

Dysfunctional Analysis & Simulation - SD9

User Guide

BPA Delivery 6 V5R19 (V5.6)



Version 6

Table of Contents

DYSFUNCTIONAL ANALYSIS & SIMULATION	1
User Guide	1
Table of Contents	2
INTRODUCTION	4
APPENDIX DOCUMENT REFERENCES	5
HARDWARE ARCHITECTURE	6
HARDWARE CONFIGURATION	6
INSTALLATION	6
LAUNCH DAS TOOL	7
USERS MANAGEMENT	9
ADMINISTRATION WINDOW	9
USERS MANAGEMENT	9
WORKGROUP MANAGEMENT	12
PRINTING OF ADMINISTRATION INFORMATION	15
DATA ACCESS RIGHTS MANAGEMENT	16
WORK ENVIRONMENT	24
PRESENTATION	24
MENU BAR	25
MAIN TOOL BAR	25
WORK AREA	28
INFORMATION BAR	28
MENU TREE STRUCTURE	30
MENU BAR	30
SUMMARY LIST FOR MENUS, SHORTCUTS AND ICONS	33
STANDARD MODELS LIBRARY	35
GENERAL	35
EDITION ON COMPONENT	35
EDITION ON EQUIPMENT	63
EDITION ON OPERATOR	81
EDITION ON TYPE	93

SEARCH MODEL.....	103
MANAGEMENT OF PROJECT.....	104
EDITION ON PROJECT.....	104
EDITION ON SYSTEM.....	106
MODELLING A SYSTEM ARCHITECTURE.....	113
SEARCH COMPONENT MODEL IN A SYSTEM	122
INFORMATIONS.....	124
SYSTEM LINKS.....	129
LINKS COLOR	131
PRINTING.....	134
EXPORT/IMPORT DATA INTO XML FORMAT.....	137
EXPORT IN XML FORMAT.....	137
IMPORT IN XML FORMAT.....	139
THE PLUGIN MANAGER	141
INTRODUCTION.....	141
GENERAL DESCRIPTION OF THE PLUGIN MANAGER PANEL.....	141
MANAGE PLUGINS.....	142
MANAGE ACTIONS	146
MANAGE PLUGIN ITEMS	148
ADJUSTMENT OF PREFERENCES.....	152
GENERAL.....	152
ENVIRONMENT RUBRIC	152
DESKTOP RUBRIC	154
TOOLS BARS RUBRIC	156
EDITION RUBRIC.....	158
INPUTS/OUTPUTS RUBRIC	161
SIMULATOR RUBRIC	162
PLUGIN OPTIONS	163
QUIT DAS TOOL	164
GLOSSARY OF TERMS	166
APPENDIXES.....	168
APPENDIX 1: SYNTAX OF Altarica CODE	168
APPENDIX 2: SYNTAX OF PROBABILITY LAWS IN ARALIA FORMAT.....	177

INTRODUCTION

This user's manual describes the work environment of DAS (System Design and Analysis Tool) tool which enables the following functions:

- Modelling of functional and dysfunctional behaviour of systems by means of reusable components in library (Cf. p.33),
- Construction and interactive graphical simulation of system architectures.
- Automatic control of the consistency of system behaviour,
- Automatic generation of fault trees concerning targeted unexpected events.

Appendix 1 concerns the Syntax of Altarica Language.

Appendix 2 concerns the Syntax of probability laws in Aralia format.

APPENDIX DOCUMENT REFERENCES

- [SD9_Appendix_ABC_R19_D6.pdf](#)
- [SD9_Appendix_SeqGen_R19_D6.pdf](#)
- [SD9_Appendix_Translator_R19_D6.pdf](#)
- [SD9_Appendix_Utility_R19_D6.pdf](#)
- [SD9_Appendix_Simulation_R19_D6.pdf](#)
- [SD9_Appendix_FRB_R19_D6.pdf](#)
- [SD9_Appendix_Attributes_Named_Parameters_R19_D6.pdf](#)

HARDWARE ARCHITECTURE

HARDWARE CONFIGURATION

Cf : Installation guide

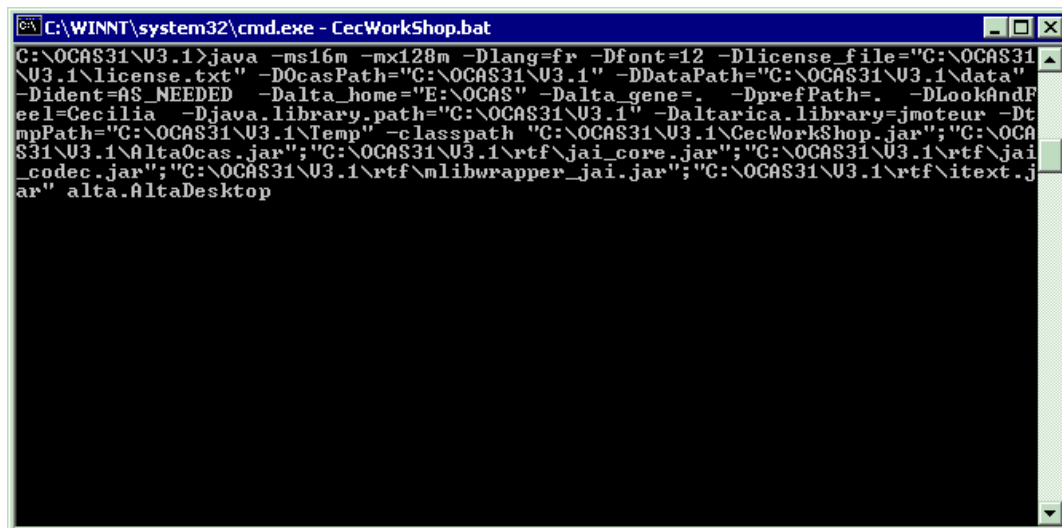
INSTALLATION

Cf. Installation guide

LAUNCH DAS TOOL

Double click on the DAS icon on the desktop to launch the DAS tool; the following displays appear:

An initialization MS DOS (Microsoft Disk Operating System) window (Figure 1).



```
C:\WINNT\system32\cmd.exe - CecWorkShop.bat
C:\OCAS31\U3.1>java -ms16m -mx128m -Dlang=fr -Dfont=12 -Dlicense_file="C:\OCAS31\U3.1\license.txt" -DOcasPath="C:\OCAS31\U3.1" -DDataPath="C:\OCAS31\U3.1\data" -Dident=AS_NEEDED -Dalta_home="E:\OCAS" -Dalta_gene=. -DprefPath=. -DLookAndFeel=Cecilia -Djava.library.path="C:\OCAS31\U3.1" -Daltarica.library=jmoteur -DtempPath="C:\OCAS31\U3.1\Temp" -classpath "C:\OCAS31\U3.1\CecWorkShop.jar";"C:\OCAS31\U3.1\AltaOcas.jar";"C:\OCAS31\U3.1\rtf\jai_core.jar";"C:\OCAS31\U3.1\rtf\jai_codec.jar";"C:\OCAS31\U3.1\rtf\mlibwrapper_jai.jar";"C:\OCAS31\U3.1\rtf\itext.jar" alta.Altadesktop
```

Figure 1 : MS DOS DAS window

- A Window display (Figure 2).

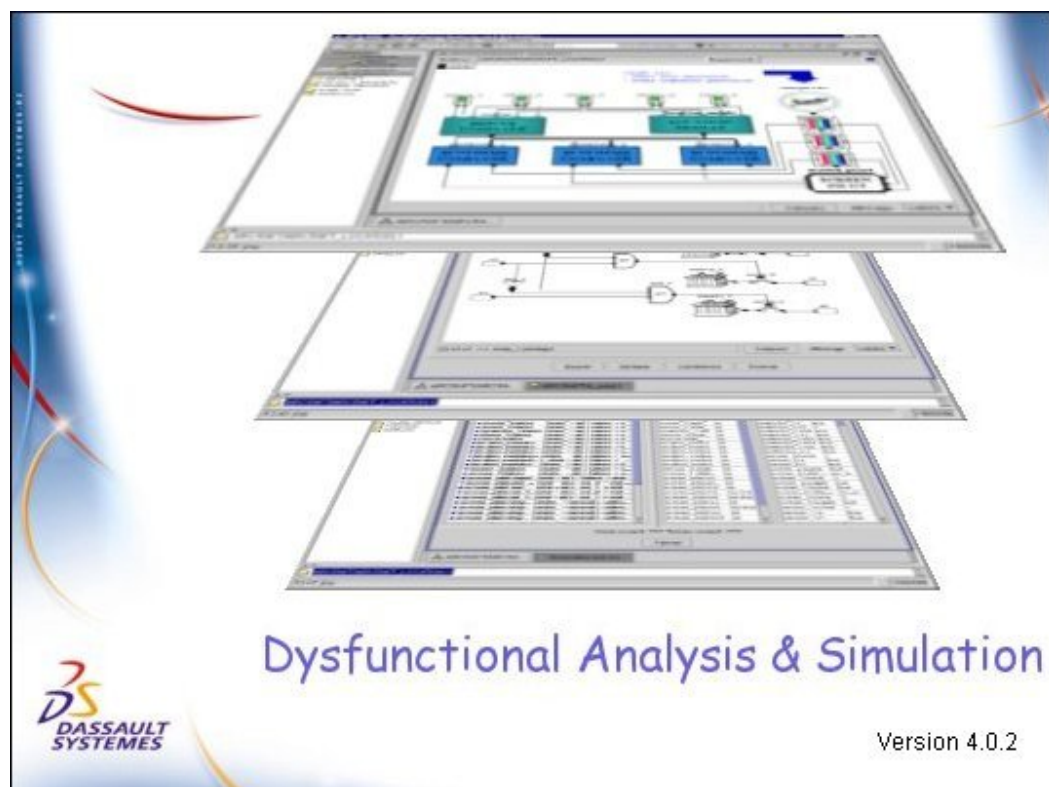


Figure 2 : DAS tool launching window

In the database configuration mode (Access ©, Oracle ©, or MySQL ©), in order to access the DAS work environment, the user name and the password associated are required (Figure 3).

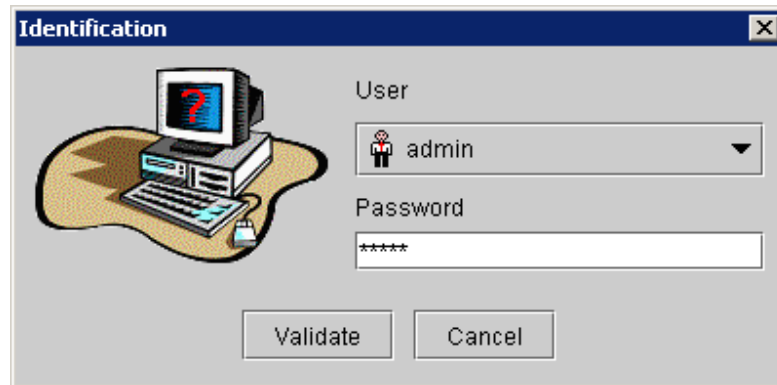


Figure 3 : Login window - DAS user identification

Note: In the database configuration every user likely to use the DAS tool, must be defined by the administrator in the Access ©, Oracle ©, or MySQL © databases.

USERS MANAGEMENT

ADMINISTRATION WINDOW

The management of the users is accessible only by the administrator or any user having the administrator profile.

An administration window is provided to the administrator profile by the DAS tool, in order to allow him to manage the access control of DAS users:

- Users definition,
- User profile definition (common user, administrator),
- Users group definition.

This information will be used to manage the access controls to DAS data (libraries, architecture system).

The administration window appears by activating the **File - Administration** command (Figure 4).

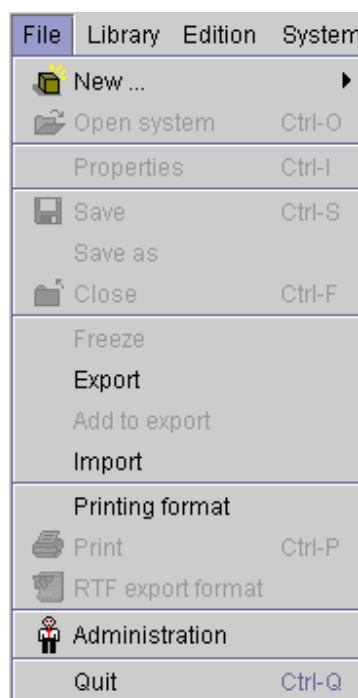


Figure 4 : View of the Administration window

USERS MANAGEMENT

The management of the users is accessible through the tab **User** from the **Users Administration** window (Figure 5).

