


Fig 6.30 Petri net Editor working window.

The type of model (PETRI) and the name of the working window (Wk1 or Wk2) are indicated in the uppermost part of the window.

Edited model (whole or part) can be saved temporarily in a working window.

A selected model can be read in a working window, this model (whole or part) can be copied in the editing window while drawing a new model. It is a mean to import predefined patterns.

.Markov chains derived from Petri nets can directly be reached through this function (see also the ***CREATE LIKE*** *function).*

READmodel Command

-> (At first click) the following message is displayed:

Message | Confirm the Model read-access by second click on READmodel.

-> (After confirmation by Re-click) display Select model dialog box (Fig 6.31):

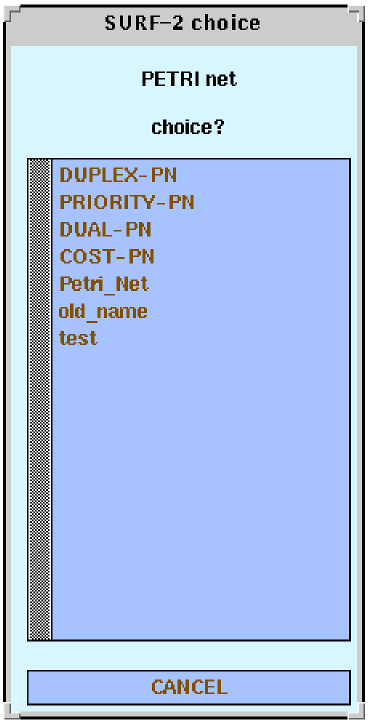


Fig 6.31 Dialog box to Select the model to be read.

Model Selection

- *Dialog box to Select the model to be read. (Fig 6.31)*

Select the name of the model to be read.

-> Reads the model.

*Clicking **CLOSE** closes the working window but no modification of the graph will be saved on disk since the EDITOR is in read-only access mode.*

Chapter 7

ASSIGNED MODEL MENU

Color: Green

7.1 PURPOSE

ASSIGNED MODEL MENU allows to manage Assigned Models in the SURF2 data base and give access to ASSIGNED Models Editor.

7.2 ACCES.

ASSIGNED MODEL Command of MAIN MENU provides access to ASSIGNED MODEL MENU.

Fig 7.1 shows both aspects of ASSIGNED MODEL MENU for MARKOV model or PETRI model.

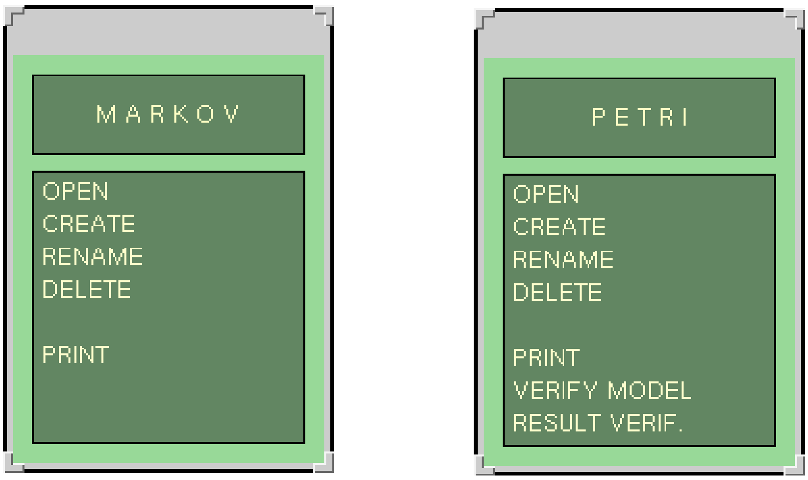


Fig 7.1 **Assigned Model menu**.

7.3 DESCRIPTION.

> *An assigned model is always associated with one model only, therefore to select an Assigned Model the user must previously select a Model.*

MARKOV / PETRI Switch

- Assigned Model menu. (Fig 7.1)

-> Switches between MARKOV or PETRI modelization: the initial value is PETRI.

OPEN Command

- Assigned Model menu. (Fig 7.1)

-> Displays Dialog box to select a model for an Assigned Model opening (Fig 7.2). The model type depends on the switch state: MARKOV or PETRI.

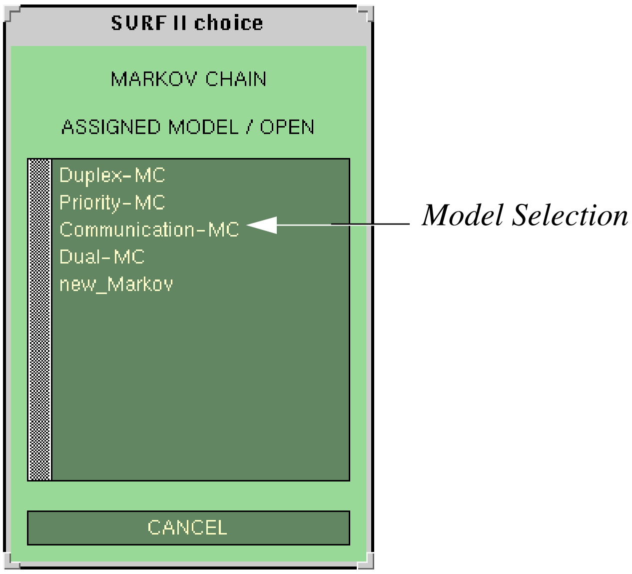


Fig 7.2 Dialog box to select a model for an Assigned Model opening

Model Selection

- Dialog box to select a model for an Assigned Model opening (Fig 7.2)

Choose a Model in the list of available Models (Markov chain or Petri net).

-> The dialog box to select an Assigned Model for the Model selected appears on the screen (Fig 7.3).

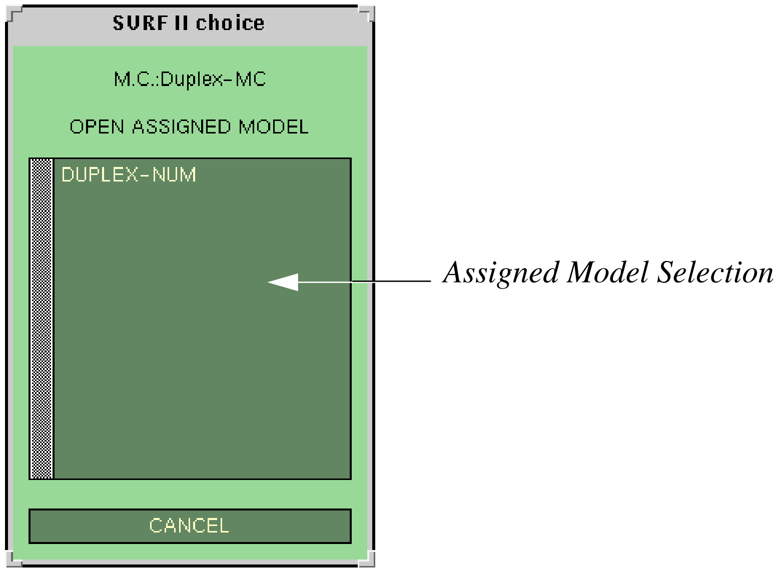


Fig 7.3 Dialog box to select an Assigned Model.

Assigned Model Selection

- Dialog box to select an Assigned Model. (Fig 7.3)

Choose an Assigned Model in the list of available Assigned Models.

-> The Assigned Model editor appears on the screen. See chapter 8 to use Assigned Model Editor

CREATE Command

- Assigned Model menu. (Fig 7.1)

-> Displays dialog box to select a Model (Fig 7.4)

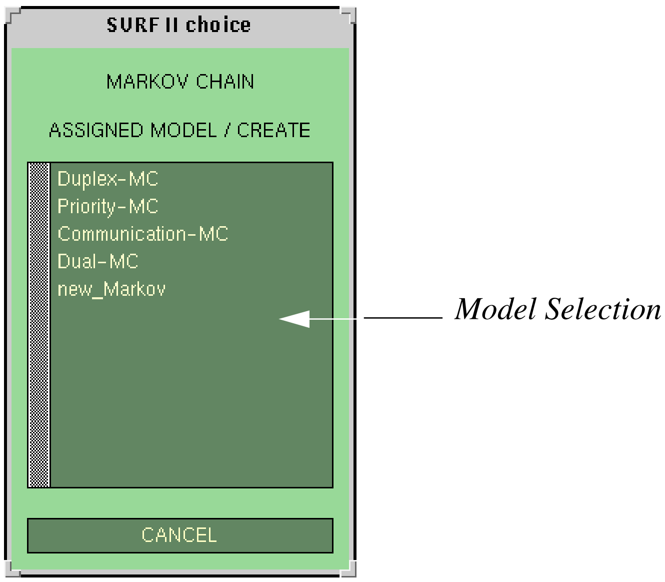


Fig 7.4 Dialog box to select a model for an Assigned Model creation.

Model Selection

- Dialog box to select a model for an Assigned Model creation. (Fig 7.4)

Choose a Model in the list of available Models (Markov chain or Petri net).

-> Displays dialog box to Assigned Model creation (Fig 7.5).

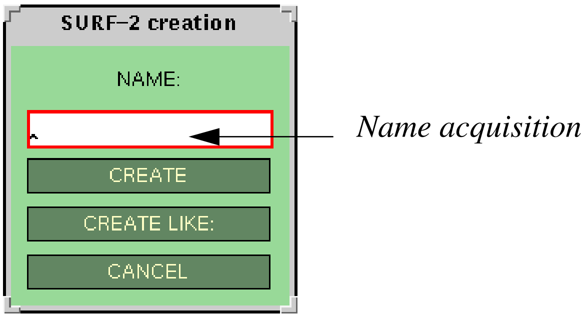


Fig 7.5 Dialog box to Assigned Model creation.

Name Acquisition

- Dialog box to Assigned Model creation. (Fig 7.5)

Enter the name of the new Assigned Model.

> *Only single names are authorized for a given Assigned Model, irrespective of the Model to which it refers.*

CREATE Command

- Dialog box to Assigned Model creation. (Fig 7.5)

-> Creates a new Assigned Model. The Assigned Model Editor appears on the screen. See chapter 8 to use Assigned Model Editor

CREATE LIKE Command

- Dialog box to Assigned Model creation. (Fig 7.5)

-> Displays dialog box to Select the Assigned Model to be duplicated (Fig 7.6).

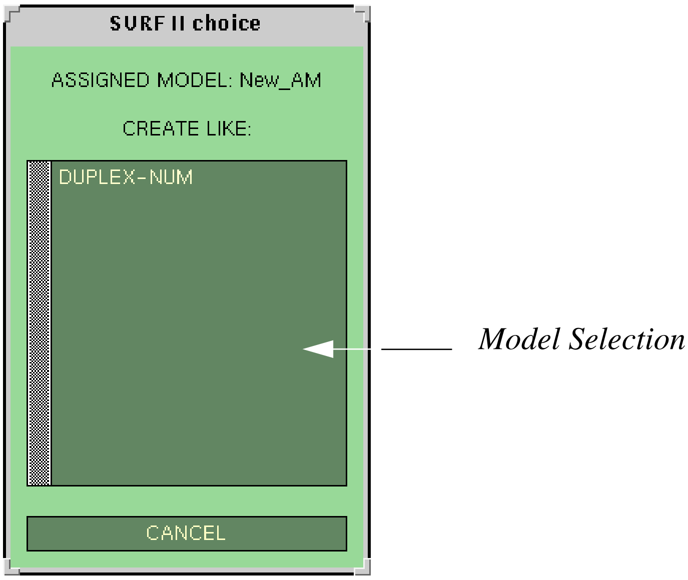


Fig 7.6 Dialog box to Select the Assigned Model to be duplicated.

Assigned Model Selection

- *Dialog box to Select the Assigned Model to be duplicated. (Fig 7.6)*

Choose the name of the Assigned Model that will be replicated in the list of existing Assigned Models.

-> Creates a copy and open the Assigned Model Editor on this copy. See chapter 8 to use Assigned Model Editor

Report (CREATE, CREATE LIKE).

Message	**GIVE THE NAME TO THE NEW ASSIGNED MODEL**
Message	**Error: name already exists**
Message	**NAMES WITH A '/' ARE NOT ALLOWED**

RENAME Command

- *Assigned Model menu. (Fig 7.1)*

-> Displays Dialog box to select a Model (Fig 7.7).

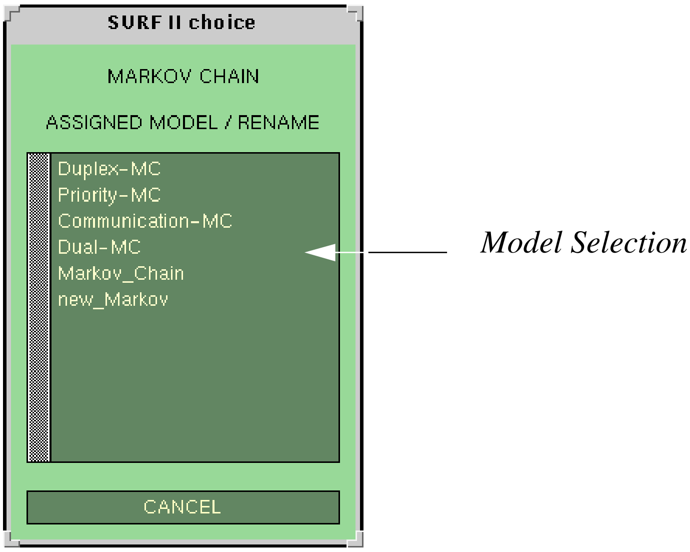


Fig 7.7 Dialog box to select a Model for Assigned Model renaming.

Model Selection

- *Dialog box to select a Model for Assigned Model renaming. (Fig 7.7)*

Choose a Model in the list of available Models (Markov chain or Petri net).

-> The dialog box to select an Assigned Model for the Model selected appears on the screen (Fig 7.8).

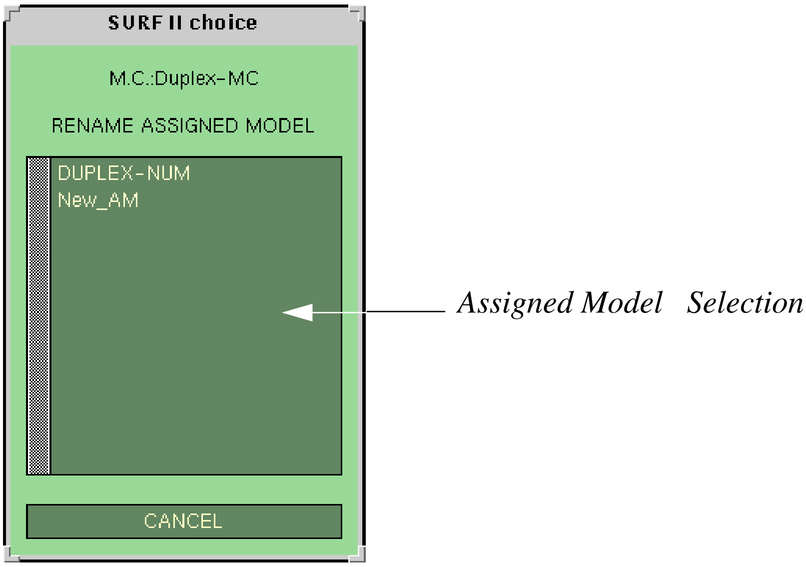


Fig 7.8 Dialog box to select an Assigned Model for renaming.

Assigned Model Selection

- *Dialog box to select an Assigned Model for renaming. (Fig 7.8)*

Choose an Assigned Model in the list of available Assigned Models.

-> Displays dialog box to Assigned Model renaming (Fig 7.9).

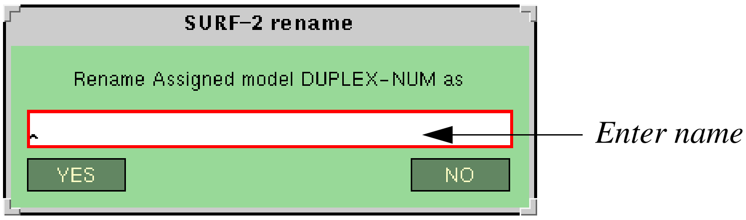


Fig 7.9 Dialog box to Assigned Model renaming

Name Acquisition

- Dialog box to Assigned Model renaming (Fig 7.9)

Enter the new name of the Assigned Model.

> *Only single names are authorized for a given Assigned Model, irrespective of the Model to which it refers.*

YES Command

- Dialog box to Assigned Model renaming (Fig 7.9)

-> Renames the Assigned Model.

NO Command

- Dialog box to Assigned Model renaming (Fig 7.9)

->Rename dialog box disappears. Assigned Model name is not modified.

DELETE Command

- Dialog box to Select the model to the Assigned Model deletion. (Fig 7.10) (-)

-> Displays dialog box to select the model for the Assigned Model deletion (Fig 7.10).

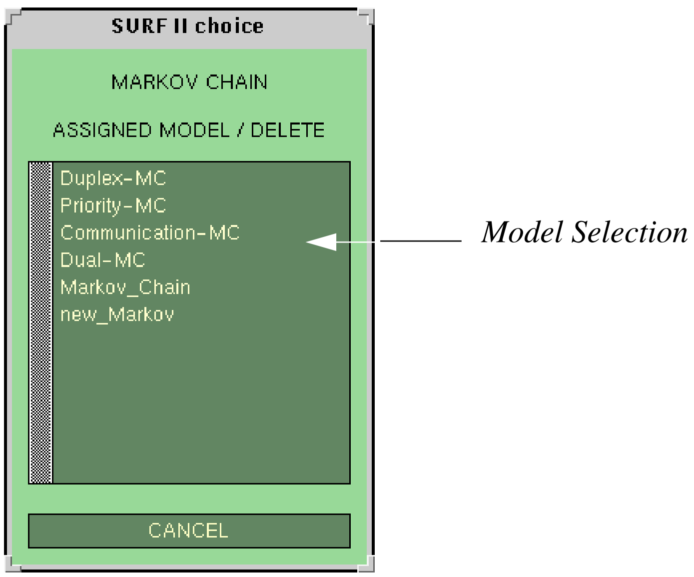


Fig 7.10 Dialog box to Select the model to the Assigned Model deletion.

Model Selection

- *Dialog box to Select the model to the Assigned Model deletion. (Fig 7.10)*

Choose a Model in the list of available Models (Markov chain or Petri net).

-> The dialog box to select an Assigned Model for the Model selected appears on the screen (Fig 7.11).

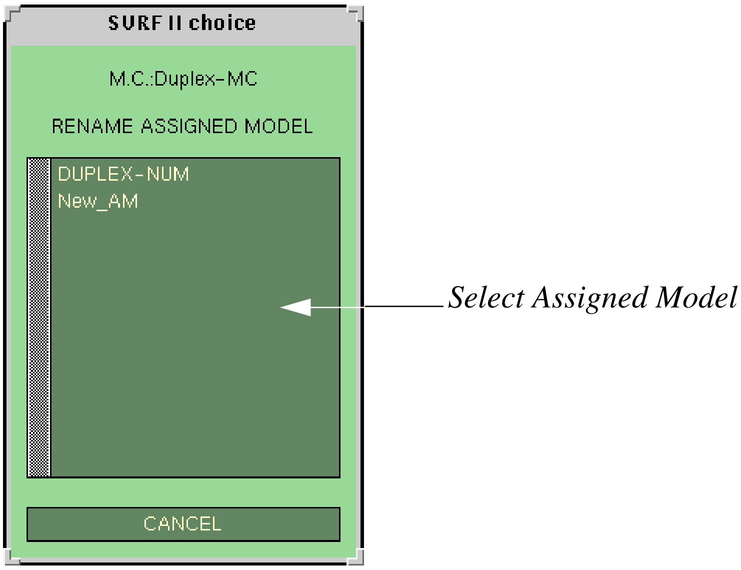


Fig 7.11 Dialog box to select an Assigned Model for deletion.

Assigned Model Selection

- *Dialog box to Select the model to the Assigned Model deletion. (Fig 7.10)*

Choose an Assigned Model in the list of available Assigned Models.

-> Displays dialog box to confirm Assigned Model deletion (Fig 7.12).

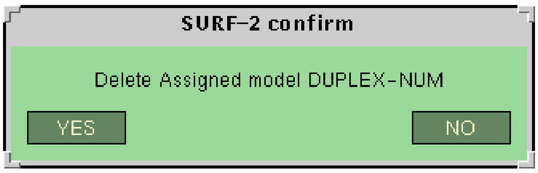


Fig 7.12 Dialog box to confirm Assigned Model deletion

YES Command

- *Dialog box to confirm Assigned Model deletion (Fig 7.12)*

-> Dialog box disappears and Assigned Model is deleted.

Report

After Assigned Model deletion the following message is displayed:

Message | **DELETING assigned model: . . . TERMINATED**

NO Command

- *Dialog box to confirm Assigned Model deletion (Fig 7.12)*

-> Dialog box disappears and Assigned Model is not deleted.

PRINT Command

- *Assigned Model menu. (Fig 7.1)*

The environment variable SURF2_IMP specifies the printer to use. If "SURF2_IMP" is not defined a PostScript file is generated in the home directory. His name is automatically generated as follows: "ETUDE__name_of_model.ps"

-> Displays dialog box to select the model for Assigned Model printing (Fig 7.13).

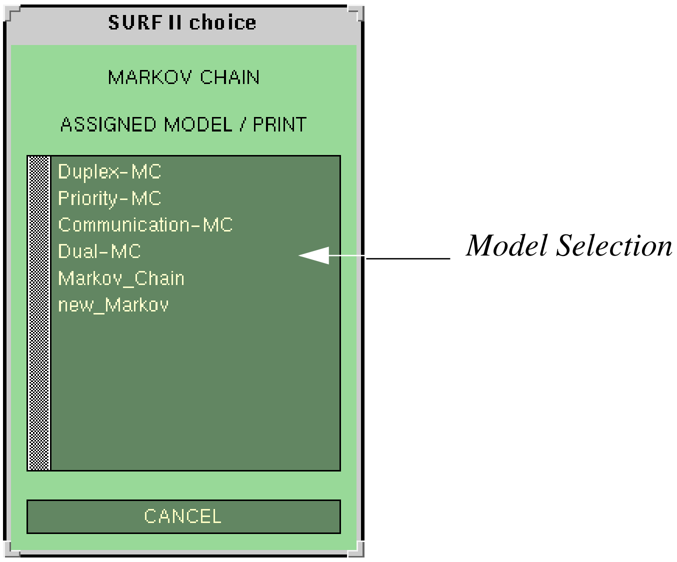


Fig 7.13 Dialog box to Select the model for Assigned Model printing.

Model Selection

- *Dialog box to Select the model for Assigned Model printing. (Fig 7.13)*

Select the name of the model for Assigned Model printing.

-> The dialog box to select an Assigned Model for the Model selected appears on the screen (Fig 7.14).

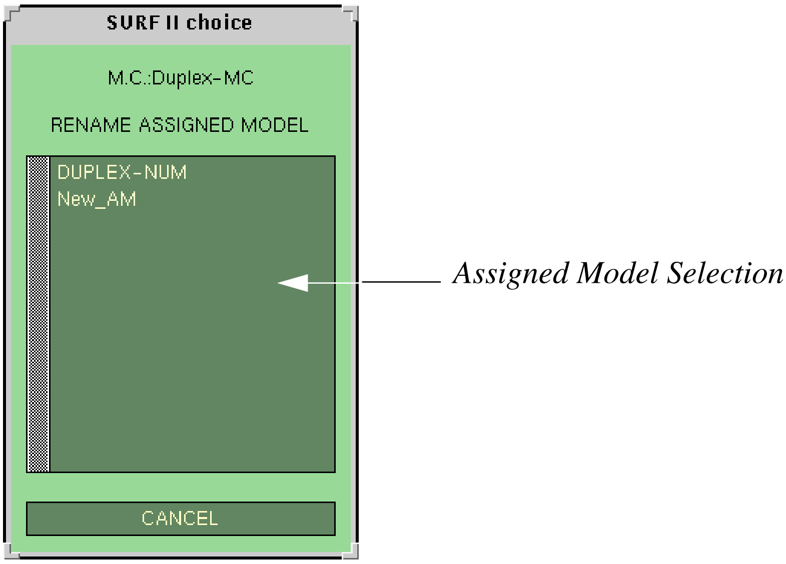


Fig 7.14 Dialog box to select an Assigned Model to print.

Assigned Model Selection

- Dialog box to select an Assigned Model to print. (Fig 7.14)

Select the name of the model for Assigned Model printing.

-> The selected Assigned Model is printed.

After printing the following message is displayed:

Message | **PRINTOUT OF ASSIGNES MODEL<...> TERMINATED**

Print is performed without confirmation.

VERIFY MODEL Command (Petri net)

- Assigned Model menu. (Fig 7.1)

-> Displays dialog box to select the model for verification (Fig 7.15).

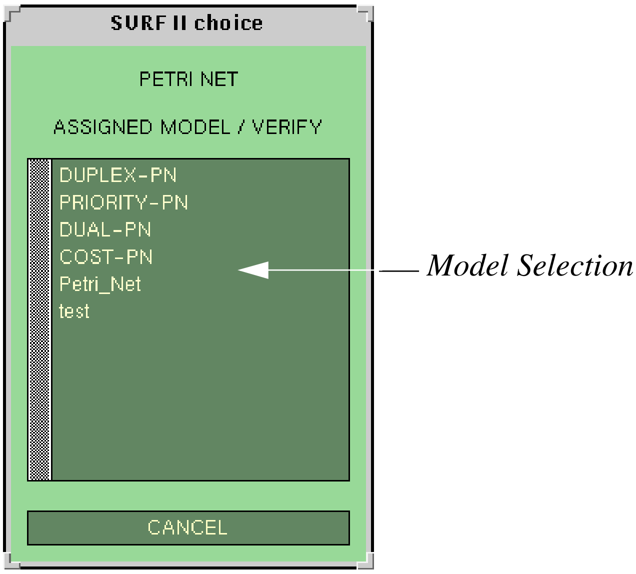


Fig 7.15 Dialog box to Select the model for verification.

Model Selection

- Dialog box to Select the model for verification. (Fig 7.15)

Select the name of the model for Assigned Model printing.

-> The dialog box to select an Assigned Model for the Model selected appears on the screen (Fig 7.16).

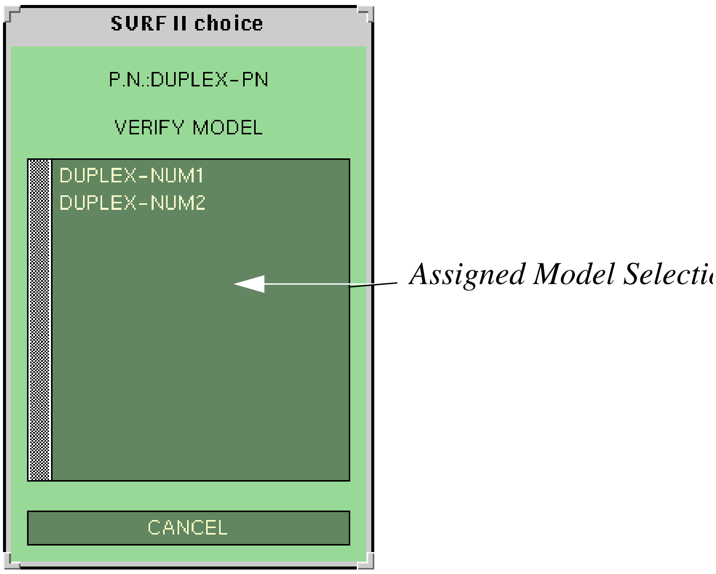


Fig 7.16 Dialog box to select an Assigned Model to verify.

Assigned Model Selection

- Dialog box to select an Assigned Model to verify. (Fig 7.16)

Select the name of the model for Assigned Model verification.

-> Processes Petri Net Verification tool, see chapter 9.

RESULT VERIFY Command (Petri net)

- Assigned Model menu. (Fig 7.1)

-> Displays dialog box to select the Model for Result Verification.

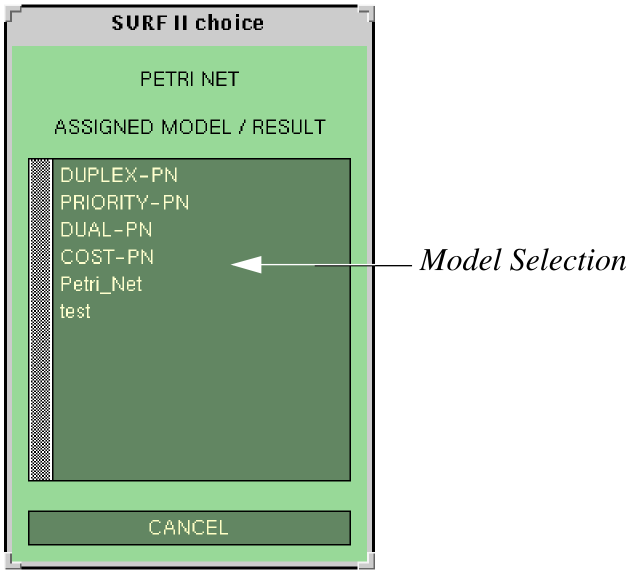


Fig 7.17 Dialog box to Select the model for result verification.

Model Selection

- Dialog box to Select the model for result verification. (Fig 7.17)

Select the name of the model for Assigned Model printing.

-> The dialog box to select an Assigned Model for the selected Model appears on the screen.

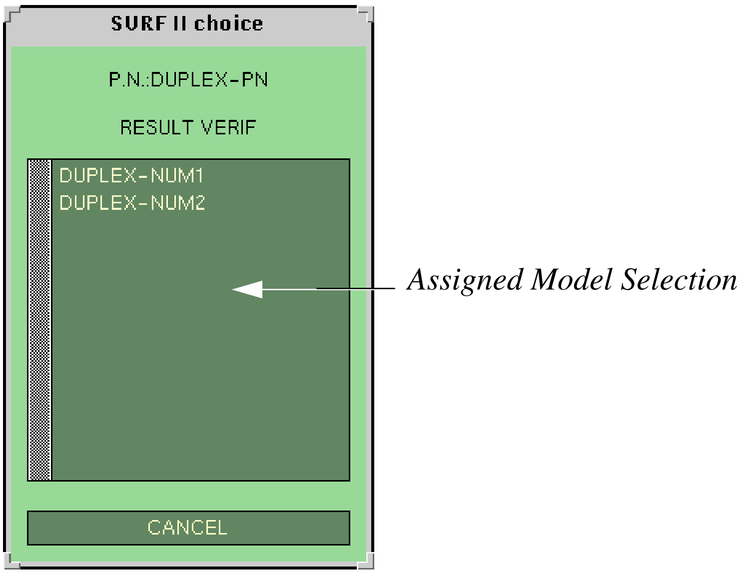


Fig 7.18 Dialog box to select an Assigned Model for result verification.

Assigned Model Selection

- Dialog box to select an Assigned Model for result verification. (Fig 7.18)

Select the name of the Assigned Model for Assigned Model printing.

-> Processes Petri Net Verification Results Viewer, chapter 10 depict this tool.

Chapter 8

ASSIGNED MODEL EDITOR

Color: yellow

8.1 PURPOSE

An Assigned Model consists of a mapping between the symbolic parameters of a Model and a set of numerical or symbolic values. An Assigned Model is defined for a Model previously created.

8.2 ACCESS

To access the Assigned Model Editor, select one of the following two commands in the ASSIGNED MODEL MENU:

- OPEN Command in the case of an existing Assigned Model, see chapter 12 page 2,
- CREATE Command if the user wishes to create a new one, see chapter 12 page 3.

8.3 ASSIGNED MODEL EDITOR

OPEN or CREATE commands of MODEL MENU give access to the window shown below:

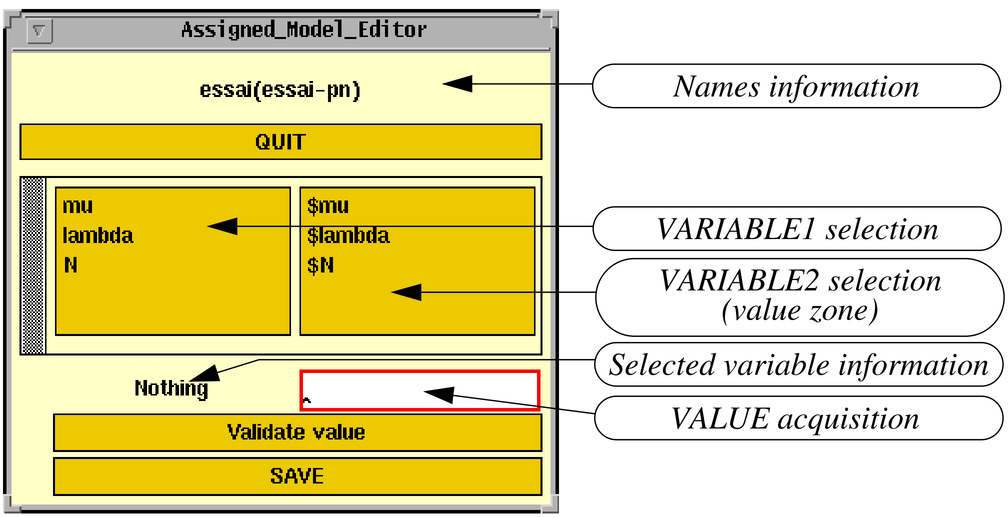


Fig 8.1 Assigned Model Editor window

Names information

- Assigned Model Editor window (Fig 8.1)

-> Displays the name of the Assigned Model and the name of the associated model put in between brackets.

(VARIABLE1) selection

- Assigned Model Editor window (Fig 8.1)

Choose a variable.

-> the name of the selected variable appear in the SELECTED VARIABLE information. Values acquisition area is not modified see Fig 8.3.

(VARIABLE2) selection

- Assigned Model Editor window (Fig 8.1)

Choose a variable (click the variable line in value zone)

-> the name of the selected variable appears in the SELECTED VARIABLE information. Values acquisition area is updated.

Selected variable information

- Assigned Model Editor window (Fig 8.1)

Gives the name of the current selected variable. As long as nothing is selected the word "nothing" is displayed.

VALUE acquisition

- Assigned Model Editor window (Fig 8.1)

Enter numerical value or global value corresponding to the selected variable. see Fig 8.4.

Validate value Command

- Assigned Model Editor window (Fig 8.1)

-> The new value is assigned to the current selected variable. The VARIABLE1 andVARIABLES2 selection area of editor are updated: the value of current variable is displayed, see Fig 8.5.

SAVE Command

- Assigned Model Editor window (Fig 8.1)

-> Saves values in SURF2 data base.

QUIT Command

- Assigned Model Editor window (Fig 8.1)

-> (if nothing is modified since last save) The Assigned Model Editor disappear.

-> (if values have been modified since last save) the following dialog box is displayed Fig 8.2.

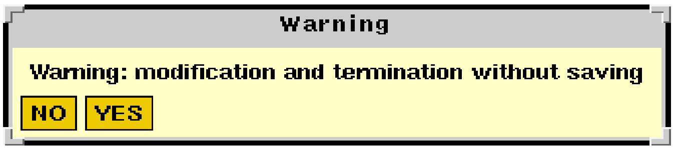


Fig 8.2 Dialog box to quit Assigned Model Editor.

No Command

- Dialog box to quit Assigned Model Editor. (Fig 8.2)

If you want to save values before quitting Assigned Model Editor.

Yes Command

- Dialog box to quit Assigned Model Editor. (Fig 8.2)

If you want to leave Assigned Model Editor without saving.

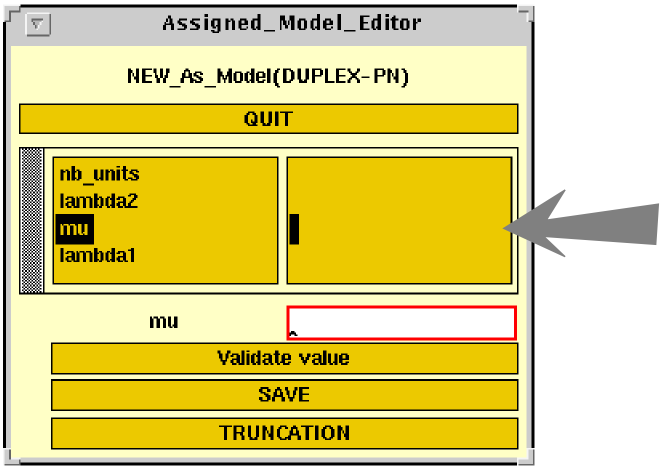


Fig 8.3 Selection of variable mu

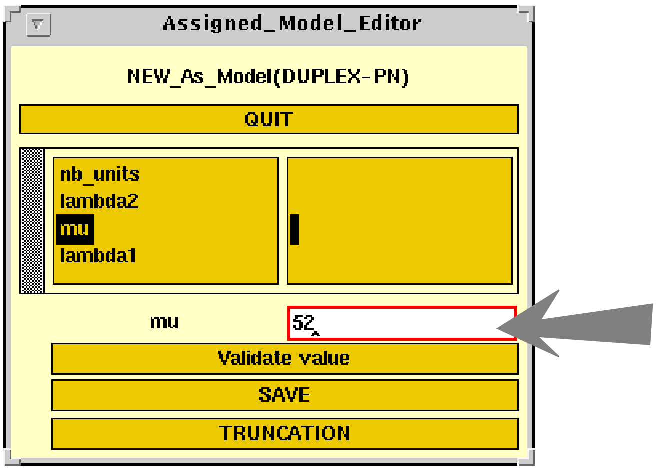


Fig 8.4 Value acquisition of variable mu

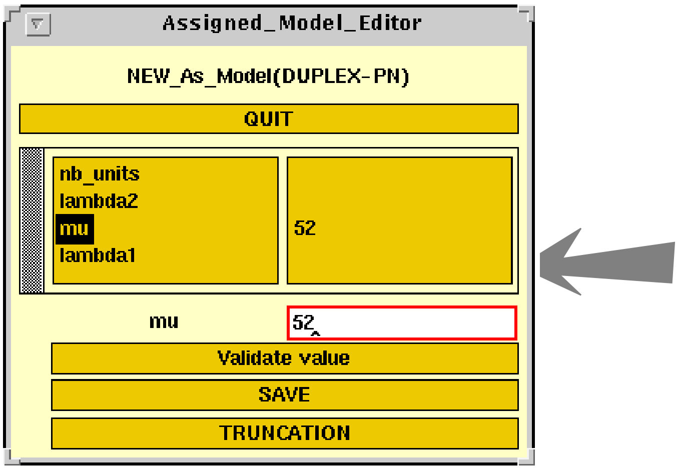


Fig 8.5 (VARIABLE,VALUE) is updated.

8.4 VALUE FORMAT

• For real-type numbers, the correct syntax is as follows:

x.xx.....xxor **xE xx...xxx**

(17 Fig.s max.) (15 Fig.s max.)

• For integer-type numbers, the correct syntax is as follows:

xx.....xx

(19 Fig.s max.)

• For global value format, the correct syntax is as follows:

$ab...yz The global value name must begin with the **$** character.

(19 characters max.)

The next figure shows an example of a naming of global value:

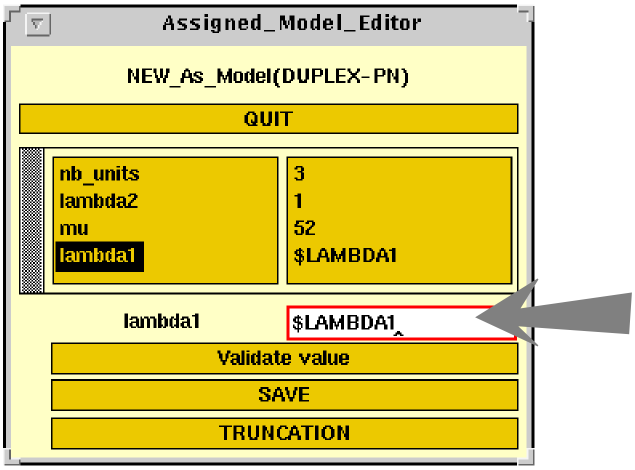


Fig 8.6 A global value is assigned to variable lambda1

Numerical assignment to global values is described in chapter 12 page 7.