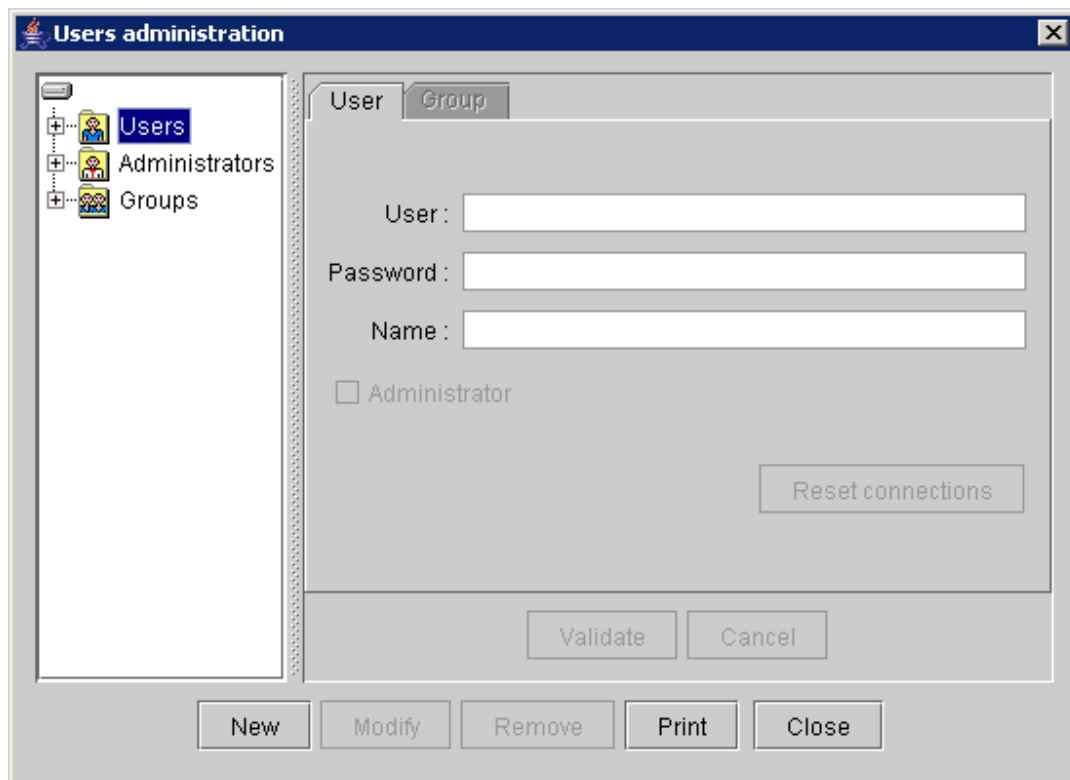


Figure 5 : Users Administration window – User tab

In order to create a new user, follow the following steps:

- ◆ Select Users or Administrators
- ◆ Click on the button **New**,
- ◆ Fill the new user in the text zone located below the field **User**,
- ◆ Fill the password associated to the new user in the text zone located after the field **Password** :,
- ◆ Fill the user name corresponding to the user in the text zone located after the field **Name** :,
- ◆ Check the Administrator field box to authorize if necessary this user to access the user's administration functions (Administrator privilege).
- ◆ Click on the button **Validate** to validate the creation of the new user profile, or on the button **Cancel** to cancel the creation of this profile.

When the user is created, he is inserted in the Administration tree structure, either in the **Users list** as a common user (Figure 6), or in the **Administrators** list if he has the administrator privilege.

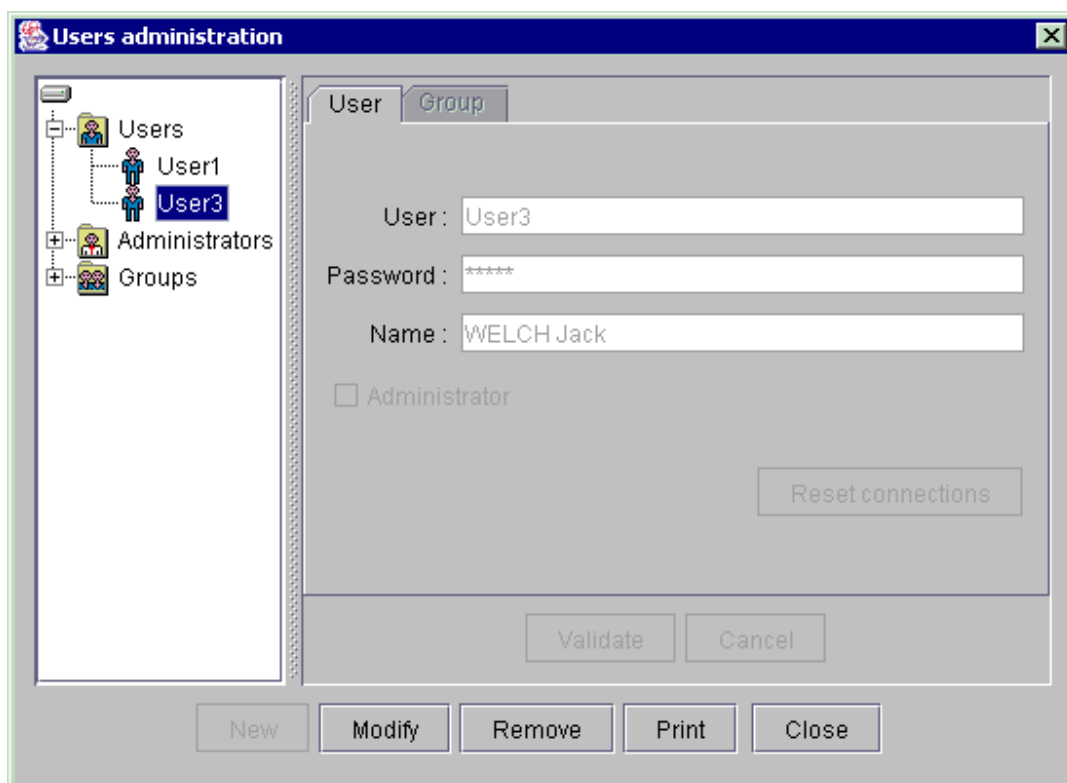


Figure 6 : Creation of a new user

In order to modify an existing user:

- ◆ Select (click left on the mouse) in the **Users** or **Administrators** list, the user to modify,
- ◆ Click on the button **Modify**,
- ◆ Make all the necessary modifications to the user profile (User, password, name, privilege),
- ◆ Click on the button **Validate** to validate the modifications, or on the button **Cancel** to cancel the modifications on this profile.

For removing an existing user:

- ♦ Select (click left on the mouse) the user to delete in the **Users** or **Administrators** list.
- ♦ Click on the button **Remove**, the following window appears on the screen (Figure 7):

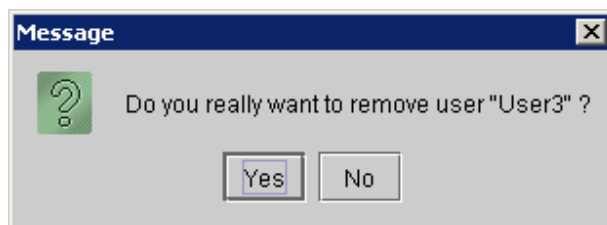


Figure 7 : Message for removing a user

- ♦ Click on the button **Yes** for deleting definitely the user profile or on the button **No** for cancelling the destruction of this profile.
- ♦ If the removed user is the owner of a DAS Object (Models, Systems, etc ...), the administrator must specify the one who will be the owner of this object. During removal, a box (Figure 8) asks Administrator for the future owner.

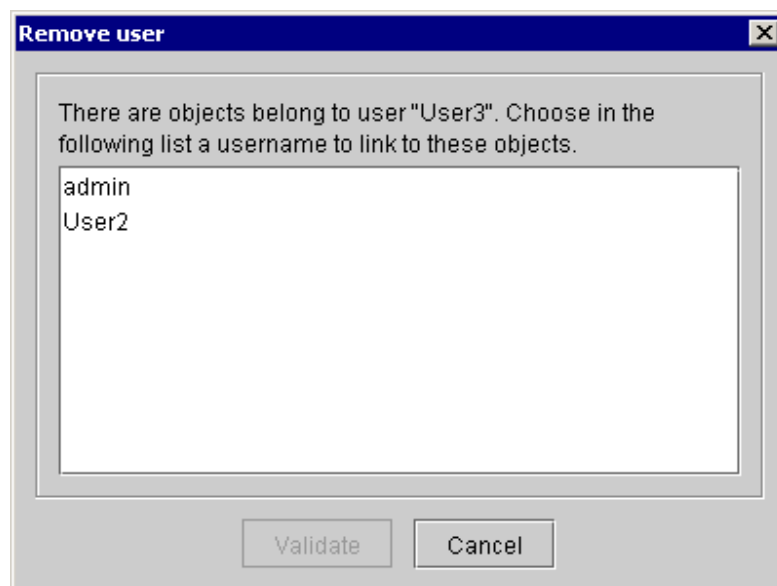


Figure 8 : New Owner

Note: Only the **Master Administrator** profile created during the initialization of DAS database (Access ©, Oracle ©, or MySQL ©) can not be destroyed.

WORKGROUP MANAGEMENT

The users workgroup management is accessible through the tab **Groups** from the **Users administration** window (Figure 9).

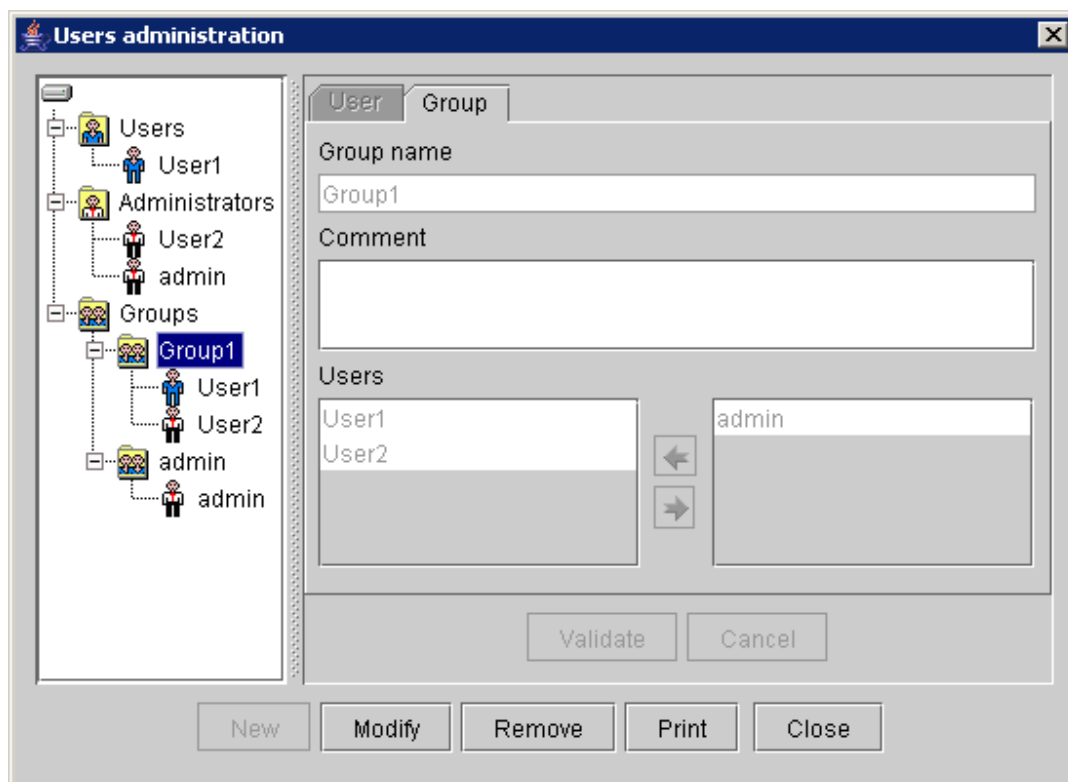



Figure 9 : Users administration window - Tab Groups

To create a new workgroup:

- ◆ Click on the button **New**,
- ◆ Fill the group name in the text zone located after the field Nom,
- ◆ In the tab **Users**, select one by one the users we wish to insert in this group. Use the icon  to move on the left list,
- ◆ Fill if required a description of this group in the zone Description,
- ◆ Click on the button **Validate** to validate the creation of this group, or on the button **Cancel** to cancel the creation.
- ◆ When the group is created, it appears in the **Groups** list (Figure 10).

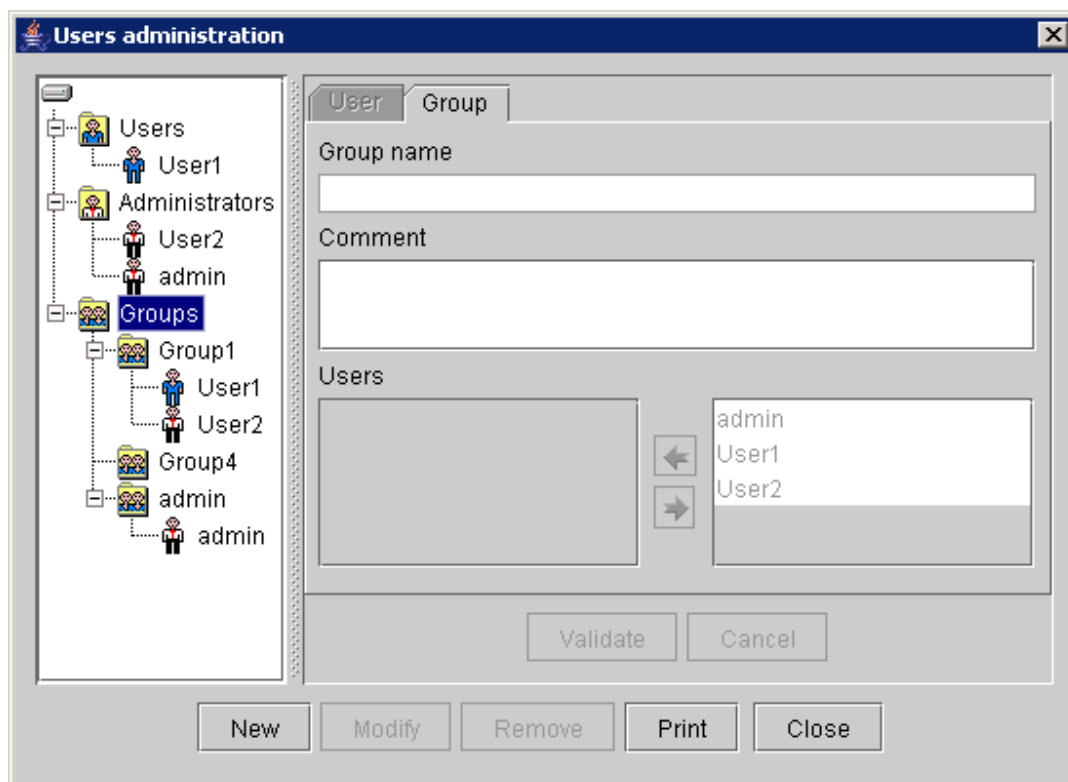


Figure 10 : Users group creation

To modify an existing group:

- ◆ Select the group to modify in the Groups list (click left on the mouse),
- ◆ Click on the button **Modify**,
- ◆ Do the required modifications to this group. To withdraw a user, select in **Users** area the user in the left list and use the icon ➡ to move it in the right list. To add a user proceed as indicated in the creation of a group,
- ◆ Click on the button **Validate** to validate the modifications, or on the button **Cancel** to cancel the modification on this group.

To delete an existing group:

- ◆ Select the group to delete in the Groups list (click left on the mouse),
- ◆ Click on the button **Remove**, the following window appears on the screen (Figure 11) :

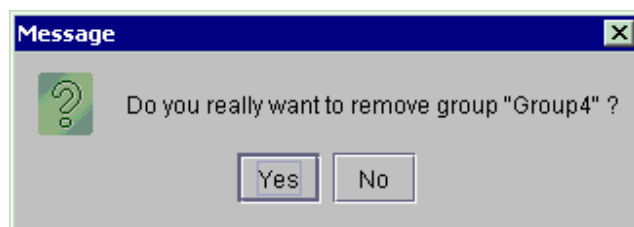


Figure 11 : message of the user group destruction

- ◆ Click on the button **Yes** to validate the modifications, or on the button **No** to cancel the deleting of this group.

CAUTION: Only a group with no users can be deleted.

Note: Only the **Master Administrator** profile created during the initialization of DAS database (Access ©, Oracle ©, or MySQL ©) can not be destroyed.

PRINTING OF ADMINISTRATION INFORMATION

The administrator can print the information contained in the Administration database by using the button Print from the *Users administration* window (Figure 12):

- The list of users,
- The associated workgroups.

Users list

Administrators

Login	Name
User2 admin	DURAND Jean Master administrator

Users

Login	Name
User1	DUPONT Pierre

Groups

Group name	Users
Group1 admin admins	

Figure 12 : Printing of Administration information

DATA ACCESS RIGHTS MANAGEMENT

In the database configuration mode (Access ©, Oracle ©, or MySQL ©), DAS allows the access right management to data (generic models in library and System architectures) thanks to dependencies and visibility attributes defined on each DAS object.

DEPENDANCIES

Any DAS object, generic model in library or system architecture is owned by user who created it. If this user belongs to several Workgroups, this one must specify during the creation of a DAS object, the Workgroup to which this object must be reattached.

Only the administrator of DAS Data Base or the owner of a given object can modify its dependencies by using the Editor of properties (Figure 12).

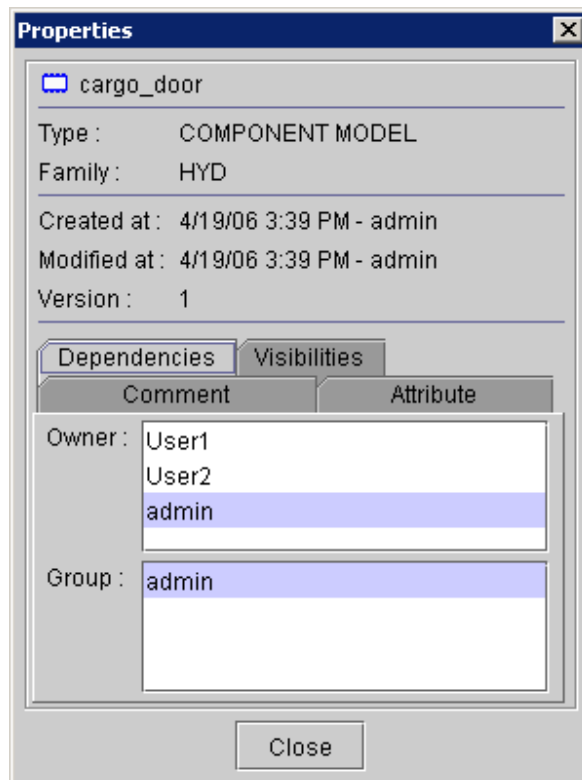


Figure 13 : Editor of Properties – Dependencies tab

To modify the owner of an DAS object:

- Select DAS-object (model or system) in the tree.
- Display Properties Editor with the File/Properties menu or with contextual menu.
- Click on the **Dependencies** tab,
- In the **Owner** area, select the user who must be the new owner of this object.
- If the new owner of this object belongs to several Workgroups,

- In the **Group** area, select the workgroup to which this object must be reattached
- Moreover, if the DAS-object is a project or a family, user can specify if the rights must be propagated to each included object with **Apply to included objects ...**

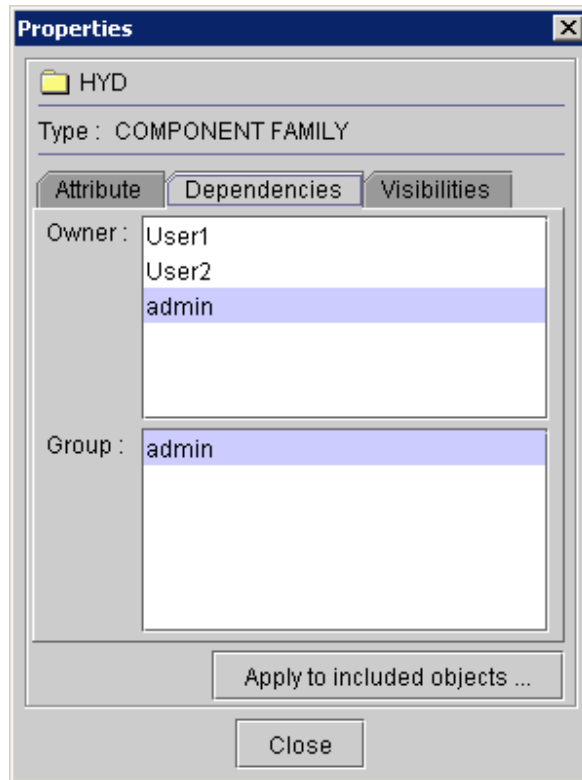


Figure 14 : Apply Dependencies to included objects

DATA ACCESS RIGHTS (ATTRIBUTES)

During the creation of an DAS object, the access rights defined by default are:

- Read Access Right for users of the workgroup to which this object is reattached (only the owner of this object has an Read/Write Access Right),
- No access for the users of the other workgroups.

Only the administrator of DAS Database or the owner of a given object can modify its attributes of access rights by using the Editor of properties (Figure 17).

To modify the access rights to an DAS object which are defined for workgroups:

- Select the DAS Object (model or architecture)
- Activate Editor of properties by clicking on Properties in File Menu
- Click on the **Attribute** tab,
- In the **Group access** area, select by using the popup menu the associated access rights for the workgroup to which this object is reattached,
- In the **Other access** area, select by using the popup menu the associated access rights for the other workgroups.

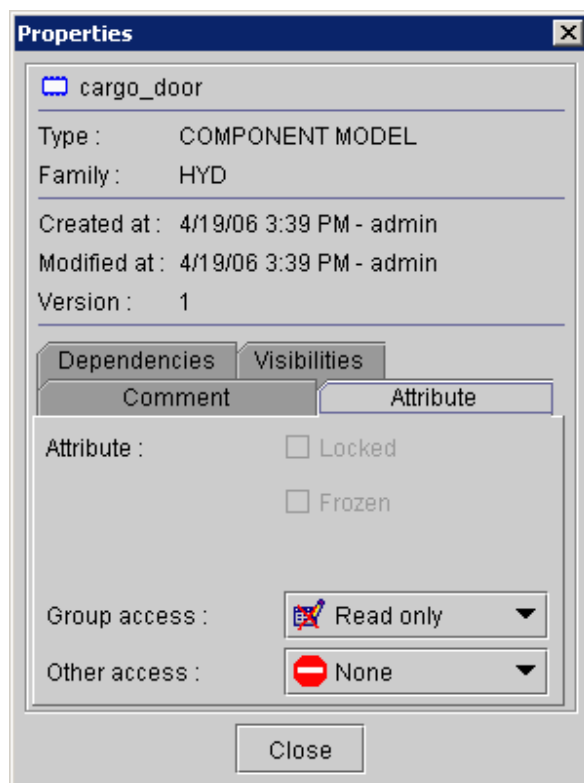


Figure 15 : Properties edition - Attribute

- Moreover, if you edit a Project or a Family of Models, you can specify if the rights must be applied to all included elements by clicking on Apply to included objects (Figure 16).



Figure 16 : Apply change to included Objects

VISIBILITIES DEFINITION

The **Visibilities** tab of **Properties** editor allow user to specify access rights to an DAS Object (generic models in library and system architecture) for each workgroup defined by using the administration module (See § 0). The access rights defined by default in the **Visibilities** tab are those defined in the **Attribute/Dependencies** tabs.

Only the administrator of DAS Data Base or the owner of a given object can modify its attributes of access rights by using the Editor of properties (Figure 17).

To modify the access rights to an DAS object which are defined for a workgroup:

- Select the DAS Object (model or architecture)
- Activate Editor of properties by clicking on Properties in File Menu
- Click on the **Visibilities** tab,
- In the Access Right area for this workgroup, select by using the popup menu the associated access rights for this workgroup (no access, read only, read/write).

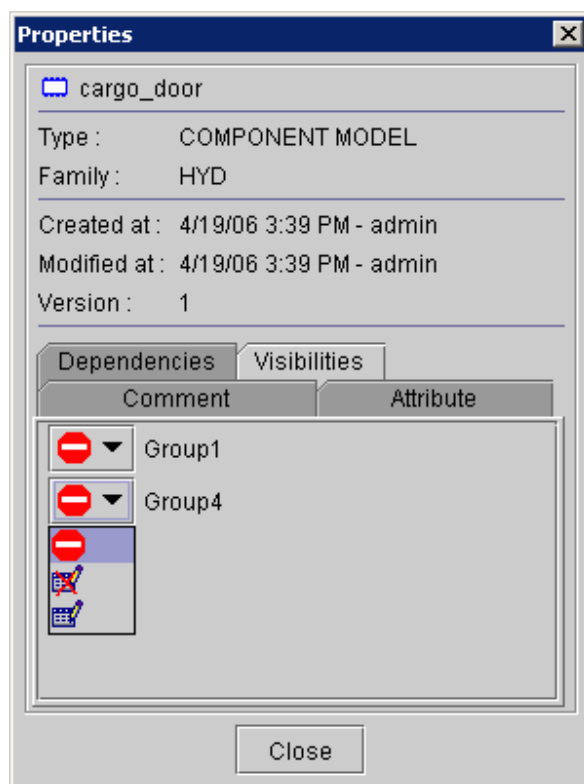
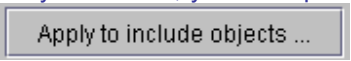


Figure 17 : Properties edition – Visibilities

- Moreover, if you edit a Project or a Family of Models, you can specify if the visibilities must be applied to all included elements by clicking on .

CONFIGURATION DEFINITION SAMPLE

The following diagram (Figure 18) configuration to define:

- Different users with their rights
- Users Group
- DAS Models Family and associated attributes (dependencies, rights, visibility)

The definition of FAMILY_B attributes (Figure 18) which are associated with this configuration, are shown Figure 19.