Chapter 9

PETRI NET VERIFICATION

9.1 PURPOSE

This command enables you to run a structural verification (verifying such Petri net properties as boundedness, liveliness or pseudo-liveliness, as well as specific assertions on the markings). This is why there is no equivalent command in the assigned model menu when a Markov chain has first been selected in the model menu.

SURF-2 allows the user to verify Petri nets structural properties such as marking invariant and firing invariant from an invariant base automatically generated by the application. In order to perfect a Petri net, the user can also provide specific invariants which have previously been defined in the Petri net editor.

Structural verification performed on a Petri net for which a total marking invariant has been found indicates that this Petri net is bounded.

Similarly, a structural verification performed on a Petri net for which a total firing invariant has been found indicates that this Petri net is alive.

Structural verification assumes the Petri net has been dimensioned. In other words, no global value has been assigned to the Petri net dimensioning variables.

9.2 ACCESS.

Use command VERIFY MODEL of Assigned Model Menu.

9.3 DESCRIPTION.

The two following processes are run:

- Petri Net Dimensioning Process,
- Invariant Verification.

A Dial is displayed to inform the user about the process in progress during Petri net structural verification. Following indication of the correct verification end, the Dial is closed before returning to the ASSIGNED MODEL menu:

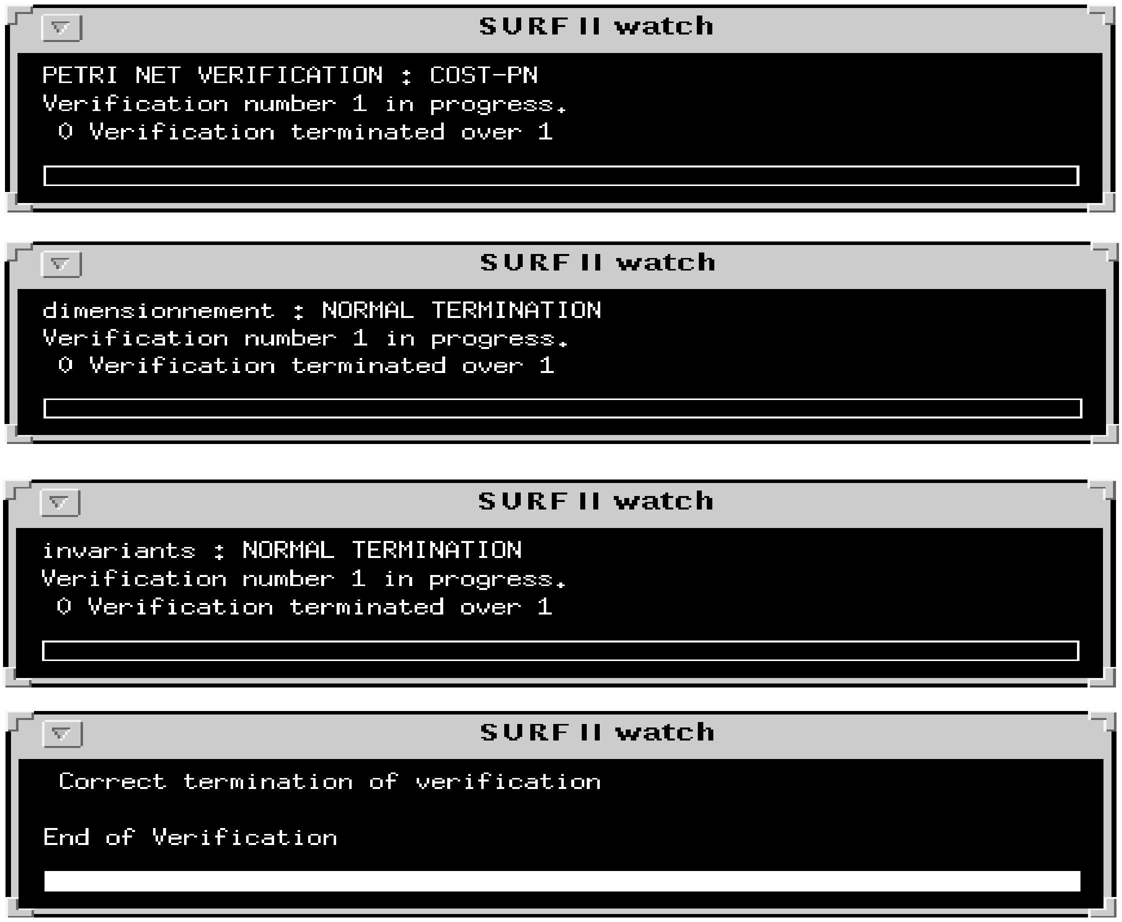


Fig 9.1 Dial displaying the information on current process during the Petri net Verification.

Chapter 10

PETRI NET VERIFICATION RESULTS VIEWER

Color: Light blue

10.1 PURPOSE

This tool allows to access results of Petri Net verification.

10.2 ACCESS

Use command RESULT VERIF of Assigned Model Menu.

10.3 DESCRIPTION

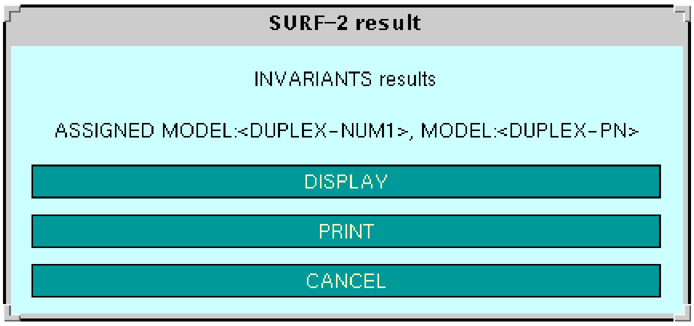


Fig 10.1 Dialog box for property verification results.

DISPLAY Command

- *Dialog box for property verification results. (Fig 10.1)*

-> Displays the following window (Fig 10.2.):

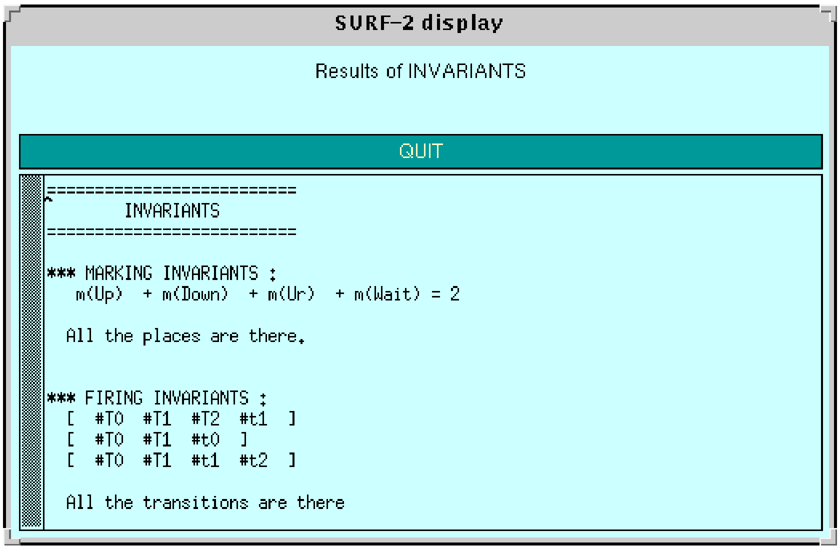


Fig 10.2 Window displaying the results.

Marking Invariants

Firing Invariants

PRINT Command

- *Dialog box for property verification results. (Fig 10.1)*

> *The environment variable SURF2_IMP specifies the printer to be used. If "SURF2_IMP" is not defined a PostScript file is generated in the home directory. Its name is automatically generated as follows: "invariants__name_of_model.ps"*

-> The property verification results are printed

After printing the following message is displayed:

Message | **PRINTOUT OF INVARIANTS TERMINATED**

> *Print is performed without confirmation.*

.

Chapter 11

MODEL FOLDER MENU

Color: Green

11.1 PURPOSE

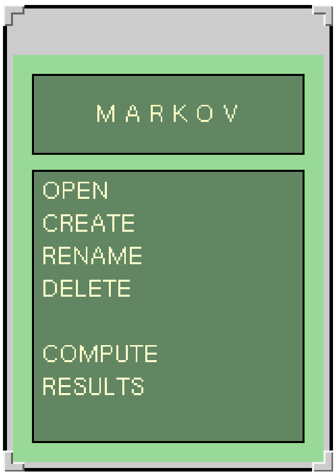
MODEL FOLDER MENU allows to manage Models Folder in the SURF2 data base and gives access to:

- Models Folder Editor,
- Results Viewer,
- Intermediate Results Viewer.

11.2 ACCESS

MODEL FOLDER *Command* of MAIN MENU provides access to MODEL FOLDER-MENU.

Fig 11.1 shows both aspects of Model Folder menu for MARKOV modelization or PETRI modelization.



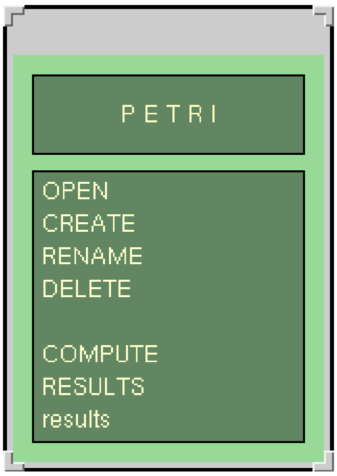


Fig 11.1 Model Folder menu.

11.3 DESCRIPTION

MARKOV / PETRI Switch

- *Model Folder menu. (Fig 11.1)*

-> Switches between MARKOV or PETRI modelization, the initial value is PETRI.

OPEN Command

- *Model Folder menu. (Fig 11.1)*

-> Displays dialog box to select the Model Folder to be opened. The model type depends on the switch state: MARKOV or PETRI.

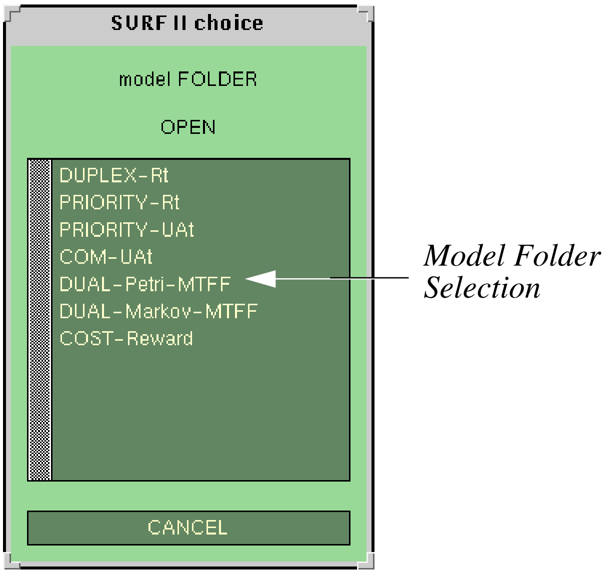


Fig 11.2 Dialog box to select the Model Folder to be opened.

Model Folder Selection

- *Dialog box to select the Model Folder to be opened. (Fig 11.2)*

Choose a Model Folder in the list of available Models Folder.

-> The Models Folder Editor appears on the screen.

See chapter 12 to use Model Folder Editor.

CREATE Command

- *Model Folder menu. (Fig 11.1)*

-> Displays dialog box to Model Folder creation (Fig 11.3):

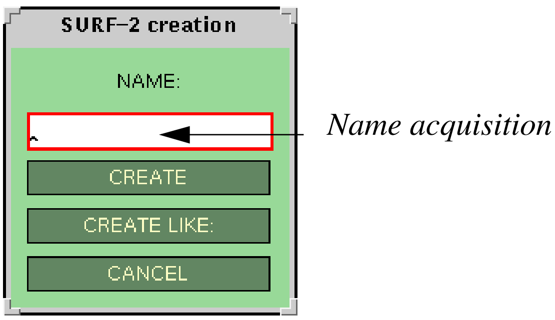


Fig 11.3 Dialog box to Model Folder creation.

Name Acquisition

- *Dialog box to Model Folder creation. (Fig 11.3)*

Enter the name of the new Model Folder

CREATE Command

- *Dialog box to Model Folder creation. (Fig 11.3)*

-> Creates a new Model and displays Model Folder Editor.

See chapter 12 to use Model Folder Editor.

CREATE LIKE Command

- *Dialog box to Model Folder creation. (Fig 11.3)*

-> Displays dialog box to Select the Model Folder to be duplicated chapter Fig 11.4.

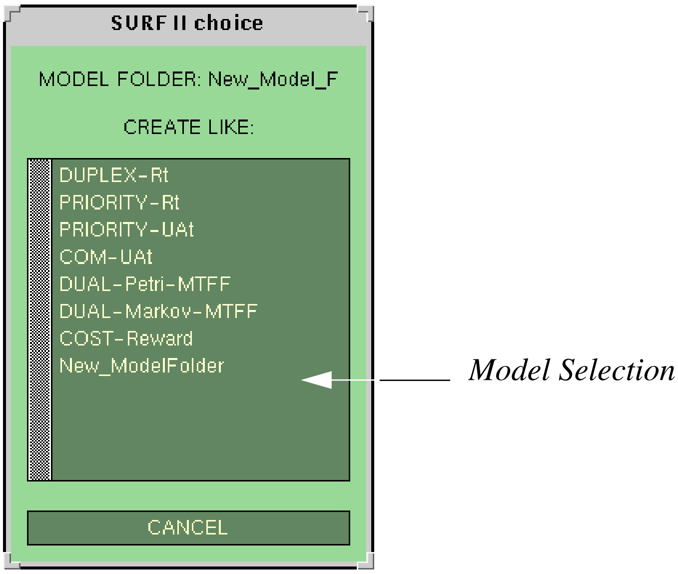


Fig 11.4 Dialog box to Select the Model Folder to be duplicated.

Model Selection

- *Dialog box to Select the Model Folder to be duplicated. (Fig 11.4)*

Choose the name of the Model Folder that will be replicated in the list of existing Models Folders.

-> Creates a new Model folder by duplication and displays Models Folders Editor.

See chapter 12 to use Model Folder Editor.

Report (CREATE, CREATE LIKE).

Message |	**GIVE THE NAME TO THE NEW MODEL FOLDER**
Message |	**Error: name already exists**
Message |	**NAMES WITH A '/' ARE NOT ALLOWED**

RENAME Command

- Model Folder menu. (Fig 11.1)

-> Display Dialog box to select the Model Folder to be renamed (Fig 11.5):

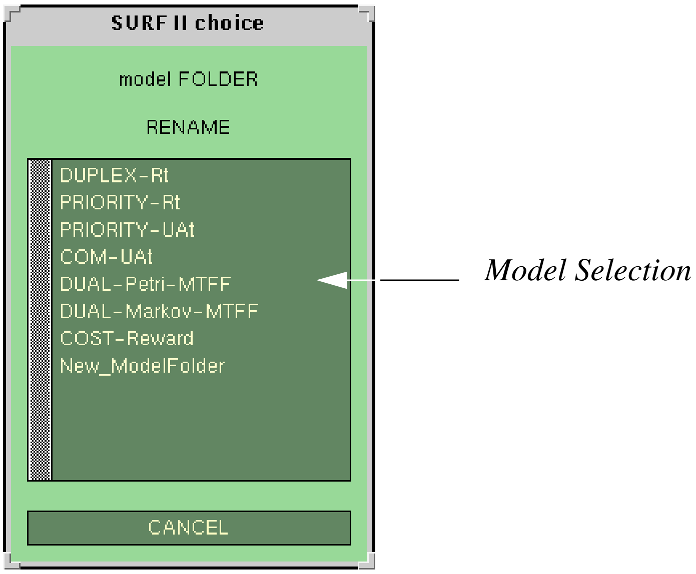


Fig 11.5 Dialog box to select the Model Folder to be renamed.

Model Selection

- Dialog box to select the Model Folder to be renamed. (Fig 11.5)

Choose a Models Folder in the list of available Models Folders.

-> Displays dialog box to Rename (Fig 11.6):

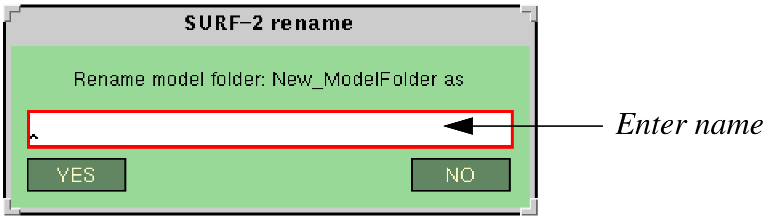


Fig 11.6 Dialog box to Rename Models Folder.

Name Acquisition

- Dialog box to Rename Models Folder. (Fig 11.6)

Enter the new name of the Model Folder

YES Command

- Dialog box to Rename Models Folder. (Fig 11.6)

-> Renames the Models Folder.

NO Command

- Dialog box to Rename Models Folder. (Fig 11.6)

-> Dialog box to rename disappears. Rename is canceled.

DELETE Command

- Model Folder menu. (Fig 11.1)

-> Display Dialog box to select the Model Folder to be deleted (Fig 11.7):

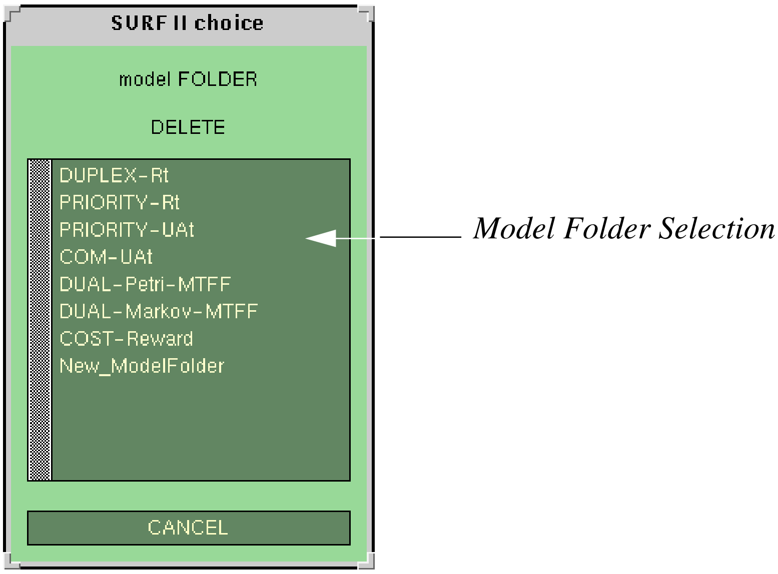


Fig 11.7 Dialog box to Select the Models Folder to be deleted.

Models Folder Selection

- Dialog box to Select the Models Folder to be deleted. (Fig 11.7)

Select the name of Model Folder to be deleted.

-> Displays dialog box to confirm Models Folder deletion (Fig 11.8):

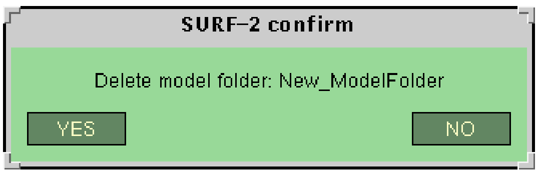


Fig 11.8 Dialog box to confirm Models Folder deletion

YES Command

- Dialog box to confirm Models Folder deletion (Fig 11.8)

-> Deletes Model Folder. Dialog box to confirm Models Folder deletion disappear.

After Model Folder deletion the following message is displayed:

Message	**DELETING model folder: . . . TERMINATED**

NO Command

- Dialog box to confirm Models Folder deletion (Fig 11.8)

-> Dialog box to confirm Model Folder deletion disappear, deletion is canceled.

COMPUTE Command

- *Model Folder menu. (Fig 11.1)*

-> the dialog box to select a Model Folder in order to compute a measure appears (Fig 11.9):

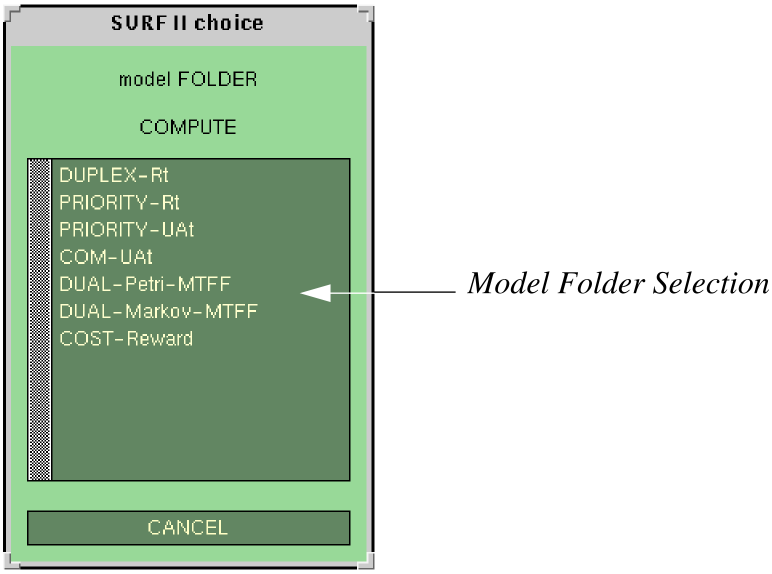


Fig 11.9 Dialog box to select a model folder in order to compute a measure.

Model Folder Selection

Select the model folder.

-> The computation process starts. See chapter 14.

As soon as a computation is executed the **COMPUTE** command is substituted for the **STOP COMPUTATION** command in the MODEL FOLDER menu (see Fig 11.10). With this facility, the user can abort a computation in progress. However, for consistency, a computation cannot be halted as long as a whole computation cycle has not been completed.

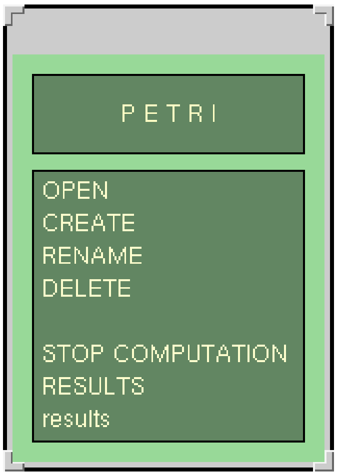


Fig 11.10 Modified Model Folder menu.

STOP COMPUTATION Command

- *Modified Model Folder menu. (Fig 11.10)*

-> Displays dialog box to confirm computation halt (Fig 11.11):

Fig 11.11 Dialog box for stop confirmation.

Stop CONFIRMED Command

- *Dialog box for stop confirmation. (Fig 11.11)*

Stop CANCELLED Command

- *Dialog box for stop confirmation. (Fig 11.11)*

-> The following message is displayed:

Message | **The computation will be terminated at the end of the running process.**

As soon as the Computation reaches the end of the process under way:

-> The computation stops. The **STOP COMPUTATION** command is substituted for the **COMPUTE** command in the MODEL FOLDER menu

RESULTS Command

- *Model Folder menu. (Fig 11.1)*

-> the dialog box to select a Model Folder in order to display results appears (Fig 11.12):

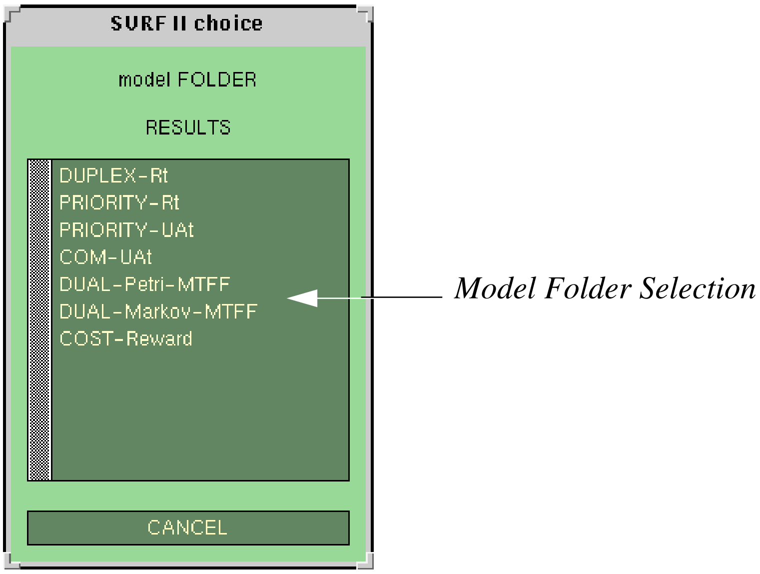


Fig 11.12 Dialog box to select a model folder in order to compute a measure.

Model Folder Selection

Select the model folder.

-> Processes Results Viewer. See chapter 15.

results Command

- Model Folder menu. (Fig 11.1)

You must access Intermediate Results just after Model Folder computation.
To avoid data base saturation, Intermediate results are destroyed:
- when you leave Surf2,
- when a new computation is done for another Model Folder.

-> the dialog box to select an Assigned Model among Assigned Model selected in the Model Folder in order to display intermediate results appears (Fig 11.13):

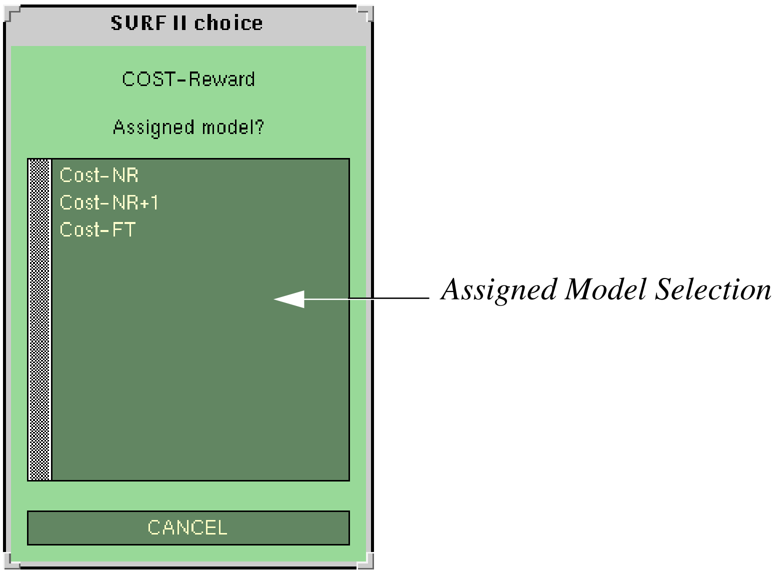


Fig 11.13 Dialog box to select a Model Folder in order to compute a measure.

Assigned Model Selection

Select an Assigned Model

->Processes Intermediate Results Viewer. See chapter 16.

Chapter 12

MODEL FOLDER EDITOR

Color: purple

12.1 PURPOSE

The Model Folder allows to define:

- the dependability measure to be evaluated,
- the computation environment,
- a list of triplets: (Assigned Model, Partition, Reward Structure).

12.2 ACCESS

To access the Assigned Model Editor, select one of the two following commands in the MODEL FOLDER MENU ()

- OPEN Command in the case of an existing Model Folder, see Chapter 11 page 2,
- CREATE Command if the user wishes to create a new one, see Chapter 11 page 3.

12.3 MODEL FOLDER EDITOR

OPEN or CREATE commands of MODEL FOLDER MENU give access to the window shown above:

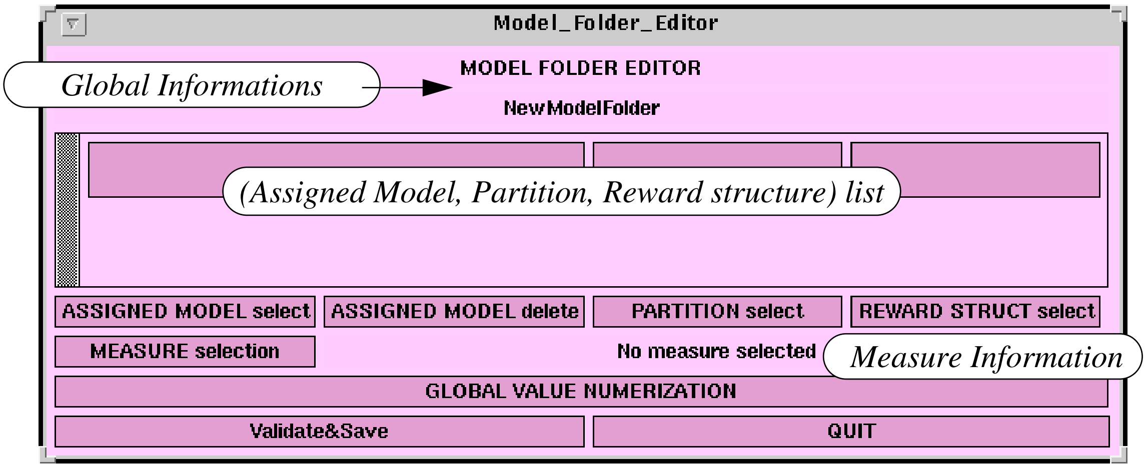


Fig 12.1 Model Folder Editor window

ASSIGNED MODEL select Command

\- *Model Folder Editor window (Fig 12.1)*

-> Displays dialog box to select an Assigned Model.

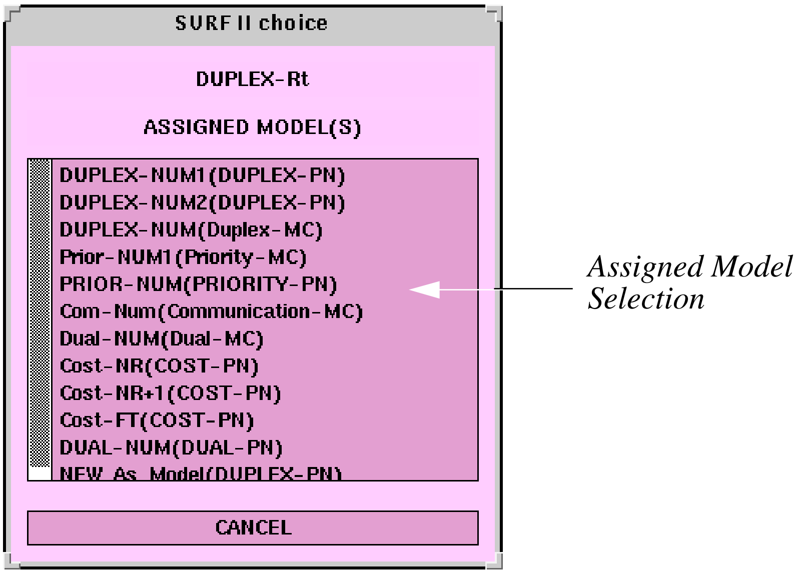


Fig 12.2 Dialog box to select an Assigned Model.

ASSIGNED MODEL Selection

\- *Dialog box to select an Assigned Model. (Fig 12.2)*

Choose an Assigned Model in the list of available Assigned Models.

-> a new entry is created in (Assigned Model, Partition, Reward structure) list.

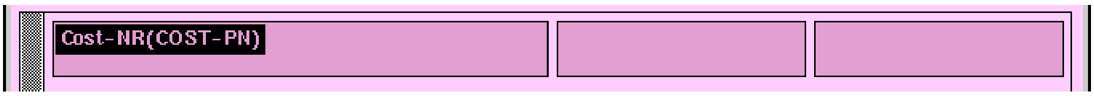


Fig 12.3 A new triplet in Assigned Model Editor.

Use following commands to complete the triplet.

PARTITION select Command

\- *Model Folder Editor window (Fig 12.1)*

-> Displays dialog box to select a Partition.

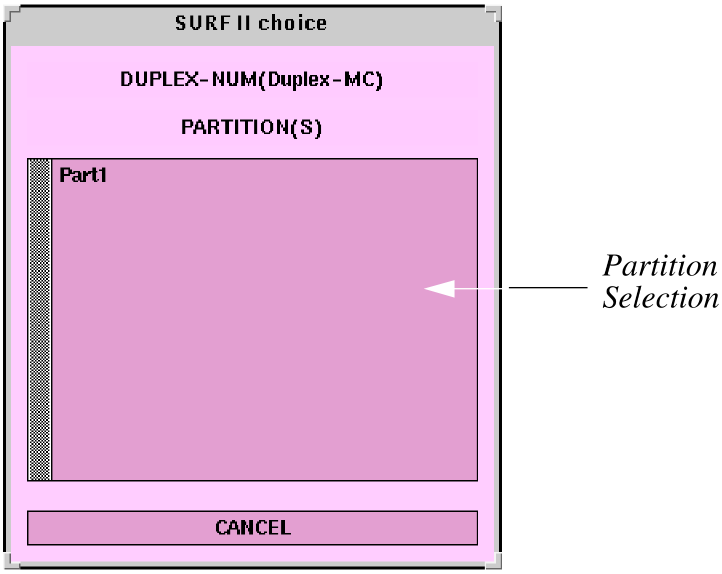


Fig 12.4 Dialog box to select a Partition.

PARTITION Selection

- Dialog box to select a Partition. (Fig 12.4)

Choose a Partition in the list of available Partitions.

-> The Partition in the current selected triplets (Assigned Model, Partition, Reward structure) is updated.

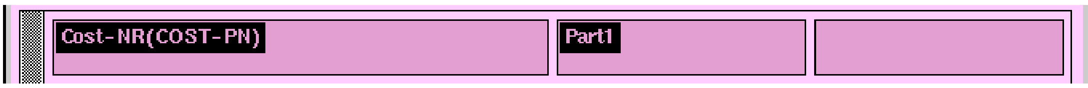


Fig 12.5 Current triplet up to date.

Use following commands to complete the triplet.

REWARD SRUCT select Command

- Model Folder Editor window (Fig 12.1)

-> Displays dialog box to select a Reward Structure.

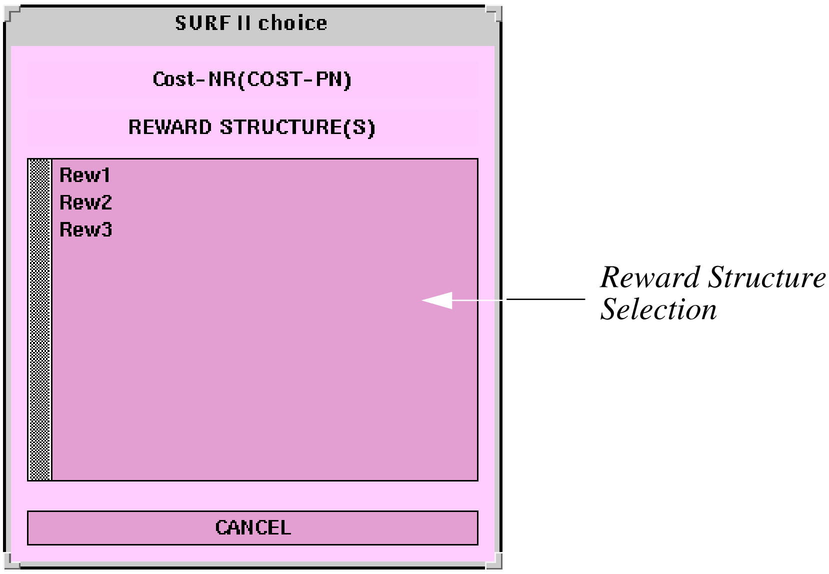


Fig 12.6 Dialog box to select a Reward Structure.

REWARD STRUCTURE Selection

- *Dialog box to select a Reward Structure. (Fig 12.6)*

Choose a Reward Structure in the list of available Reward Structures.

-> The Reward Structure in the current selected triplets (Assigned Model, Partition, Reward structure) is updated.

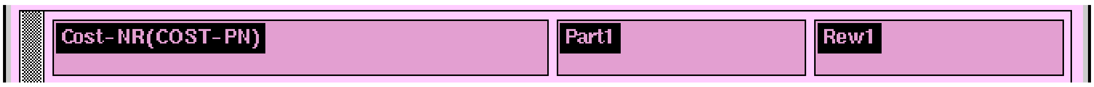


Fig 12.7 Updated current triplet.

(Assigned Model, Partition, Reward structure) Selection

- *Model Folder Editor window (Fig 12.1)*

Select a (Assigned Model, Partition, Reward structure) triplet.

-> this triplet becomes the current selected triplet:

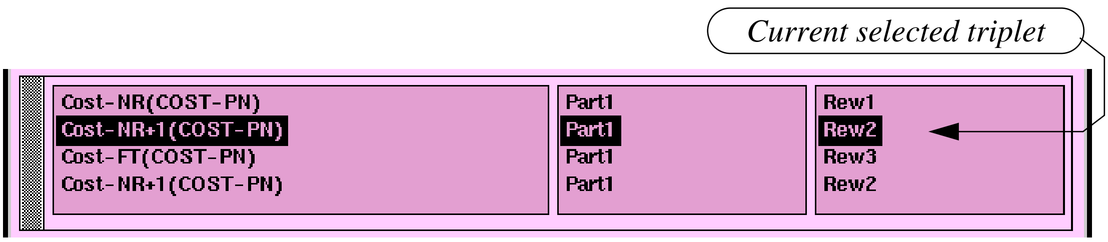


Fig 12.8 Current selected triplet in Assigned Model Editor.

ASSIGNED MODEL delete Command

- *Model Folder Editor window (Fig 12.1)*

-> Deletes the selected triplet in the list of triplets:

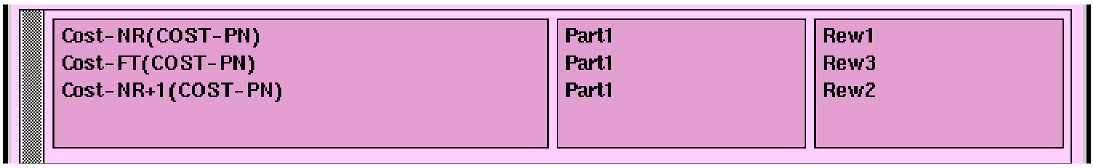


Fig 12.9 Remaining triplets after deletion.

MEASURE selection Command

- Model Folder Editor window (Fig 12.1)

-> Displays dialog box to select Measure.

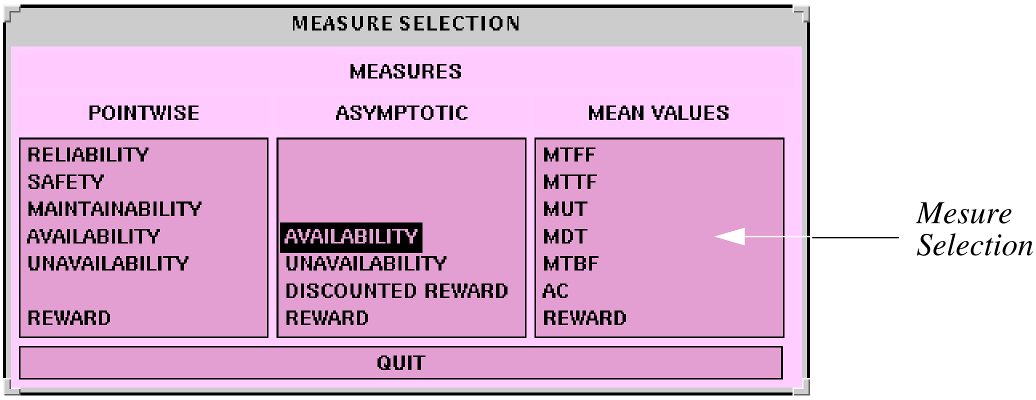


Fig 12.10 Dialog box to select a Measure.

MEASURE Selection

- Dialog box to select a Measure. (Fig 12.10)

Choose a Measure in the three lists (POINTWISE, ASYMPTOTIC, MEAN VALUES).

-> The Measure information area of the Model Folder Editor window is updated with the name of the selected measure and its type:

Fig 12.11 Updated Measure.