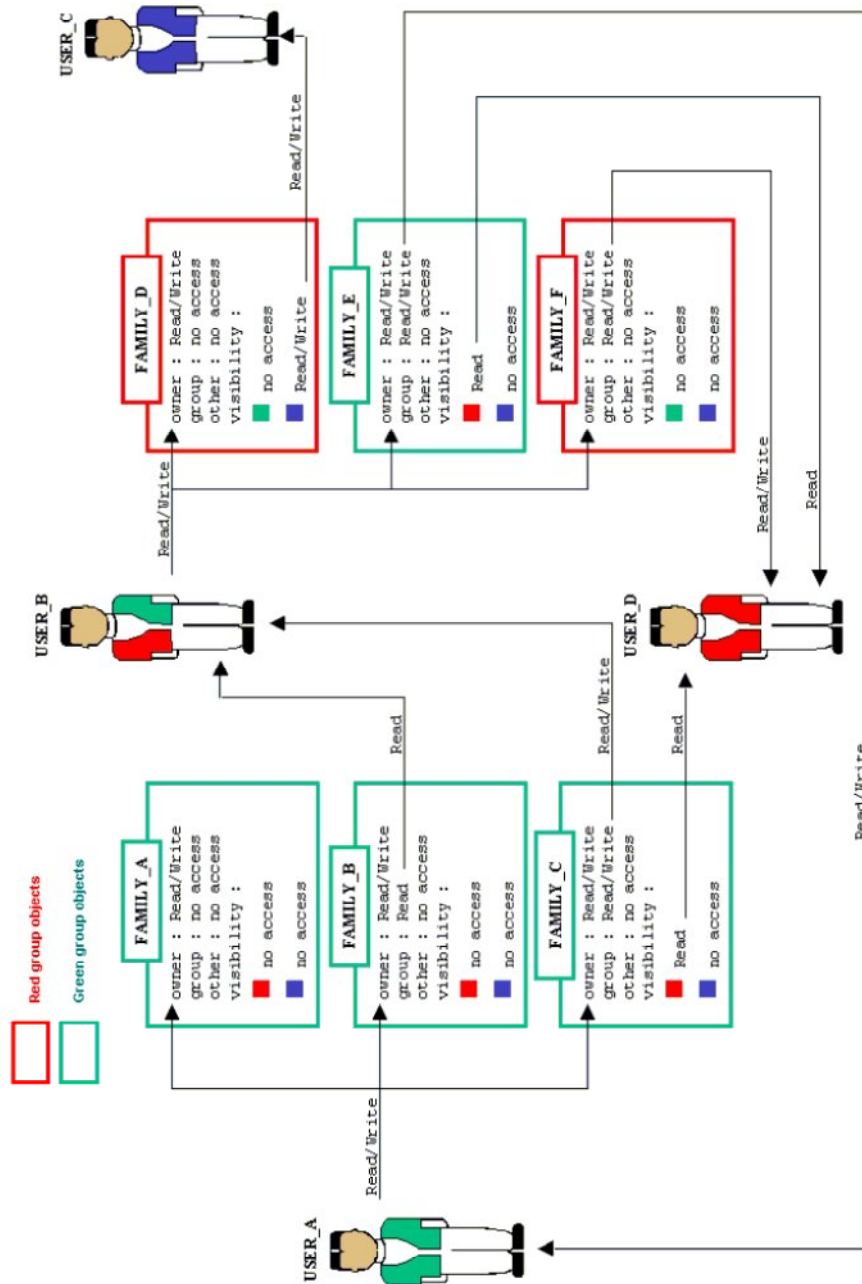


Figure 18 : Rights, Dependencies, Visibilities

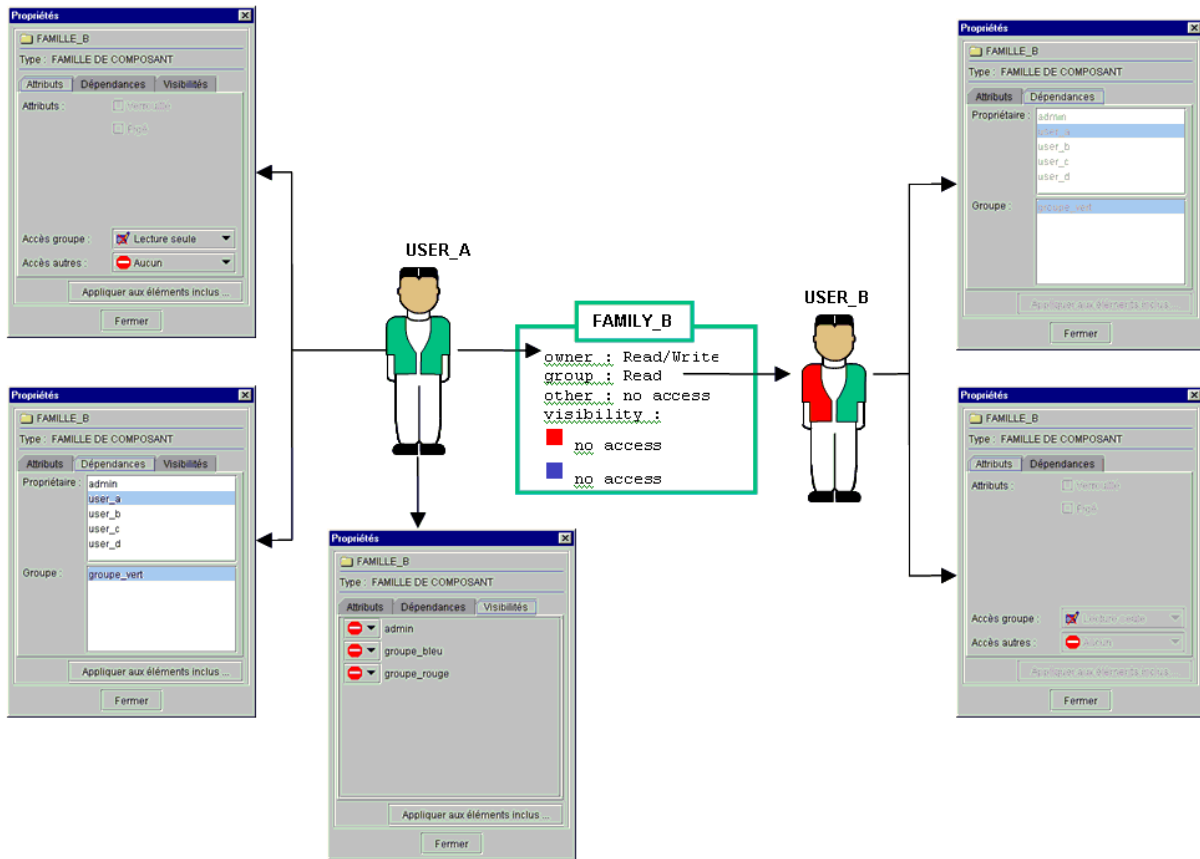


Figure 19 : FAMILY_B

WORK ENVIRONMENT PRESENTATION

The DAS tool work environment is divided into seven parts (Figure 15):

- ◆ A title bar (1).
- ◆ A menu bar (2) giving access to the tool functions.
- ◆ A main tool bar (3) enabling activation of some tool functions.
- ◆ A work area (4) in which are displayed the views and windows associated with the opened objects; the lower bar (5) indicates the objects opened in the work area. When the mouse cursor is positioned on an object in this bar, an information bubble indicates the project or model name.
- ◆ An information bar (6) indicating the time and date.
- ◆ An area (7) indicating the list of opened systems as well as their breakdown; the dimensions of this area can be adjusted by means of the arrows located on the top right part.
- ◆ An object (system, component, equipment, type, operator) handling area (8), enabling management of objects in the library and systems in the projects; the dimensions of this area can be adjusted by means of the arrows located on the top right part.

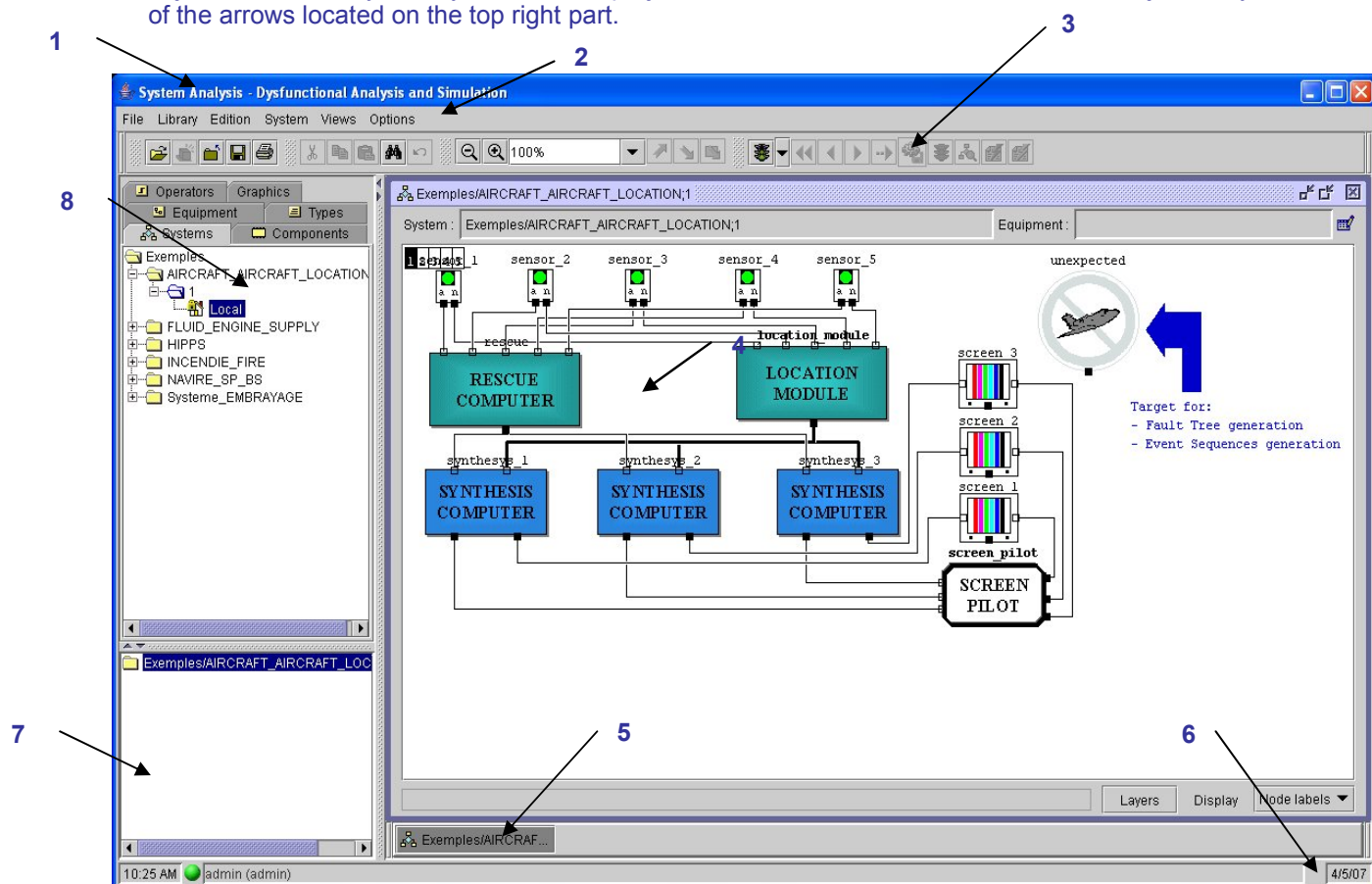


Figure 20 : Work environment

MENU BAR

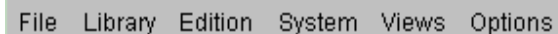


Figure 21 : Menu bar

The menu bar (Figure 21) is located at the top of the main window. It gives access to the various DAS tool functions. It consists of five menus:

- ♦ **File** menu: it contains all commands about projects or systems (Create, Open, Unlock, Save, Close, Formatting, Print and Quit).
N.B: The Unlock command is active on such models as components, equipment and operators.
- ♦ **Library** menu: it contains all commands about families and models (Create, Remove, Duplicate, Rename, Delete, Edit, Add, Copy and Paste).
- ♦ **Edition** menu: it contains all commands about object edition (Cut, Copy and Paste).
- ♦ **System** menu: it contains all commands about the exploitation of the system model in the course of edition (consistency check, simulation, dataflow generation, Java conversion).
- ♦ **Views** menu: it contains all commands about the views (Architecture, Consistency control, Simulation, Information, System links and Colours).
- ♦ **Options** menu: it contains all the configuration functions about the DAS tool (Preferences, Language choice and Help).

MAIN TOOL BAR

The main tool bar is located underneath the menu bar. It can be customized and can be composed of seven tool bars each of which enable a quick activation of the DAS tool functions:

- ♦ Standard tool bar (Figure 22): it gives access to the following functions:
 - * Open system,
 - * Create new system,
 - * Close system,
 - * Save system,
 - * Print architecture (this command is not specific to an architecture)



Figure 22 : Main tool bar

- ♦ Edition tool bar (Figure 23): it gives access to the following functions:
 - * Cut,
 - * Copy,
 - * Paste,
 - * Search model or component,
 - * Undo last command.



Figure 23 : Edition tool bar

- ◆ Design tool bar (Figure 24): it gives access to the rotation commands on a component or an equipment:
 - * Vertical mirror,
 - * Horizontal mirror,
 - * 90° rotation.

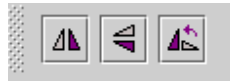


Figure 24 : Design tool bar

- ◆ Links tool bar (Figure 25) : it gives access to various kinds of links between components or equipment in an architecture:
 - * Direct,
 - * Right angle,
 - * Right angle from right to bottom,
 - * Right angle from left to down,
 - * Right angle from right to top,
 - * Right angle from left to top,
 - * Possibility of specifying an orientation for the graphic link symbolizing the connection (an arrow is drawn to indicate the direction of the flow propagation),
 - * Possibility to modify the body of the graphic link (thick line, double line)



Figure 25 : Links tool bar

- ◆ Alignment tool bar (Figure 26) : it gives access to the alignment commands between components or equipments in an architecture :
 - * Alignment on the left of the selected elements,
 - * Alignment on the right of the selected elements,
 - * Alignment on the top of the selected elements,
 - * Alignment on the bottom of the selected elements,
 - * Centred alignment of the elements selected on a vertical axis,
 - * Centred alignment of the elements selected on a horizontal axis,
 - * Centre the selected elements.

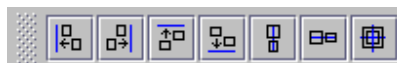


Figure 26 : Alignment tool bar

In all the cases the alignment is carried out the first selection element.

- ◆ Navigation tool bar (Figure 27) : it zooms in and out on the various graphs displayed in the work area and helps to navigate in the architectures:
 - * Backward zoom,
 - * Forward zoom,
 - * Zoom definition,
 - * Up,
 - * Down,
 - * Down in another view.



Figure 27 : Navigation tool bar

- ◆ The tool bar dedicated to the simulation of the system model gives access to :
 - * Go,
 - * Initialize,
 - * Return to begin,
 - * Continue simulation,
 - * Save as initial state,
 - * Stop simulation,
 - * Variables display,
 - * Selection of an event to be started,
 - * Modification of a state variable,



Figure 28 : Simulation tool bar

The main tool bar can be customized in the following way:

- ◆ Use the **Options – Preferences...** menu; the Preferences window is displayed (Figure 182).
- ◆ Select the **Tools bars** tab and tick the tool bars which will be displayed on the main tool bar on top of the screen.

WORK AREA

The work area is the main part of the DAS tool graphic interface. It enables display of the various views associated with the objects or system architectures in current edition.

This area comprises, in its lower part, a bar which displays buttons corresponding to all opened windows (Figure 22).

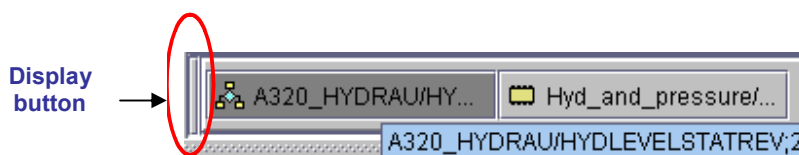


Figure 29 : DAS objects in course of edition

Note: If the number of views is important, a display mechanism associated to the two rectangular buttons located on both sides of the bar (Figure 29) allows displaying of the information bar respectively to the right or the left, and thus to reach the buttons previously invisible.

Possible actions:

- A Popup menu accessible by a right click on one of the buttons, can restore, reduce, increase or close the edition window of the associated view.
- The moving of the mouse cursor on one of the icons located on the open objects bar, allows us to have an information bubble indicating the system or the model name in course of edition.
- The keyboard command **ALT+<SPACE>** can shift from the current edition of a view to another (Figure 30).



Figure 30 : Edition shifting mechanism

INFORMATION BAR

The information bar is located at the bottom of the workspace. It shows the following information's: time, up to date indicator, user name, user group, current date.