-> If a measure of type POINTWISE is selected the following dialog box is displayed:

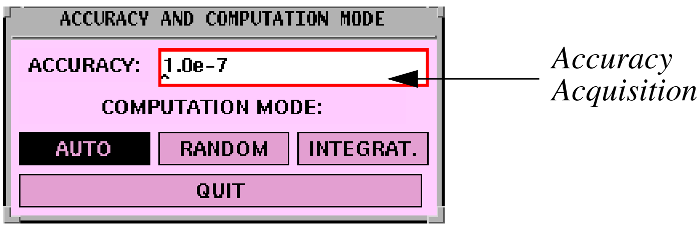


Fig 12.12 Dialog box to define accuracy and computing mode.

Selecting a pointwise measure necessitates that the computational mode to be employed during processing be selected. This is why the "computation mode" window is automatically displayed each time the user has selected the type of measure.

ACCURACY Acquisition

- *Dialog box to define accuracy and computing mode. (Fig 12.12)*

Enter accuracy.

AUTO Command

- *Dialog box to define accuracy and computing mode. (Fig 12.12)*

-> Selects AUTO mode of computation.

RANDOM Command

- *Dialog box to define accuracy and computing mode. (Fig 12.12)*

-> Selects RANDOMIZATION mode of computation.

INTEGRAT. Command

- *Dialog box to define accuracy and computing mode. (Fig 12.12)*

-> Selects INTEGRATION mode of computation.

QUIT Command

- *Dialog box to define accuracy and computing mode. (Fig 12.12)*

-> Dialog box to define accuracy and computation mode disappears.

The "automatic" computation mode AUTO *is initially selected. Evaluation of Pointwise measures is based on transient probability (or rewards) computations involving either a randomization method* RANDOM *(also called uniformization method) or a differential systems resolution* INTEGRAT .

The AUTO *method is automatically chosen according to the λt_{max} value, where λ represents the absolute value of the greatest element of the processed matrix, and* t_{max} *the maximum duration (derived from the mean time to failure or mean time to repair). λt_{max} enables the user to evaluate the stiffness of differential systems. The method of resolution of the differential systems is automatically chosen if λt_{max} is greater than 1000, otherwise the randomization method is selected. The user can compel SURF-2 to use one of the two methods with* RANDOM *or* INTEGRAT.

Accuracy of the numerical results displayed is initially set to 1.0e-7.

> *If the integration method does not work (i.e. no convergence), SURF-2. switches automatically to the other. It has to be noticed that the randomization method may not converge in some cases, computation never halting. The user must stop it. and values must then be changed before starting a new computation.*
>
> *In case of Mean and Asymptotic measures, the fast CGS method which is first used may not converge either. In this case, Gauss method is used instead. The computation is then slower.*

GLOBAL VALUE NUMERISATION Command

- Model Folder Editor window (Fig 12.1)

-> Displays dialog box to global value numerisation. Allows to assign numerical values to global values.

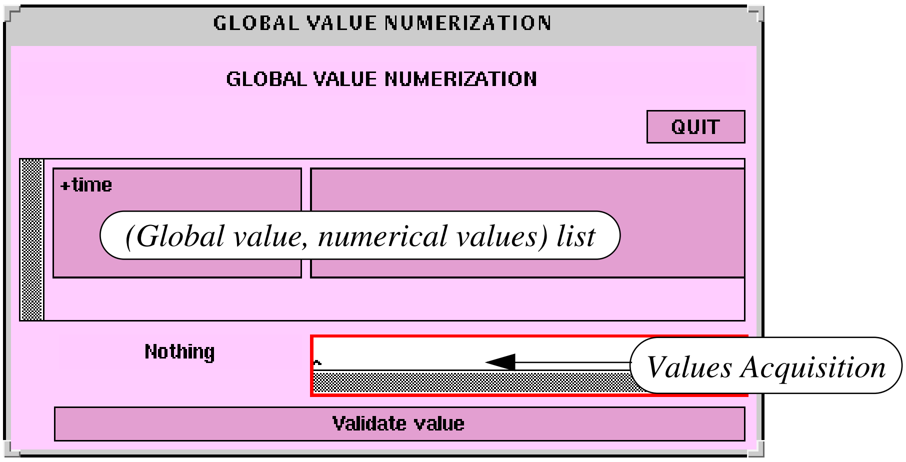


Fig 12.13 Dialog box to global value numerization.

QUIT Command

- Dialog box to global value numerization. (Fig 12.13)

-> The Dialog box to Global Values Numerization disappears.

(Global value, numerical values) Selection

- Dialog box to global value numerization. (Fig 12.13)

Choose a Global Value.

-> this Global Value is the current selected Global Value:

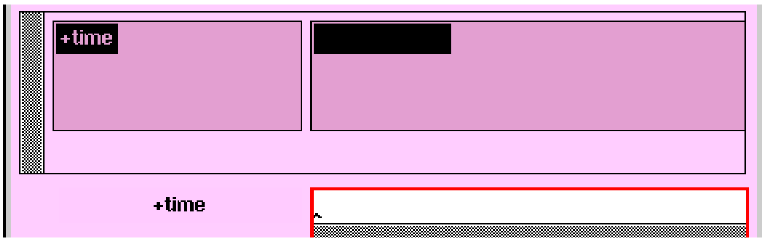


Fig 12.14 Current selected global value.

Values Acquisition

- Dialog box to global value numerization. (Fig 12.13)

Enter Numerical values.

See NUMERICAL VALUES SYNTAX page: 12.9

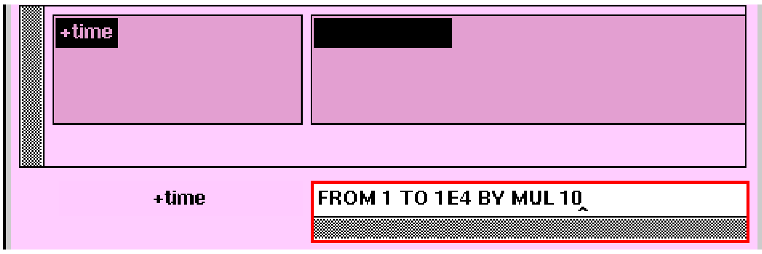


Fig 12.15 Enter Numerical value.

Validate value Command

- Dialog box to global value numerization. (Fig 12.13)

-> Assigns the Numerical Value to selected Global Values Numerization.:

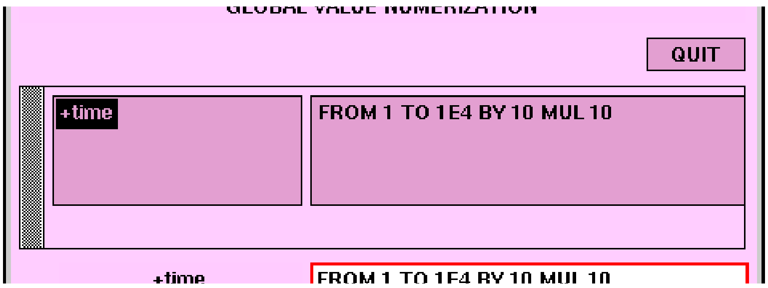


Fig 12.16 Numerical values are assigned to Global value +time.

Global values numerization must be used in the following cases:

- if global values have been defined in at least one of the assigned model selected in the model folder

- whenever a pointwise measure has been selected, even if no global value has been defined in the assigned model selected in the model folder. In this particular case, the time value is indeed considered as a global value. Note that if a reward pointwise measure has been selected, the discount rate is considered as the global value as well.

12.4 NUMERICAL VALUES SYNTAX

A set of numerical values can be defined as:

- a list of numerical values separated by a space character:

  example: 12 14 16 18 20

- An arithmetic progression

  Syntax: **from** 1st_value **to** last_value **by** increment

  the j^{th} value is obtained as follows:

  j^{th} value = 1st_value + (j-1^{th}) value X step.

  example: from 1 to 100 by 2 would give: **1 3 5** **99**

- A geometrical progression:

  Syntax: **from** 1st_value **to** last_value **by** increment <MUL> <inc.>

The step corresponds to the "ratio" of the geometrical progression and <inc.> indicates the number of intermediate values between two successive values.

example: from 1 to 100 by 10 MUL 4 would give: **1** 2 4 6 8 **10** 20 40 60 80 **100**

Indeed the geometrical progression give: 1 1×10^1 1×10^2, but in this example, this list is completed with four intermediate values at each step since <inc.> is equal to 4.

Chapter 13

MISCELLANEOUS MENU

Color: Green

13.1 PURPOSE

MISCELLANEOUS MENU allows to perform miscellaneous actions:

- checking database,
- removing old results,
- modifying environment.

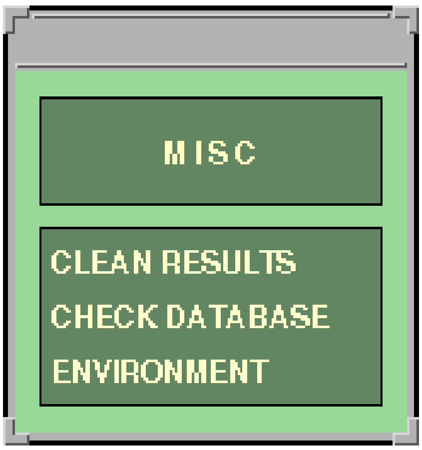


Fig 13.1 Misc menu.

13.2 CLEAN RESULTS

-> For all existing model folders, keeps only the results of the last computations. The older ones are deleted. When SURF-2 makes a new computation, old results are saved in order to let the user be able to access them. Thus, old results must be removed manually when their size become too important.

13.3 CHECK DATABASE

In case where:

- you have some problems when accessing models, assigned models or model folders, with warning messages indicating inconsistencies in the database,

- or lists are incomplete and some elements disappeared,

it is thus recommended to apply this command, which checks the links between models, assigned models and model folders and makes the whole consistent.

13.4 ENVIRONMENT

-> Allows to modify your environment variables which are set in your start-up script,

- the database path,
- the import/export directory,
- the printer name.

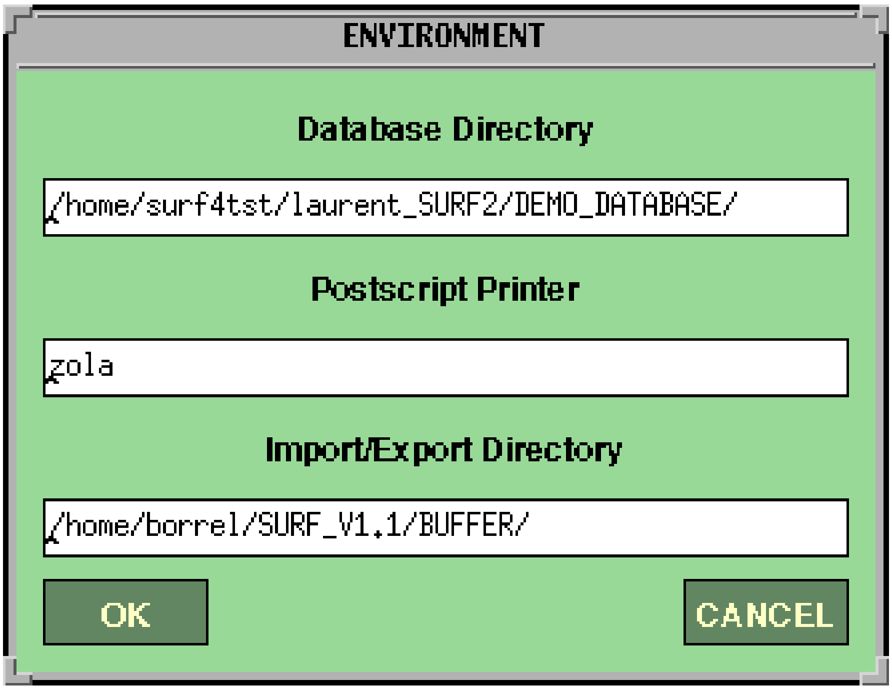


Fig 13.2 Environment window.

The database directory string must contain the full pathname or your database. You are not allowed to change it if you are doing another action in the same time.

- If you validate, the new database is checked. If it is incorrect, the dialog box reappears and you are asked to change the database name.
- If you click on CANCEL, the initial database is re-loaded.

Printer name and Import/Export directory cannot be checked. You are to be sure they are correct by yourself.

-

Chapter 14

COMPUTATION

14.1 PURPOSE

Computation processes the Dependability Measure selected in the Model Folder and stores Results in the data base.

14.2 ACCESS

COMPUTE Command of MODEL FOLDER MENU allows to run computation. See Chapter 11 page 6.

14.3 CONDITIONS TO PERFORM COMPUTATION

A computation can be performed when the following requirements have been met:

- the model folder has been modified and validated since the last computation performed on that model folder,
- the model folder contains at least one assigned model that has been modified and validated since the last computation on that model folder,
- the model folder contains at least one model that has been modified and validated since the last computation on that model folder.

A computation cannot be done if one of the following conditions is fulfilled:

- no measure has been defined,
- no assigned model with its associated partition (reward structure if computing reward measures) has been defined,
- one or several variables have not been assigned values in one or more assigned model selected in the model folder,
- the model folder contains global values that have not been assigned numerical values

If the computation cannot be done a message is displayed; for example:

Message	**ERROR: no assigned model has been defined for this model folder**

14.4 COMPUTATION CYCLES GENERATION

A computation is made up of several processes successively handled by the Supervisory task.

A dependable measure computation is composed of:

- a preliminary phase of parameter files creation used by each elementary task during one computation cycle,

- N cycles of computation, where N corresponds to the total number of Parameter Files first generated.

The generation of parameter files is handled for each Assigned Model selected in the Model Folder. If Global Values have been defined in the Model Folder, there will be as many Parameter Files as there are combinations of Numerical Values. By way of example, consider the following:

global_value_1: 10 20 70

global_value_2: 3 8.

This would generate six different Parameter Files since six possible combinations can be obtained: (10-3), (10-8), (20-3), (20-8), (70-3), (70-8).

If the Parameter File generation fails, computation will not be initiated. The application monitor will then display the following error message in the window from which the user started SURF-2.

Message | COMPUTATION REJECTED: ERROR FOUND BY THE MONITOR

The cause of the error is indicated by a message. Usually these errors are associated with the access to some files internal to the database which for some reason has been damaged.

14.5 MARKOV COMPUTATION CYCLE

If the model selected in the Model Folder is a Markov Chain elementary processes are:

- State and event-class processing,
- Partition processing
- Reward Structure processing (depending on the Measure evaluated),
- Markov Chain parameterization. Variables used in the expressions defined in the Model are substituted for their Numerical Values from the Assigned Model. The expressions are then computed (processed).
- The Dependable Measure selected in the Model Folder is then computed.

14.6 PETRI COMPUTATION CYCLE

If the model selected in the Model Folder is a Petri Net elementary processes are:

- Petri net dimensioning. The variables associated with arc weights and place markings (so-called dimensioning variables) are substituted for their numerical values,
- Computation of the reachability graph from the initial marking and Petri Net transformation in a symbolic Markov Chain,
- Symbolic Markov chain decompiling (i.e., is transformation into a graphical format readable by the Markov Chain Editor),
- Computation of firing and marking invariants,
- State and event-class processing, Partition processing and (depending on the Measure evaluated) Reward Structure processing,
- Symbolic Markov Chain parameterization. Variables used in the expressions defined in the model are substituted for their Numerical Values from the Assigned Model. The expressions are then computed (processed).
- Computation of the dependable Measure selected in the Model Folder.

Each computation cycle is optimized since the first five processes are not systematically run when several assigned models are involved in the computation. Indeed, for a given model, a computation cycle is only fully executed if at least one dimensioning variable has changed value.

14.7 COMPUTATION MESSAGES

A window informs the user of the process under way (including parameter file generation) as well as its status returned at the end of processing. The user is able to know at any time whether or not a computation cycle has been correctly completed.

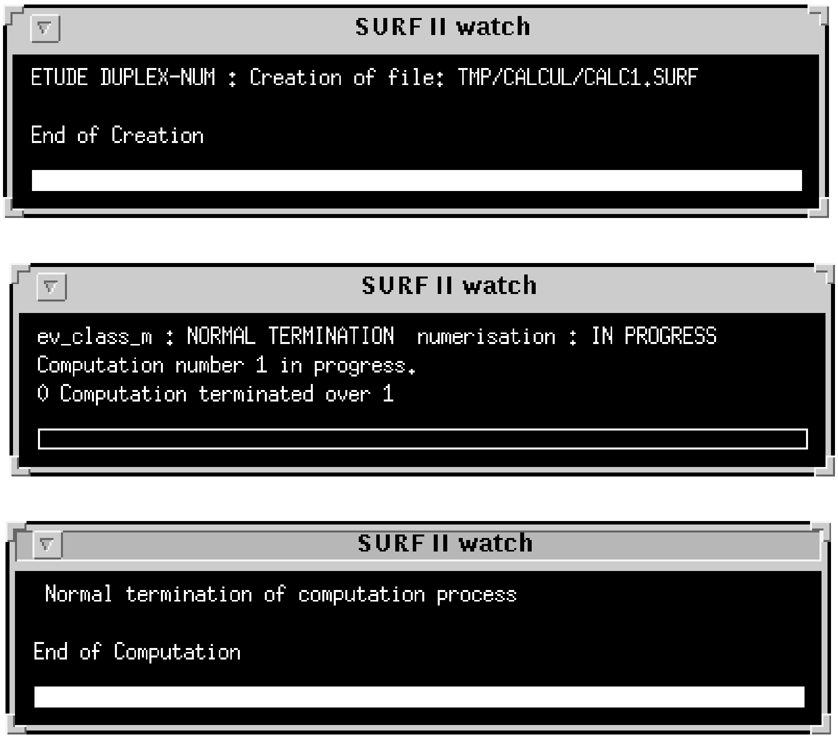


Fig 13.1 Dial displaying the process under way.

Errors detected during a computation as well as the name of the involved processing are also displayed in the Dial box. The computation is then aborted before returning to the MODEL FOLDER menu.

As soon as the computation is halted, a message stating the cause for discontinuity and giving details on the computation environment is displayed in the window from which SURF-2 was started. The information is displayed as shown in the following example:

```
-------------------------------------------
measures: No convergence
-------------------------------------------
Error occurred in computation No: 1
Assigned model:  num-PN (No: 52)
Petri net: example-of-PN (No: 78)
Model folder: Mod_Fold_PN (No: 13)
Partition No: 0
Parameter value:1.00000000e-06
Parameter value:1.00000000e-05
Parameter value:1.50000000e-04
Parameter value:1.50000000e-02
```

Fig 13.2 Exemple of computation halt message.

This information allows the user to analyze the reasons which led to the discontinuity of the computation in particular through the edition of Numerical Values assigned to the Global Values in the incriminated computation (the number of the computation in progress is edited in this message). Using the Assigned Model and the Model Folder editors (see para. 3.3.7 ASSIGNED MODEL EDITOR and para. 3.4.7 MODEL FOLDER EDITOR), the user will be able to determine those parameters that may have caused the error. The name of the Model associated with the Assigned Model (this name appears in the Model Folder Editor) together with the number of the Partition permit the user to know the Model and the State Class involved in the erroneous computation.

Chapter 15

RESULTS VIEWER

Color: Beige

15.1 PURPOSE

This tool allows Results Visualization.

15.2 ACCESS

RESULTS Command of MODEL FOLDERMENU provides access to Result Visualization. The Display Curve Window is displayed, see Fig 14.1.

15.3 DESCRIPTION

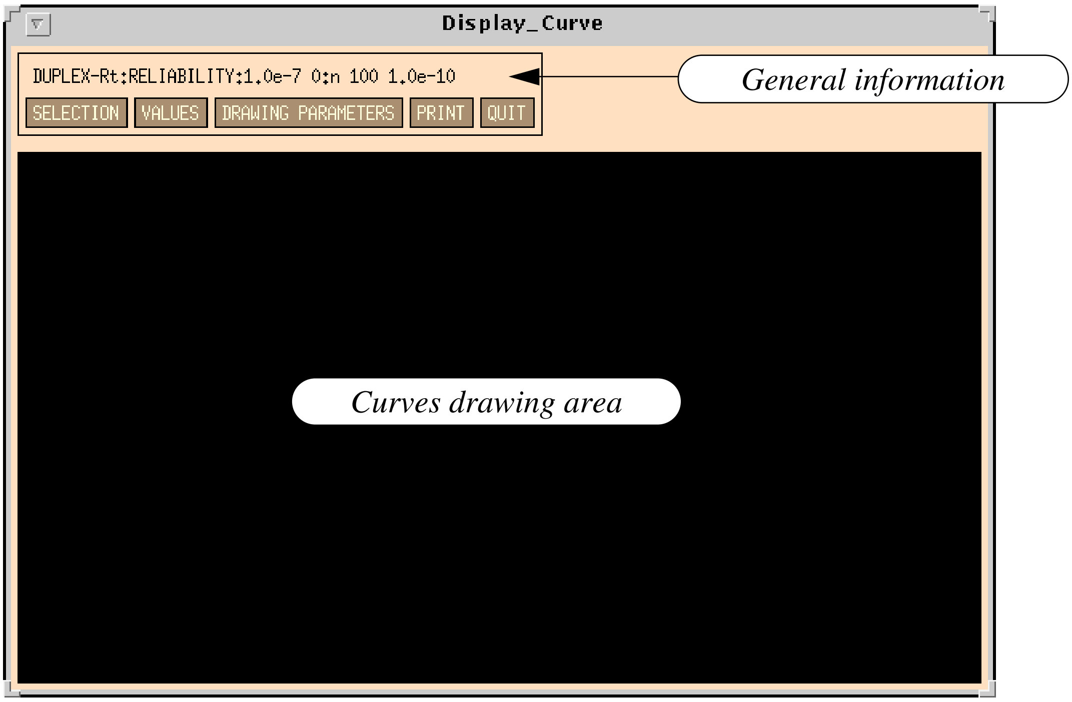


Fig 14.1 Display Curve Window.

General Information

- *Display Curve Window. (Fig 14.1)*

Displays the following information:

- the Model Folder name,
- the Measure Name,
- the accuracy of the value printouts (only for pointwise measures) initially set to 1.0e-7,
- a displaying mode (verbose or not) initially set to "**n**" (not verbose mode)
- the number of maximum iterations required for the optimization procedure used in the computations (initially set to 20).
- computation accuracy (initially set to 1.0e-15).[1]

SELECTION Command

- Display Curve Window. (Fig 14.1)

-> The Dialog box to select Assigned Model and Curves is displayed:

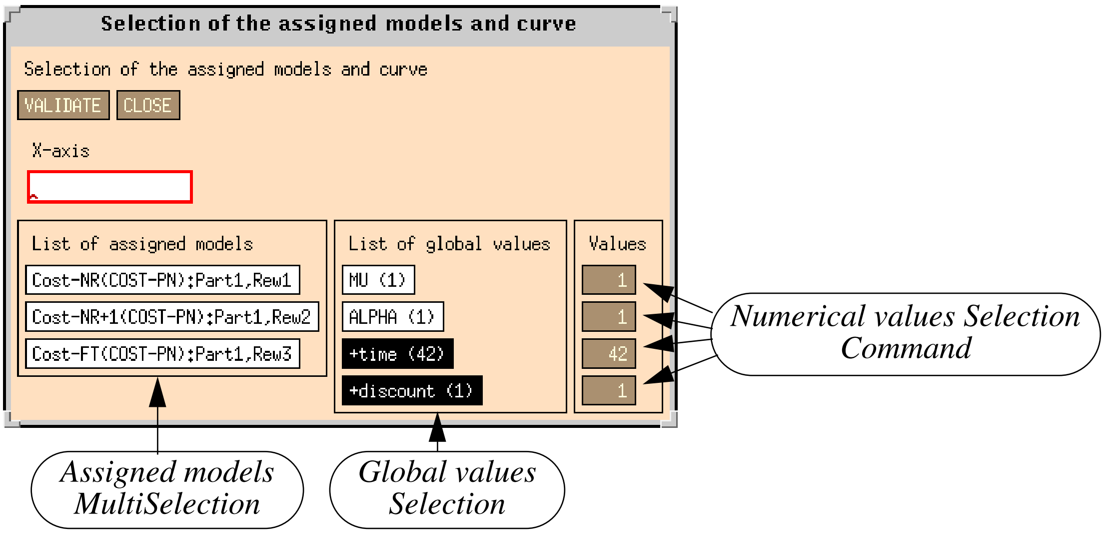


Fig 14.2 Dialog box to select Assigned Model and Curves

VALIDATE Command

- Dialog box to select Assigned Model and Curves (Fig 14.2)

-> Validates the configuration selected by the user. Curves are displayed, see Fig 14.8:

[1] These three parameters can be changed in the start-up script. They are defined in the environment variable SURF_WIZARD.

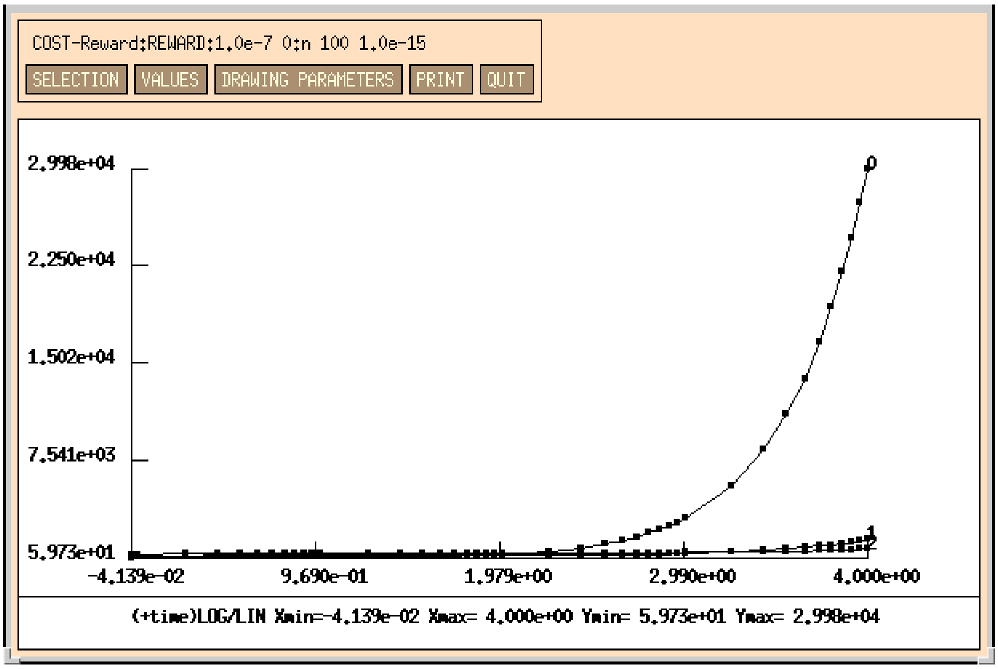


Fig 14.3 Plot of graphical results.

CLOSE Command

- *Dialog box to select Assigned Model and Curves (Fig 14.2)*

-> Closes this window.

Assigned models MultiSelection[2]

- *Dialog box to select Assigned Model and Curves (Fig 14.2)*

All the Assigned Models involved in the computations are listed. Their names are followed by the model name (between brackets) as well as the partition and reward structure names (if such options have been selected in the Model Folder). Choose one or several Assigned Models.

Global value Selection

- *Dialog box to select Assigned Model and Curves (Fig 14.2)*

All Global Values are listed. The user must select the "sensitivity variable" (X-axis values). Only values with black background can be selected. For point-wise measures, the time will be automatically chosen by the system if no other value has been selected.

The number of numerically computed values is displayed in front of each global value names.

Numerical Values Selection Commands

- *Dialog box to select Assigned Model and Curves (Fig 14.2)*

For each Global Value, on the same line, the number of Numerical values selected is displayed. Each value works like a command:

-> Displays a dialog box to select numerical values for the corresponding global value:

2. See MultiSelection definition in 2 page 4.

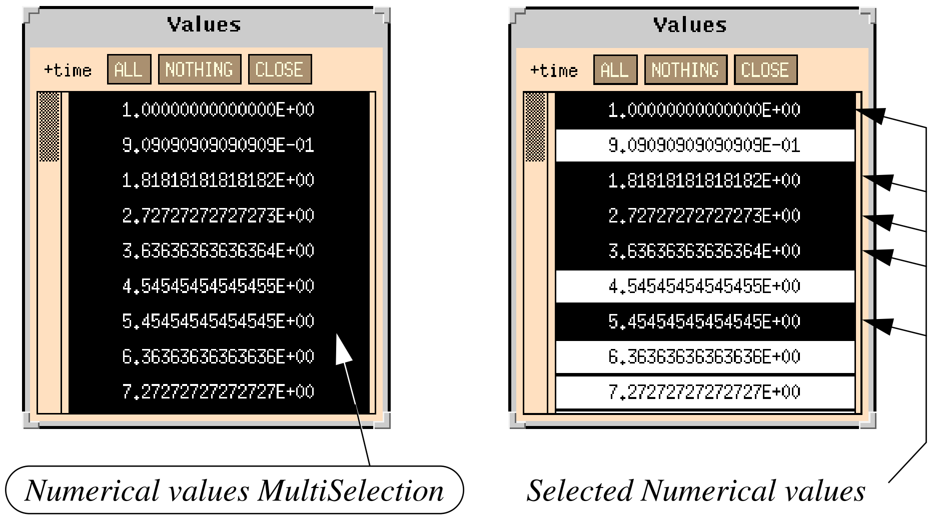


Fig 14.4 Dialog box to select numerical values of global values

Selected values are displayed on a black background. Unselected values are displayed on a white background.

ALL Commands

- Dialog box to select numerical values of global values (Fig 14.4)

-> Selects the whole numerical values therefore background is black.

NOTHING Commands

- Dialog box to select numerical values of global values (Fig 14.4)

-> Unselects the whole numerical values therefore background is white.

Numerical values MultiSelection

- Dialog box to select numerical values of global values (Fig 14.4)

Choose Numerical Values for the selected Global Value.

CLOSE Commands

- Dialog box to select numerical values of global values (Fig 14.4)

-> Dialog box to select numerical values disappears.

VALUES Command

- Display Curve Window. (Fig 14.1)

-> Displays numerical results, see Fig 14.5:

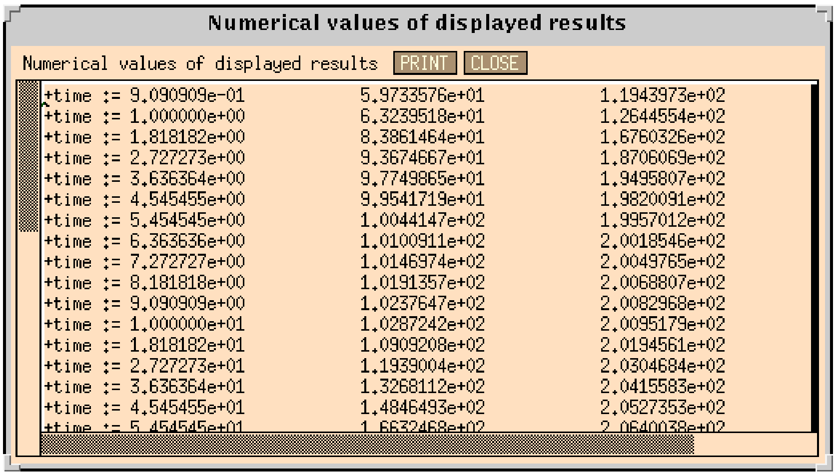


Fig 14.5 Numerical results window.

<u>DRAWING PARAMETERS</u> Command

- *Display Curve Window. (Fig 14.1)*

Display dialog box to set drawing parameters, see Fig 14.5:

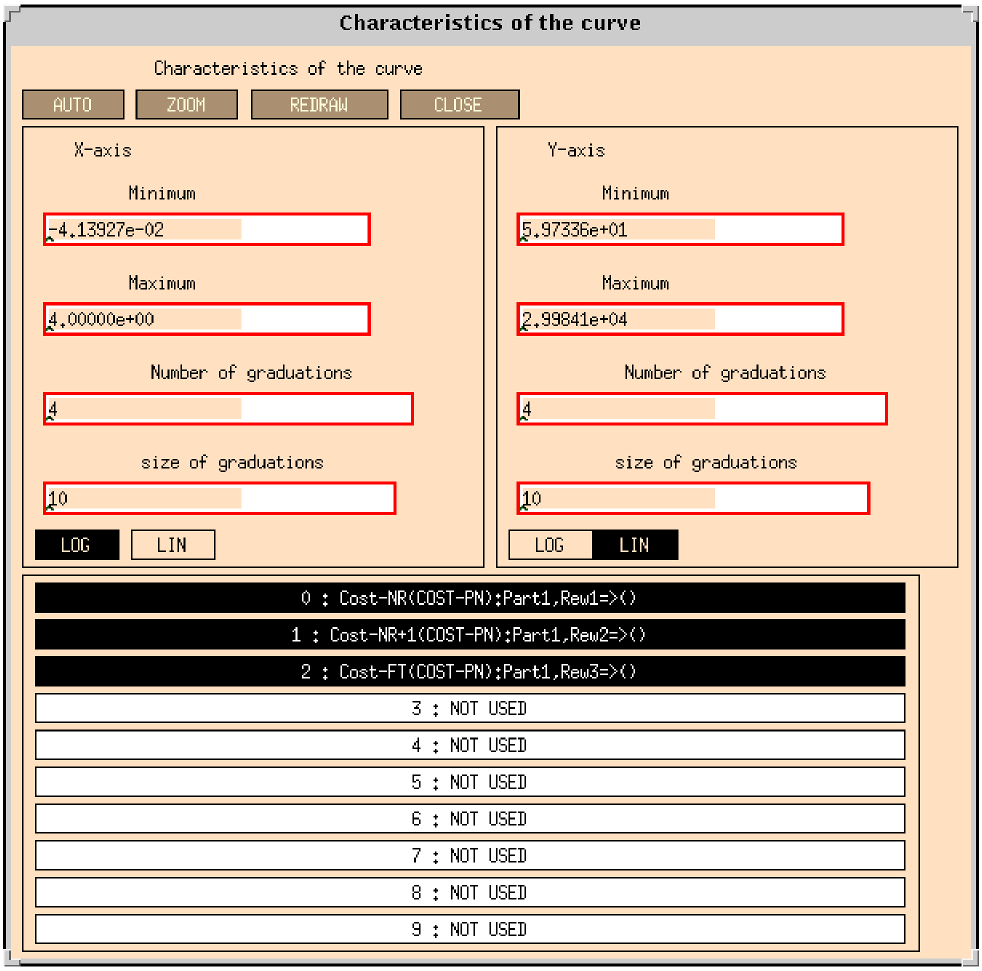


Fig 14.6 dialog box to set drawing parameters.

AUTO Command

- *dialog box to set drawing parameters. (Fig 14.6)*

-> Restores parameters to their initial values.

ZOOM Command

- *dialog box to set drawing parameters. (Fig 14.6)*

-> Allows to zoom in on an area of the plot.

Move pointer into Display Curve Window. The pointer takes the shape: ⌜

Move the cursor at the top left vertex of the area and press button the pointer takes the ⌟ shape. Hold the button down and move the pointer to the bottom right vertex of the area, then release the button:

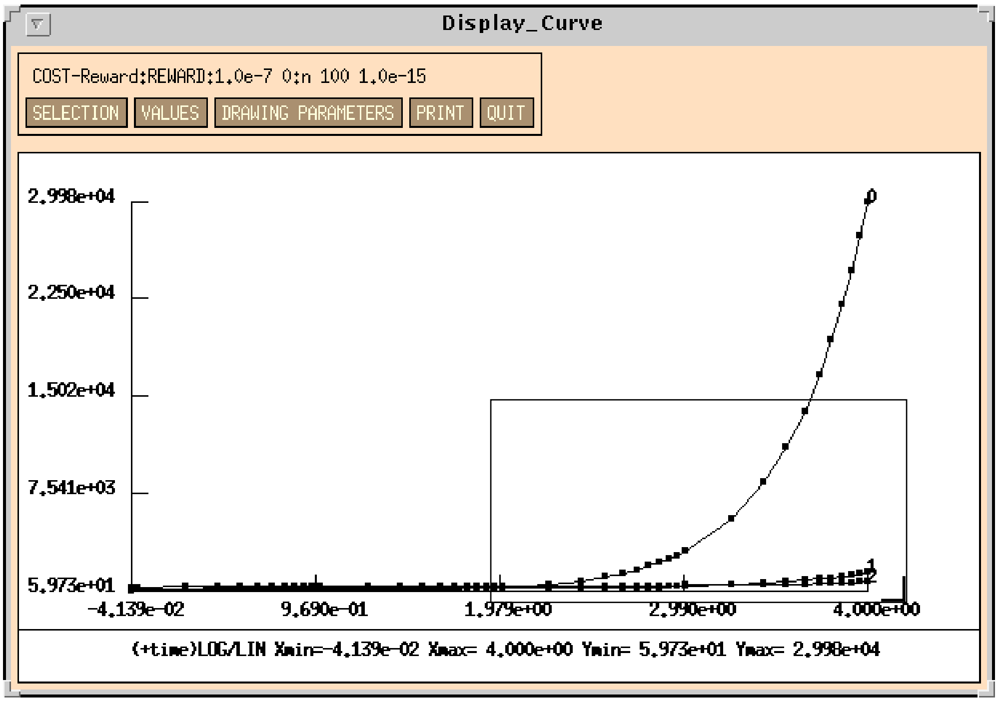


Fig 14.7 Example of use of command ZOOM.

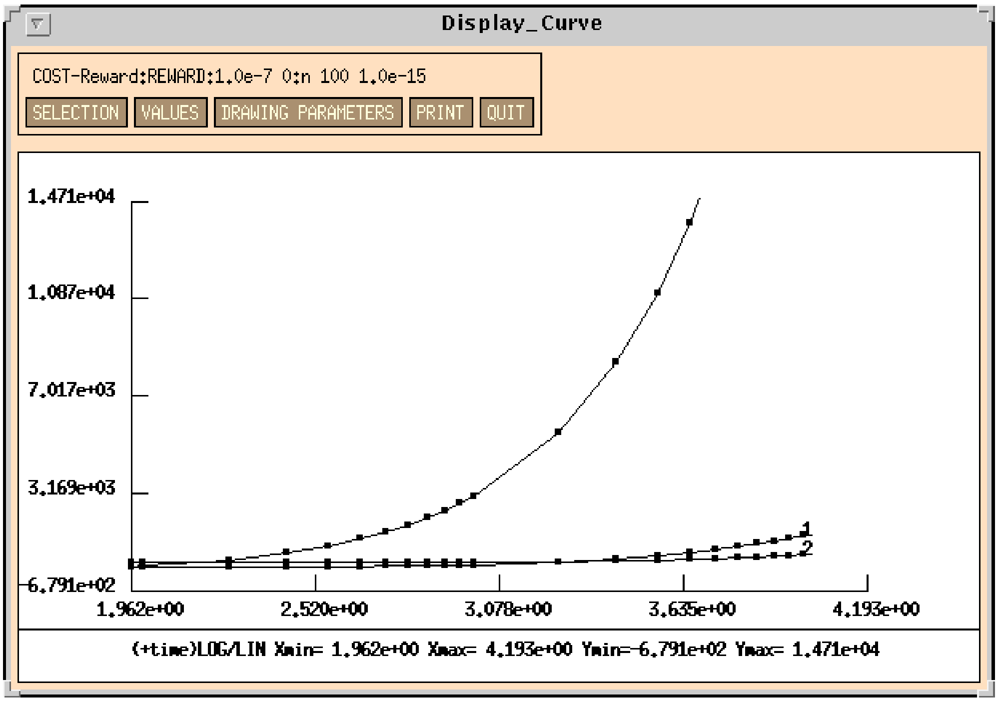


Fig 14.8 Zoom effect.

REDRAW Command

- *dialog box to set drawing parameters. (Fig 14.6)*

-> Re-draws the plot within the Display Curve Window, in accordance with the new values assigned to the drawing parameters.

CLOSE Command

- *dialog box to set drawing parameters. (Fig 14.6)*

-> The dialog box to set drawing parameter disappears.

PRINT Command

- *Display Curve Window. (Fig 14.1)*

-

The environment variable SURF2_IMP specifies the printer to be used. If "SURF2_IMP" is not defined a PostScript file is generated in the home directory. His name is automatically generated as follows: "MARKOV_name_of_model.ps"

-> Prints curves.

QUIT Command

- *Display Curve Window. (Fig 14.1)*

-> The Display Curve Window disappears.

Chapter 16

PETRI NET INTERMEDIATE RESULTS VIEWER

Color: Light blue

16.1 PURPOSE

In the case of a Petri Net model this tool allows visualization of:

- Markov Chain,
- Marking Graph,
- List of transition conflicts.

16.2 ACCESS

Choose Petri Net modelization and use the command result of MODEL FOLDER MENU.

You must access Intermediate Results just after Model Folder computation.
To avoid data base saturation Intermediate results are destroyed:
- when you leave Surf2,
- when a new computation is done for another Model Folder.

If Model Folder includes Markov and Petri Assigned Model, Intermediate Results for Petri Assigned Model can't be accessed.

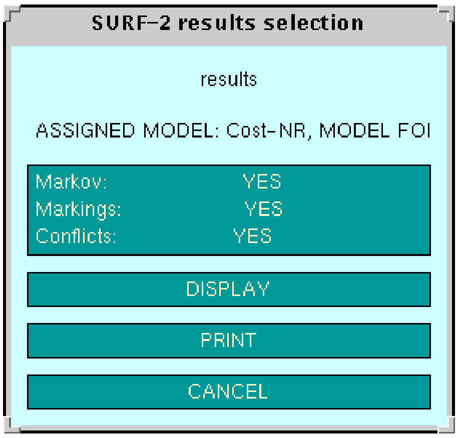


Fig 15.1 Dialog box for Intermediate Results Visualization.

(Markov) YES / NO Switch

- *Dialog box for Intermediate Results Visualization. (Fig 15.1)*

Choose YES if you want to display or print Markov Chain.

(Marking) YES / NO Switch

- *Dialog box for Intermediate Results Visualization. (Fig 15.1)*

Choose YES if you want to display or print Marking Graph.

(Conflicts) YES / NO Switch

- *Dialog box for Intermediate Results Visualization. (Fig 15.1)*

Choose YES if you want to display or print list of transition conflicts.

DISPLAY Command

- *Dialog box for Intermediate Results Visualization. (Fig 15.1)*

-> Displays the selected Intermediate Results, see Fig 15.2.

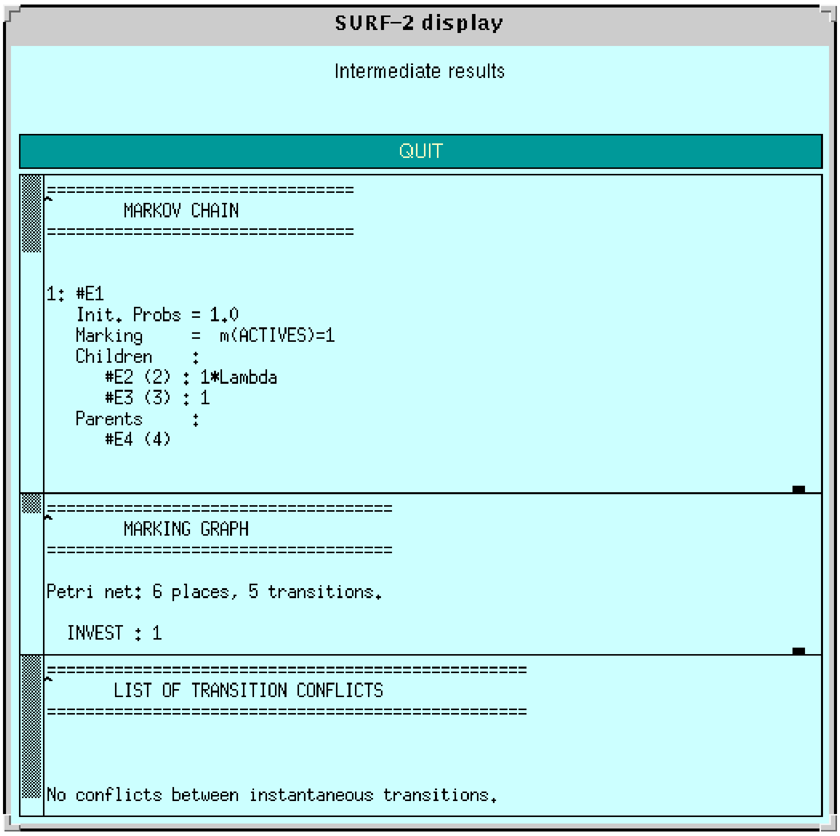


Fig 15.2 Window displaying the Intermediate Results.

PRINT Command

- Dialog box for Intermediate Results Visualization. (Fig 15.1)

The environment variable SURF2_IMP specifies the printer to use. If "SURF2_IMP" is not defined a PostScript file is generated in the home directory. His name is automatically generated as follows: "invariants__name_of_model.ps"

-> The property verification results are printed

After printing the following message is displayed:

Message | **PRINTOUT OF INTERMEDIATE RESULTS TERMINATED**

Print is performed without confirmation.

Appendix 1

PARTITION SYNTAX

A1.1 BASIC DEFINITIONS

Key-words

STATE_CLASS

EVENT_CLASS

PARTITION	IMPROPER	CATASTROPHIC
REWARDS	YIELD_RATE	BONUS

Operators

Arithmetic operators

+ - * /

Boolean operators

and or not

Relational operators

> >=

< <=

= ><

Specific operators

m(marking place

-> set of objects

Separators

(…) grouping objects

{…} list

, items separator of a list

; end of a state- or event-class definition, or end of partition or rewards definition

Space, end-of-line and tabulation characters are ignored by the compiler.

Comments

/* ... */ anything between these characters will be ignored by the compiler.

Identifiers

An identifier is a sequence of non-digit characters and digits. The first character must be a non-digit character.

Any character not used as operator or separator is allowed (for example: ! @ # $…).

An identifier must not be neither a key-word nor one of the following reserved words: **@AE** or **@AR**.

Precedence rules

The precedence rules make the interpretation of expressions unambiguous.

Here are the main rules that must be applied:

- Operators that have the highest precedence are applied first.
- Same precedence operators are applied from left to right. In other words, when an operand is written between two operators of the same precedence, it is bound to the operator to the left.
- Operators inside parenthesized sub-expressions are applied first.

Arithmetic operators are applied according to a decreasing precedence rule.

- (unary minus sign operator)

* /

\+ -

Other operators have same precedence.

The grammar used to write expressions requires to be strict, preventing thus unambiguous expressions, particularly when using operators that have not the same precedence.

<u>Example</u>:

The grammar used to define state- or event-classes prevents combining logical-OR operator or logical-AND operator in a same expression. Thus, such expressions must be written using parenthesized sub-expressions.

The following expression is syntactically incorrect :

cl1 and cl2 or cl3 and cl2

Such an expression must be written as:

cl1 and (cl2 or cl3) and cl2

or:

(cl1 and cl2) or (cl3 and cl2)

A1.2 EXPRESSIONS DEFINING OBJECTS

MARKOV chains

Rate associated to an edge

Initial probability of a state

real-arithmetical expr.

PETRI nets

Place marking

integer-arithmetical expr.

Weight associated to an edge

Rate associated to a timed transition

Initial probability associated to an immediate transition

real-arithmetical expr.

A1.3 DEFINITION OF CLASSES, PARTITIONS AND REWARDS

State- or event-classes, Partitions and Reward structures (rewards) of Markov chains or Petri nets must be defined according to the grammar described below, with respect to the following order:

State-classes,

Event-classes,

Partitions,

Rewards.

On the class concept

The approach used in SURF-2 to model the system behavior is the representation of the system by **states** and **events** (that enable transition of one state to another state of the system).

States and events can be gathered in different sets that have a specific interest for the user, either to carry out measure computations, or to associate to them rewards (performance indices). Here are some examples of sets:

- **states** that provide a proper service (system is up),
- **states** that provide an improper service (system is down),
- repairing or restoration-service **events**, etc…

The concept of **class** is used to constitute these sets of states or events. Two different classes are used in SURF-2: **state-classes** and **event-classes**.

3.1.1 State-classes

States classes are identified by the key-word STATE_CLASS followed by the definition of each state-class. Every state-class is constituted of an identifier and an expression that define the set of states belonging to this state-class. The syntax used depends on the type of model (Markov chain or Petri net).

Here are several examples of state-class definition for a Markov chain:

```
STATE_CLASS
    State_Class1_Markov = {State1,State2};
    State_Class2_Markov = not ({State3} or {State4});
    State_Class3_Markov = (State_Class1_Markov and {State5}) or (not (State_Class2));
```

Here are several examples of state-class definition for a Petri net:

```
STATE_CLASS
    State_Class1_Petri = {m(Place1) >< 0};
    State_Class2_Petri = (({m(Place2) = 1}) and ({m(Place3) >< m(Place4)}))
                         or ({m(Place5) <=k});
    State_Class3_Petri = State_Class1_Petri or (not {m(Place2) >= 5});
```

3.1.2 Event-classes

Event-classes are identified by the key-word EVENT_CLASS followed by the definition of each event-class. Every event-class is constituted of an identifier and an expression that define the set of events belonging to this event-class. The syntax used depends on the type of model (Markov chain or Petri net).

Here are several examples of event-class definition for a Markov chain:

EVENT_CLASS

Event_Class1_Markov = State_Class1_Markov -> State_Class2_Markov;

Event_Class2_Markov = not {State1->State2,State2->State4};

Event_Class3_Markov = (Event_Class1_Markov and (not (Event_Class2_Markov)));

Here are several examples of event-class definition for a Petri net:

EVENT_CLASS

Event_Class1_Petri = {transition1, transition2, transition3};

Event_Class2_Petri = not ({transition4});

Event_Class3_Petri = State_Class1_Petri ->State_Class2_Petri;

Event_Class4_Petri = Event_Class1 or ((not ({transition2})) and {transition1,transition3}));

On the partition concept

The user must specify which subset of states will be involved in the dependability measure computation. He has thus to define a **partition** constituted of a failed-states class (called **improper** class) and a class of all unacceptable (because they are unsafe) states, called **catastrophic** class. The later class must always be included in the *improper* class since it corresponds to the subset of states for the *improper* class that lead to a catastrophic system failure. *Catastrophic* class is only used for safety-measures evaluation.

Partitions are identified by the key-word PARTITION followed by the definition of each *partition*. Several *partitions* for a given model enable comparative evaluation of competing architecture designs. Every *partition* is constituted of an identifier and an expression that define the *improper* class and the *catastrophic* class. The *catastrophic* class must systematically be present in the partition. When measures computation does not require to define a *catastrophic* class, the user will then specify an empty *catastrophic* class in the expression of the partition.

Here are several examples of partitions :

PARTITION

Part2 = IMPROPER system_failed CATASTROPHIC fatal_breakdown;

Part3 = INPROPER system_failed CATASTROPHIC empty;

The "empty-class" must previously be defined as:

STATE_CLASS

empty = { };

On the rewards concept

The ability of superposing a reward structure onto a behavior model with SURF-2 enables computation reward measures such as expected performance or cost[1].

The reward structure is settled by the vector of the yield rates associated to the presence in the states and by the bonus matrix associated to the transitions between states.

Evaluation of performance-related dependability measures requires to define **yield rates** and

1. All the dependable and performance- or cost-related dependability measures are described in Appendix2 of this User Guide. Mathematical expressions associated to these measures will be also found in Appendix2

bonus that correspond to the quantification of the interest and the drawbacks to be in state-classes and events-classes, respectively. This is performed by assigning an arithmetic expression to each state-class for a yield rate and to each event-class for a bonus.

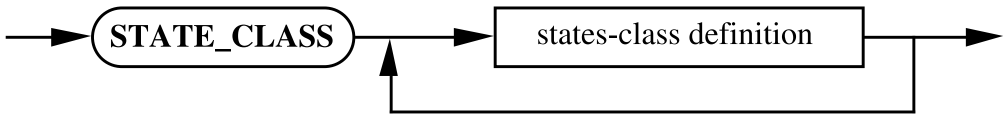
Reward structures are identified by the key-word REWARDS followed by the definition of one or several *reward structures*. Every reward structure is constituted of an identifier and an expression that define the *yield rate* and the *bonus* associated to the reward structure.

Here are several examples of several rewards:

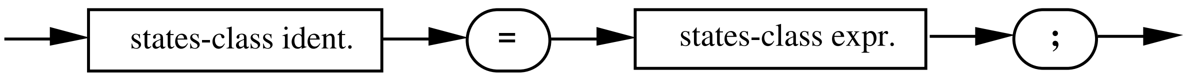
```
REWARDS
        Rew1 =
        YIELD_RATE State_class1 :(yield - 5),
                   State_class2 :(2 * yield + 1),
                   State_class3 :-30;
        Rew2 =
        BONUS      Evt_Class1 :100,
                   Evt_Class3 :-50;
        Rew3 =
        YIELD_RATE State_class1 : Unit
        BONUS      Evt_Class4 :(- Unit * 0.5);
```

A1.3.1 MARKOV / PETRI STATE-CLASSES

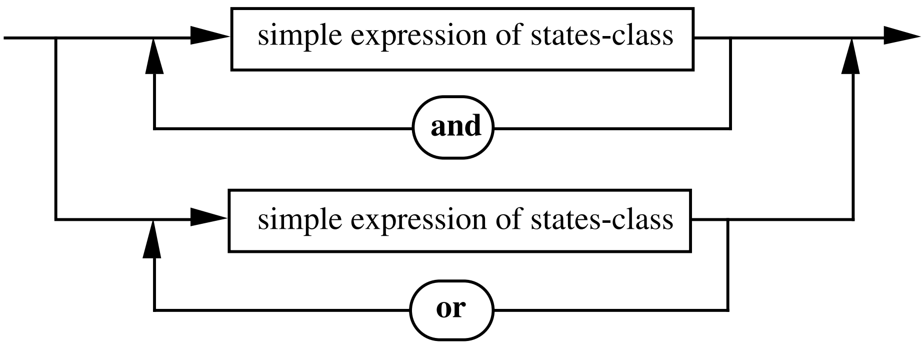
State-class



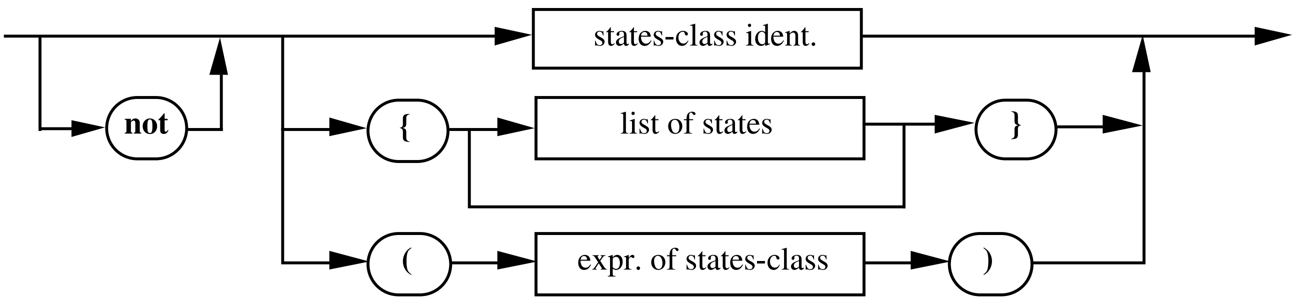
State-class definition



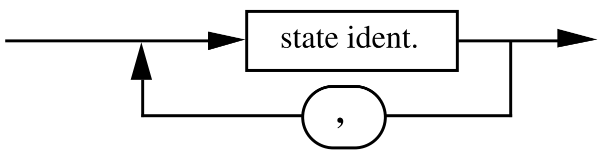
state-class expr.



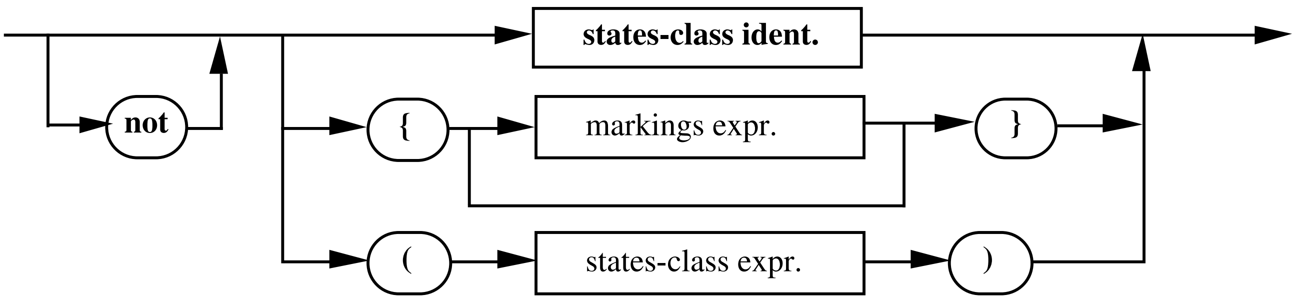
simple expression of MARKOV state-class



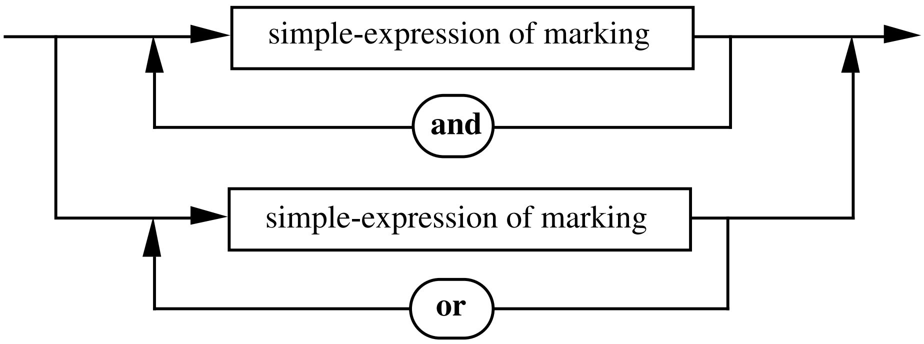
List of states (MARKOV)



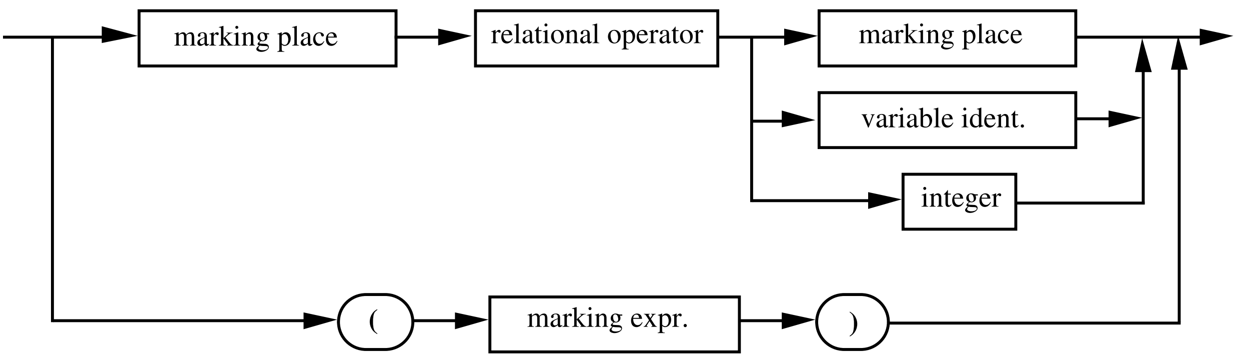
Simple expr. of PETRI state-class



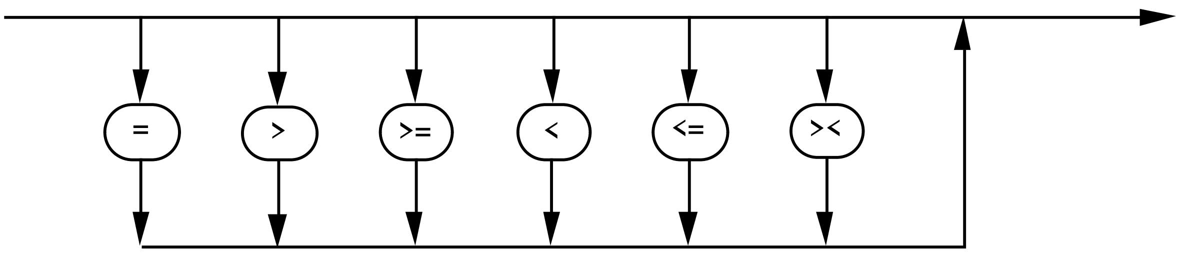
Marking expr. (PETRI)



Simple expr. of marking (PETRI)



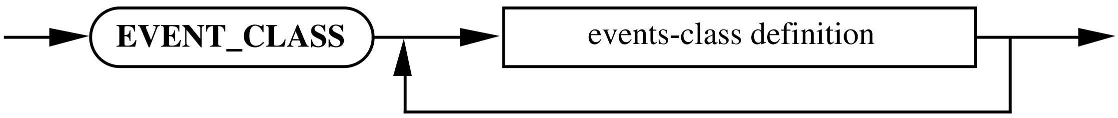
Relational operators (PETRI)



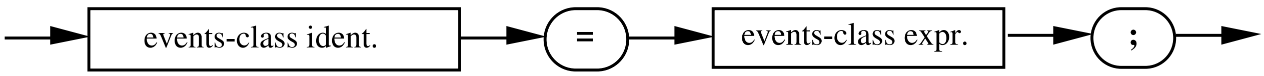
Marking place PETRI

A1.3.2 MARKOV / PETRI EVENT-CLASSES

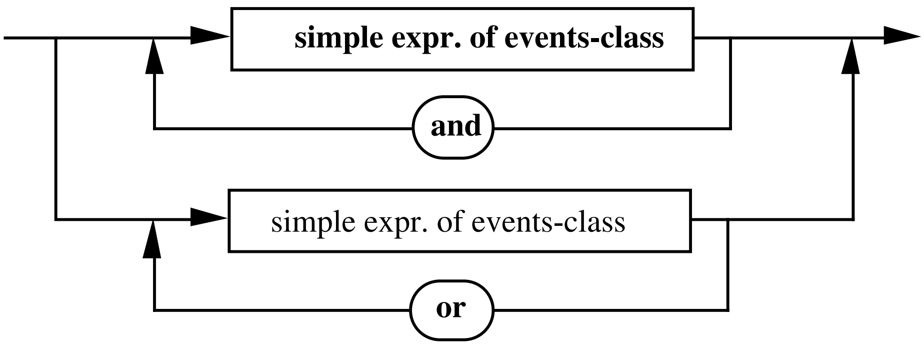
Event-class



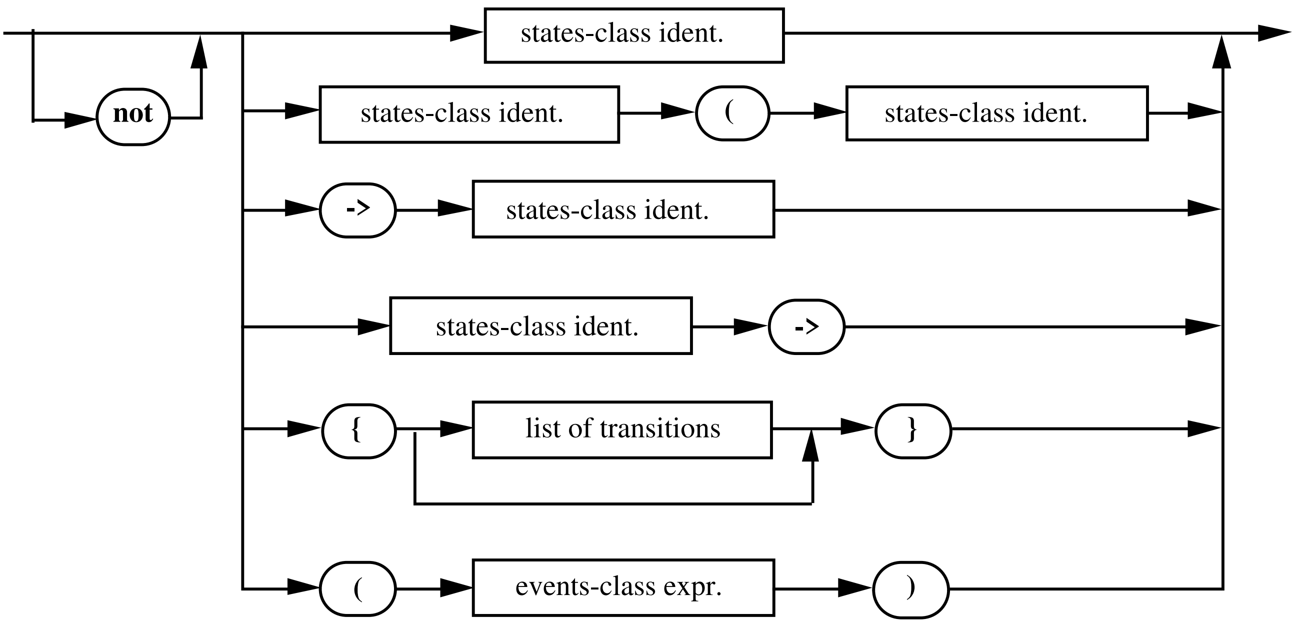
Event-class definition



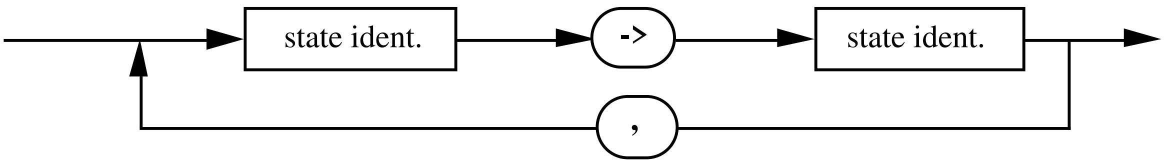
Event-class expr.



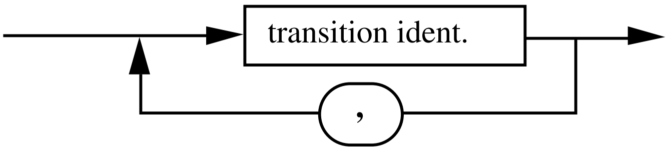
Simple expr. of event-class



List of transitions (MARKOV)

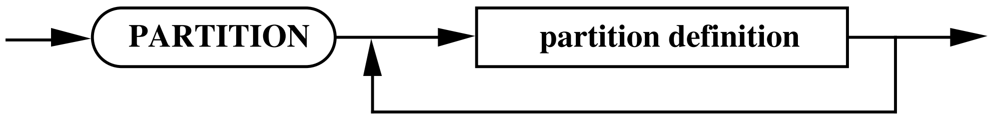


List of transitions (PETRI)

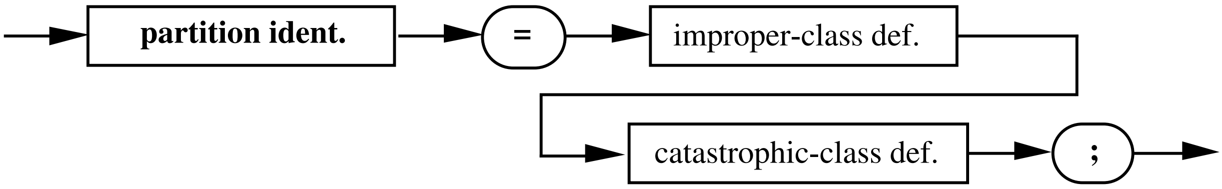


A1.3.3 PARTITIONS (MARKOV / PETRI)

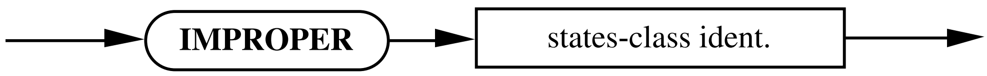
Partitions



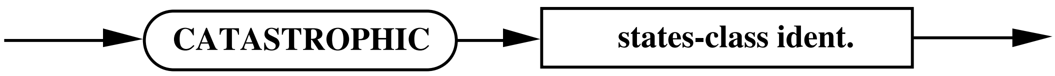
Partition definition



Improper-class def.

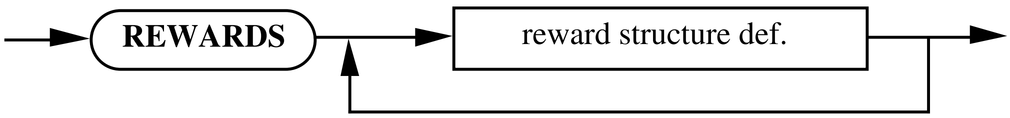


Catastrophic-class def.

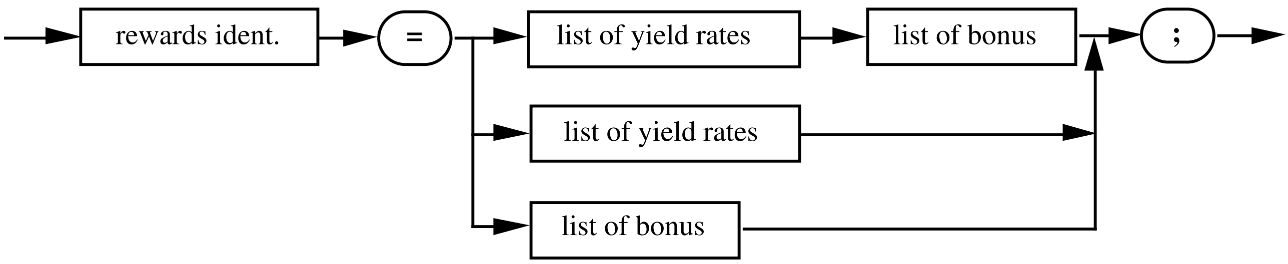


A1.3.4 REWARDS (MARKOV / PETRI)

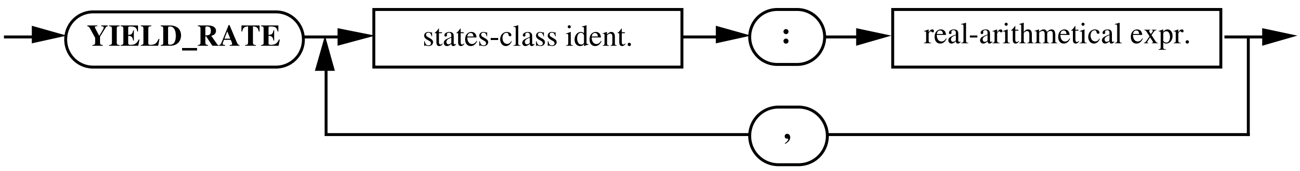
Reward structure



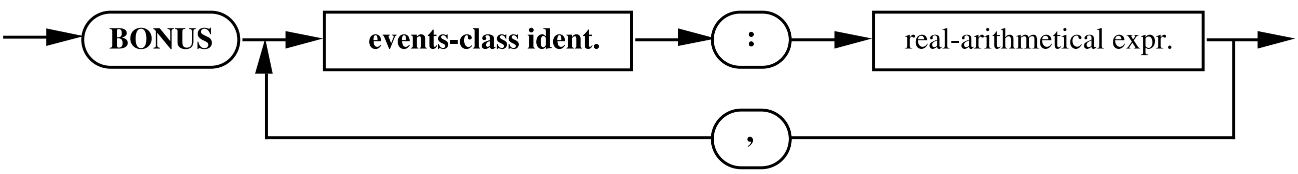
Rewards definition



List of yield rates

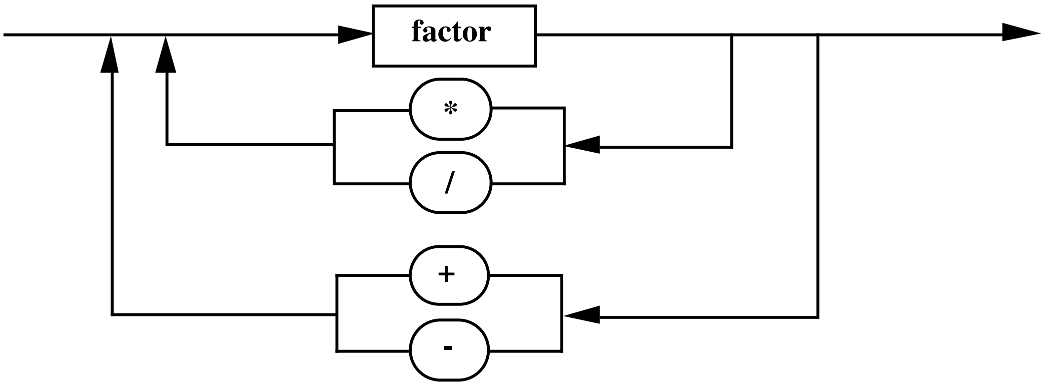


List of bonus

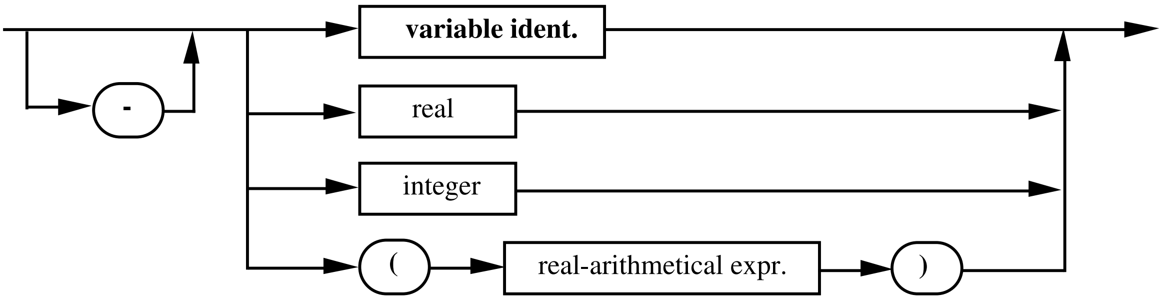


A1.3.5 ARITHMETIC EXPRESSIONS (INTEGER / REAL)

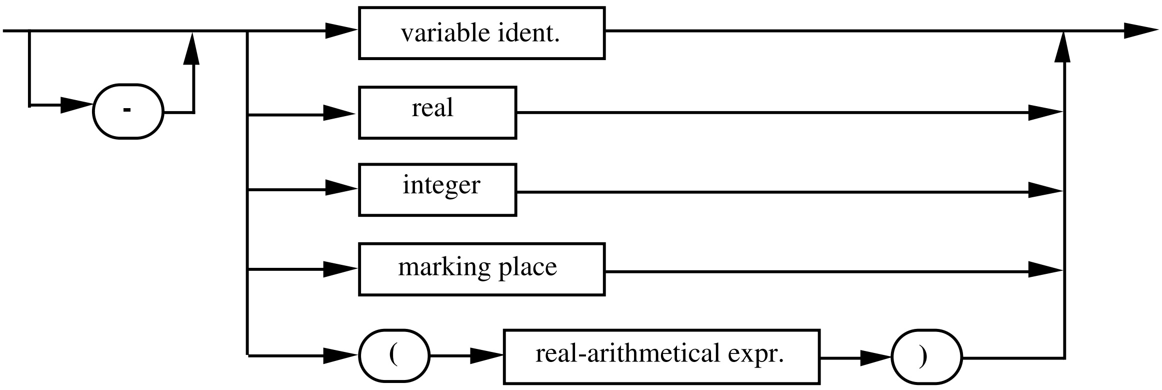
Arithmetic expr. (integer / real)



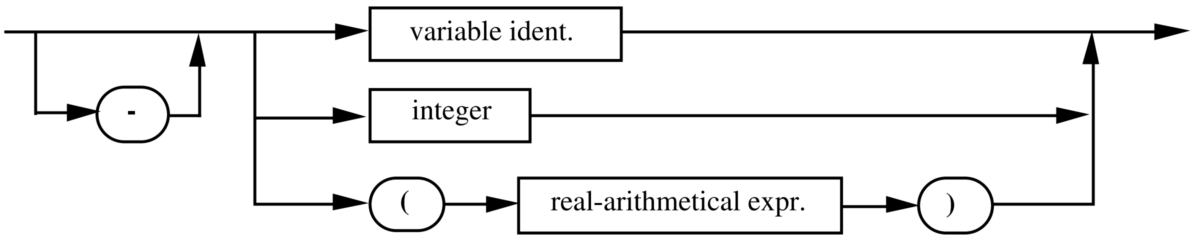
Real-type factor (MARKOV)



Real-type factor (PETRI)

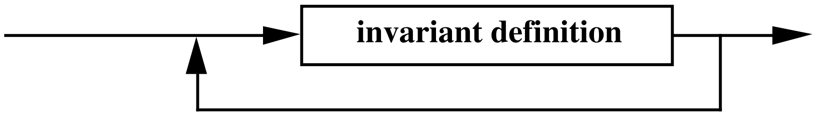


Integer-type factor

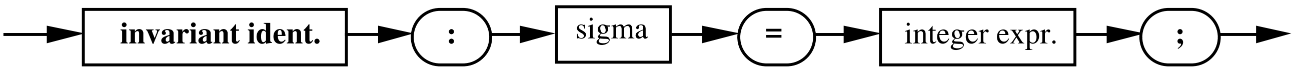


A1.4 INVARIANTS DEFINITION

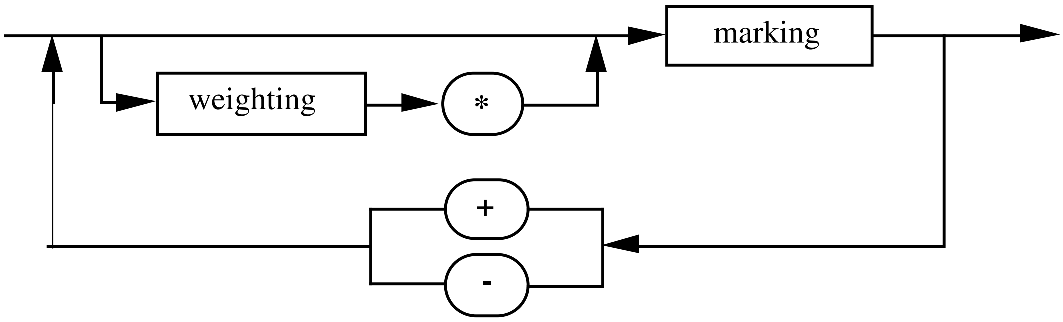
INVARIANTS



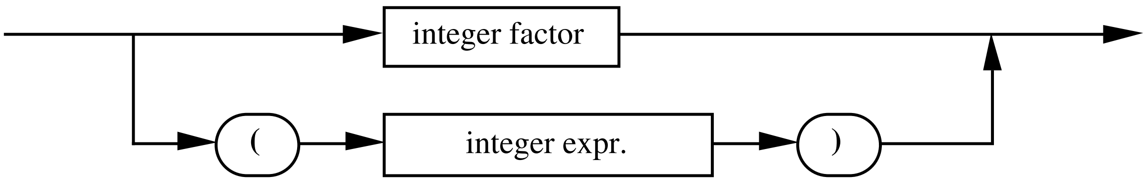
Invariant definition



Sigma

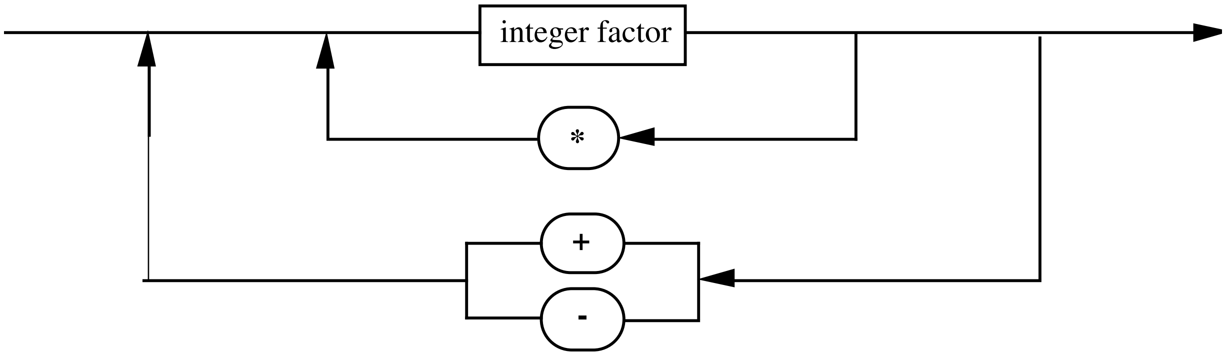


Weighting

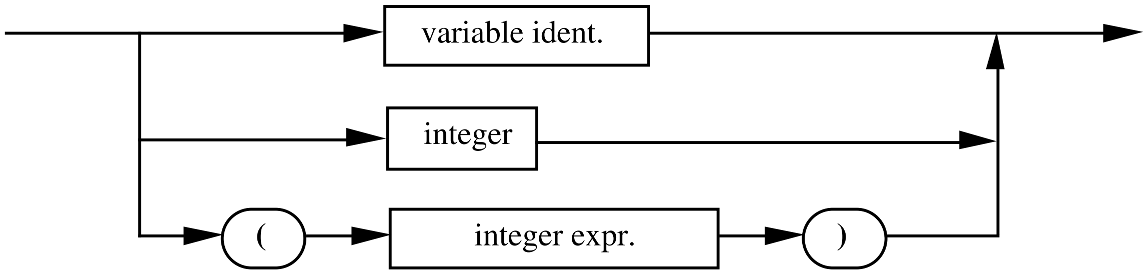


Marking

Integer-type expr.