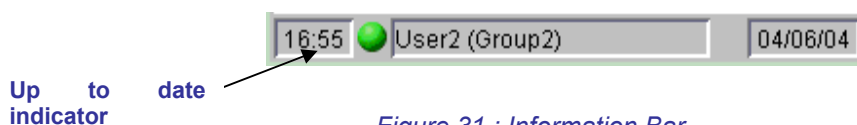


Figure 31 : Information Bar

When DAS is used in Database mode (Access ©, Oracle ©, or MySQL ©), it allows to manage concurrent access in a multi-user environment.

The indicator shows the validity of data in object manager. Icon is green when the content of the object manager is the same as the content of the database (usually, until users have created, modified or deleted an object).

When the contents are different, the icon becomes red.

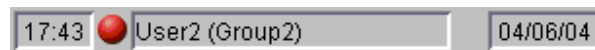


Figure 32 : Not Up To Date

To update the object manager, the user must follow these steps.

- Left-Click on red icon to validate the update. Object manager will take into account all modifications made by other users connected to database.

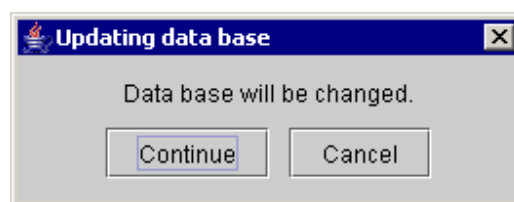


Figure 33 : Update object manager

- Click on Continue to confirm the update.

MENU TREE STRUCTURE

MENU BAR

The menu bar comprises the following menus:
File menu (Figure 34):

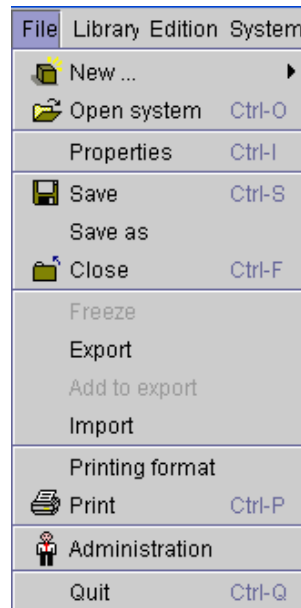


Figure 34 : File menu

- ♦ Library menu (Figure 35):

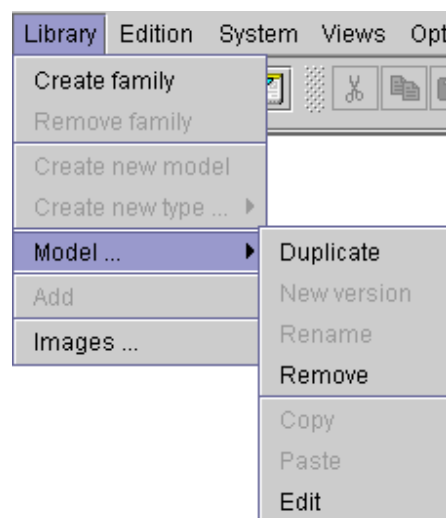


Figure 35 : Library menu

- ♦ Edition menu (Figure 36):



Figure 36 : Edition menu

- ♦ System menu (Figure 37):

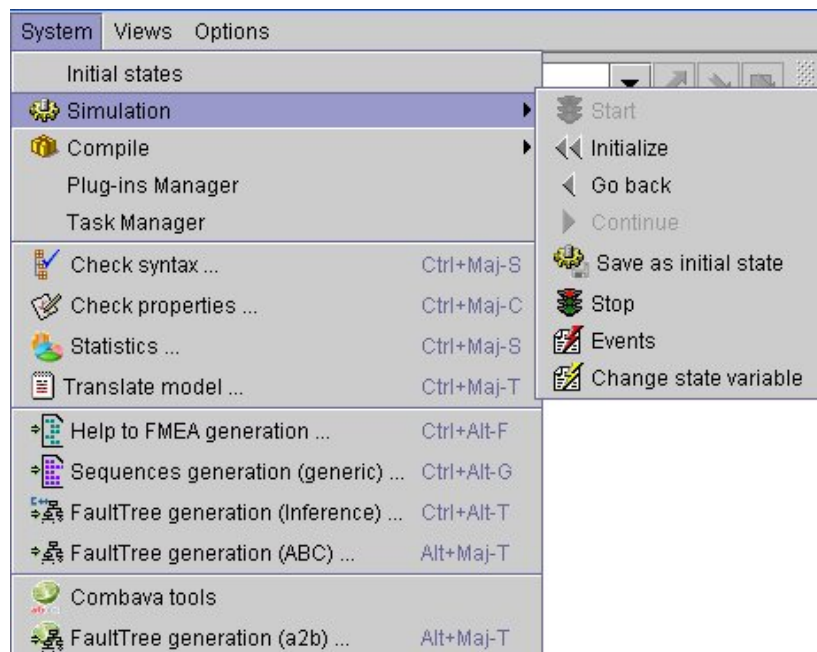


Figure 37 : System menu

- ♦ Views menu (Figure 38):

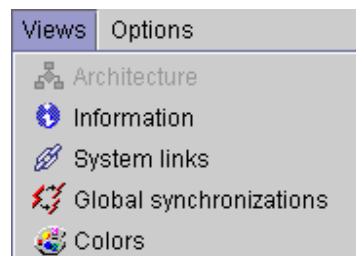


Figure 38 : Views menu

- ◆ Options menu (Figure 39):

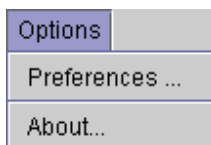













Figure 39 : Options menu

SUMMARY LIST FOR MENUS, SHORTCUTS AND ICONS

The menu breakdown is the following:

- ◆ File:
 - * New ...
 - Project
 - System 
 - * Open system  CTRL + O
 - * Properties CTRL + I
 - * Save  CTRL + S
 - * Save as
 - * Close  CTRL + F
 - * Freeze
 - * Export
 - * Add to export
 - * Import
 - * Printing format
 - * Print  CTRL + P
 - * Administration 
 - * Quit CTRL + Q
- ◆ Library:
 - * Create family
 - * Remove family
 - * Create new model
 - * Create new type ...
- Enumerate
- Record
 - * Model ...
- Duplicate
- New version
- Rename
- Remove
- Copy
- Paste
- Edit
 - * Add
 - * Images ...
- ◆ Edition :
 - * Cancel CTRL + Z
 - * Cut  CTRL + X
 - * Copy  CTRL + C
 - * Paste  CTRL + V
 - * Search  CTRL + B
 - * Select All  CTRL + A

◆ System:

* Initial states

* Simulation

- Start
- Initialize
- Go back
- Continue
- Save as initial state
- Stop
- Events
- Change state variable



* Compile

• Compile with inference rules ...

• Compile in Byte Code Java [1] ...

◆ Views:

* Architecture

* Information

* System links

* Global synchronizations

* Colors



◆ Options:

* Preferences ...

* About ...

STANDARD MODELS LIBRARY

GENERAL

The standard models library contains the following main families:

- ♦ Types,
- ♦ Operators,
- ♦ Components,
- ♦ Equipments,

N.B : Projects are displayed in the **Systems** tab, but are not included in the library of reusable components.

EDITION ON COMPONENT

Create family of components

To create a new family of components as follows:

- In the left part of the screen, click on the **Components** tab to gain access to the list of families of components.
- Click on the mouse right button to display the following Popup menu (Figure 40)

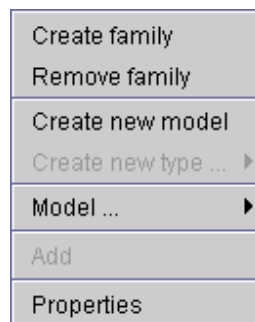


Figure 40 : Family – Popup menu

- Click on the **Create family** command.
- The Create components family window is displayed (Figure 41):

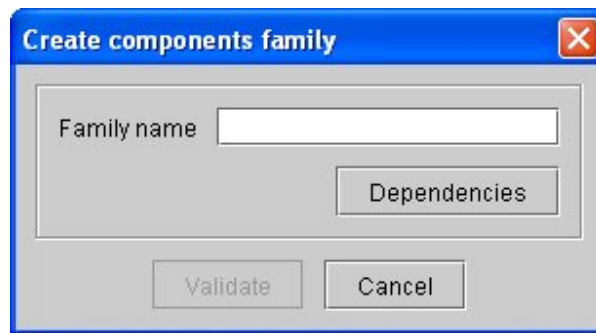


Figure 41 : Components - Create family

- In the **Family name** field, type the name of the new family.
- If the family's owner belongs to several workgroups (See § 0), he must specify the group to which must be reattached the family created.
- Click on the button **Dependencies** and select in the groups list which belongs the user, the owner group on which must belong the family created. Click on the **Close** button to validate the selection.



Figure 42 : Selection of the owner group

- Click on the **Validate** button, otherwise on the **Cancel** button; the new family appears in the **Components** tab in the left part of the screen.

Remove a family of components

Remove a family of components as follows:

- ♦ In the left part of the screen, click on the Components tab to gain access to the list of families of components.
- ♦ Select the family name to be removed (left click).
- ♦ Click on the mouse right button to display the contextual menu (Figure 40).
- ♦ Click on the Remove family command.
- ♦ If the family is not empty, an error message indicates that the family removal is impossible.
- ♦ Click on the OK button.
- ♦ Remove each component in the family and then remove the family

Create a new component model

Create a new component model as follows:

- ♦ In the left part of the screen, click on the Components tab to gain access to the list of families of components.
- ♦ Select the family in which the component will be created.
- ♦ Click on the mouse right button to display the contextual menu (Figure 40).
- ♦ Click on the Create new model command; the following bar graph is displayed (Figure 43)

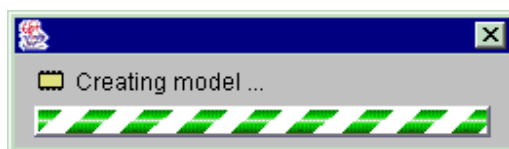


Figure 43 : Creating model bar graph

- ♦ Then the Model editor window is displayed (Figure 44).
- ♦ Type the corresponding information in the various tabs

A component model under DAS can be considered as a flow processor which:

- has an internal state
- reacts to events
- receive and/or send information through (**I/O**) interfaces which enable him to communicate with other components (by flow propagation).

When an event occurs, the component determines the internal change of state induced by his release, and the new flow values emitted by its interfaces.

The editor model for the component model allows the user to describe his mechanisms by the means of the following tabs:

- The tab **General** allows to fill the general information associated to the model.
- The tab **I/O** allows to fill the list of the information flow variables transmitted or received by the component.
- The tab **States** allows to fill the list of the component state variables characterizing the various states of the component : functional or dysfunctional;
- The tab **Events** allows to specify the events on which react the component (phenomena modifying the intern state of the component).
- The tab **Icons** allows to specify the graphic representations corresponding to the various states of the component (used during the graphic simulation of the system).
- The tab Altarica **code** allows to specify the state changing conditions of the component and how the information flow propagation is carried out in each one of these states.

When the new model created is saved, it is automatically inserted in the administrator (alphabetically) as well as his version.

Remark: A new component model is created in the version 1.0

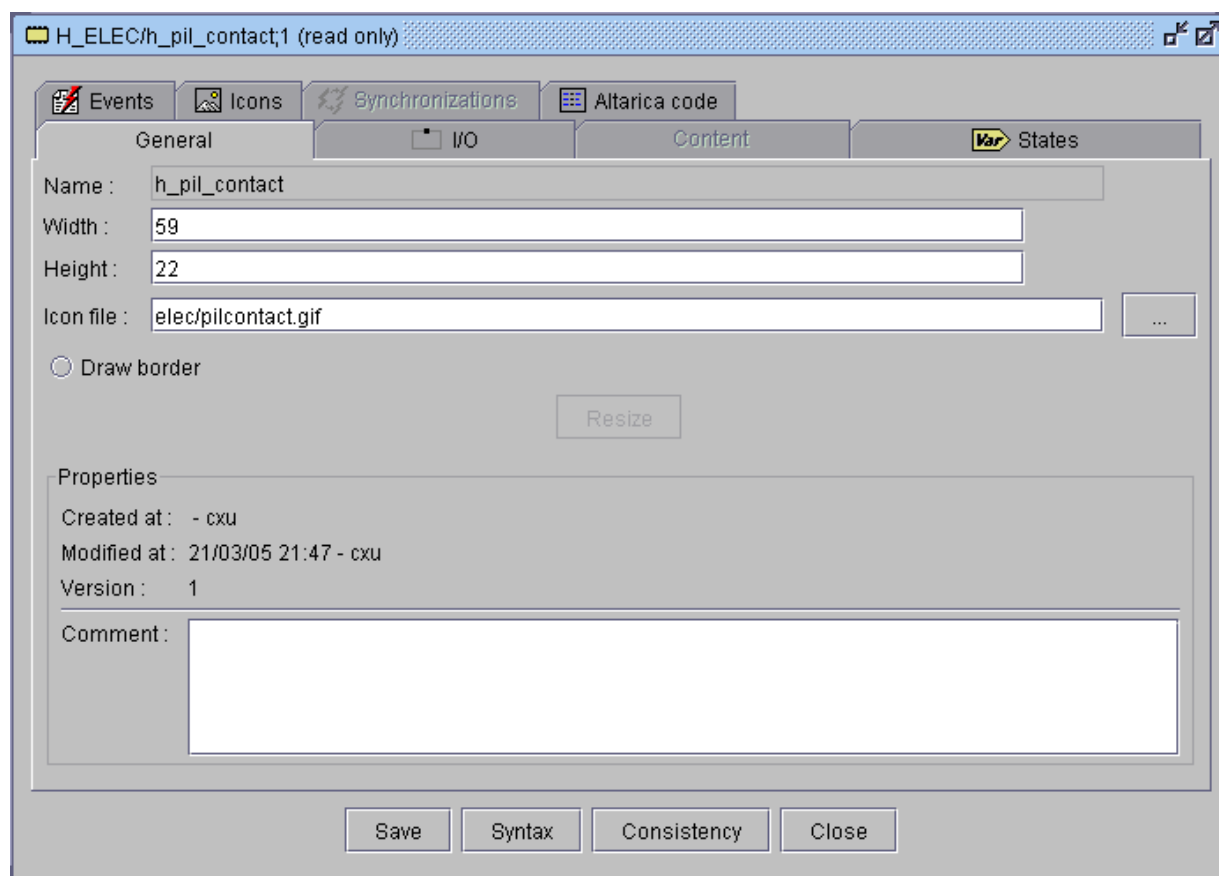
General tab

The **General** tab (Figure 44) allows the following information to be typed in:

- ♦ Name: type the model name,
- ♦ Note: In order to save the model, the name is the only obligatory field.
- ♦
- ♦ Width: ll (icon width default value),
- ♦ Height: hh (icon height default value),
- ♦ Icon file: the ... button enables selection of the icon to be assigned to the model.