Integer-type factor



Appendix 2

TWO SIMPLE EXAMPLES

A2.1 A SIMPLE MARKOV CHAIN.

We define a very simple Markov Chain Model to guide user throw SURF-2.

A2.1.1 Model creation.

We modelize a simple system composed of one machine. When the machines fails it can be repaired. System is ok if the machine is ok.

In Main Menu point MODEL <Command> to display Model Menu:

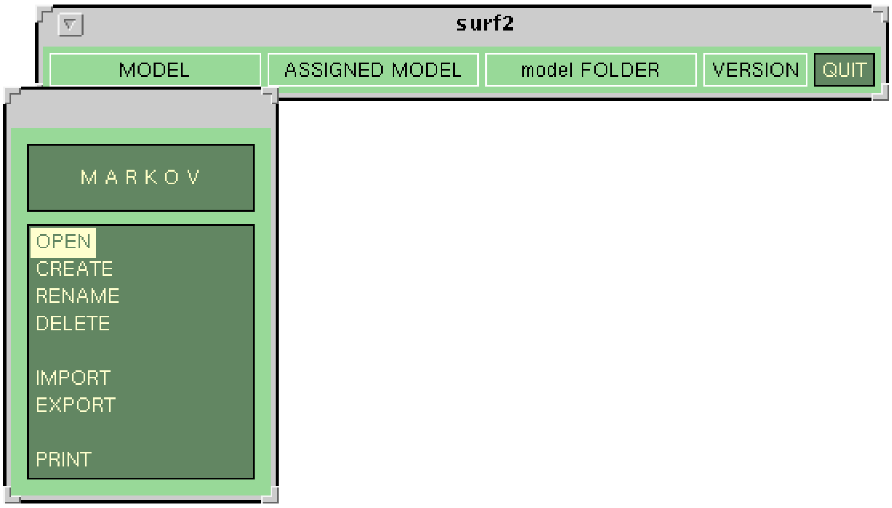


Fig A3.1 Main Menu and Model Menu.

Use CREATE Command the following dialog box appears: Fig A3.2 (a). Enter the model name: Fig A3.2 (b).

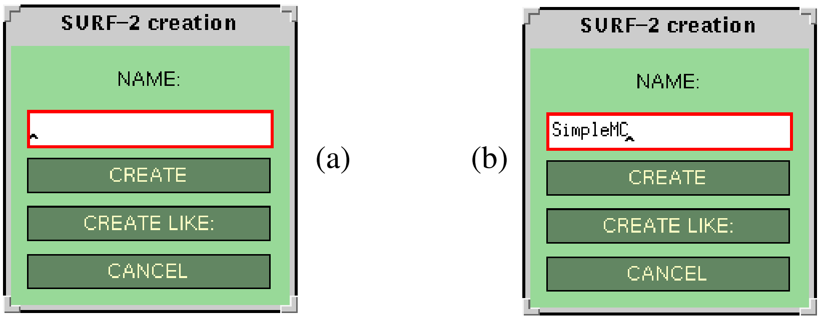


Fig A3.2 Dialog box for Model Creation.

Then use CREATE Command. The Markov Editor dialog box is displayed:

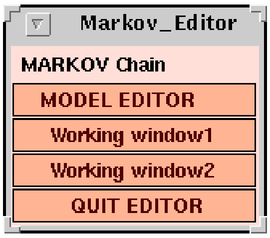


Fig A3.3 Markov Editor dialog box.

Select Command MODEL EDITOR the Markov chain editing window is displayed.

Use (STATE, EDGE, MOVE, ...) *editor functions to design model:*

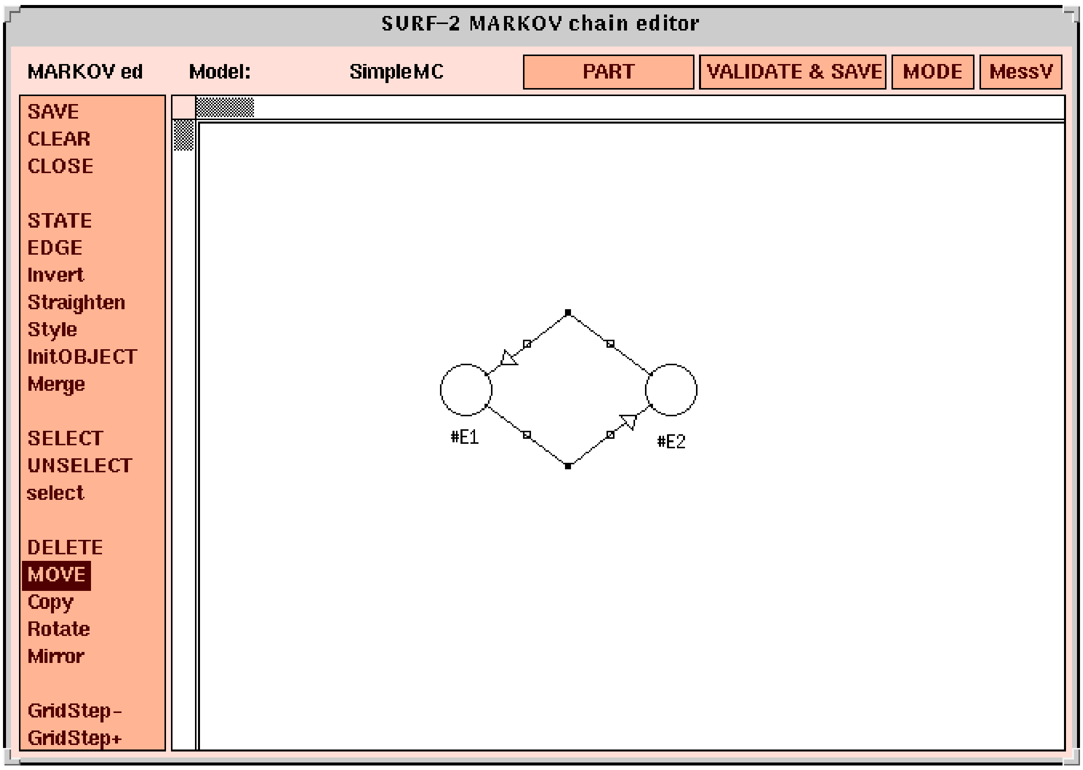
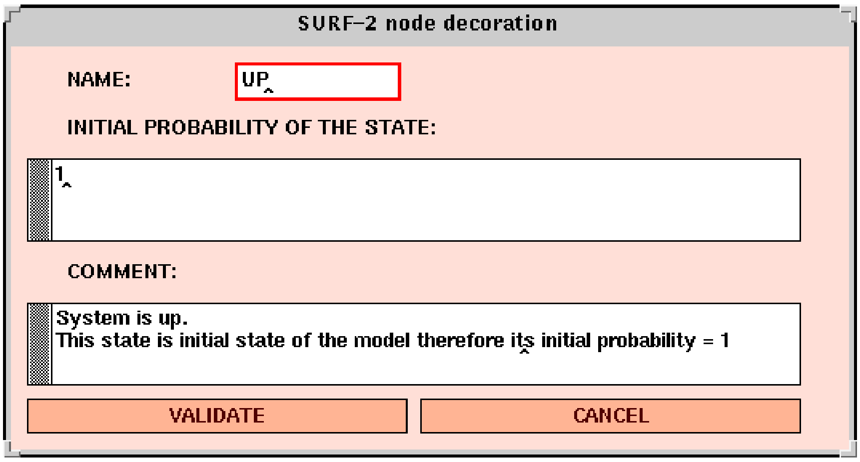


Fig A3.4 Markov chain editing window.

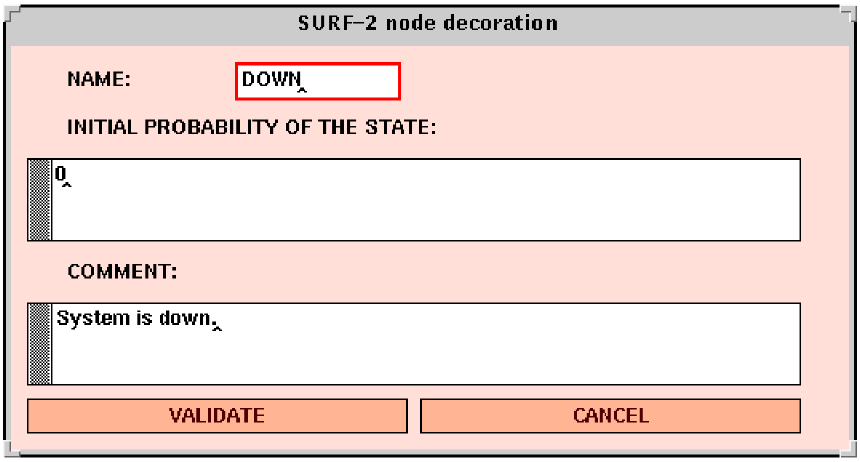
Select Editor Function **initObjet** and than pick objects to initialize:

- Pick state #E1 and use dialog box displayed to initialize it:



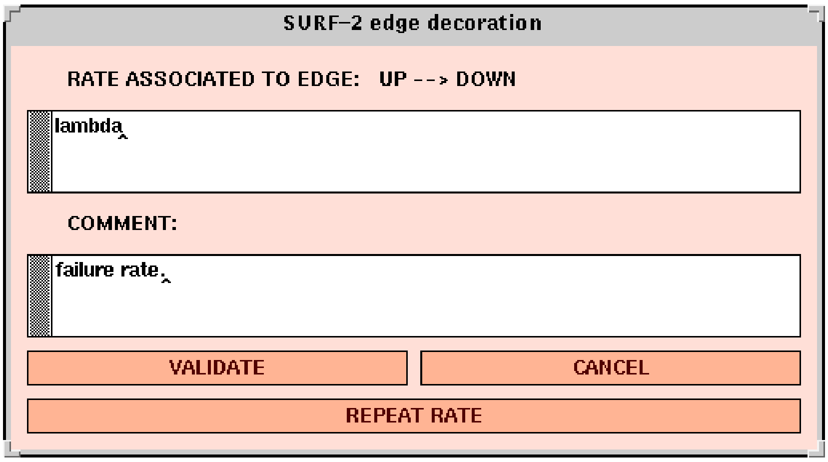
Click VALIDATE command, the dialog box disappears.

- Pick state #E2 and use dialog box displayed to initialize it:



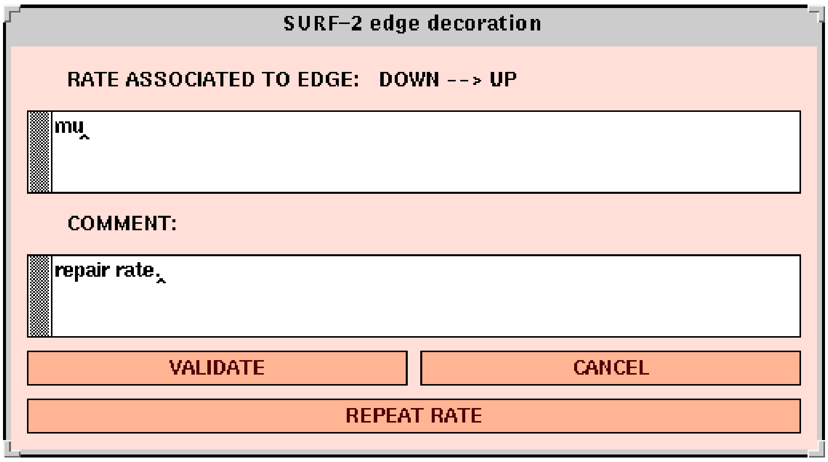
Click VALIDATE command, the dialog box disappears.

- Pick edge UP --> DOWN and use dialog box displayed to initialize it:



Click VALIDATE command, the dialog box disappears.

- Pick state #E1 and use dialog box displayed to initialize it:



Click VALIDATE command, the dialog box disappears.

Fig A3.5 show the Model after objects initialization:

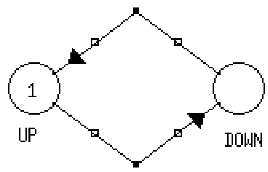


Fig A3.5 The Markov Chain Model.

State names (UP, DOWN) are meaningful for you but not for SURF-2 so we must define a partition to split the set of states into two classes:

*- **Improper class**: class of failed states (explicitly defined).*

- Proper class: the complementary class of improper class not explicitly defined.

*Moreover a **catastrophic** class must be specified to define safety measures.*
See Appendix 1 page 4

Click PART Command (Fig A3.4) partition editor appears.

Enter partition definition:

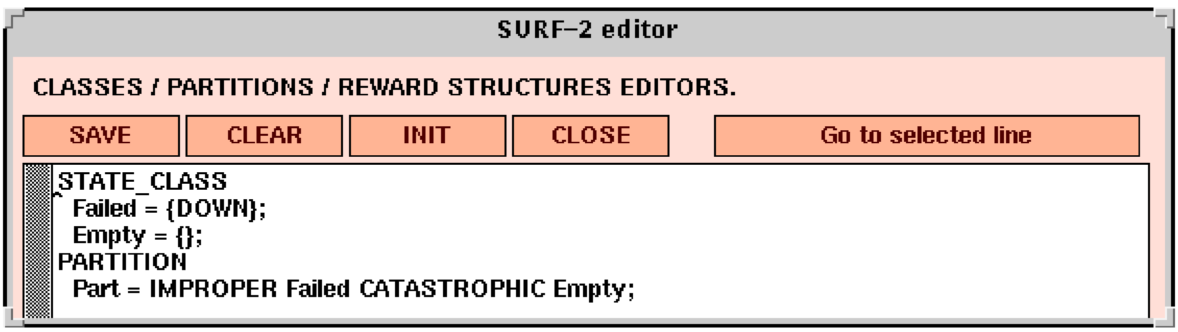


Fig A3.6 Editor classes/partitions/reward structures.

Now use VALIDATE & SAVE Command (Fig A3.4), the following message is displayed:

Message | Valid Model.

A2.1.2 Assigned Model creation:

In Main Menu point ASSIGNED MODEL <Command> to display Assigned Model Menu:

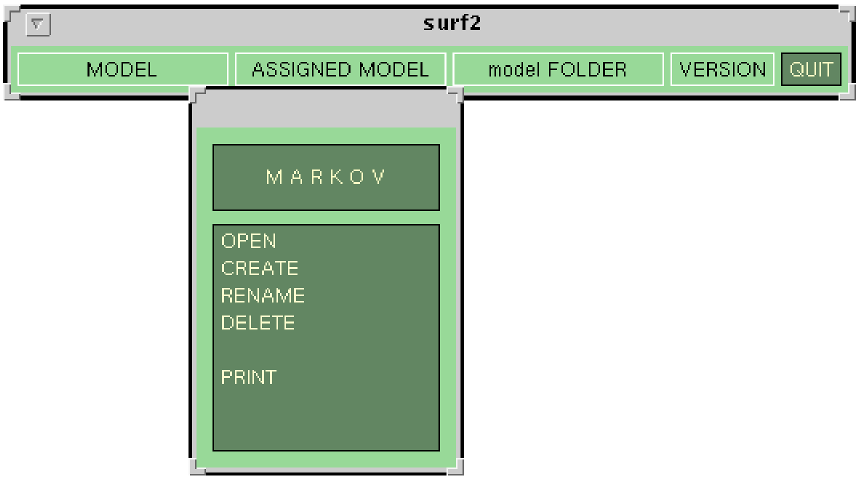


Fig A3.7 Main Menu and Assigned Model Menu.

CREATE Command allows to display the following dialog box:

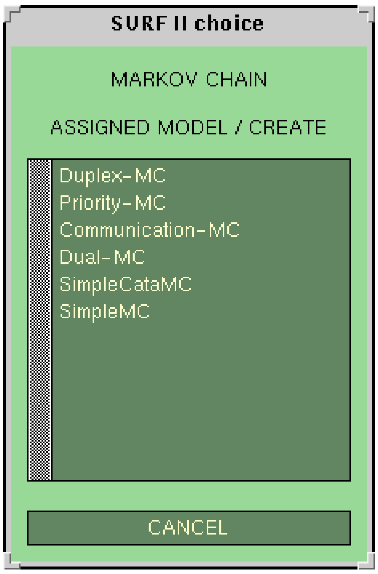
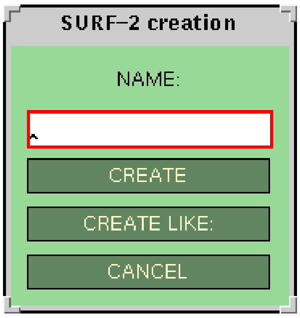


Fig A3.8 Dialog box to select model.

Select our Model: SimpleMC

The following dialog box appears: Fig A3.9 (a). Enter the model name: Fig A3.9 (b).



(a) (b)

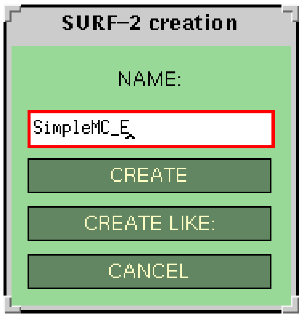


Fig A3.9 Dialog box for Model Creation.

Then use CREATE Command (Fig A3.9). The Assigned Model Editor dialog box which is displayed allows to assign numerical value or global value to each parameter:

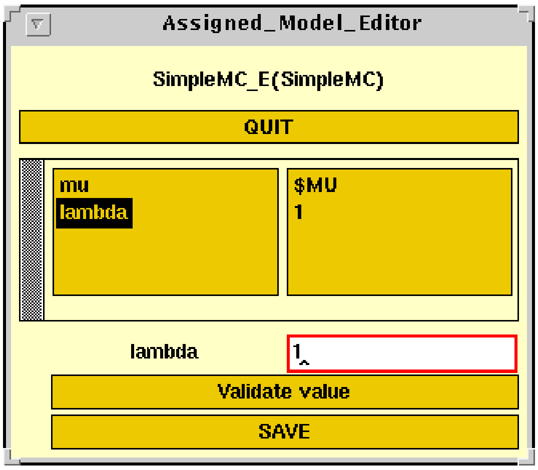


Fig A3.10 Assigned Model Editor window.

As you can see, the failure rate value is 52 failures by year. A global value is assigned to repair rate. Save this values with Command SAVE.

A2.1.3 Model Folder creation.

In Main Menu point model FOLDER <Command> to display Model Folder Menu:

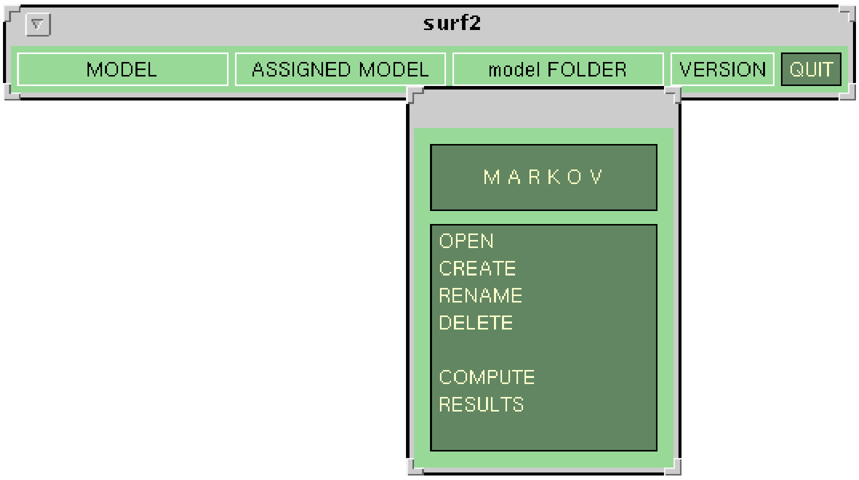


Fig A3.11 Main Menu and Model Folder Menu.

Use CREATE Command, the following dialog box appears: Fig A3.2 (a). Enter the model name: Fig A3.2 (b).

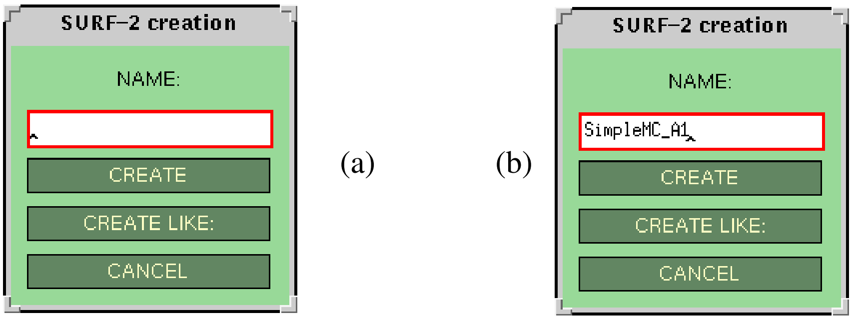


Fig A3.12 Dialog box for Model Folder Creation.

Then use CREATE Command (Fig A3.12), the Markov Editor dialog box is displayed:

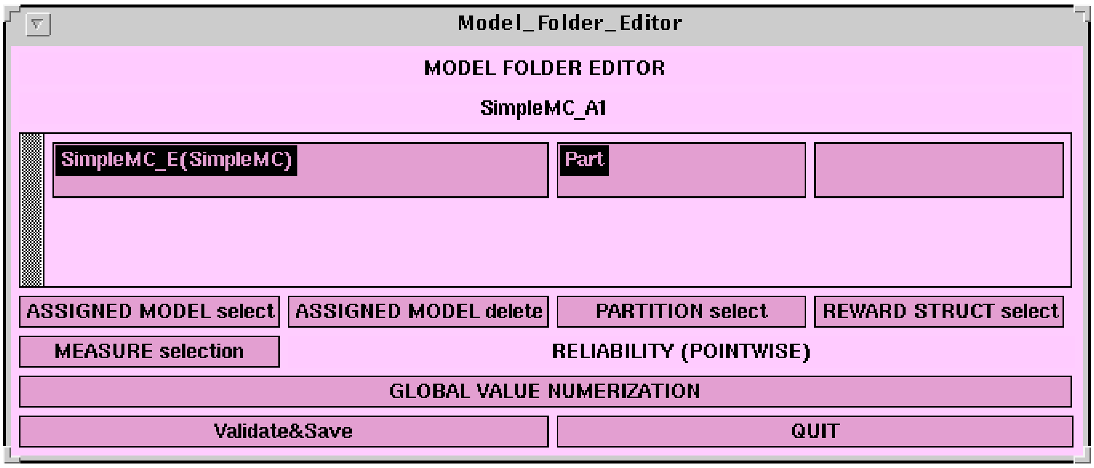


Fig A3.13 Model Folder Editor main window.

Choose Assigned Model SimpleMC_E in dialog box displayed by ASSIGNED MOD-EL select Command:

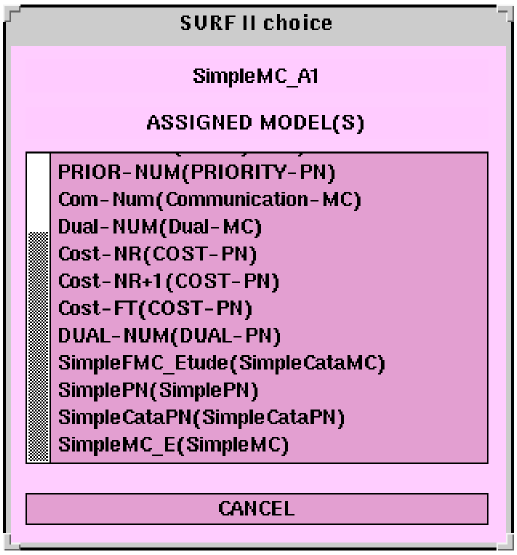


Fig A3.14 Dialog box to select Assigned Model.

Choose partition part in dialog box displayed by PARTITION select Command:

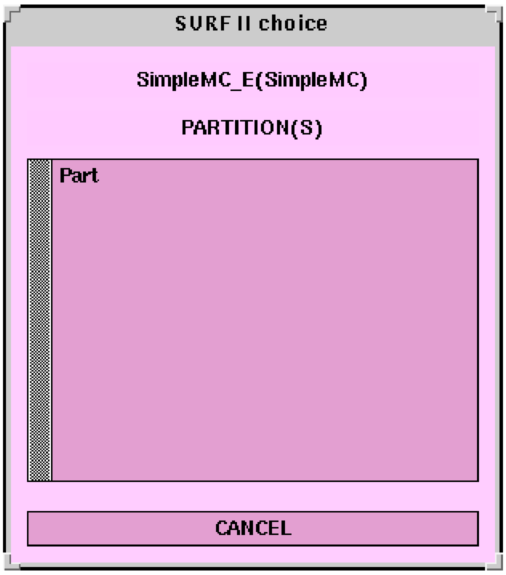


Fig A3.15 Dialog box to select a partition.

Choose Reliability in dialog box displayed by MEASURE selection Command:

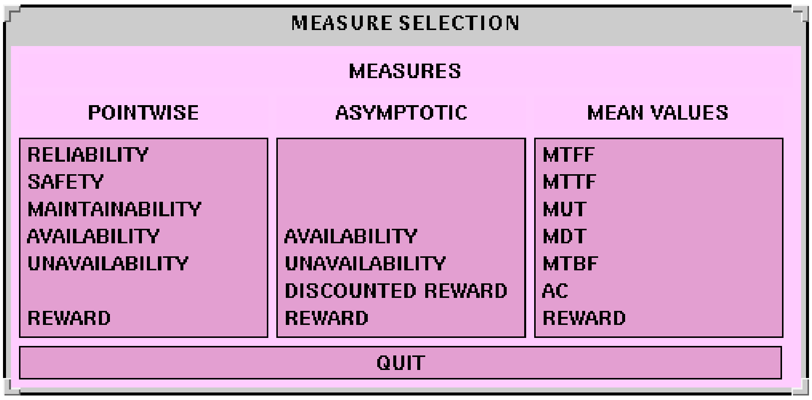


Fig A3.16 Dialog box to select a Measure.

The following dialog box is displayed. We use default value of accuracy mode and computation mode: AUTO, click command QUIT (Fig A3.17).

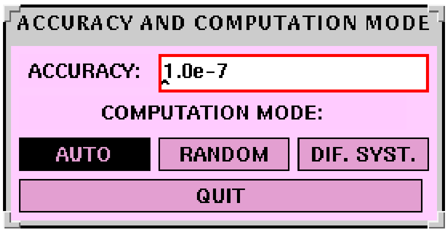


Fig A3.17 Dialog box to define accuracy and computation mode.

A global value was defined in Assigned Model Editor, we must assign it values. Use GLOBAL VALUES NUMERIZATION Command the following dialog box is displayed:

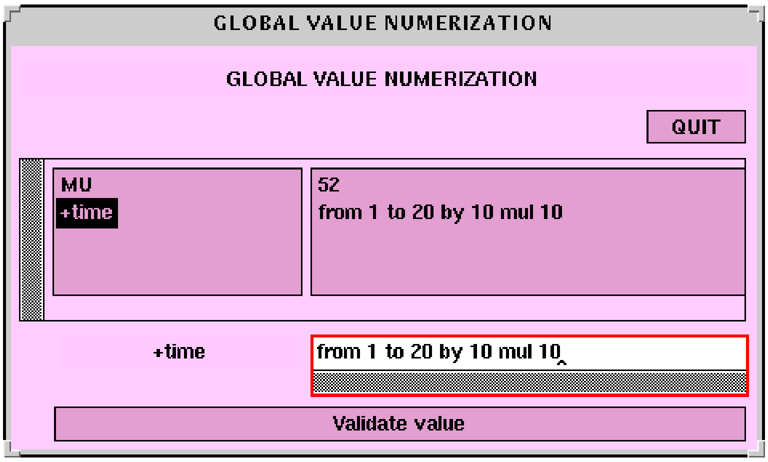


Fig A3.18 Dialog box to global value numerization.

For each global value[1] define a numerical value or a set of values.

Quit accessories windows and use Validate&Save Command of Model Folder Editor main window(Fig A3.13).

3.2.1.4 Computation

[1] Time is a global value since reliability is a pointwise measure.

Use Command COMPUTE in Model Folder menu. The following window allows to choose our Model Folder SimpleMC_A1

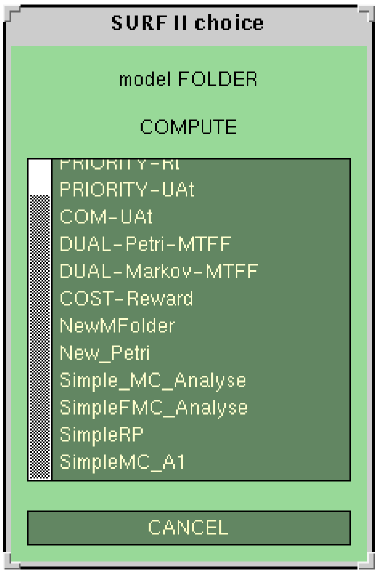


Fig A3.19 Dialog box to select a Model Folder in order to compute a measure.

Computation is done, some messages give information about computing process:

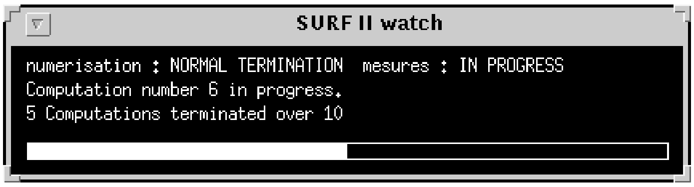


Fig A3.20 A message during computation.

Some complementary informations are printed in window where SURF-2 is executed.

3.2.1.5 RESULTS

When computation is finished we can examine results.

RESULTS Command of Model Folder menu displays dialog box (Fig A3.21); select our Model Folder SimpleMC_A1.

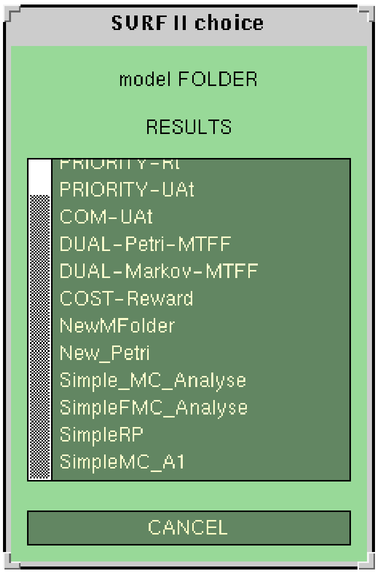


Fig A3.21 Results: Model folder selection.

The following window is displayed:

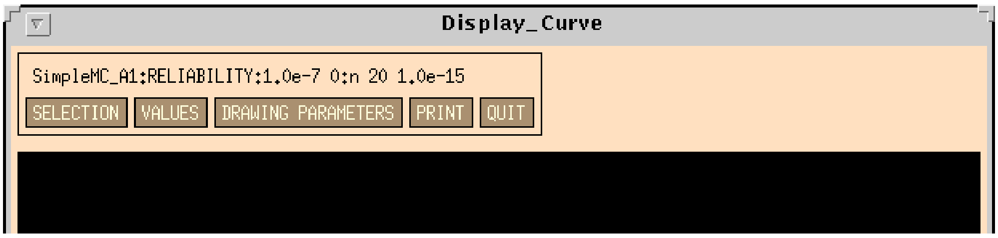


Fig A3.22 Display Curve window.

Use SELECTION Command, dialog box(Fig A3.23)is displayed. We select Assigned Model SimpleMC_E and global value time.

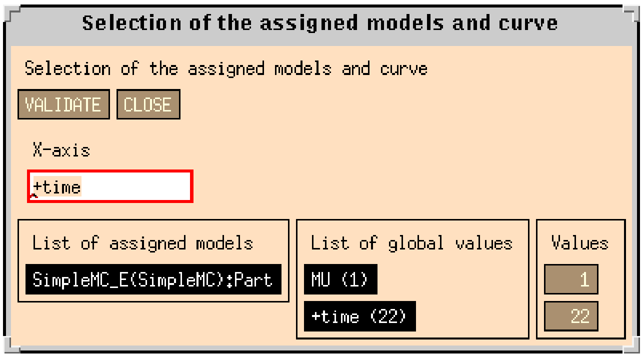


Fig A3.23 Dialog box to select Assigned Model and Curves.

Use Command VALIDATE (Fig A3.23) to display curve of Reliability:

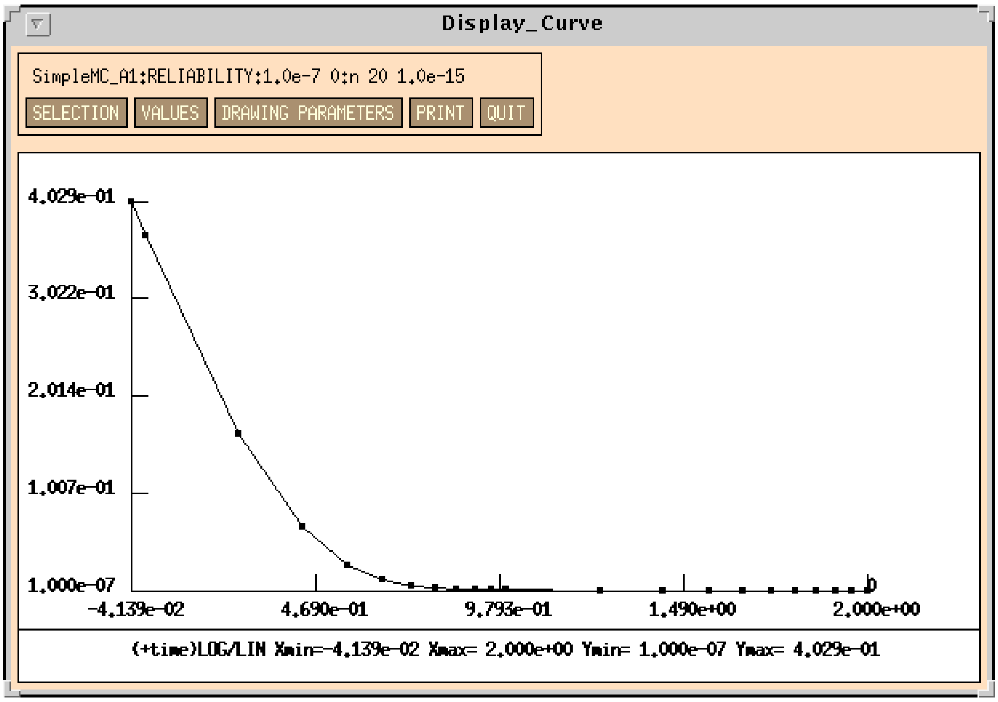


Fig A3.24 Reliability.

A2.1.6 Another computation.

We use the same Model and Assigned Model and we create a new Model Folder named: SimpleMC_A2. We choose an asymptotic measure: AVAILABILITY.

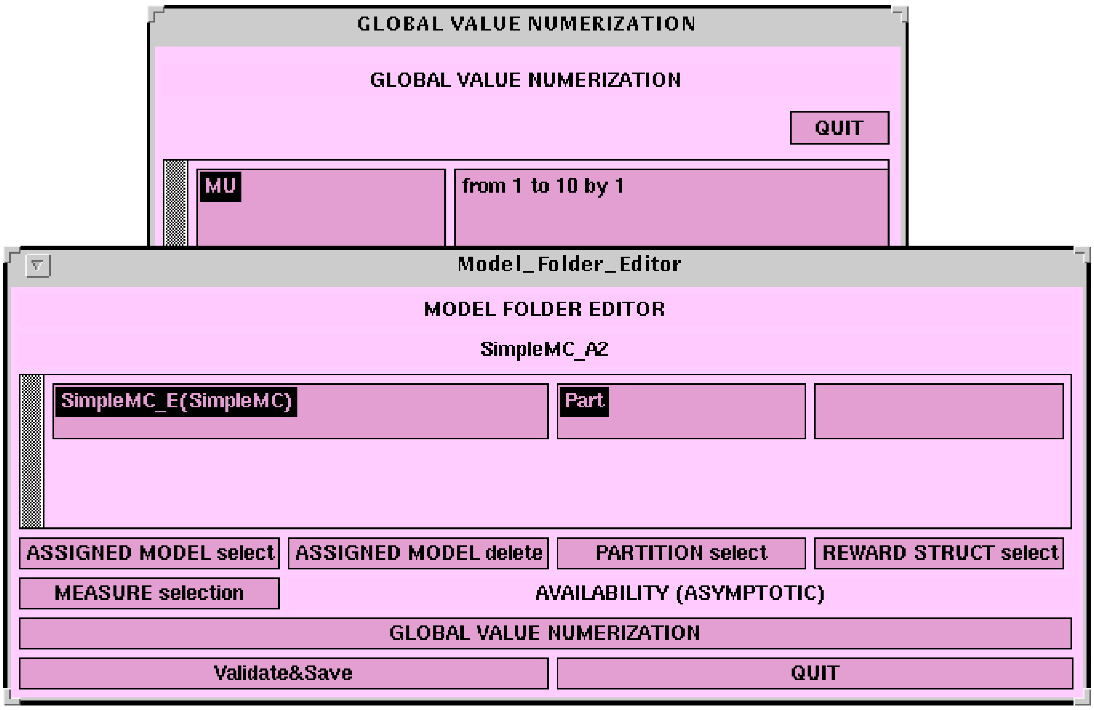


Fig A3.25 A new Model Folder.

We activate computing and examine new results:

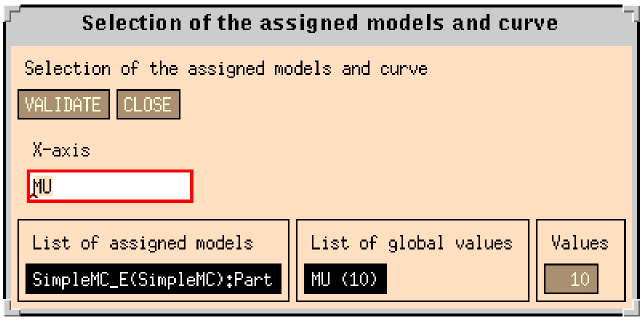


Fig A3.26 Dialog box to select Assigned Model and Curves.

We select Assigned Model "SimpleME_E" and global value MU. The curve is displayed:

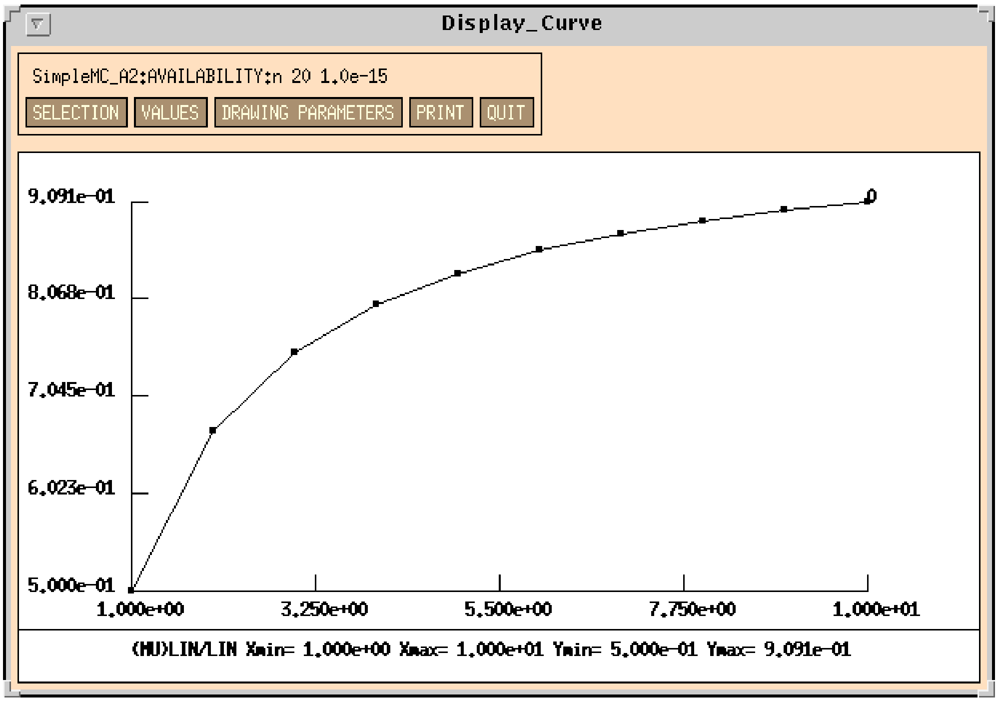


Fig A3.27 Availability.

Fig A3.27 Availability increases with the repair rate.

A2.2 A SIMPLE PETRI NET.

A2.2.1 Model creation.

We modeliize a simple system composed of a set of m machines. When a machines fail it can be repaired. System is ok if at least one machine is ok.

In Main Menu point MODEL <Command> to display Model Menu:

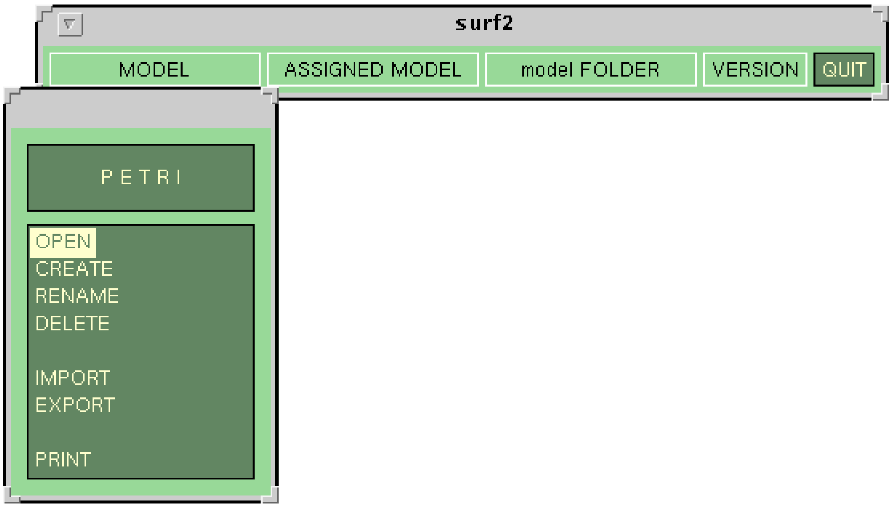
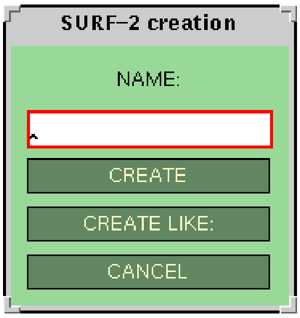


Fig A3.28 Main Menu and Model Menu.

Use CREATE Command (Fig A3.28), the following dialog box appears: Fig A3.29 (a). Enter the model name: Fig A3.29 (b).



(a)

(b)

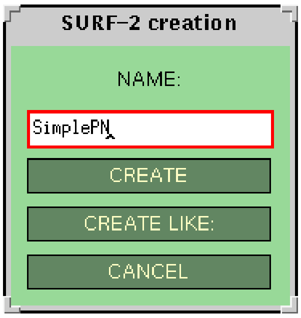


Fig A3.29 Dialog box for Model Creation.

Then use CREATE Command (Fig A3.29). The Petri Editor dialog box is displayed:

Fig A3.30 Petri Editor dialog box.

Select Command MODEL EDITOR the Markov chain editing window is displayed.

Use (PLACE, TRANSITION, EDGE, MOVE, ...) *editor functions to design model.*

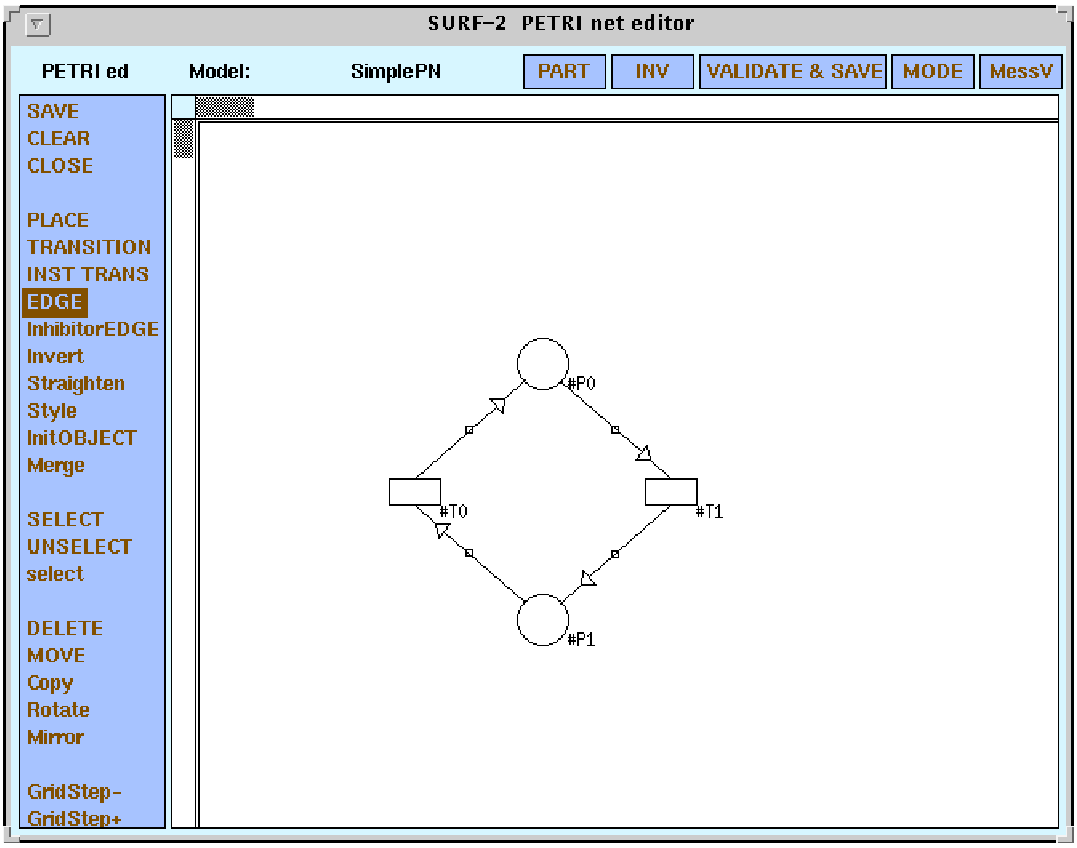
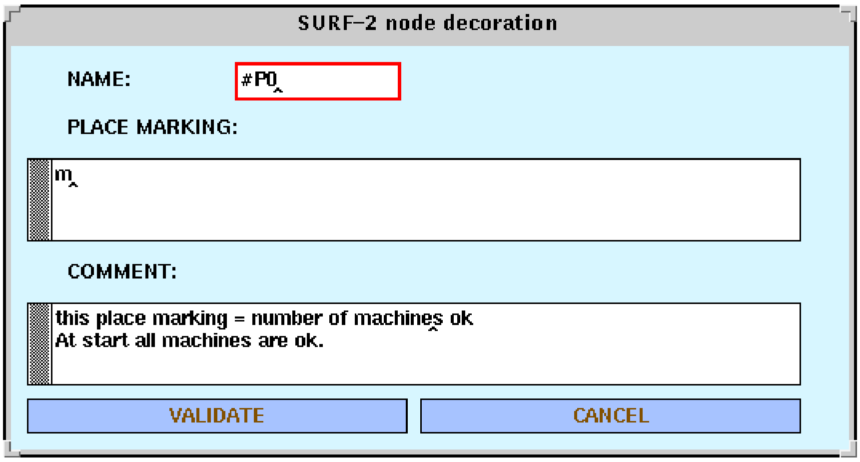


Fig A3.31 Petri Net editing window.

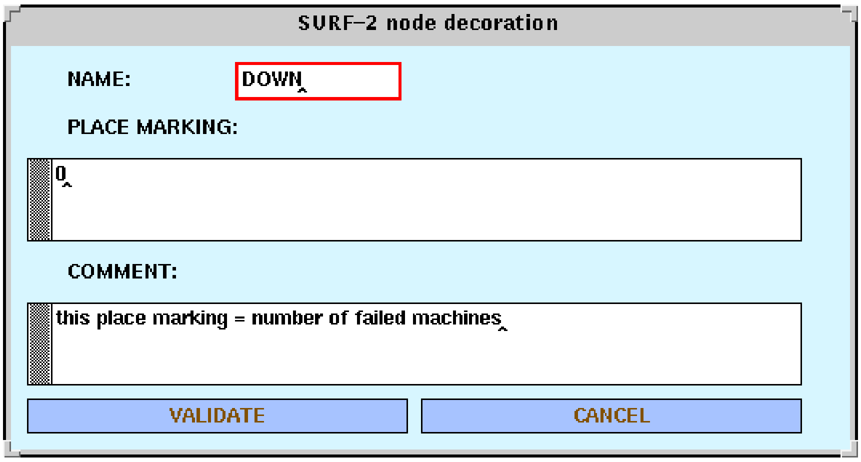
Select Editor Function **initObjet** and then pick objects to initialize then:

- Pick Place #P0 and use dialog box displayed to initialize this object:



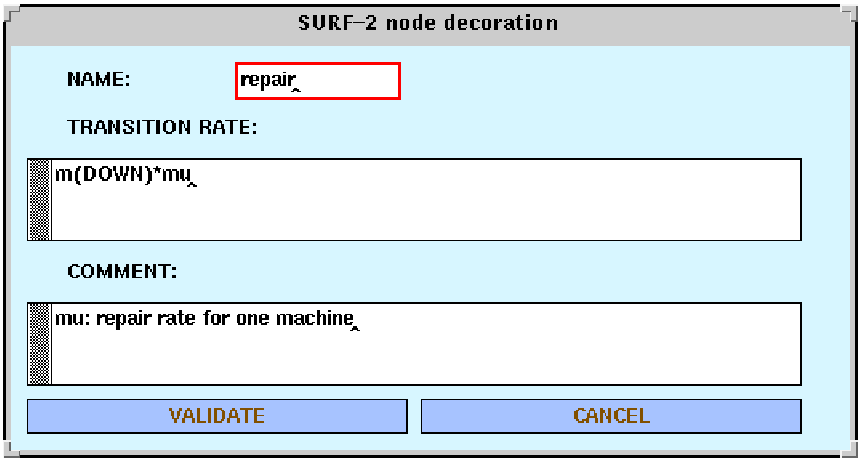
Click VALIDATE Command, the dialog box disappears.

- Pick Place #P1 and use dialog box displayed to initialize this object:



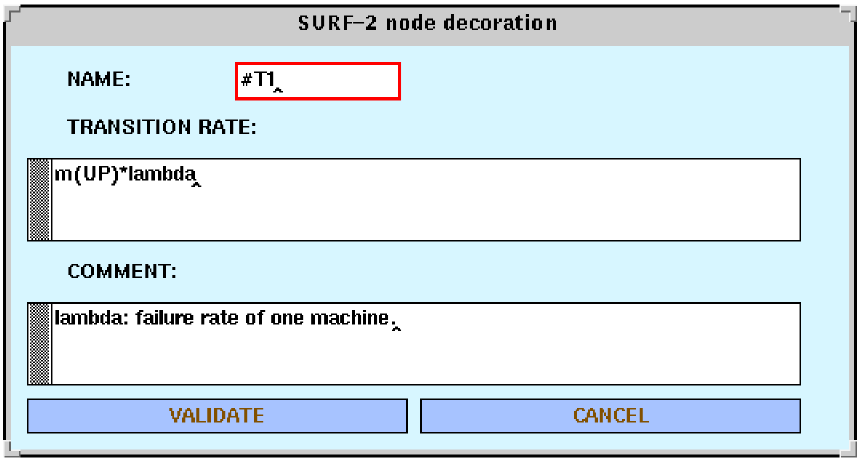
Click VALIDATE Command, the dialog box disappears.

- Pick Transition #T0 and use dialog box displayed to initialize this object:



Click VALIDATE Command, the dialog box disappears.

- Pick Transition #T1 and use dialog box displayed to initialize this object:



Click VALIDATE Command, the dialog box disappears.

Fig A3.32 shows the Model after objects initialization:

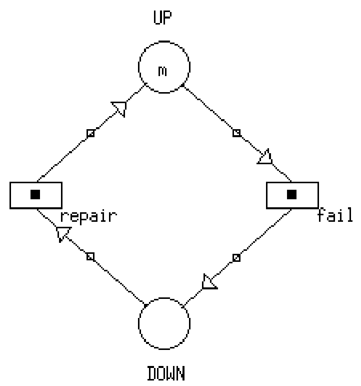


Fig A3.32 Petri Net Model.

Place names (UP, DOWN) are meaningful for you but not for SURF-2 so we must define a partition to split the set of states into two classes:

- ***Improper class****: class of failed states (explicitly defined).*
- *Proper class: the complementary class of improper class not explicitly defined.*

Moreover a ***catastrophic*** *class must be specified to define safety measures.*
See Appendix 1 page 4

We use PART Command (Fig A3.30) partition editor appear.

Enter partition definition:

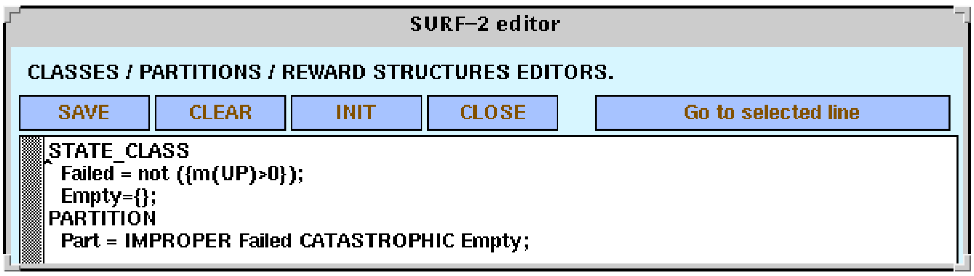


Fig A3.33 Partition.

The number of units in working order is equal to place "UP" marquing. The system is ok if at least one unit is ok: m(UP) > 0.

A2.2.2 Assigned Model creation

In Main Menu point ASSIGNED MODEL <Command> to display Assigned Model Menu:

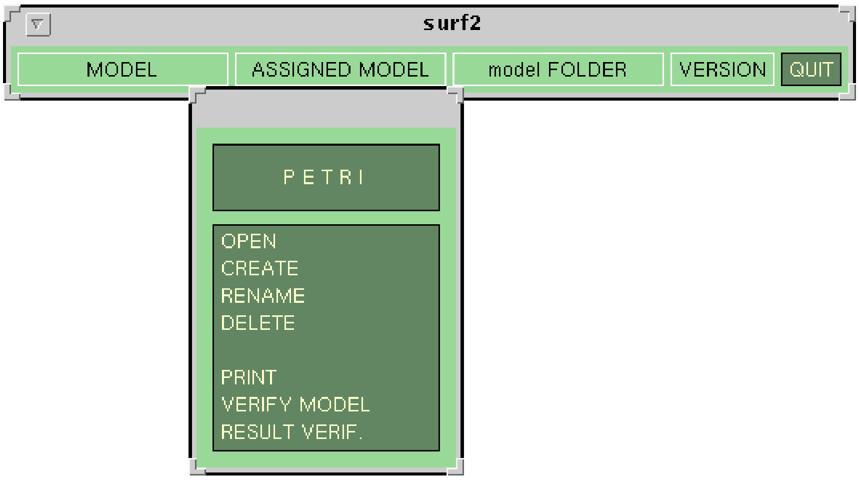


Fig A3.34 Main Menu and Assigned Model Menu.

CREATE Command allows to display the following dialog box:

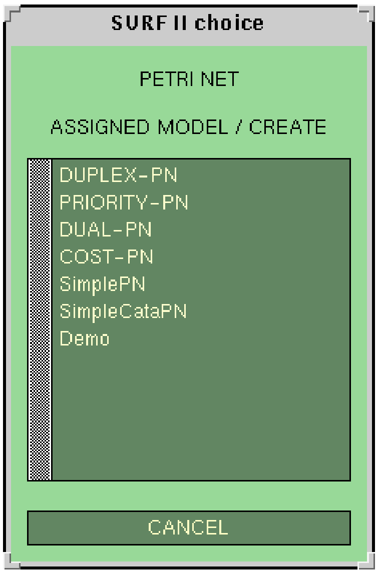


Fig A3.35 Dialog box to select model.

Select our Model: SimplePN.

The following dialog box appears: Fig A3.36 (a). Enter the model name: Fig A3.36 (b).

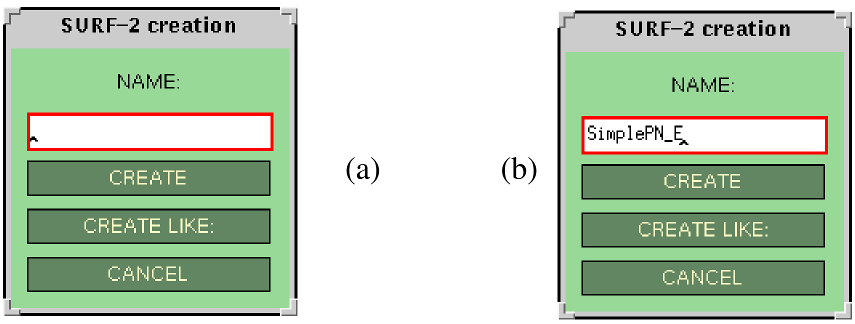


Fig A3.36 Dialog box for Model Creation.

Then use CREATE Command (Fig A3.37). The Assigned Model Editor dialog box is displayed it allow to assign numerical value or global value to each parameter:

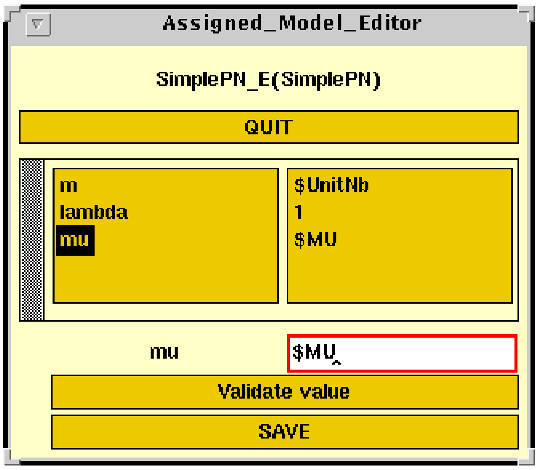


Fig A3.37 Assigned Model Editor window.

A2.2.3 Model Folder creation

In Main Menu point model FOLDER <Command> to display Model Folder Menu:

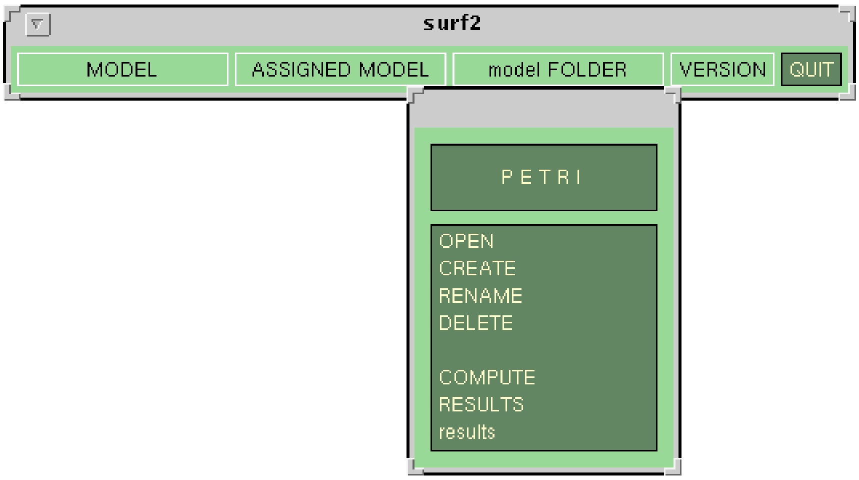
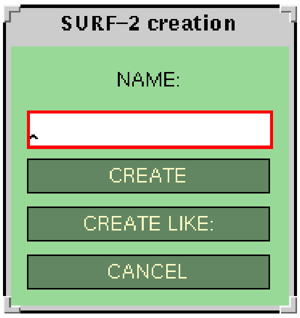


Fig A3.38 Main Menu and Model Folder Menu

Use CREATE Command and the following dialog box appears: Fig A3.2 (a). Enter the model name: Fig A3.2 (b).



(a) (b)

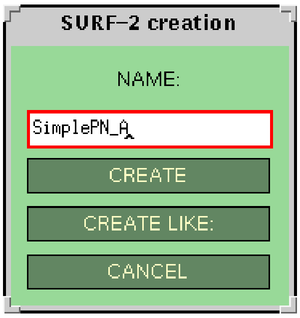


Fig A3.39 Dialog box for Model Folder Creation.

Then use CREATE Command (Fig A3.39), the Markov Editor dialog box is displayed:

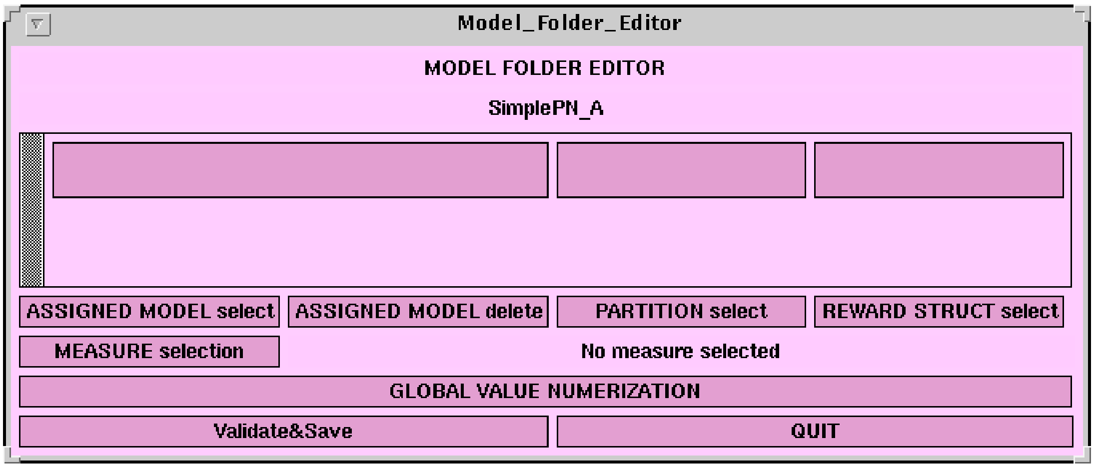


Fig A3.40 Model Folder Editor main window.

Choose Assigned Model SimplePN_E in dialog box displayed by ASSIGNED MODEL select Command:

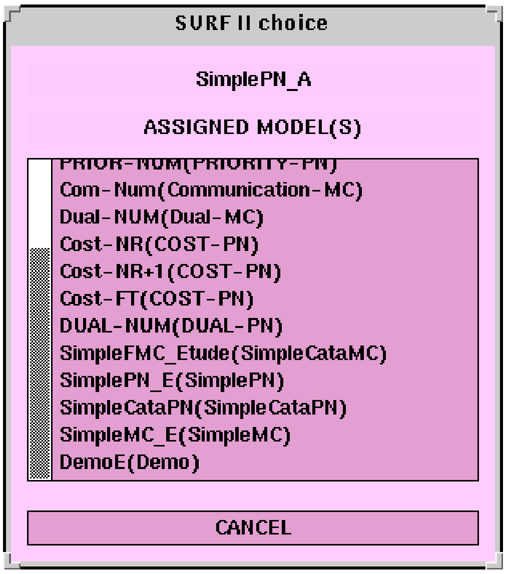


Fig A3.41 Dialog box to select Assigned Model.

Choose partition "part" in dialog box displayed by PARTITION select Command:

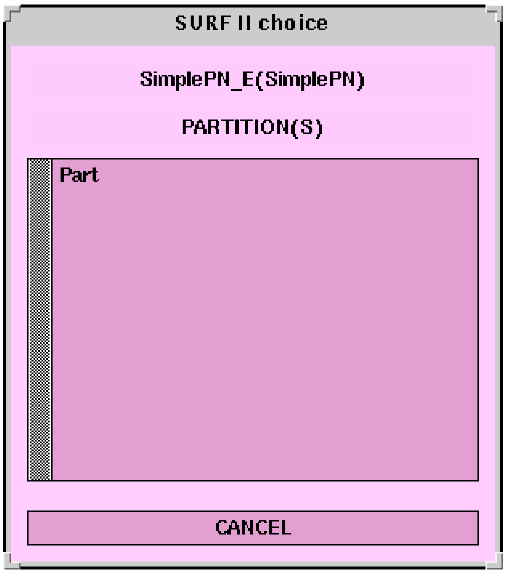


Fig A3.42 Dialog box to select a partition.

Choose Reliability in dialog box displayed by MEASURE selection Command:

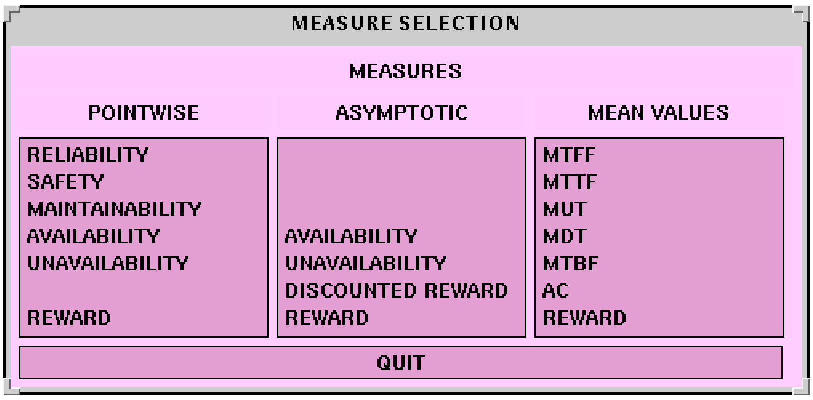


Fig A3.43 Dialog box to select a Measure.

Global values were defined in Assigned Model Editor: we must assign these values. Use GLOBAL VALUES NUMERIZATION Command the following dialog box is displayed:

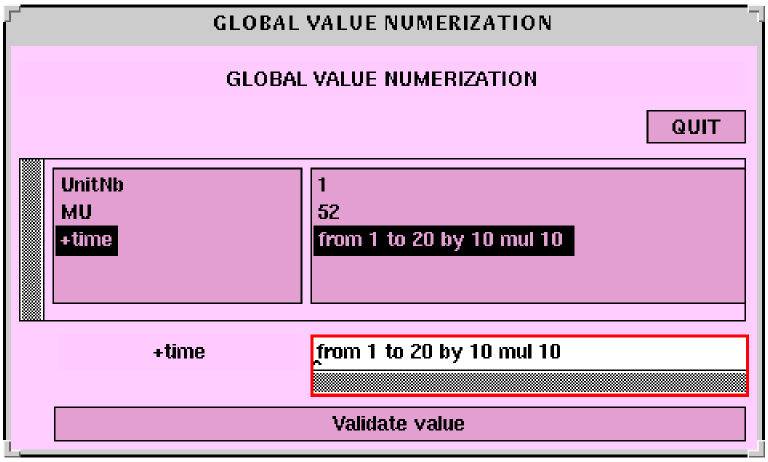


Fig A3.44 Dialog box to global value numerization.

For each global value[2] define a numerical value or a set of values.

Fig A3.45 shows the new aspect of Model Folder Editor main window:

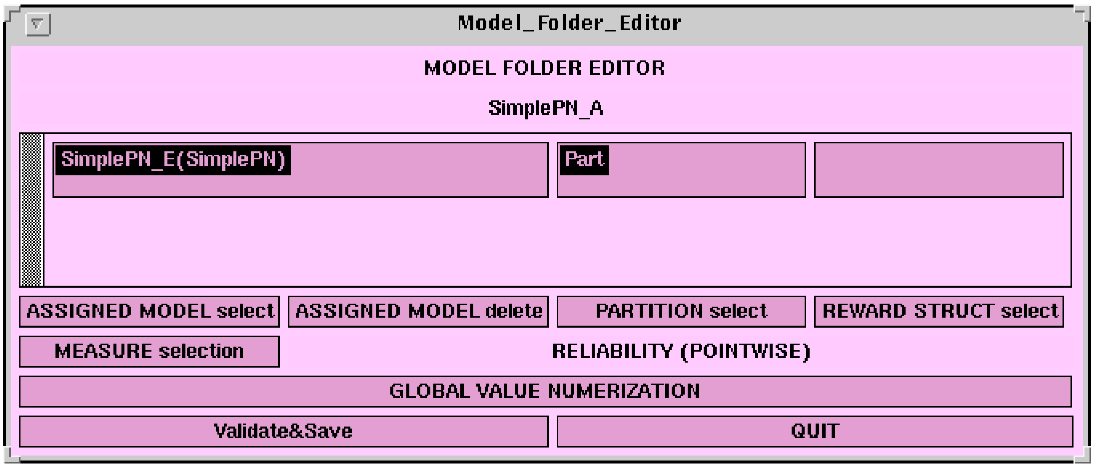


Fig A3.45 Dialog box to global value numerization.

Quit accessories windows and use Validate&Save Command of Model Folder Editor main window(Fig A3.45).

3.2.2.4 Computation

Use Command COMPUTE in Model Folder menu. The following window allows to choose your Model Folder SimplePN_A

[2] Time is a global value since reliability is a pointwise measure.

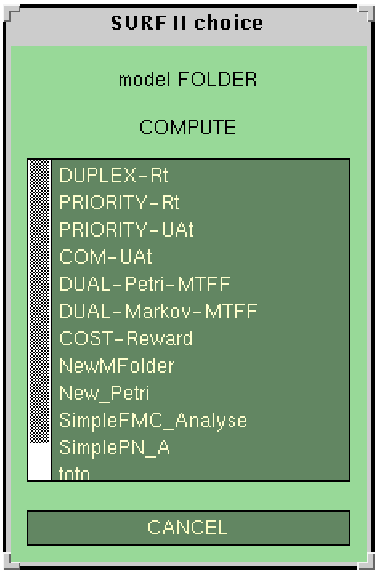


Fig A3.46 Dialog box to select a Model Folder in order to compute a measure.

Computation is done, some messages give information about computing process:

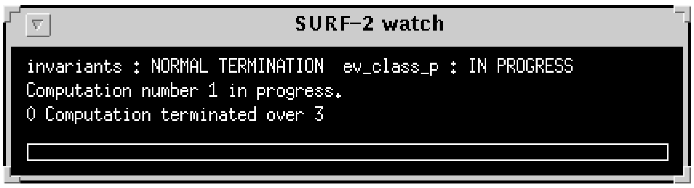


Fig A3.47 A message during computation.

Some complementary informations are printed in window where SURF-2 is executed.

A2.2.5 RESULTS

When computation is finished we can examine results.

RESULTS Command of Model Folder menu displays dialog box (Fig A3.48). Select our Model Folder SimplePN_E.

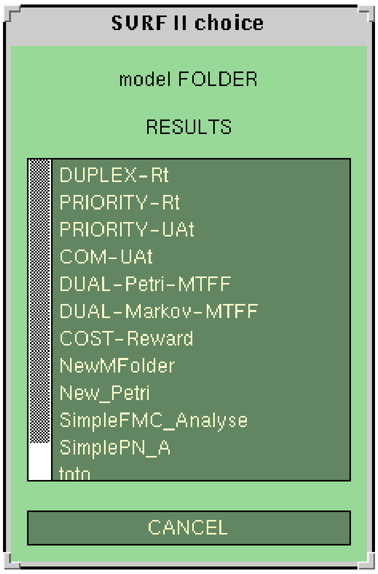


Fig A3.48 Results: Model folder selection.

The following window is displayed:

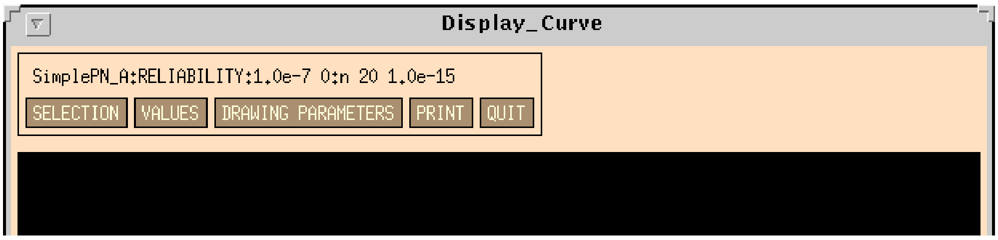


Fig A3.49 Display Curve window.

Use SELECTION Command, dialog box(Fig A3.50)is displayed. We select Assigned Model SimpleMC_E and global value time.

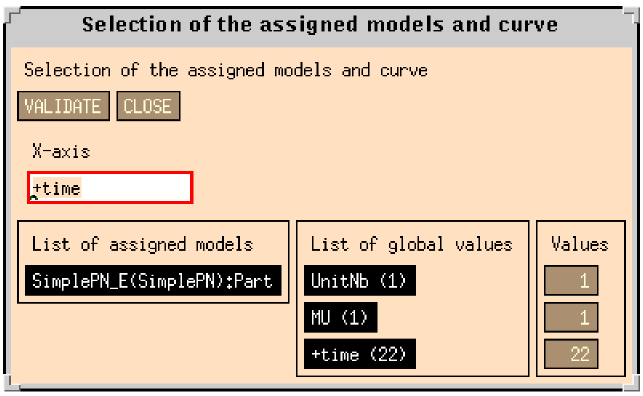


Fig A3.50 Dialog box to select Assigned Model and Curves.

Use Command VALIDATE (Fig A3.50) to display curve of Reliability:

Result is identical to the result for Markov Model (Fig A3.24 & Fig A3.51).

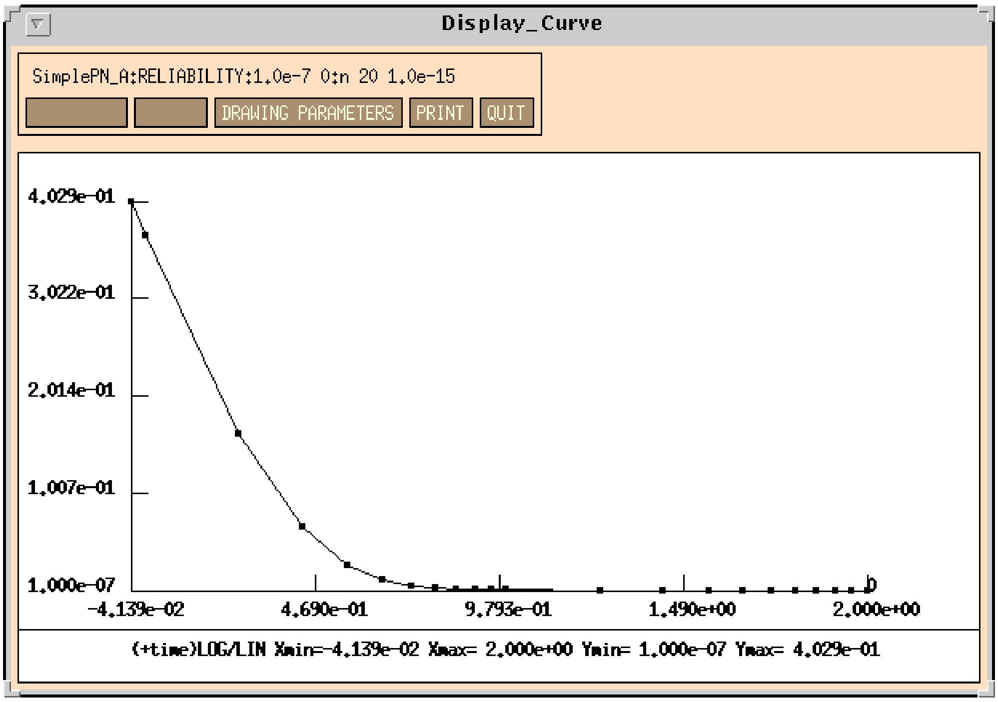


Fig A3.51 Reliability.

A2.2.6 Access to Markov Chain generated.

Generated Markov Chain can't be access directly.

In Main Menu point Model <Command> to display Model Menu. Switch on Markov Modelization.

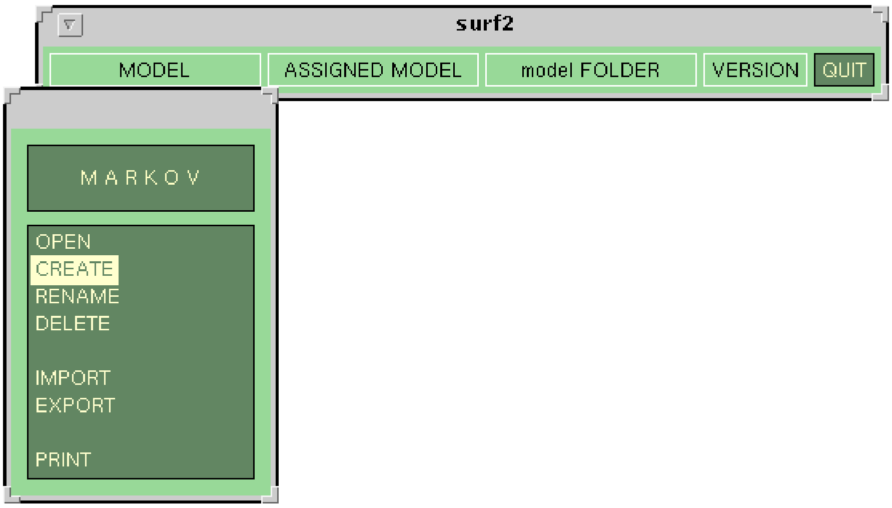


Fig A3.52 Main Menu and Model Menu.

Use CREATE Command. In the following dialog box Fig A3.53 (a) give the name to the Markov Chain: "MarkovGenerated". Click Command CREATELIKE Fig A3.53 (a) and select E-SimplePN_E_P-SimplePN Markov Chain Fig A3.53 (b).