Remark: Icons are image files in “*.gif” or “*.jpg” format.

- ♦ Tick Draw border to display or not the border associated to the icon.
- ♦ Click on the Resize button to validate, if necessary, the icon larger dimensions.
- ♦ Comment : fill the comment describing the created component.



H_ELEC/h_pil_contact;1 (read only)

Events Icons Synchronizations Altarica code

General I/O Content States

Name : h_pil_contact

Width : 59

Height : 22

Icon file : elec/pilcontact.gif ...

☐ Draw border

Resize

Properties

Created at : - cxu

Modified at : 21/03/05 21:47 - cxu

Version : 1

Comment :

Save Syntax Consistency Close

Figure 44 : Component – General tab

All the model editor tabs contain four common buttons:

- ◆ Save: allows storing of all typed information,
- ◆ Syntax: allows a syntax control on the Altarica code,
- ◆ Consistency: allows a consistency control on the Altarica code,
- ◆ Close: allows closure of the model editor.

Two icons on the top right hand side allow window size modification:

- ◆ Reduce,
- ◆ Extend.

Remark: A component may be un-modifiable, It happens when another user as locked this component, or if component is opened from the simulation view. In these cases, the **Save** button is replaced by a **Save as** button.

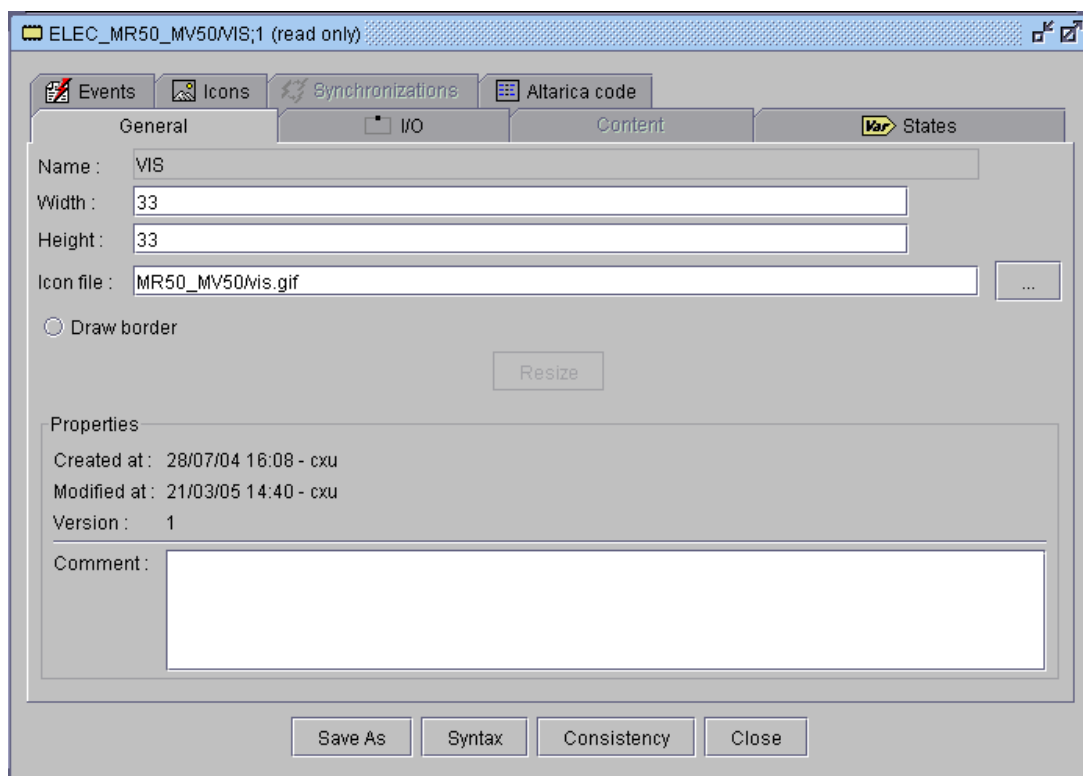


Figure 45 : Model modification during simulation

Click on **Save As**, the dialog box (Figure 46) allows users to choose the new name of the component.

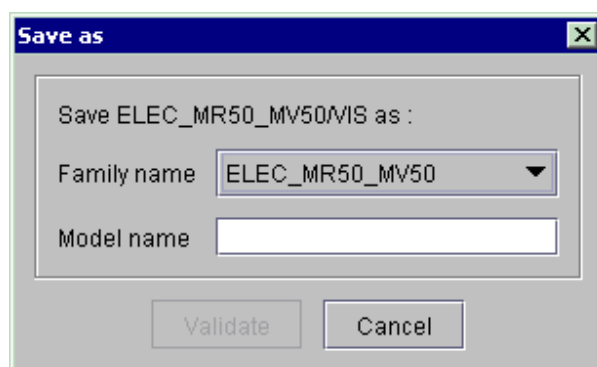


Figure 46: Model – Save As

The Save As button is available in models Editors as soon as models can't be modified.

I/O tab

The **I/O** tab (Figure 47) allows configuration of a component input/output flow variables.

The **I/O** flow variables are the support for the information flows exchanged between the different components of the system.

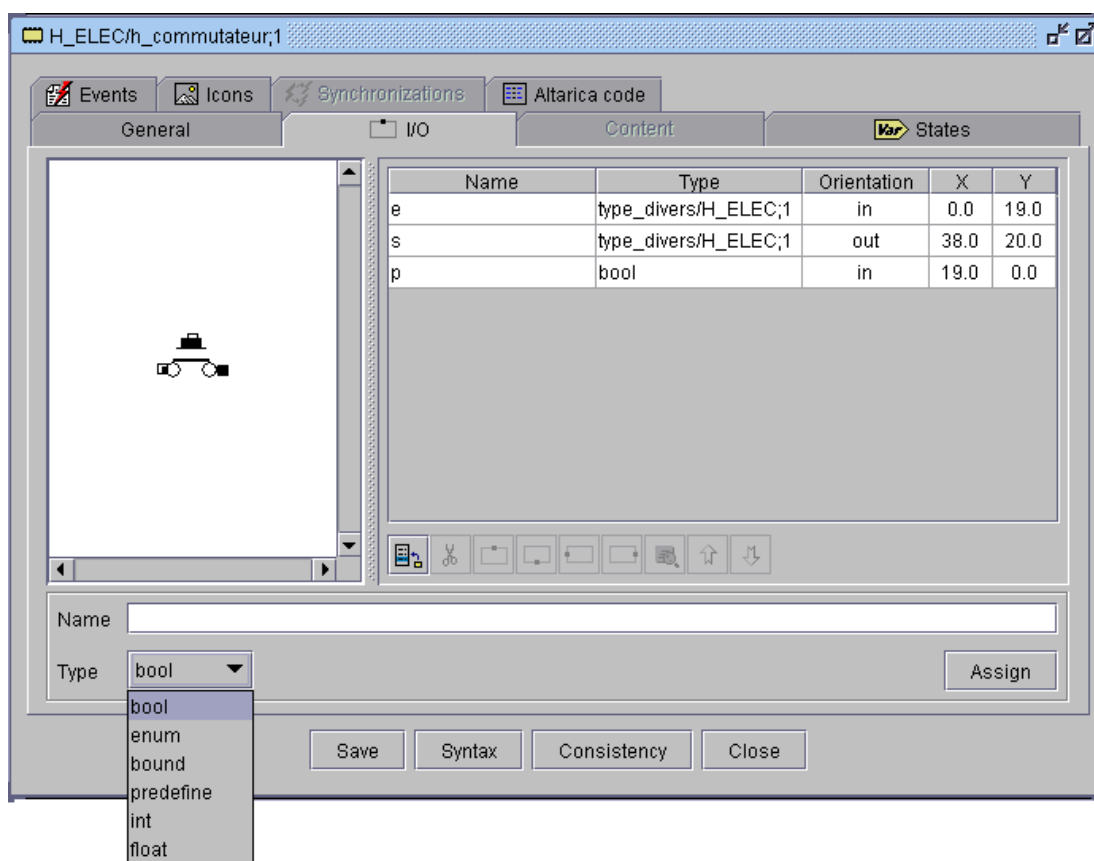


Figure 47 : I/O tab - Boolean variable

- ◆ In the Name field, type the I/O name.
- ◆ Use the Type menu and choose the variable type:
 - * **bool:** **boolean** variable (Figure 47), containing the constants **true** and **false**
 - * **enum:** **enumerate** variable (Figure 49); values **must** be separated by commas without spaces, e.g. : null,low,high,





N.B: The “space” character is forbidden when keying in the *enumerate* variable.

- * **bound:** variable whose value varies between a minimum value and a maximum value (Figure 50),
- * **predefine:** **predefined** type in library (Figure 51)
- * **int :** **integer variable**
- * **float :** **float variable**

- ◆ Click on the  icon or the Enter key; the I/O appears on the component border (the default type is “bool”).

The line corresponding to the typed **I/O** is added in the **I/O** table with the following information (*Name*, *Type*, *Orientation*, X, Y); X and Y indicate the coordinates of the input (**in**) or output (**out**) port location on the icon with respect to an origin at the top left-hand side; the default orientation is “in”.


- ◆ Position the **I/O** on the model by clicking on the corresponding icon:

- * **Top:**  icon,
- * **Down:**  icon,
- * **Left:**  icon,
- * **Right:**  icon.

N.B: Use the mouse to position an **I/O** anywhere on the component border.

N.B: An input is identified by a white rectangle and an output by a black one.

- ♦ Three other icons are available:

- * Edit I/O type: use the  icon or double click with the mouse left button on the Type cell of the selected I/O line,
- * Move up ,
- * Move down .

- Position the cursor on a line and click with the mouse right button to display the contextual menu (Figure 48).

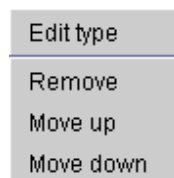



Figure 48 : Contextual menu

- ♦ Repeat the same procedure for the other **I/O**.

Remark: If an **I/O** is selected, the **I/O** added with the  icon is positioned before the selected **I/O**. If no **I/O** is selected, the **I/O** is positioned at the end of the list.

- If necessary, select the **I/O** and click on the  icon to delete any selected **I/O**