> *Generated Markov Chain mane is:*
> *E-<Assigned Model name>_P-<Petri Net Model name>*

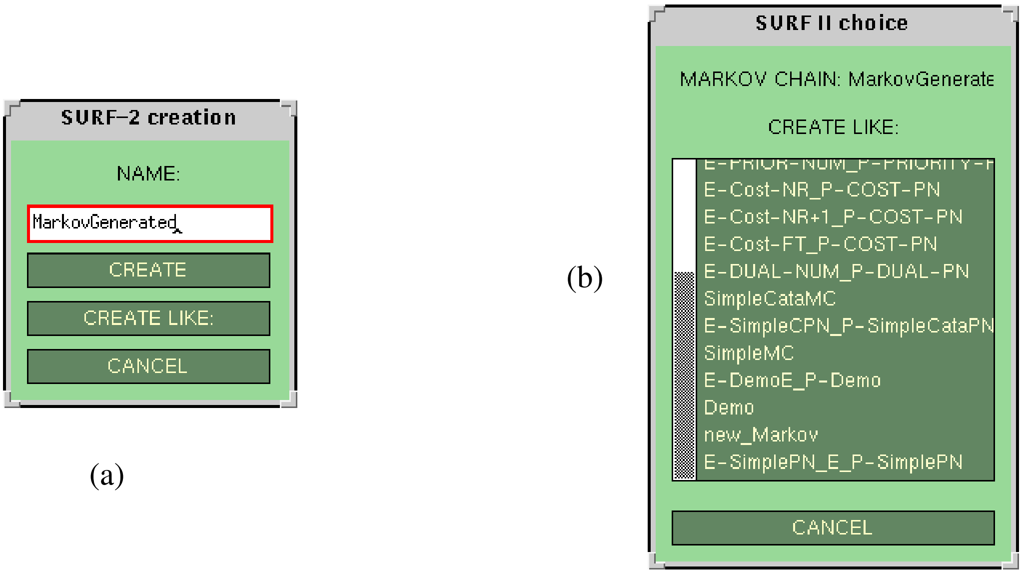


Fig A3.53 Access to generated Markov Chain.

The generated Markov Chain is displayed. Use initObject function to obtain complementary informations.

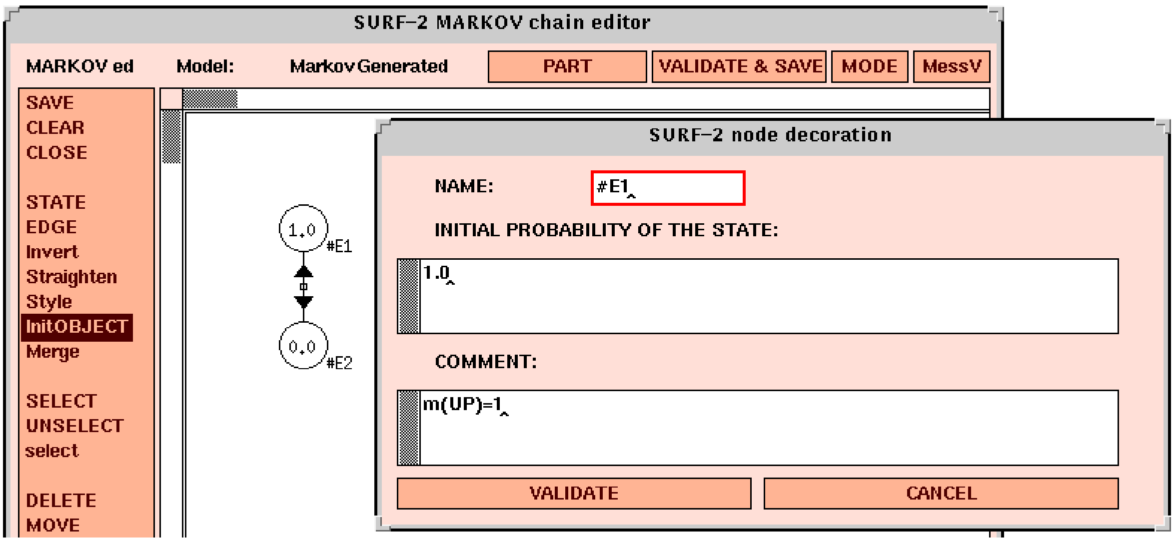
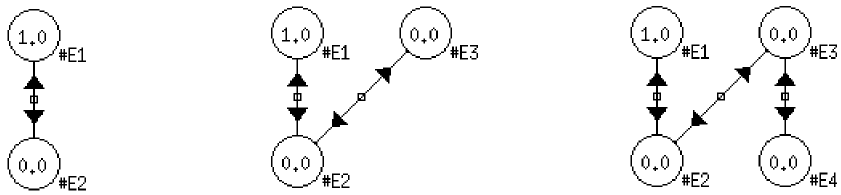


Fig A3.54 Generated Markov Chain.

To display Generated Markov Chain use a Model Folder with just one Assigned Model.

The Markov Chain generated depends on the number of units (m) in the model (init value of place UP marquing).



m= 1 m=2 m= 3

A2.2.7 Display Invariants.

In Main Menu point model FOLDER <Command> to display Assigned Model Menu:

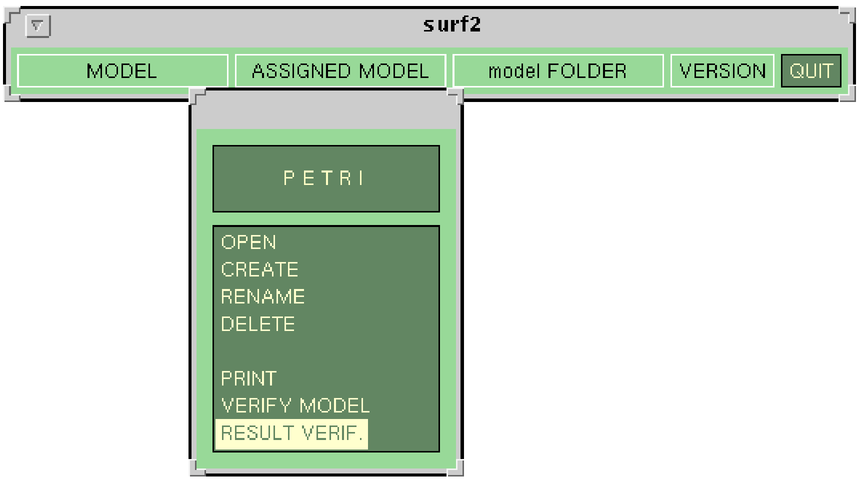
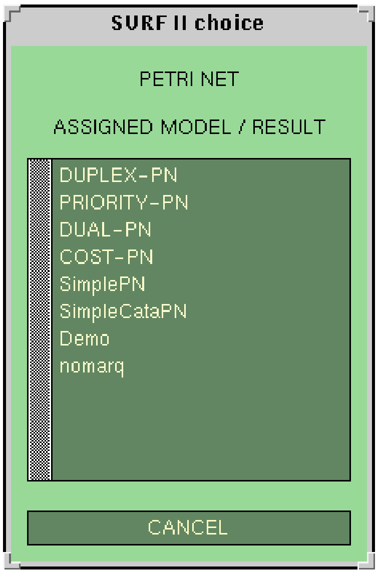


Fig A3.55 Main Menu and Assigned Model Menu.

First select Model: SimplePN (Fig A3.56 a) and after select Assigned Model: SimplePN_E (Fig A3.56 a).



(a)

(b)

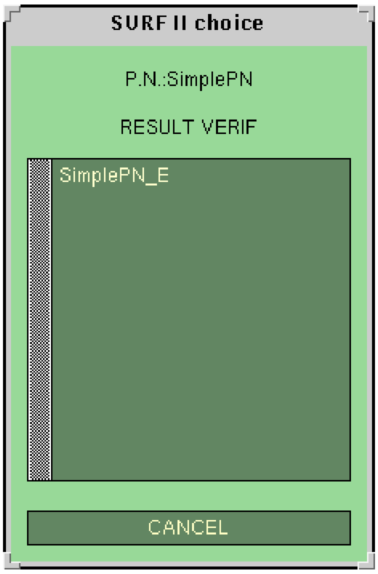


Fig A3.56 Dialog box to select Model (a) and Assigned Model (b).

The following dialog box is displayed, click DISPLAY Command.

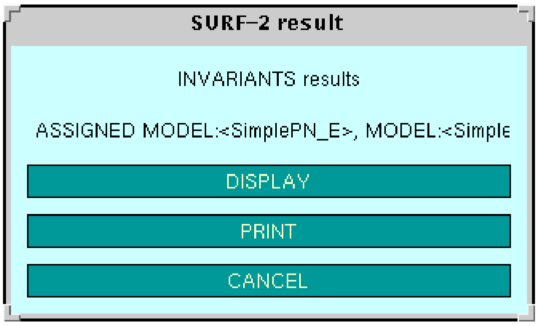


Fig A3.57 Dialog box for property verification results.

Invariants are displayed:

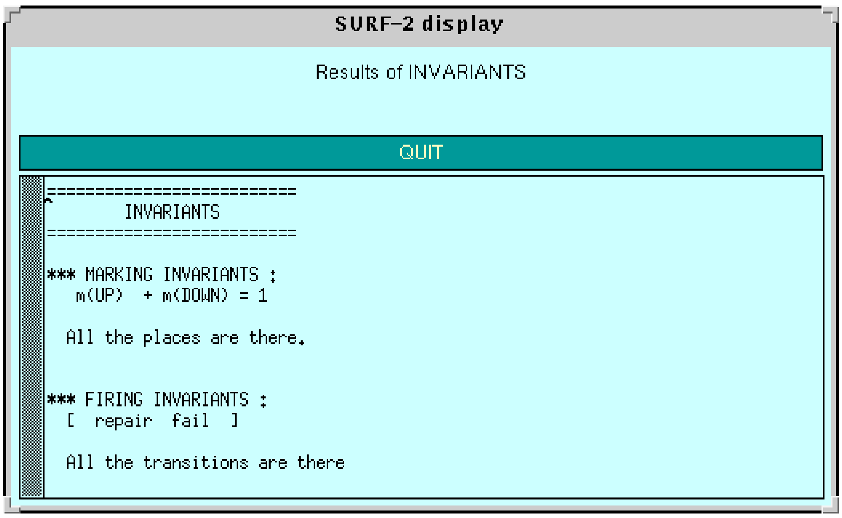


Fig A3.58 Window displaying the results.

A2.2.8 Another computation.

We redefine global values numerization in Model Folder SimplePN_A:

UnitNb	1 2 3
MU	104 52 12
+time	from 1 to 20 by 10 mul 10

Fig A3.59 Global values numerization.

We can display curves of Reliability according to number of units for a repair rate of 52 repairs in an year:

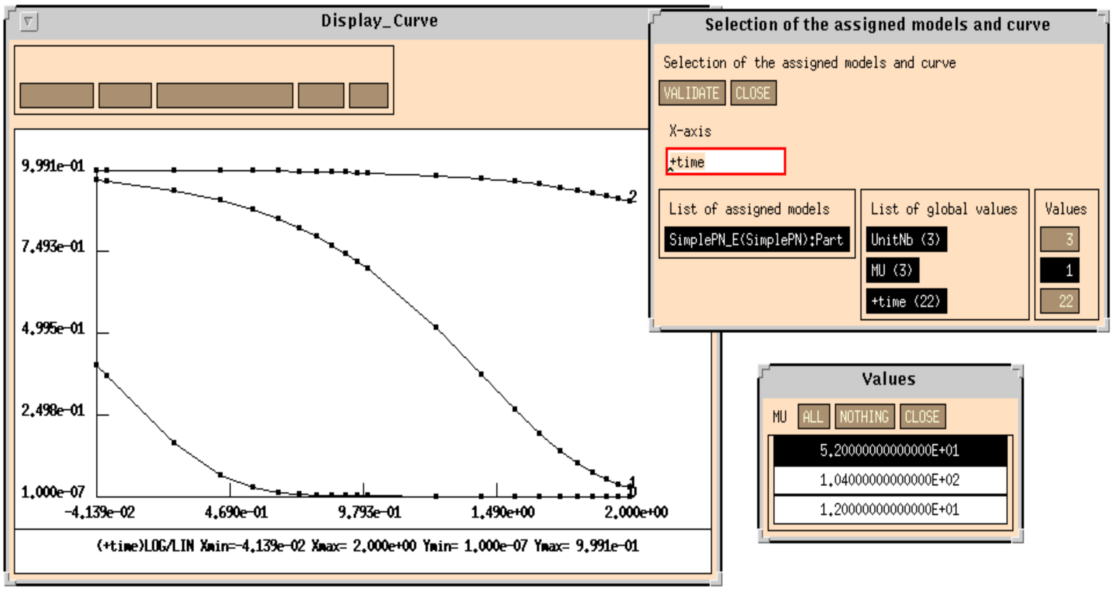


Fig A3.60 Reliability in case of 1, 2 and 3 units.

We can display curves of Reliability according to repair rate for a system of 3 units:

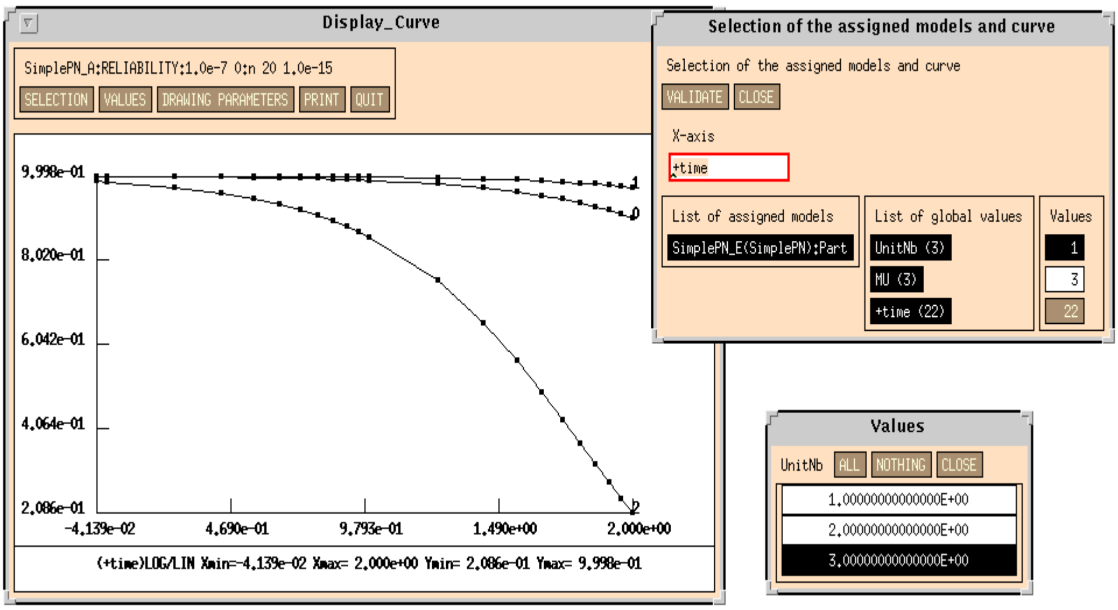


Fig A3.61 Reliability according to repair rate.

INSTALLATION PROCEDURE

WARNING !!!!!

SURF-2 V3.0 can read models created with previous versions of SURF-2 but SURF-2 V2 cannot read models created or modified with SURF-2 V3.

CONTENT OF THE CARTRIDGE/CD-ROM

The cartridge/CD-ROM contains the "**SURF_V3.0**" directory including:

- "**SURF_REF**" directory that contains the following subdirectories:
    - "**BDVIERGE**": backup of initial database.
    - "**SURF2_XINIT**" : containing two configuration files for the X-windows manager (color and back&white monitors).
    - "**EXE**": executable programs.

- "**BDSURF**" directory which is the initial database,

- "**DEMO_DATABASE**" directory containing some simple modelling examples,

- "**INSTALL**" directory containing two files for SURF-2 installation:
    - "**README**" file describing the installation procedure.
    - "**start-surf.model**" script shell file used to start SURF-2.

SYSTEM ADMINISTRATOR'S TASKS

1. INSTALL THE SURF-2 SOFTWARE:

Take the root's identity:

```
#su root
```

- if you have got a tape:

1-1 Copy the content of the tape into a temporary directory and go to the INSTALL directory.
#cd tmp; tar xpvf [/dev/rst0, /dev/rmt/0]

• if you have got a CD-ROM:
1-1 Go to the mounted cd-rom INSTALL directory.

1-2 Run the **install.surf** script.
#cd tmp/SURF_V3.0/INSTALL
or
#cd /cdrom/SURF_V3.0/INSTALL
#install.surf
This scripts allows to choose where to install SURF-2 and which system version you will use.

1.3. Choose between color or white monitor by copying the appropriate file "SURFII.bw" or "SURFII.color" into "SURFII" file in the directory SURF2_XINIT.
(*default configuration file is SURFII.color*)
:

```
#cd <SURF2_HOME>/SURF_V2.0/SURF_REF/SURF2_XINIT
#cp SURFII.bw SURFII
 or
#cp SURFII.color SURFII
```

2. INSTALL THE USER'S ENVIRONMENT FOR RUNNING SURF-2

Take the user's identity:
```
#su user
```

In SURF-22 distribution, there are two database directories:
- "BDSURF" containing the initial empty database,
- "DEMO_DATABASE" containing some simple modelling examples.

Both can be installed for user.

Go to the INSTALL directory where you have installed SURF-2 and run the **install.user** script
#cd <SURF2_HOME>/SURF_V3.0/INSTALL
#install.user

This script allows you to install one or both databases and SURF-2 scripts:
- **surf** (for BDSURF),
- **demo-surf** (for DEMO-DATABASE) in ~/bin directory

Make sure that the user has the path "~/bin" in his "PATH" environment variable.

3. TEST THE CORRECT INSTALLATION BY RUNNING SURF-2 UNDER THE USER'S ENVIRONMENT

SURF-2 must run under a properly installed X-Windows graphic environment.

Make sure that the user has the path "/usr/openwin/lib"or X-Windows equivalent in his "LD_LIBRARY_PATH" environment variable.

The "**surf**"and "**demo-surf**" script shell files can start SURF-2 on SunOS 4.1.x or Solaris2.x system depending on the installation you have chosen.

#su - user

#setenv DISPLAY host:0.0

#surf

or

#demo-surf

These scripts support optional arguments. Syntax is:

surf [database directory] [options]

options list:

-p | -P printer name

-e | -E import/export directory

-h |-H | -help this help

Addendum 1

IMPORT/EXPORT OF MODELS

1.1 OVERVIEW.

Model Importation and Exportation functions have been completely rewritten in order to make them easier to use. The exported model is more complete and can be easily moved. Main improvements are:

- assigned models are imported/exported with the model,
- names of models and assigned models are saved,
- one file contains all data corresponding to an exported model and can be easily moved or copied.

1.2 EXPORTATION.

Click on EXPORT and then select the model you want to export (Petri or Markov)(Fig. 1). When this is done, a new file named *Model_Name.petri* or *Model_Name.markov* is created in the EXPORT directory. This file contains:

- the model,
- the associated Markov model if it exists,
- all the corresponding assigned models.

This file can be moved or copied elsewhere in order to import it on another database.

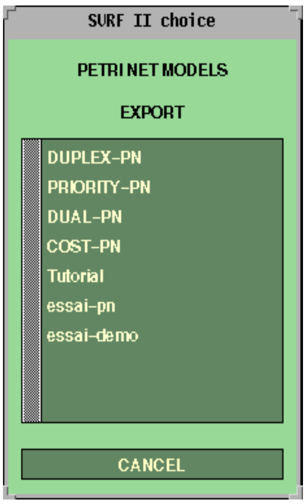


Fig. 1 Export Window.

> *Filename is used as model name during importation. If you change it, the name of the imported model will change accordingly.*
> *Do not remove the suffix .petri or .markov as it is used by the IMPORT function to recognize the exported models.*

1.3 IMPORTATION.

When you select IMPORT, a listing of all Petri Net models or Markov models inside the EXPORT directory is shown (Fig. 2). Select one of them and SURF-2 will import the model(s) and assigned model(s) in your database.

The names are identical with those in the original database except:

- if you have changed the name of the file,
- if there is a model or an assigned model in your database which has already the same name: in this case, model or assigned model is imported under a new name which is *name.suffix* (suffix is a number starting from 1).

Fig. 2 Import Window

Addendum 2

EXPORTATION OF RESULTS

2.1 OVERVIEW.

You can now export your results in a formatted text file that you can edit or import in a spreadsheet. This file contains all the information related to the model folder (names of models, assigned models, measure, ...) and the values of results and of global variables.

2.2 HOW TO.

First, verify that you have defined correctly the IMPORT/EXPORT directory either in your startup file or in the ENVIRONMENT window (MISC Menu). The results file will be saved into this directory under the name *model_folder_name.export*.

Click on the RESULTS menu and select one of the model folders. When the results window shows, just click on EXPORT button (Fig. 1) and a new file containing the results is created (Fig. 2).

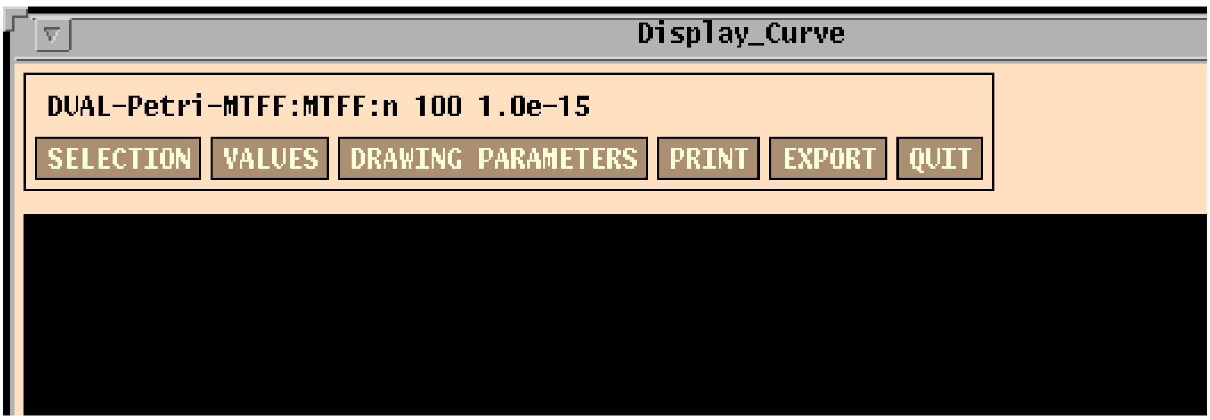


Fig. 1 Results Window

Fig. 2 Message Window

2.3 FILE DESCRIPTION

The file is composed of a set of lines, each line may contain several values/strings separated by tabs.

- first line contains the **name of the model folder** and the **measure**,
- for each assigned model, format is as follows:
  - a line containing the **name of assigned model**, the **name of model**, the **name of partition** and if necessary the **name of reward**,
  - in case of reward measures, a line with the string **"discount"** and its **value**,
  - a line with the **names of global values** and the string **"result"**,
  - a set of lines with **the combinations of global values** and of **result values**.

Here is an example with one assigned model, and one global value:

```
DUAL-Petri-MTFF MTFF
DUAL-NUM        DUAL-PN     Part1
LS      result
1.00000000000000E-04    1.19264069264069 3E+03
2.50000000000000E-04    1.12222222222222E+03
5.00000000000000E-04    1.03333333333333E+03
```

Addendum 3

MODEL FOLDER PRINTING

3.1 OVERVIEW.

You can now print a model folder and all its content (assigned models, partitions, global values, ...).

3.2 HOW TO.

You just have to click on the new PRINT button (see Fig. 1 below) which is shown on the Model Folder Editor window.

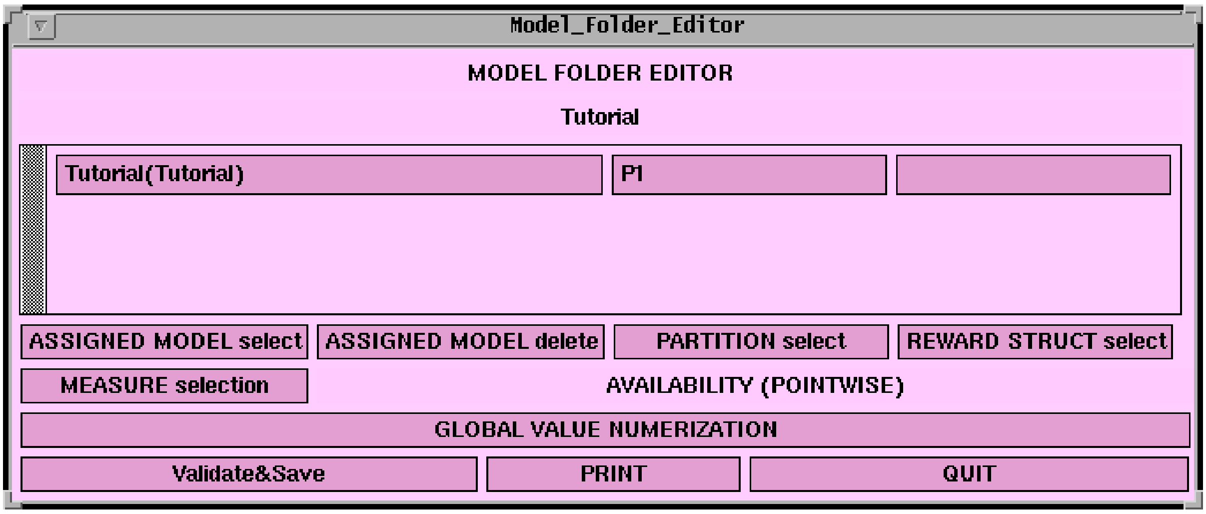


Fig. 1 Model Folder Editor window.

The printout contains the name of the model folder, the name of the measure, the set of assigned models and corresponding partitions, and all the global values.

Addendum 4

EDGES DISPLAY

4.1 OVERVIEW.

In the Model Editor, you can select a node and display only edges coming from or to this node.

4.2 HOW TO.

Select a node (place or transition), and click on mouse third button. All edges but those coming from or to this node will be hidden (Fig. 1). Select another node and click again on third button and edges related to this node will be shown.

Click on the second button and all edges are shown (Fig. 2).

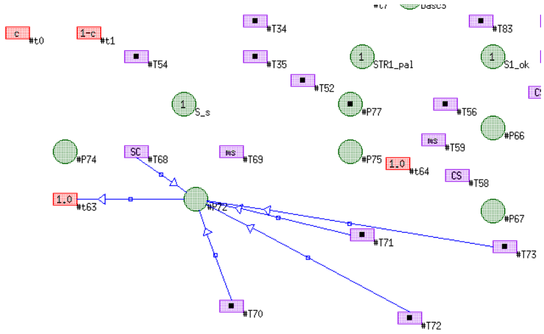


Fig. 1 Only #P72 related edges are shown.

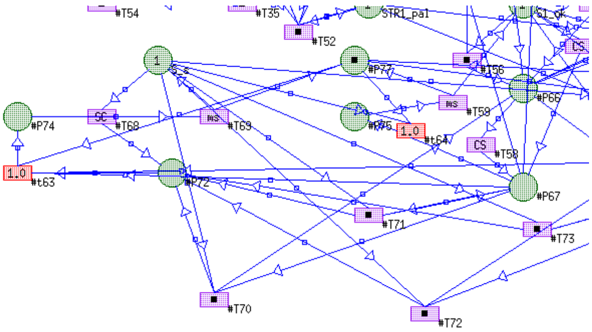


Fig. 2 All edges are shown.

Addendum 5

PREDICATES

5.1 OVERVIEW.

Predicates allow user to add textual firing conditions to transitions. Firing condition in a Petri Net depends on the input places of transitions as well as the weight of the input edges. The transition is fired when the marking of all input places are greater or equal of connecting edges in case of normal edges or when the marking is equal to zero in case of inhibitor edges.

Up to now in SURF-2, these edges and transitions were drawn by user. Thus the conditions were expressed only graphically.

But in large models, the number of edges can be dramatically high making the Petri Net unreadable and very difficult to check and modify. We have thus introduced textual firing conditions we named predicates (actually they are predicates). These predicates enable to define conditions without the need of drawing all. The firing condition is thus an AND function of predicate and existing graphic condition.

A predicate is composed of boolean variables (place names with weight attributes) connected by logical operators NOT, AND, OR (full syntax is described further). When the predicate becomes true and the graphic conditions (marking of places graphically connected) are true, the transition is fired. Boolean variables become true according to their weights and the marking of related places.

Predicates do not fully substitute graphic edges and transitions, they must be added to an existing correct Petri Net.

5.2 HOW TO.

Select Init Object command and click on a transition, the transition attributes window appears (Fig. 1). You can now enter your firing condition in the predicates field according to the syntax described below.

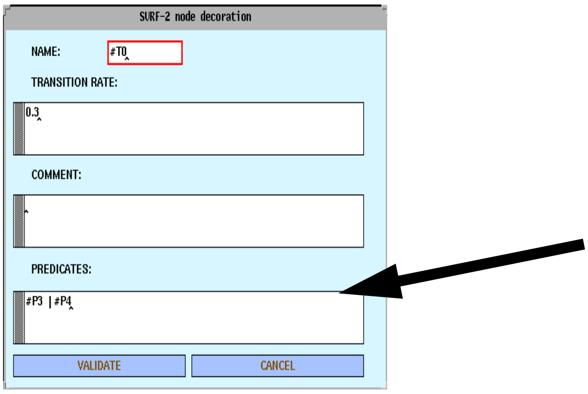


Fig. 1 Transition attributes with predicates.

Transitions containing predicates are highlighted as shown in Fig. 2.

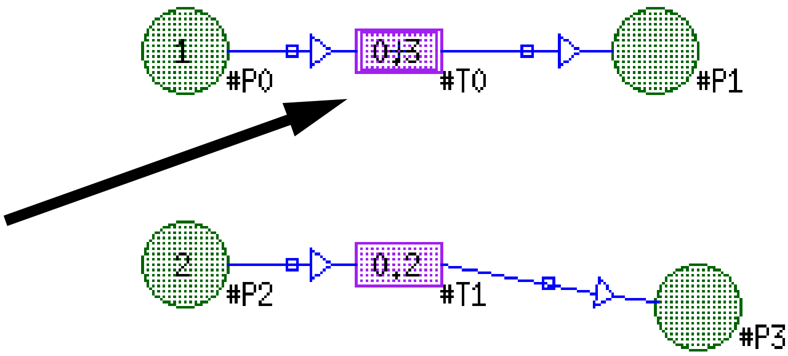


Fig. 2 Transition #T0 contains predicates.

5.3 SYNTAX

A predicate defines the firing condition in terms of a boolean combination of place marking test. The marking test of one place can be:

- "marking of this place is equal to 0",
- "marking of this place is greater or equal than a non-null positive number".

The marking of a place is represented by an optional weight attribute associated with the name of the place (*name*:*weight*). The default value of marking is 1.

P is TRUE when marking of *P* is greater or equal than 1.

P:n, with *n* > 0, is TRUE when marking of *P* is greater or equal than *n*.

P:0 is TRUE when marking of *P* is equal to 0.

Connect operators used in predicates are **&** (AND), **|** (OR) and **~**(NOT).

The priority order of operators is - from highest to lowest - **~**, **&**, **|**.

A predicate must be written as a sum of min-term, the full syntax is as follow:

<predicate> ::= <min-term> [**|** <predicate>]

<min-term> ::= <place-info> [**&** <min-term>]

<place-info> ::= <place-name>

<place-info> ::= <place-name>**:**<marking>

<place-info> ::= **~** <place-name>

<marking> ::= 0...

Note: Parenthesis are not allowed.

Predicate evaluation:

place-infoA & place-infoB

is TRUE if place-infoA and place-infoB are TRUE.

min-termA | min-termB

is TRUE if min-termA is TRUE or min-termB is TRUE

~ place-name

is TRUE if marking of place-name is 0 (= *place-name***:***0)*

Examples:

P1:2 & P2 | P3

is TRUE if marking of P1 is greater or equal than 2 and marking of P2 is greater or equal than 1, or marking of P3 is greater or equal than 1.

~ P1 | P2:4 & P3:3

is TRUE if marking of P1 is equal to 0 or, if marking of P2 is greater or equal than 4 and marking of P3 is greater or equal than 3.

5.4 SEMANTICS

The different components of a predicate do not change the semantics of Petri Nets. They are only a textual way to represent graphic elements, such as edges and transitions. The semantics is described below:

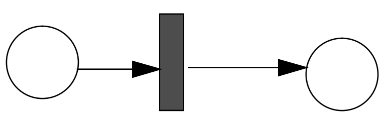


Fig. 3 Original graphic.

place:n

an input bidirectional edge, whose weight is n, between the named place and the selected transition (Fig. 4).

place

an input bidirectional edge whose weight is 1(Fig. 4).

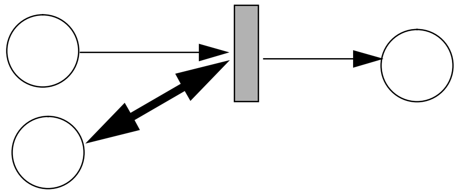


Fig. 4 Simple predicate.

~place or **place:0**

an input inhibitor edge (Fig. 5).

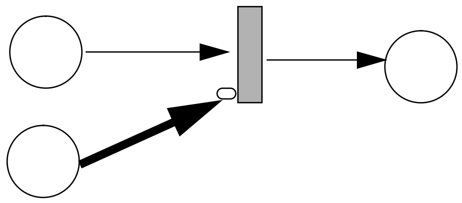


Fig. 5 NOT.

place1:n1 & place2:n2 & ...

a set of input bidirectional edges between named places and selected transition (Fig. 6).

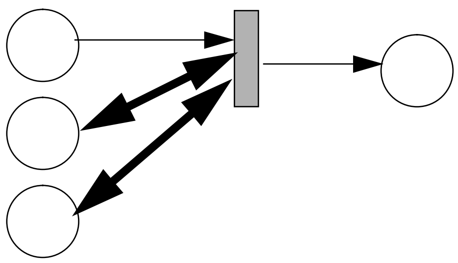


Fig. 6 AND.

place1:n1 & place2:n2 & ... | place11:n11 & ...

the first min-term represents what is described above, the second min-term is a new transition, whose rate is equal to selected one's, a set of bi-directional edges between named places and this transition, and output edges from this transition to output places of selected transition. New output edges have the same weights as existing ones (Fig. 7).

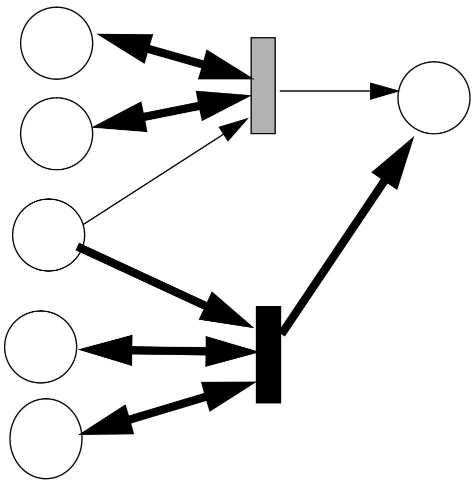


Fig. 7 OR.

Predicates do not create new tokens nor delete existing ones. The marking of referenced places in the predicate do not change when transition is fired.

The actual firing condition of a transition is "graphic condition AND textual condition". Both must be TRUE to fire it.

Do not add a predicate, to a transition:

- *which does not have any input edges,*
- *which has only input bidirectional edges.*

Addendum 6

COMPUTATION

6.1 COMPUTATION METHODS.

There are 3 computation methods in SURF-2: CGS, Integration, Randomisation

6.1.1 CGS

CGS method is used for asymptotic and mean measures. It is a bounded method whose limits are the maximum number of steps and the accuracy defined by the SURF2_WIZARD variable in surf startup script. Default values are 200 and 1e-15.

CGS method may not converge in some cases, if values are too small or the difference between them is too large. An error message is then displayed.

First solution consists in increasing the number of steps (you should not decrease accuracy), but it does solve the problem in a few number of cases. A better method would be to change the parameters of your model and of your model folder.

6.1.2 Integration

Integration method is used to compute pointwise measures. It is generally faster than randomisation for large differential systems. Based on CGS, it is also bounded by the same parameters: accuracy and number of steps.

Same problem may arise than in CGS, method may not converge. Solutions are identical.

But in case of no-convergence of integration method, SURF-2 automatically switches to randomisation method and tries to redo the computation.

6.1.3 Randomisation

Randomisation method is generally used is small differential systems. Slower than integration method, it is more robust and can solve some sytems unsolved by integration. Unfortunately the computation is not bounded and when method cannot succeed to give results or when elapsed time seems too long, user has to stop the computation manually.

An information is then displayed about the step (time) computation stopped. User can then split the computation interval according to theses indications.

Do not abort the computation too early as the the randomisation process can be long.

6.2 RESULTS

Most results are probabilities and must be in the interval [0,1], but in some cases, due to the inaccuracy of floating point computation, results can be greater than 1 or lower than 0. In this case, a warning message is displayed. Be careful, because results are automatically rounded to 0 or 1 by SURF-2.

If the measures are such low or high, it means that your parameterized model is too or not enough dependable and that you have to change it.

Addendum 7

INVARIANTS

7.1 USER-DEFINED INVARIANTS.

User-defined place invariants can be explicitly defined in the Petri Net model. Select the INV button (See Chapter 6, Fig 6.25) and enter the invariants you want to be verified following syntax shown in Appendice 1. The place invariants are verified by the verify function (See Chapter 9).

It has to be to noticed that all invariants (place and transition invariants) are automatically detected and displayed by the verify model function. But in large models, where number of invariants can be high, it may be helpful for model validation to define a priori some invariants which must be in the model.

Here is an example of user-defined place invariants:

```
inv1:m(P1)+2*m(P2)+m(P3)=5;
inv2:3*m(P2)-m(P4)=2;
```

7.2 VERIFY INVARIANTS

The Model verify function shows all transition and place invariants of the assigned model. The result is divided in three parts shown below.

First part consists in all place invariants

```
*** MARKING INVARIANTS :
m(mach_OK)  + m(Wait)  + m(Workshop) = 4
m(Free_Tech)  + m(Workshop) = 1

All the places are there.
```

Second part consists in all transition invariants

```
*** FIRING INVARIANTS :
[  Failure  Repair  RendezVous  ]

All the transitions are there
```

Third part consists in user-defined invariants.

```
Invariant inv :   ok !

Invariant inv2 :   not invariant !
```

Appendix 4

THE MATHEMATICAL EXPRESSIONS OF THE DEPENDABILITY MEASURES

1. Characterization of the basic dependability measures

1.1 Notation

E is the set containing the states of the Markov chain describing the model behavior, where **|E|** is the cardinal number of **E**.

Concerning the expression of the measures, we assume the knowledge of:

- the transition rates between states of the graph modeling the system behavior: λ_{ij}, $\forall$ i, j $\in$ {1, 2, …, |E|}, i j,
- the initial distribution in the states: p_{0i}, $\forall$ i $\in$ {1, 2, …, |E|}
- the associated reward structure (when it has been defined):

  - yield rates proportional to the time spent in each state of the graph: y_i , $\forall$ i $\in$ {1, 2, …, |E|},

  - bonus acquired for each transition between the states of the graph: b_{ij}, $\forall$ i, j $\in$ {1, 2, …, |E|}, i j.

Subsequently, X is subset the states of the graph for which the measure will be calculated, and Y the complementary subset ; X and Y define a partition of E. The measures are significant only if X corresponds to a set of connected states (ergodic Markov subchain). These measures are expressed using the following matrix notation.