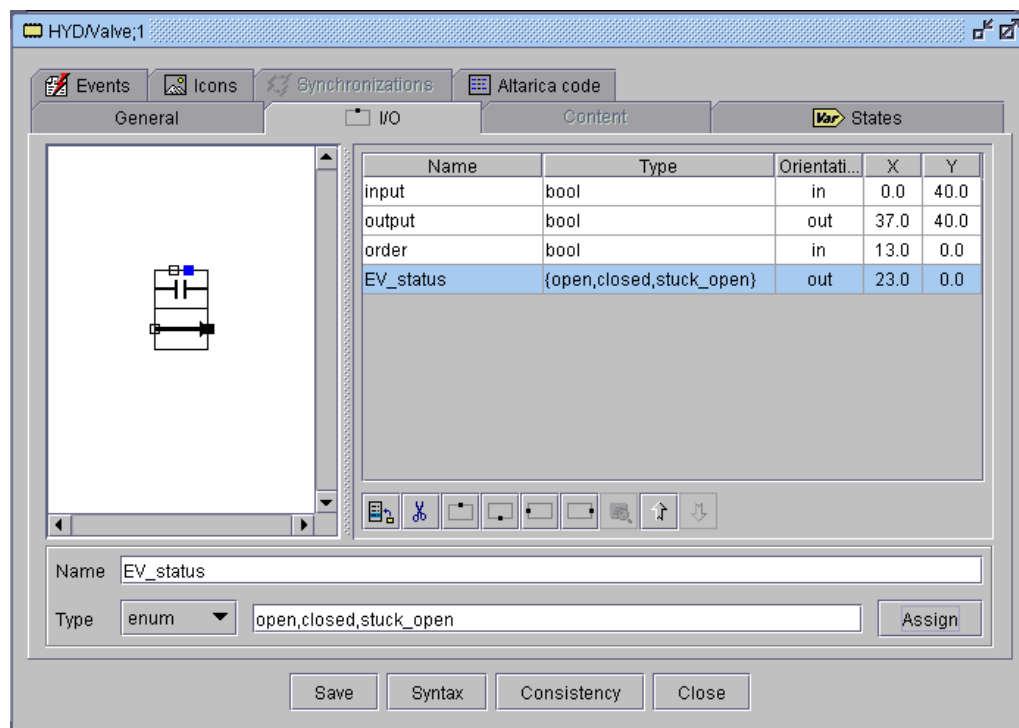


Figure 49 : I/O tab – Enumerate variable

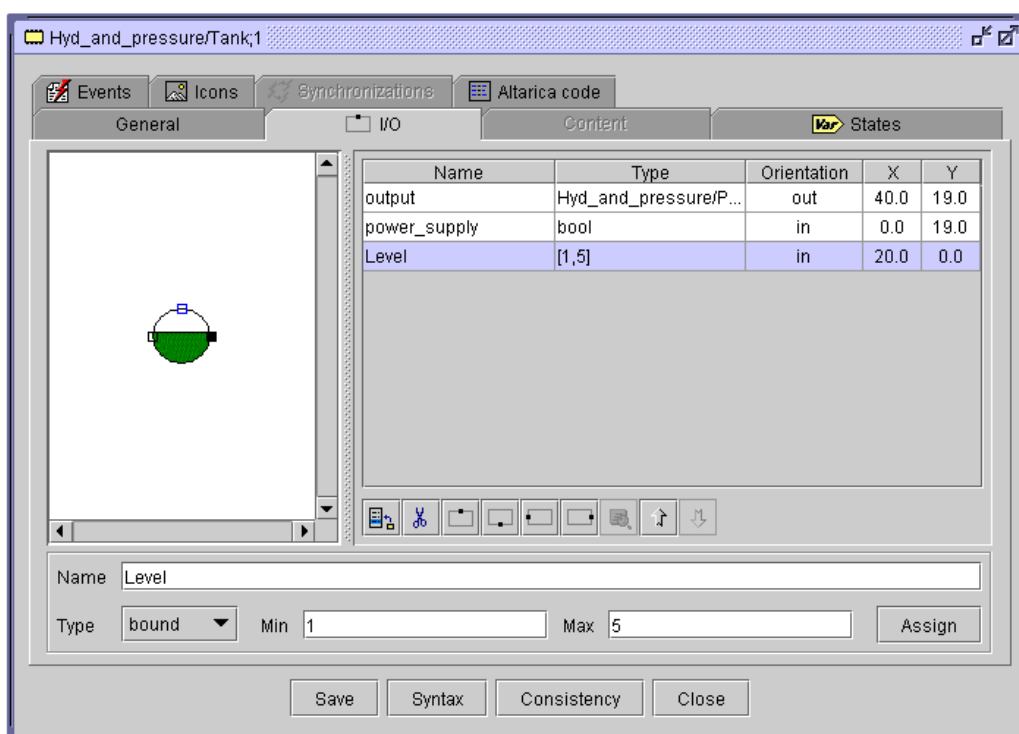


Figure 50 : I/O tab – Bound variable

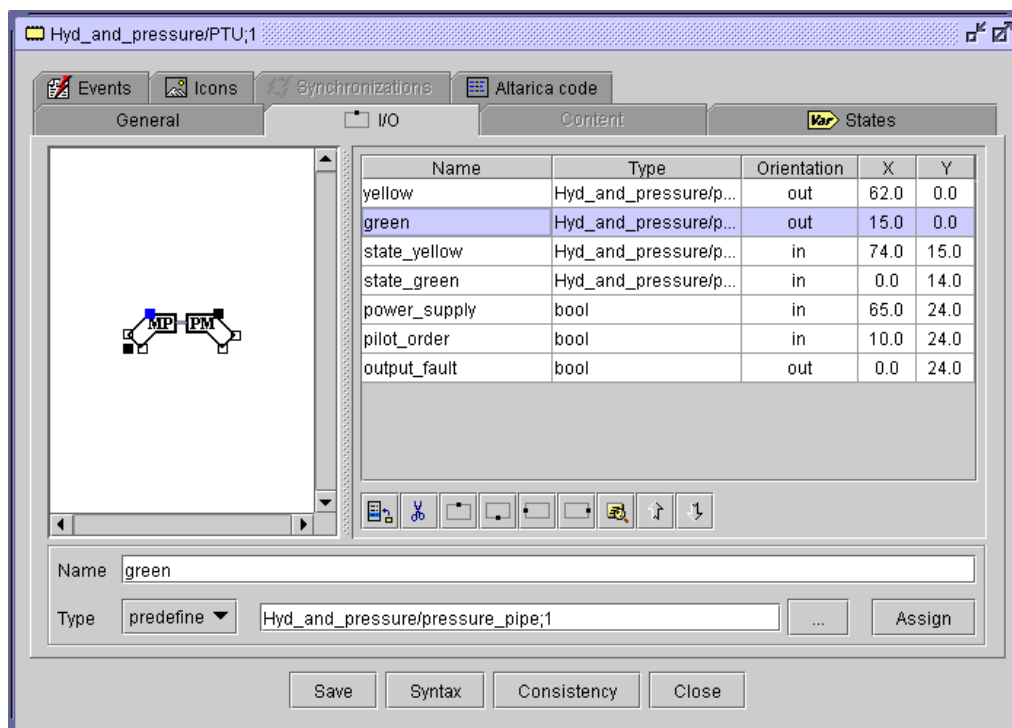


Figure 51 : I/O tab – Predefined variable

- ◆ If necessary, select the **I/O** and click on the **Assign** button; the type corresponding to the selected **I/O** is then updated.
- ◆ To modify the orientation of an **I/O**, click on the **Orientation** cell in the associated line with the mouse left and then right button; in the pull-down menu, assign the flow orientation (Figure 52). The possible values are the following:
 - * **In**: for an input interface.
 - * **Out**: for an output interface.
 - * **local**: for a local interface with no relation with the outside. This local value allows the use of the associated flow variable as a “memory variable” in the Altarica code.

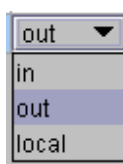



Figure 52 : I/O orientation

- To modify the name of an existing **I/O**, click on the **Name** cell; modify the name and validate with the **Enter** key.

States tab

The **States** tab (Figure 53) allows typing of a component states:

- ◆ In the **Name** field, type the name of the state variable.
 - ◆ Use the **Type** menu and choose the variable type:
 - ◆ **bool**: boolean variable,
- Dysfunctional Analysis & Simulation- User Guide- V5.6- Business Process Accelerator
Do not reproduce, copy or use without a license from Dassault Systèmes
© 2008. Dassault Systèmes, All Rights Reserved

- ♦ **enum**: enumerate variable, e.g. ok,failure,
- ♦ **bound**: variable whose value varies between a minimum value and a maximum value,
- ♦ **predefine**: predefined type in library (Figure 51). For the version 3.0 only the enumerate types must be used for the state variables.
- ♦ **int** : integer variable
- ♦ **float** : float variable
- ♦ Click on the  icon or the **Enter** key.
- ♦ For example: As shown in figure 41, the state is displayed on a line which indicates: its **Name** (state_), the values domain of its **Type** (ok, failure) and its default **Value** (ok). The ok value is the one taken by default before the simulation, during the initialization of the state variable.

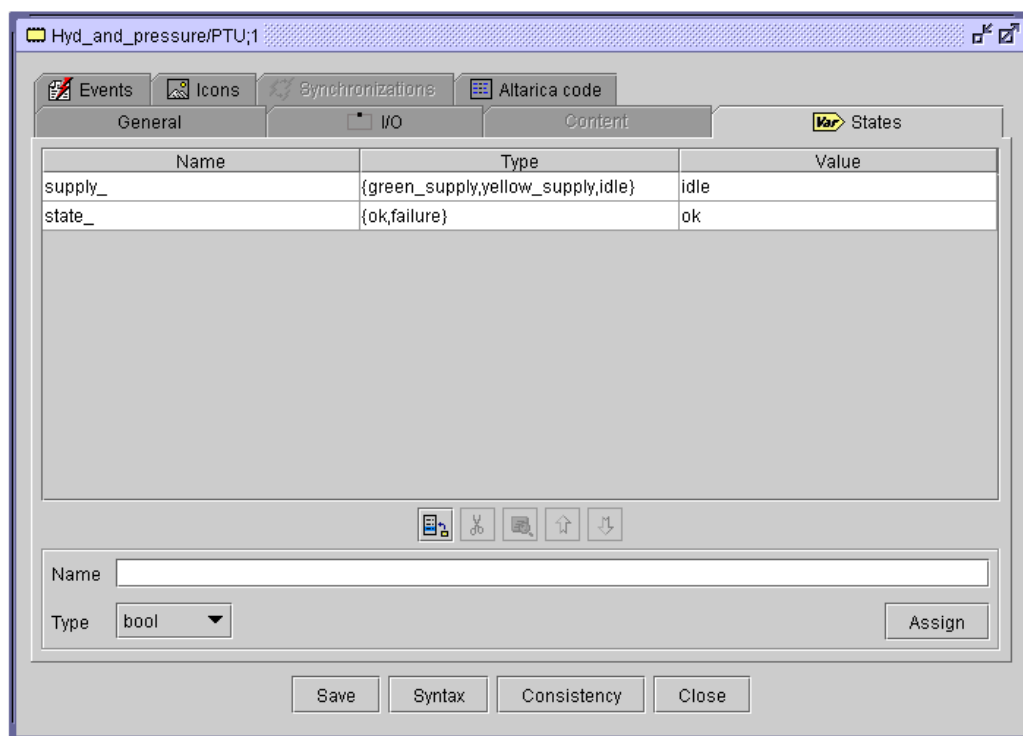



Figure 53 : Component – State tab

Remark: If one of state variables is selected, the added state variable (with the  icon) is positioned before this selected variable. If no state variable is selected, the added state variable is positioned at the end of the list.

- ♦ To modify the type of an existing state variable, select the state variable: select a new type for this variable and click on the **Assign** button to associate the selected type to the state variable.
- ♦ To modify the name of an existing state, click on the **Name** cell; modify the name and validate with the **Enter** key.
- ♦ Click on the **Value** cell of the selected state variable to modify, if necessary the default value.
- ♦ Three following icons are available:
- Edition of the preset type of the state variable

Edit the preset type of the state variable: use the icon , or do a left double click on the cell **Type** of the selected state variable,

Modification of the definition order of the state variable

Move to the top of the line of the selected state variable : 

Move to the bottom of the line of the selected state variable : 

Events tab

The  **Events** tab (Figure 54) allows typing of component events:

- ♦ In the *Name* field, type the event name.
- ♦ Click on the  icon or the **Enter** key.

For example: The “*Failure*” event is displayed on a line

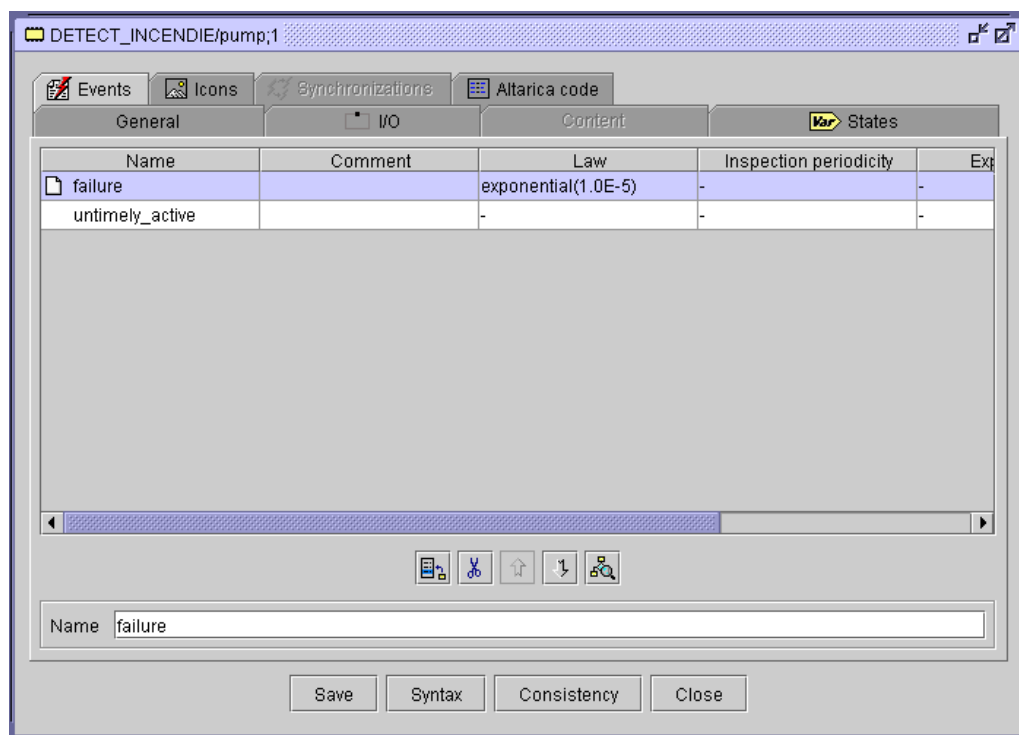



Figure 54 : Events tab

To modify the name (or law) of an existing event, double click on the *Name* (or *Law*) cell; modify the name and validate the new name (or law) with the **Enter** key

Remark: If an event is selected, the added event is positioned before the selected event. If no event is selected, the added event is positioned at the end of the list

- Click on the icon  to delete the selected event.
- Click on the icon  to move the event line selected.
- Click on the icon  to move the event line selected.

- Click on the icon  to edit the properties of the selected event (Figure 55).

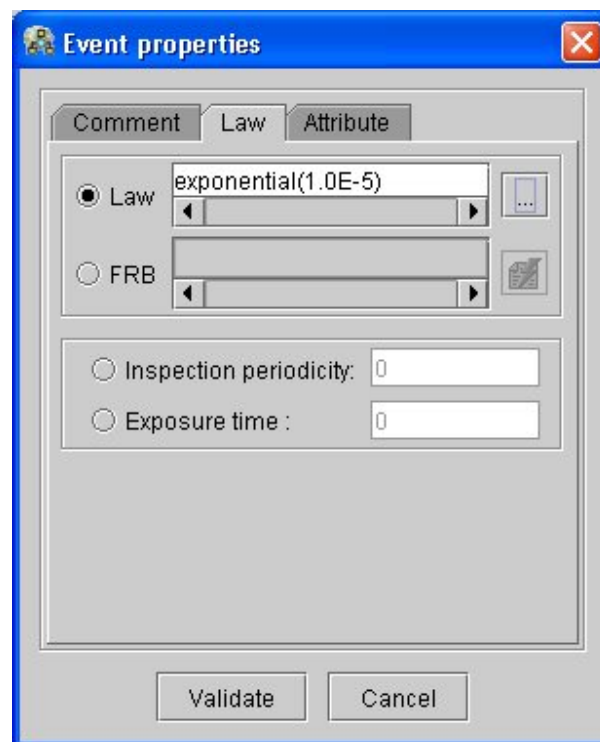


Figure 55 : Event properties

- Specify a comment in the field Comment.

Remark : The comment filled in this zone will be used as the comment of the corresponding basic event in the default tree generated by the DAS tool.