Transition rates matrix: $\Lambda = [\ \lambda_{ij}\ ] = \begin{bmatrix} \Lambda_{xx} & \Lambda_{xy} \\ \Lambda_{yx} & \Lambda_{yy} \end{bmatrix}$

λ_{ij} : events /unit of time ; $\lambda_{ij} \in \Re^{+}$ (i j) ;$\lambda_{ii} \in \Re^{-}$ et $\lambda_{ii} = - \sum_{j=1,\, j\neq 1}^{|E|} \lambda_{ij}$

thus : $\sum_{j=1}^{|E|} \lambda_{ij} = 0$

Initial probability vector[1] : $\mathbf{P}_0 = [\ p_{0i}\ ] = [\ \mathbf{P}_{X0}\ \ \mathbf{P}_{Y0}]$

p_{i0} : probability ; $0 \le p_{i0} \le 1$; $\sum_{i=1}^{|E|} p_{i0} = 1$;

otherwise: $\mathbf{P}_0 = \mathbf{P}_0$ so that $p_{10} = 1$ and $p_{j0} = 0$, $j \in \{2, 3, \ldots, |E|\}$

Yield rate vector (sojourn time in each state) : $\mathbf{Y} = [\ y_i\ ] = [\ \mathbf{Y}_X\ \ \mathbf{Y}_Y]$

y_i : reward/unit of time; $y_i \in \Re$

Bonus matrix (transitions between states): $\mathbf{B} = [\ b_{ij}\ ] = \begin{bmatrix} \mathbf{B}_{XX} & \mathbf{B}_{XY} \\ \mathbf{B}_{YX} & \mathbf{B}_{YY} \end{bmatrix}$

b_{ij} : reward ; $b_{ij} \in \Re$; $b_{ii} = 0$ (matrix with a zero diagonal)

Discount rate (cost analysis of the reward — discounted reward) : γ

γ : depreciation/unit of time ; $\gamma \in \Re^+$; $s(0) = \exp(-\gamma\, t)\, s(t)$.

Transient-state probabilities matrix: $\Omega(t) = [\ \omega_{ij}(t)] = \begin{bmatrix} \Omega_{XX}(t) & \Omega_{XY}(t) \\ \Omega_{YX}(t) & \Omega_{YY}(t) \end{bmatrix}$

$\omega_{ij}(t)$: probability to be in the state **j** at **t** time, the process being in state **i** at the initial time 0),

$0 \le \omega_{ij}(t) \le 1$; $\sum_{j=1}^{|E|} \omega_{ij}(t) = 1$

Transient-state probabilities vector: $\mathbf{P}(t) = [\ p_i(t)] = [\ \mathbf{P}_X(t)\ \ \mathbf{P}_Y(t)] = \mathbf{P}_0\ \Omega(t)$

$p_i(t)$: probability ; $0 \le p_i(t) \le 1$; $\sum_{i=1}^{|E|} p_i(t) = 1$

Asymptotic-state probabilities vector: $\mathbf{P} = \lim_{t\to} \mathbf{P}(t) = [\ p_i] = [\ \mathbf{P}_X\ \ \mathbf{P}_Y]$

p_i : probability ; $0 \le p_i \le 1$; $\sum_{i=1}^{|E|} p_i = 1$

Mean sojourn times vector: $\mathbf{T} = [\ t_i] = [\mathbf{T}_X\ \ \mathbf{T}_Y]$$t_i$: unit of time ; $t_i \in \Re^+$

1. All the considered vectors are line-vectors.

Sojourn times in-state matrix in interval [0;t] : (t) = [$\sigma_{ij}(t)$] $= \int_0^t \Omega(u)du$

$$= \begin{bmatrix} \Sigma_{XX}(t) & \Sigma_{XY}(t) \\ \Sigma_{YX}(t) & \Sigma_{YY}(t) \end{bmatrix}$$

$\sigma_{ij}(t)$: unit of time (expected sojourn time in state **j** in the interval [0,t], the process being in state **i** at time 0) ; $0 \le \sigma_{ij}(t) \le t$ and $\sum_{i=1}^{|E|} \sigma_{ij}(t) = t$

Sojourn-times-in-state vector for a given interval: $\mathbf{L}(t) = [l_i(t)] = \mathbf{P}_0(t) = \int_0^t \mathbf{P}(u)du$

$$= [\ \mathbf{L}_X(t)\ \ \mathbf{L}_Y(t)]$$

$l_i(t)$: unit of time (expected sojourn time in state **i** in the interval [0;t]);

$0 \le l_i(t) \le t$ and $\sum_{i=1}^{|E|} \mathbf{l}_i(t) = t$

Identity vector (elements equal to 1) : $\mathbf{1} = [\ \mathbf{1}_X\ \ \mathbf{1}_Y\ ]$

Zero vector (elements equal to 0) : $\mathbf{0} = [\ \mathbf{0}_X\ \ \mathbf{0}_Y\ ]$

Summation vector : $\mathbf{U} = [\ \mathbf{1}_X\ \ \mathbf{0}_Y\ ]$ (elements equal to 1 for X, elements equal to 0 for Y)

$\mathbf{U}' = [\ \mathbf{0}_X\ \ \mathbf{1}_Y\ ]$ (elements equal to 0 for X, elements equal to 1 for Y)

Identity matrix: $\mathbf{I} = [\ i_{ij}] = \begin{bmatrix} \mathbf{I}_{XX} & \mathbf{I}_{XY} \\ \mathbf{I}_{YX} & \mathbf{I}_{YY} \end{bmatrix}$ $\quad i_{ij} = \begin{cases} 1 & \text{if} \quad i = j \\ 0 & \text{if} \quad i \neq j \end{cases}$

$\mathbf{I}_{XY} = 0$; $\mathbf{I}_{YX} = 0$ and $\mathbf{I}_{XX}$, $\mathbf{I}_{YY}$ identity matrix.

Laplace transform: $F(s) = \mathcal{L}\{F(t)\}$

Inverse of the Matrix M : M^{-1}

Transposed matrix $\mathbf{M}$ (or transposed vector of $\mathbf{V}$) : $\mathbf{M}^T$ ($\mathbf{V}^T$)

1.2. States probabilities

The Chapmann-Kolmogorov equation can be written as:

$$\frac{d}{dt}\mathbf{P}(t) = \mathbf{P}(t)\,\Lambda \quad (1.1)$$

The use of Laplace transform enables to write (1.1) in the following form:

$$\mathbf{P}(s) = L\,\{\mathbf{P}(t)\} = \mathbf{P}_0\,(s\,\mathbf{I} - \Lambda)^{-1} \quad (1.2)$$

1.2.1. Pointwise states probabilities

a) Case of an ergodic Markov chain

Solving (1.1) leads to: $\mathbf{P}(t) = \mathbf{P}_0 \exp(\Lambda\, t) = \mathbf{P}_0\, \Omega(t)$(1.3)

The instantaneous probability to be in the states of subset **X** with respect to vector $\mathbf{P}_0$ is thus given by:

$$\mathbf{P}_X(t) = \mathbf{P}_X(t)\,(\mathbf{1}_X)^T = \mathbf{P}_0 \exp(\Lambda\, t)\,(\mathbf{U})^T \quad (1.4)$$

$\mathbf{U}$ is a summation vector whose the $|\mathbf{X}|$ first elements are equal to 1, the other being equal to 0.

$(\bullet)^T$ designates the transposition operation.

b) Case of a Markov chain with an absorbing state

Consider that **Y** is a subset of absorbing states. This case corresponds to the computation of the probability to be in the states of **X** before absorbtion by the states of subset **Y**, with respect to vector $\mathbf{P}_0 = [\,\mathbf{P}_{0X}\ \ \mathbf{0}_Y\,]$. From (1.1) and from the partition of Λ we can write:

$$\mathbf{P}_X(t) = \mathbf{P}_{X0} \exp(\Lambda_{XX}\, t) = \mathbf{P}_{X0}\, \Omega_{XX}(t) \quad (1.5)$$

$$\mathbf{P}_X(t) = \mathbf{P}_X(t)\,(\mathbf{1}_X)^T = \mathbf{P}_{0X} \exp(\Lambda_{XX}\, t)\,(\mathbf{1}_X)^T \quad (1.6)$$

1.2.2. Asymptotic states probabilities

a) Case of an ergodic Markov chain

From relation (1.1), according to an asymptotic behavior, we have:

$$\lim_{t\to} \frac{d}{dt}\mathbf{P}(t) = \mathbf{0} = \mathbf{P}\ \Lambda \quad (1.7)$$

$\mathbf{0}$ is the null vector.

Moreover $\mathbf{P}$ being a stochastic vector,

$$\sum_i p_i = \mathbf{P}\ \mathbf{1}^T = \mathbf{1} \quad (1.8)$$

$\mathbf{1}$ is the identity vector.

To make the computation simpler, relations (1.7) and (1.8) can be combinated as follows:

$$\mathbf{0}_m = \mathbf{P}\ \Lambda_m$$

$$\mathbf{P} = \mathbf{0}_m\ (\Lambda_m)^{-1} \quad (1.9)$$

where $\mathbf{0}_m$ is the modified null vector for which element m only is equal to 1,

Λ_m is the modified transition rate matrix where the m^{th} column is replaced by vector $\mathbf{1}^T$,

$(\bullet)^{-1}$ is the matrix inversion operation.

The asymptotic probability to be in the states of subset $\mathbf{X}$ is then given by:

$$\mathbf{P}_X = \mathbf{P}_X\ (\mathbf{1}_X)^T = \mathbf{P}\ \ (\mathbf{U})^T \quad (1.10)$$

b) Case of a Markov chain with an absorbing state

Consider that Y is a subset of absorbing states; in the asymptotic behavior, we have:

$$\mathbf{P}_X = \mathbf{P}_X\ (\mathbf{1}_X)^T = 0$$

$$\mathbf{P}_Y = \mathbf{P}_Y\ (\mathbf{1}_Y)^T = \mathbf{1} \quad (1.11)$$

This case deserves a practical interest only if we consider the probability to be in a subset of absorbing states of $\mathbf{Y}$; let $\mathbf{Z}$ be this subset and $\mathbf{H}$ the complementary subset. We then have the following partitions:

Transition rates matrix:

$$\Lambda = \begin{bmatrix} \Lambda_{XX} & \Lambda_{XY} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \Lambda_{XX} & \Lambda_{XZ} & \Lambda_{XH} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Asymptotic-state probabilities vector: $\mathbf{P} = [\ \mathbf{0}\ \ \mathbf{P}_Y] = [\ \mathbf{0}\ \ \mathbf{P}_Z\ \ \mathbf{P}_H]$

In this case, considering the form and the partition of Λ , relation (1.1) can be written as:

$$\frac{d}{dt}\mathbf{P}_X(t) = \mathbf{P}_X(t)\,\Lambda_{XX}$$

$$\frac{d}{dt}\mathbf{P}_Y(t) = \mathbf{P}_X(t)\,\Lambda_{XY} \Rightarrow \frac{d}{dt}\,[\,\mathbf{P}_Z(t)\;\;\mathbf{P}_H(t)\,] = \mathbf{P}_X(t)\,[\,\Lambda_{XZ}\;\;\Lambda_{XH}\,]$$

Using the Laplace transform leads to:

$$\mathbf{P}_X(s) = \mathbf{P}_{0X}\,(s\,\mathbf{I}_{XX} - \Lambda_{XX})^{-1}$$

$$s\,\mathbf{P}_Y(s) = \mathbf{P}_{0X}\,(s\,\mathbf{I}_{XX} - \Lambda_{XX})^{-1}\;\Lambda_{XY}$$

that is :

$$s\,[\mathbf{P}_Z(s)\;\;\mathbf{P}_H(s)] = \mathbf{P}_{0X}\,(s\,\mathbf{I}_{XX} - \Lambda_{XX})^{-1}\;[\Lambda_{XZ}\;\;\Lambda_{XH}]$$

The theorem of the final value: $\mathbf{F} = \lim_{t\to}\mathbf{F}(t) = \lim_{s\to 0} L\,\{\frac{d}{dt}\;\mathbf{F}(t)\} = \lim_{s\to 0} s\,\mathbf{F}(s) - \mathbf{F}(0)$, allows, for example, to write:

$$\mathbf{P}_Z = -\,\mathbf{P}_{0X}\,(\Lambda_{XX})^{-1}\,\Lambda_{XZ} \quad (1.12)$$

$\mathbf{P}_Z$ is the vector of the asymptotic probabilities to be in the subset **Z** with respect to $\mathbf{P}_{0X}$. Considering that $\mathbf{P}_Z$ corresponds to the asymptotic probability of presence in **Z**, we further have:

$$\mathbf{P}_Z = \mathbf{P}_Z\,(\mathbf{1}_Z)^T = -\,\mathbf{P}_{0X}\,(\Lambda_{XX})^{-1}\,\Lambda_{XZ}\;\;(\mathbf{1}_Z)^T \quad (1.13)$$

1.3. Mean sojourn Times

To obtain a significant measure, when computing the **mean sojourn time in the states of subset X**, we consider that Y is a subset of absorbing states[2]. The vector of the mean sojourn time in the states of X before absorbtion by the states of Y is defined as:

$$\mathbf{T}_X = \int_0^{\infty}\mathbf{P}_X(t)\,dt \;= \mathbf{lim}_{s\to 0}\,\mathbf{P}_X(s) \quad (1.14)$$

and thus, from relation (1.2):

$$\mathbf{T}_X = -\;\mathbf{P}_{0X}\,(\Lambda_{XX})^{-1} \quad (1.15)$$

$$\mathbf{T}_X\,\Lambda_{XX} = -\;\mathbf{P}_{0X}$$

2. In the opposite case, the mean sojourn time would be infinite.

The mean sojourn time in the states of subset **X** with respect to vector $\mathbf{P}_{0X}$ is thus given by:

$$\mathbf{T}_X = \mathbf{T}_X\,(\mathbf{1}_X)^T = -\ \mathbf{P}_{0X}\,(\Lambda_{XX})^{-1}\,(\mathbf{1}_X)^T \quad (1.16)$$

$(\mathbf{T}_X)^{-1} = \rho_X$ can be interpreted as the mean absorbing rate by subset **Y** (flow from X to Y). When **Y** is partitioned into two subsets **Z** and **H** (see paragraph 1.2.2.b.), this rate can be split up into two rates $\rho_{X/Z}$ and $\rho_{X/H}$, defined as:

$$\rho_{X/Z} = (\mathbf{T}_{X/Z})^{-1} = \mathbf{P}_Z\,(\mathbf{T}_X)^{-1} \quad (1.17)$$

$$\rho_{X/H} = (\mathbf{T}_{X/H})^{-1} = \mathbf{P}_H\,(\mathbf{T}_X)^{-1}$$

where, $\mathbf{P}_Z$ and $\mathbf{P}_H$ represent the absorbing probabilities by **Z** and **H** respectively, and so that:

$$\rho_X = \rho_{X/Z} + \rho_{X/H}$$

Relations (1.13) and (1.16) allow, for example, to write:

$$\mathbf{P}_Z = \mathbf{T}_X\ \Lambda_{XZ}\,(\mathbf{1}_Z)^T \quad (1.18)$$

From (1.17), **the mean sojourn time in X before absorbtion by Z [$T_{X/Z}$]** is:

$$\mathbf{T}_{X/Z} = \frac{\mathbf{T}_X}{\mathbf{P}_Z} = \frac{\mathbf{T}_X(\mathbf{1}_X)^T}{\mathbf{P}_Z(\mathbf{1}_Z)^T} = \frac{\mathbf{T}_X}{\mathbf{P}_Z(\mathbf{1}_Z)^T}(\mathbf{1}_X)^T = \frac{\mathbf{P}_{0X}(\Lambda_{XX})^{-1}}{\mathbf{P}_{0X}(\Lambda_{XX})^{-1}\Lambda_{XZ}}(\mathbf{1}_X)^T \quad (1.19)$$

Vectors $\mathbf{T}_{X/Z}$ and $\mathbf{T}_{X/H}$ can also be introduced to characterize the mean sojourn times in the states of **X** before absorbtion by **Z** and **H**;, respectively. It is quite easy to deduce $T_{X/Z}$ from relation (1.19) and from the following definitions:

$$\mathbf{T}_{X/Z} = \mathbf{T}_{X/Z}(\mathbf{1}_X)^T$$

$$\mathbf{T}_{X/H} = \mathbf{T}_{X/H}\,(\mathbf{1}_X)^T$$

1.4. Reward

The **reward W(t) calculated in the interval [0, t]** takes into account the cumulative time effect on the whole states of the model using the **reward structure** settled by the vector of **yield rates** in relation to the presence in the states $\mathbf{Y} = [\ y_i\ ]$ and by the matrix of **bonuses** associated to transitions $\mathbf{B} = [\ b_{ij}\ ]$.

In the case of a cost analysis, where the notion of reward involved is a **discounted reward**, it is quantitatively different to gain or to spend an amount of money immediately or later on; we thus assume that an amount of money invested at time zero with the intention to yield interests, would earn more money after **N** periods of time.

The model of depreciation of money used in SURF-2 is an exponential model detailed hereafter:

Assuming that s(t) corresponds to an amount of money at time t, the value of this money at time zero taken as a reference, is given by:

$$s(0) = \exp(-\gamma t)\, s(t) \quad (1.20)$$

where γ is the discount rate.

In this case, the reward parameters must account for this depreciation; the yield rates and the bonuses at time **t**, brought back to the initial reference time of the interval, are thus computed as:

$$\mathbf{Y}(t,\gamma) = [y_i(t,\gamma) = y_i\ e^{-\gamma t}] \text{ and } \mathbf{B}(t,\gamma) = [b_{ij}(t,\gamma) = b_{ij}\ e^{-\gamma t}] = \mathbf{B}\, I\ e^{-\gamma t} \quad (1.21)$$

We thus obtain for the cumulative reward $W(t,\gamma)$:

$$W(t,\gamma) = \sum_i y_i \int_o^t e^{-\gamma u} V_i(u)du + \sum_i \sum_{j! = i} b_{ij} \int_0^t e^{-\gamma u} S_{ij}(u)du \quad (1.22)$$

$$W(t,\gamma) = \sum_i y_i \int_o^t y_i(u,\gamma) V_i du + \sum_i \sum_{j! = i} \int_0^t b_{ij}(u,\gamma) S_{ij}(u)du \quad (1.23)$$

where, $V_i(t) =$ 1 if the state variable X(t) = i,

0 otherwise))

and $S_{ij}(t)\, dt =$ 1 if X(t) = i and X(t +) = j

0 otherwise))

Relation (1.23) is the cumulative reward earned in the [0, t] interval expressed with respect to the initial time zero:

- during the sojourn in each state of the model,
- during each transition between states of the model.

Note that if the discount rate γ is equal to zero, relation (1.23) can be simply written as:

$$W(t,0) = W(t) = \sum_i y_i \int_o^t V_i(u)du + \sum_i \sum_{j! = i} b_{ij} \int_0^t S_{ij}(u)du \quad (1.24)$$

We can thus write (1.24) as follows:

$$W(t) = W(t,\gamma) = \sum_i y_i t_i(t) + \sum_i \sum_{j!=i} b_{ij} n_{ij}(t) \quad (1.25)$$

where $t_i(t)$ is the cumulative time elapsed in state **i** in the interval [0, t], and $n_{ij}(t)$ is the number of transitions occuring from state **i** to state **j**.

Under a matrix form : $W(t) = \mathbf{Y}\,[\mathbf{T}(t)]^T + \mathbf{B}\,[\mathbf{N}(t)]^T\,\mathbf{1}^T (1.26)$

where $\mathbf{Y}$ =[y_i] and $\mathbf{T}(t)$ =[$t_i(t)$] are vectors and $\mathbf{B}$ = [b_{ij}] and $\mathbf{N}(t)$ = [$n_{ij}(t)$] are matrix (with $b_{ii} = 0$ and $n_{ii}(t) = 0$).

1.4.1. Pointwise reward

a) Taking into account the discount rate

The expected pointwise reward in the interval [0, t] [EW(t;γ)] corresponds to:

$EW(t,\gamma) = E[W(t,\gamma)]$

$$= \sum_i [y_i \int_o^t p_i(u) e^{-\gamma u} V_i(u) du + \sum_{j \neq i} b_{ij} \int_0^t p_i(u) \lambda_{ij} p_i(u) e^{-\gamma u} S_{ij}(u) du\,] \quad (1.27)$$

where $p_i(u)$ is the probability to be in state **i** at time **u**,

and $p_i(u)\ \lambda_{ij}$ is the density probability of a transition from **i** to **j** in the interval[u;u +].

$$EW(t,\gamma) = \sum_i \left\{ \left[\int_0^t p_i(u) e^{-\gamma u} du \right] \left[y_i + \sum_{j \neq i} b_{ij} \lambda_{ij} \right] \right\} \quad (1.28)$$

In a matrix fom: $EW(t,\gamma) = \left[\int_0^t P(u) \exp(\gamma I u) du \right] (\mathbf{Y} + \mathbf{B} \square \Lambda \mathbf{1}^T)^T \quad (1.29)$

The symbol ❑ corresponds to the matrix operation "product of congruence" defined as:
$\mathbf{Z} = \mathbf{X} \square \mathbf{Y} \Leftrightarrow [z_{ij}] = [x_{ij} \cdot y_{ij}]$.

From relation (1.3) we find:

$$EW(t,\gamma) = \left\{ \mathbf{P}_0 \int_0^t \exp[(\Lambda - \gamma \mathbf{I}) u] du \right\} (\mathbf{Y} + \mathbf{B} \square \Lambda \mathbf{1}^T)^T \quad (1.30)$$

EW(t,γ) is thus obtained by multiplying the integration of the exponential function of matrix:

$$\Lambda^* = \Lambda - \gamma\,\mathbf{I} \quad (1.31)$$

by the vector R of the **reward rates** associated to both state occupency and state transition[3]:

$$\mathbf{R} = \mathbf{Y} + \mathbf{B} \,\square\, \Lambda\ \mathbf{1}^T (1.32)$$

hence :

$$EW(t,\gamma) = \left\{ \mathbf{P}_0 \int_0^t \exp(\Lambda^* u)\,du \right\} \mathbf{R}^T = \mathbf{P}_0\ {}^*(t)\ \mathbf{R}^T = \mathbf{L}^*(t)\ \ \mathbf{R}^T (1.33)$$

with : $\ {}^*(t) = \int_0^t \exp(\Lambda^* u)\,du$

$$\mathbf{L}^*(t) = \mathbf{P}_0\ {}^*(t) = \ \mathbf{P}_0 \int_0^t \exp(\Lambda^* u)\,du$$

$$\mathbf{R} = (\mathbf{Y} + \mathbf{B} \,\square\, \Lambda\ \mathbf{1}^T)$$

b) No discount rate is considered

If the discount rate is equal to 0, we thus have:

$$EW(t) = \left\{ \mathbf{P}_0 \int_0^t \exp(\Lambda u)\,du \right\} \mathbf{R}^T = \mathbf{P}_0\ \ (t)\mathbf{R}^T = \mathbf{L}(t)\ \ \mathbf{R}^T (1.34)$$

with : $\ (t) = \int_0^t \exp(\Lambda u)\,du$

$$\mathbf{L}(t) = \mathbf{P}_0\ \ (t) = \mathbf{P}_0 \int_0^t \exp(\Lambda u)\,du$$

$$\mathbf{R} = (\mathbf{Y} + \mathbf{B} \,\square\, \Lambda\ \mathbf{1}^T)$$

In the latter case, we also have:

$$EW(t) = E[W(t)] = E[\mathbf{Y}\ \mathbf{T}(t) + \mathbf{B}\,\mathbf{N}(t)^T\ \mathbf{1}^T] = \mathbf{Y}\ E[\mathbf{T}(t)] + \mathbf{B}\ E[\mathbf{N}(t)^T]\ \ \mathbf{1}^T (1.35)$$

This can be applied both to an ergodic Markov chain and to a state-absorbing Markov chain.

---

3. By analogy with relation (1.4), $\mathbf{R}$ can be considered as a "super" summation vector for the calculation of additional measures.

1.4.2. Asymptotic reward

a) Taking into consideration the discount rate

The **asymptotic value of the expected reward, [EW(γ)]** is defined as:

$$EW(\gamma) = \lim_{t\to} EW(t,\gamma) = \lim_{t\to} E[W(t,\gamma)] \quad (1.36)$$

From relation (1.33); one can write:

$$EW(\gamma) = \left\{ \mathbf{P}_0 \int_0^{\infty} \exp(\Lambda^* u)\,du \right\} \mathbf{R}^T$$

$$EW(\gamma) = \lim_{s\to 0} \{ P_0 \ (s\,\mathbf{I} - \Lambda^*)^{-1} \} \mathbf{R}^T$$

$$EW(\gamma) = - \mathbf{P}_0 (\Lambda^*)^{-1} \ \mathbf{R}^T = \mathbf{P}_0 (\gamma \mathbf{I} - \Lambda)^{-1} \ \mathbf{R}^T \quad (1.37)$$

b) No discount rate is considered

When there is no depreciation of the cumulated reward, the **asymptotic value of the expected reward** tends towards a finite constant value, only in the case of a state-absorbing Markov chain whose the associated yield rate is equal to zero. This value is not bounded in the case of an ergodic Markov chain; it is however possible to evaluate the reward variations rate.

b.1) Case of a Markov chain with absorbing states

We consider **Y** as the subset of absorbing states. In this particular case, it is interesting to evaluate the **asymptotic value of expected total reward, [EW$_X$]** in the **X** subset before absorbtion by **Y** with respect to P_{0X}. This measure is calculated from the partition that corresponds to subset **X** ; from (1.29) we thus have:

$$EW_X = \int_0^{\infty} \mathbf{P}_X(u)\,du\,(\mathbf{R}_X)^T$$

where $\mathbf{R}_X$ is a subvector from $\mathbf{R}$ defined by (1.32) that corresponds to the states of X : $\mathbf{R} = [\ \mathbf{R}_X \ \ \mathbf{0}_Y\ ]$

From (1.14) and (1.15) we thus obtain :

$$EW_X = - \mathbf{P}_{0X} \ (\Lambda_{XX})^{-1} \mathbf{R}_X{}^T \quad (1.38)$$

Note that relation (1.38) is only a particular case of relation (1.37).

b.2) Case of an ergodic Markov chain

For an ergodic Markov chain, one can show that the expected value of the reward varies asymptotically in linear way when time tends towards infinity. The **reward variation rate [r]** corresponds to the slope of this variation.

$$r = \lim_{t \to} \frac{EW(t)}{t} = \lim_{t \to} \frac{E[W(t)]}{t} \quad (1.39)$$

Considering (1.26), relation (1.30) can be writen as:

$$r = \lim_{t \to} \left[\frac{1}{t} \int_0^t \mathbf{P}(u)du \right] \mathbf{R}^T$$

hence: $$r = \mathbf{P}\,\mathbf{R}^T (1.40)$$

2. On the dependability measures in SURF-2

The dependability measures available in SURF-2 are listed in the table of Fig. 1. What follow in this paragraph, are definitions of the mathematical expressions characterizing these basic measures in relation with the basic measures given in paragraph 1.

Pointwise Measures	Asymptotic Measures	Mean Values
Reliability		MTFF
Safety		MTTF
Maintenability		MUT
Availability	Availability	MDT
Unavailability	Unavailability	MTBF
	Discounted Reward	AC[a]
Reward	Reward	Reward

a.Availability before Catastrophic failure.

Fig. 1: Dependability measures selection

In the table of Figure 1; the reward and discounted reward measures characterize the absence or the presence of a discount rate respectively.

To allow the measures to be computed, the user must associate a semantic to the subsets of states defined in paragraph 1. This definition relies on the notion of **class**. The user must define the **failed-states class** (that provide the improper service: I). The class of states that provide a proper service (P) is thus the complementary state class. Fig. 2 shows this state partition of the model.

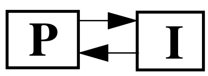


Fig. 2: Definiton of classes

2.1. Pointwise measures

2.1.1. Reliability

The **reliability [R(t)]** corresponds to the probability of a continuous presence in the subset of states **X**; the reliability is expressed from relation (1.6) where the subset **X** corresponds to the **class of states that provide a proper service P**.

$$\mathbf{R}(t) = \mathbf{P}_{0P} \exp(\Lambda_{PP}\, t)\, (\mathbf{1}_P)^T \quad (2.1)$$

Note : $\mathbf{P}_{0P}$ = vector of initial probabilities associated to the states that provide a proper service.

2.1.2. Safety

The **safety [S(t)]** is expressed in a similar way as the reliability; the difference is mainly semantic and is related to the definition of the failed state class defined by the user.

$$S(t) = \mathbf{P}_{0P} \exp(\Lambda_{PP}\, t)\, (\mathbf{1}_P)^T \quad (2.2)$$

2.1.3. Maintainability

The **maintainability [M(t)]** corresponds to the probability of presence in the class **P** when the system was initially down (Improper class). This measure can be obtained by:

- calculating the probability of the continuous presence in class **I**,
- taking the complement to 1.

$$M(t) = 1 - [\ \mathbf{P}_{0I} \exp(\Lambda_{II}\, t)\, (\mathbf{1}_I)^T] \quad (2.3)$$

Note: $\mathbf{P}_{0I}$ = vector of initial probabilities associated to the states that provide an improper service.

2.1.4. Availability

The **availability [A(t)]** is expressed from relation (1.4):

$$A(t) = \mathbf{P}_0 \exp(\Lambda\, t)\, \mathbf{U}^T \quad (2.4)$$

U is a summation vector for which the **|P| first** elements are equal to 1, the other being equal to 0.

2.1.5. Unavailability

The **unavailability [UA(t)]** is simply expressed as:

$$UA(t) = 1 - A(t) = \mathbf{P}_0 \exp(\Lambda\, t)\, \mathbf{Z}^T \quad (2.5)$$

Z is a summation vector whose **|P| first** elements are equal to 0, the other being equal to 1.

2.1.6. Reward and discounted reward

The expected total reward in the [0, t] interval [EW(t)] is expressed from relation (1.33):

$$EW(t,\gamma) = \mathbf{P}_0 \ {}^{*}(t)\ \mathbf{R}^T \quad (2.6)$$

with : $^{*}(t) = \int_0^t \exp(\Lambda^{*}u)du = \int_0^t \exp[(\Lambda - \gamma I)u]du$

$$\mathbf{R} = (\mathbf{Y} + \mathbf{B} \ \square\ \Lambda\ \mathbf{1}^T) : r_i = y_i + \sum_{j=1}^{j \neq i,\ |E|} b_{ij}\lambda_{ij}$$

2.2. Asymptotic measures

2.2.1. Availability

The **asymptotic availability [A]** is derived form expression (1.10):

$$A = \mathbf{0}_{|E|}\ (\Lambda_{|E|})^{-1}\ \mathbf{U}^T \quad (2.7)$$

where $\mathbf{0}_{|E|}$ is the modified zero vector for which only element $|E|$ (the last one) is equal to 1.

$\Lambda_{|E|}$ is the modified transition rate matrix replacing column $|E|$ by vector $\mathbf{1}^T$.

2.2.2. Unavailability

The **asymptotic unavailability [UA]** can be simply expressed as:

$$UA = \mathbf{0}_{|E|}\ (\Lambda_{|E|})^{-1}\ \mathbf{U}'^T \quad (2.8)$$

where $\mathbf{U}' = [\ \mathbf{0}_X\ \ \mathbf{1}_Y\ ]$.

One can also note that: $UA = 1 - A = \mathbf{1} - \mathbf{0}_{|E|}\ (\Lambda_{|E|})^{-1}\ \mathbf{U}^T$

2.2.3. Discounted reward

From relation (1.37), the **asymptotic value of the expected discounted reward [EW(γ)]** can be written as:

$$EW(\gamma) = -\mathbf{P}_0\ (\Lambda^{*})^{-1}\ \mathbf{R}^T = P_0\ (\gamma \mathbf{I} - \Lambda)^{-1}\ \mathbf{R}^T \quad (2.9)$$

2.2.4. Reward rate

The **rate of the expected reward (r) for an ergodic Markov chain with an asymptotic behavior** is calculated from (1.40):

$$r = \mathbf{P}\ \mathbf{R}^T \quad (2.10)$$

with $\mathbf{P} = \mathbf{0}_{|E|}\ (\Lambda_{|E|})^{-1}$

$$\mathbf{R} = (\mathbf{Y} + \mathbf{B}\square\ \Lambda\ \mathbf{1}^T) : r_i = y_i + \sum_{j = 1,\, j \neq i}^{|E|} \lambda_{ij}$$

2.3. Mean Time values

The various mean sojourn times are calculated from expression (1.16):

$$\mathbf{M}\bullet\bullet\bullet = -\ \mathbf{P}_{X0}\ (\Lambda_{XX})^{-1}\ (\mathbf{1}_X)^T \quad (2.11)$$

The main difference lies essentially on the choice of vector $\mathbf{P}_{X0}$.

2.3.1. MTFF

Concerning the **Mean Time to First Failure**, $\mathbf{P}_{X0} = \mathbf{P}_{P0} = [1\ 0\ \ldots\ 0]$; that is the vector used to calculate the reliability:

$$\text{MTFF} = -\ \mathbf{P}_{P0}\ (\Lambda_{PP})^{-1}\ (\mathbf{1}_P)^T \quad (2.12)$$

2.3.2. MTTF

In the case of an ergodic Markov chain; this measure corresponds to the **mean time elapsed in the proper class P (Fig. 2) whereas the system is randomly reset in an asymptotic behavior [Mean Time To Failure]**.

This measure can be obtained from calculating:

- the vector of probabilities of the asymptotic states in the proper class **P**: P_P, from relation (1.9) with $\mathbf{P} = [\mathbf{P}_P\ \ \mathbf{P}_I]$,
- the vector of conditional probabilities to be in the proper class **P**, in the asymptotic behavior: $\mathbf{P}_{CP} = \mathbf{P}_P\ /\ [\mathbf{P}_P\ (\mathbf{1}_P)^T]$,
- the mean sojourn time in proper class **P** before absorbtion by the improper class **I**, using the vector $\mathbf{P}_{CP}$ as the vector of initial probabilities P_{0X} in (2.11).

$$\text{MTTF} = -\ \frac{\mathbf{P}_P}{\mathbf{P}_P \mathbf{1}_P}\ (\Lambda_{PP})^{-1}\ (\mathbf{1}_P)^T \quad (2.13)$$

2.3.3. MUT

In the case of an ergodic Markov chain, this measure corresponds to **the mean time elapsed in the proper class P (Fig. 2) whereas the system is reset after a transition from class I to class P, in an asymptotic behavior [Mean Up Time]**.

This measure can be obtained by calculating:

- the vector of probabilities of the asymptotic states in the improper class **I** : $\mathbf{P}_I$, from relation (1.9) with $\mathbf{P} = [\mathbf{P}_P \ \mathbf{P}_I]$,
- the vector of probabilities of transitions from class **I** to class **P** in the asymptotic behavior: $\mathbf{P}_{TP} = (\mathbf{P}_I \ \Lambda_{IP}) \ / \ [\mathbf{P}_I \ \Lambda_{IP} \ (\mathbf{1}_P)^T]$
- the mean sojourn time in proper class P before absorbtion by the improper class **I**, using the vector $\mathbf{P}_{TP}$ as the vector of initial probabilities $\mathbf{P}_{0X}$ in (2.11).

From

(i) relation 1.7 [$\mathbf{0} = \mathbf{P} \Lambda$]

(ii) the induced partition by **P** and **I**:

$$\mathbf{P}_P \Lambda_{PP} + \mathbf{P}_I \Lambda_{IP} = 0 \quad (2.14)$$

$$\mathbf{P}_P \Lambda_{PI} + \mathbf{P}_I \Lambda_{II} = 0 \quad (2.15)$$

(iii) the fact that the lines of Λ sum up to zero, in particular knowing that:

$$\Lambda_{PP} (\mathbf{1}_P)^T + \Lambda_{PI} (\mathbf{1}_I)^T = 0 \quad (2.16)$$

we obtain:

$$\text{MUT} = \frac{\mathbf{P}_P (\mathbf{1}_P)^T}{\mathbf{P}_P \Lambda_{PI} (\mathbf{1}_I)^T} \quad (2.17)$$

2.3.4. MDT

There is a duality between this measure **[Mean Down Time]** and the MUT (permutation of **P** and **I** classes).

$$\text{MDT} = \frac{\mathbf{P}_I(\mathbf{1}_I)^T}{\mathbf{P}_I \Lambda_{IP}(\mathbf{1}_P)^T} \quad (2.18)$$

2.3.5. MTBF

This measure characterizes the mean cycle time between two failures **[Mean Time Between Failure]** that is:

$$\text{MTBF} = \text{MUT} + \text{MDT} \quad (2.19)$$

and thus, from relations (2.17) and (2.18), and the fact that $\mathbf{P}(\mathbf{1})^T = \mathbf{P}_P(\mathbf{1}_P)^T + \mathbf{P}_I(\mathbf{1}_I)^T = 1$:

$$\text{MTBF} = \frac{1}{\mathbf{P}_P \Lambda_{PI}(\mathbf{1}_I)^T} = \frac{1}{\mathbf{P}_P \Lambda_{IP}(\mathbf{1}_P)^T} \quad (2.20)$$

From the asymptotic availability definition (paragraph 1.1) we also have:

$$A = \frac{\text{MUT}}{\text{MTBF}} \quad (2.21)$$

2.3.6. Mean reward

The expected reward (with a null discount rate) cumulated in subset **P**, before absobtion by **I**, **[EWp]**; is calculated from (1.38) as:

$$\text{EW}_P = - \mathbf{P}_{P0} \, (\Lambda_{PP})^{-1} \, \mathbf{R}_P^T \quad (2.22)$$

where $\mathbf{R}_P$ is the fraction of vector $\mathbf{R}$ corresponding to the states of P: $\mathbf{R} = [\ \mathbf{R}_P \quad \mathbf{0}_I\ ]$

2.3.7. Examples of specific measures

Here are some examples derived from the use of EW_P. One can note, in particular, that from (2.12) and (2.22):

$$\text{MTFF} = \text{EW}_P \quad \text{for } \mathbf{Y}_P = \mathbf{1}_P \text{ and } \mathbf{B}_P = \mathbf{0}_P \quad (2.23)$$

Other interesting measures can be obtained using vector $\mathbf{Y}$ as a parametrable summation vector. This is particularly applicable in the case of safety system, where the states of the graph can be partionned into three different classes (Fig.3):

- class A : Service Accomplishment,
- class B : Benign failure,
- class C : Catastrophic failure.

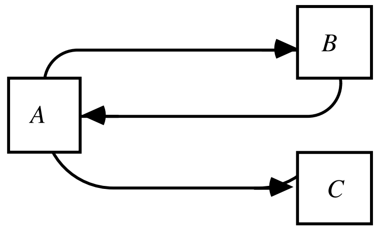


Fig. 3 : Generic graph for a safety system.

The Catastrophic class **C** corresponds to the Improper class when the safety measure S(t) is evaluated. It can easily be verified that expression (2.12) enables to calculate the **mean time before occurrence of a catastrophic failure [Mean Safe Time : MST]**. Moreover, relation (2.22) enables to calculate:

- using $\mathbf{Y}_P = [\ \mathbf{1}_A\ \mathbf{0}_B\ ]$ and $\mathbf{B}_P = \mathbf{0}_P$, the **mean accomplishment time before occurrence of a catastrophic failure [Mean Accomplishment Time : MAT]**,
- using $\mathbf{Y}_P = [\ \mathbf{0}_A\ \mathbf{1}_B\ ]$) and $\mathbf{B}_P = \mathbf{0}_P$, the **mean benign failure time before occurrence of a catastrophic failure [Mean Benign failure Time : MBT]**;

respectively defined as:

$$\text{MAT} = - \mathbf{P}_{P0}\ (\Lambda_{PP})^{-1}\ (\mathbf{1}_A\ \mathbf{0}_B)^T \quad (2.24)$$

$$\text{MBT} = - \mathbf{P}_{P0}\ (\Lambda_{PP})^{-1}\ (\mathbf{0}_A\ \mathbf{1}_B)^T \quad (2.25)$$

In particular, this allows to deduce the **mean availability before occurrence of a catastrophic failure [Availability before Catastrophic failure : AC] as well as the mean unavailability before occurrence of a catatrophic failure [UnAvailability before Catastrophic failure : UAC]** :

$$\text{AC} = \frac{\text{MAT}}{\text{MST}} = \frac{\text{MAT}}{\text{MAT} + \text{MBT}} \quad (2.26)$$

$$\text{UAC} = 1 - \text{UAC} = \frac{\text{MBT}}{\text{MST}} = \frac{\text{MBT}}{\text{MBT} + \text{MAT}} \quad (2.27)$$