- The field Law allows the definition of a probability law for the event. This law will be used during the quantification of the tree generated by the DAS tool.
- We have 2 methods to specify the probability law :

1. Select the Law radio button and fill the field in or alternatively, click on the right button to access to the law editor assistant window (See Figure 56). The law entered in the field must be In AltaRica format (cf. Annexe 2). The associated syntax is the following:

<Law_Name> (<Parameters_List>); parameters are separated with comma.

Currently the available probability laws are:

- constant** (<probability>) ; example : constant(0.5) ;
- exponential** (<lambda>) ; example : exponential(1e-6) ;
- GLM** (<gamma>, <lambda>, <mu>) ; example : GLM(0.5,1e-6,1e-3) ;
- Weibull** (<alpha>, <beta>) ; example : Weibull(0.5,1.5) ;
- Periodic_test** (<lambda>, <tau>, <t0>) ; example : periodic_test(1e-3,10,0) ;
- GLM_asymptotic** (<lambda>, <mu>) ; example : GLM_asymptotic(1e-6,1e-3) ;
- CMT** (<lambda>, <duration>, <probability>) ; example : CMT (1e-3, 72, 0);
- Dirac** (<delay>); example : Dirac(10.0);
- dormant** (<lambda>, <MTTR>, <period>) ; example : dormant (1e-3, 10, 25) ;
- periodic** (<Delay1>, <Delay2>) ; example : periodic(100.0,10.0);
- unif** (<Delay1>, <Delay2>) ; example : unif(10.0,20.0);

2. Select the FRB radio button, then fill the field in or alternately click on the right button to access to the FRB (Failure Rate Bank) tree manager that shows the existing event models (See Figure 58). The path entered in the field must be an event model from the FRB (Failure rate Bank) defined in FaultTreeAnalysisSystem (See User's Guide - FaultTreeAnalysisSystem Module: Failure Rate Bank).

- The field Attributes allows the user to specify attributes (or properties) in order to characterize the event more precisely. These properties can be used during the simulation phases in order to filter the events. The associated syntax is the following:
- '<Attribute_Name>' <Attribute_Value> ; Example : 'Type=' "Valve"

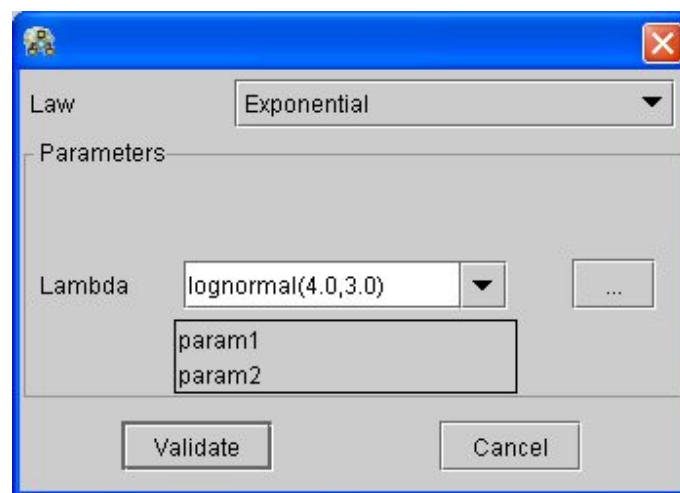


Figure 56 : Law editor assistant window

If the user clicks on the arrow of a parameter combo box, then the available existing named parameters become selectable.

If the user click on the buttons located on the right of each parameter line, then the following window appears (Figure 57):

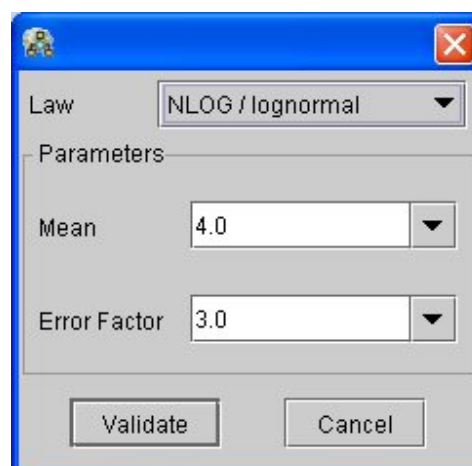


Figure 57 : Parameter law editor assistant window

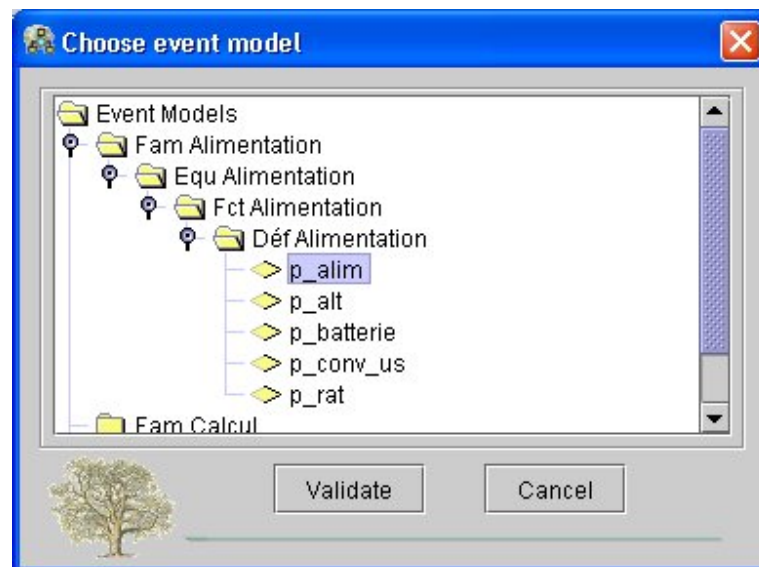


Figure 58 : Event model link editor assistant window


Icons tab

General

The **Icons** tab (Figure 59) allows selection of the icon(s) which graphically describe(s) the component behaviour during the simulation of the system architecture:

- ◆ In the right part of the tab, select the directory in which the icon is stored.

Remark : the last reached directory is memorized and is automatically displayed during the following selection.

- ◆ Select the desired icon.
- ◆ Click on  to transfer the selected icon from the right to the left part of the tab which displays at the end of the list the icon number (used in the Altarica code), and the icon graphical representation.

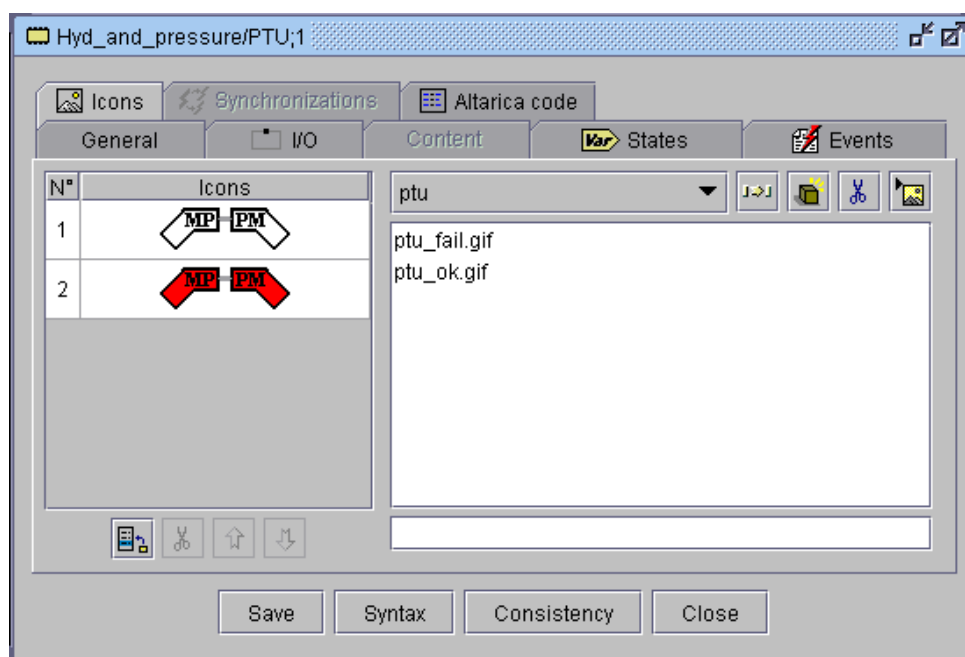


Figure 59 : Icons tab

- Click on , to remove the selected icon from the left side of the tab.

By using the two icons ( et ) we can move the selected icon upwards or downwards and thus modify the sequence order in the icon list.





WARNING : within the Altarica code, the icon displayed for a given state is identified by a number. Thus, modifying the icons order in the list induces indirectly a change of icon in the Altarica code.

Remark : The selection of the icon graphical representation or its number in the left part of the tab, automatically selects in the right part of the tab the name of the icon in the corresponding directory.

Note : When the component doesn't change its graphical representation, do not create icons in the **icons** tab : select the icon only in the **General** tab.

Remark : It is **recommended** that all the icons of the same component have the same size during the graphical simulation of the state changes.

Four icons are available at the top right part of the tab:

- Update: 
- New library: 
- Remove icons: 
- Import icons: 


Update

Proceed as follows:

- ♦ Click on the  icon.
- ♦ The library in the right part of the screen is updated with the new added icons.


Create new icons library

Proceed as follows:

- Click on the  icon.
- Type the name of the new library, e.g. "library_xxx".
- On the keyboard, press the **Enter** key.
- Check that "library_xxx" has been created in the Icons directory

Delete icons

Proceed as follows:

- Select from the library the icon to delete,
- Click on the  icon to delete the selected icon.

Import icons

Proceed as follows :

- Click on the  icon.
- In the Select window (Figure 60), click on the Look in selector and move down in the architecture up to the directory containing the icons to be imported.

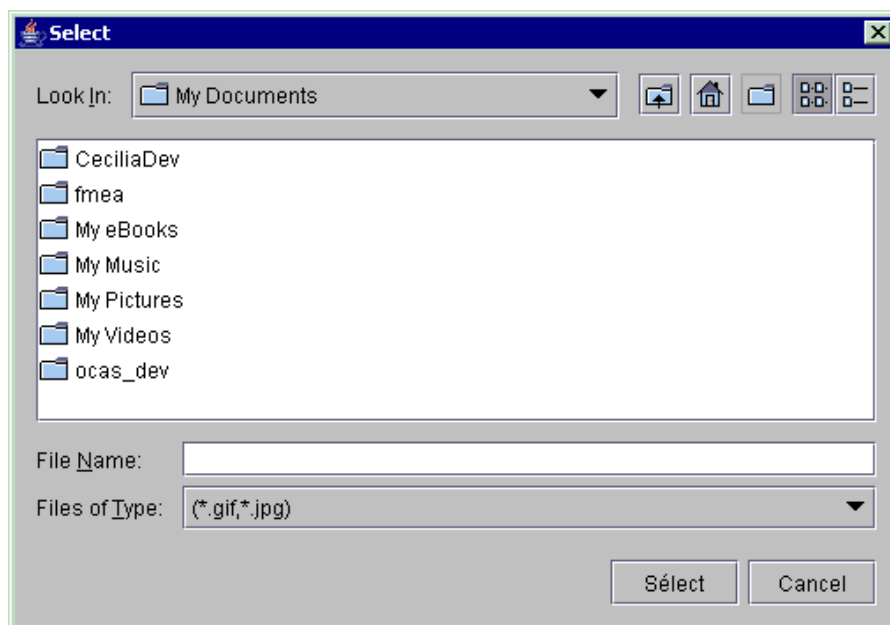


Figure 60 : Select window

- ♦ Use the following icons located at the top right: *Up one level*, *Home*.

Remark: The *Create new folder*, *List* and *Details* icons are not used in this functionality.

- ♦ Select the file type (“*.gif” or “*.jpg”).
- ♦ Click on the **Select** button; the selected icons are imported in the new library (library_xxx) created previously.

Note: The item **Images ...** from the **Library** menu allows launching the icons manager (Figure 61) dedicated to the search and visualization of the icons stored within the icons libraries.

This icons manager allows visualization of the graphical representation of icons stored within the icons libraries, and also to carry out the following commands:

- Update :
- Create a new library :
- Delete icons :
- Import icons :



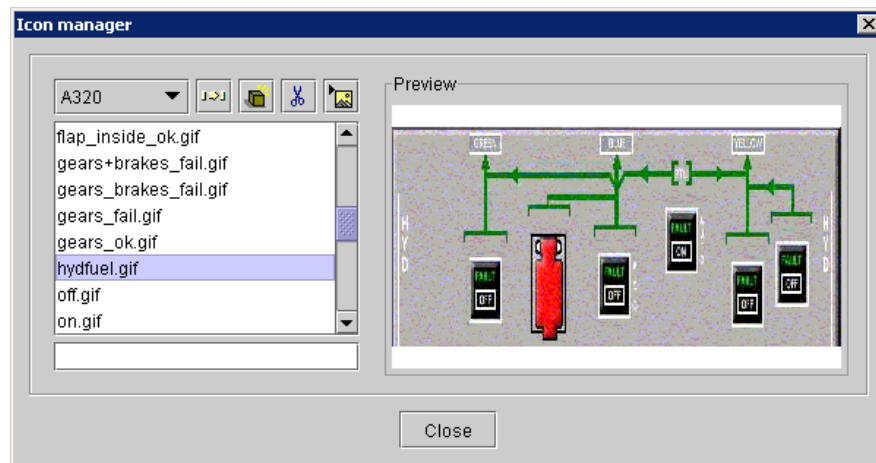



Figure 61 : Icons Manager

Altarica code tab

The Altarica **code** tab (Figure 62) allows writing of the Altarica code which conditions the component operation (Cf. Appendix 1: Syntax of Altarica code for component models).

The **Edition** area comprises from top to down:

- An icon bar (Copy, Cut, Paste, Backward zoom, Forward zoom, Display initial states).
- The initial states display area is non-editable and the  icon is used to display/mask this area.
- A data summary area (non-editable) concerning the current component (node name, icons used, input and output variables as well as their type (boolean, ...), the state variables as well as their type, the events, icon variable [1,3]).
- The Altarica code typing area (for the assertions (**assert**) and transitions (**trans**); the typing order for these assertions or transitions is indifferent).

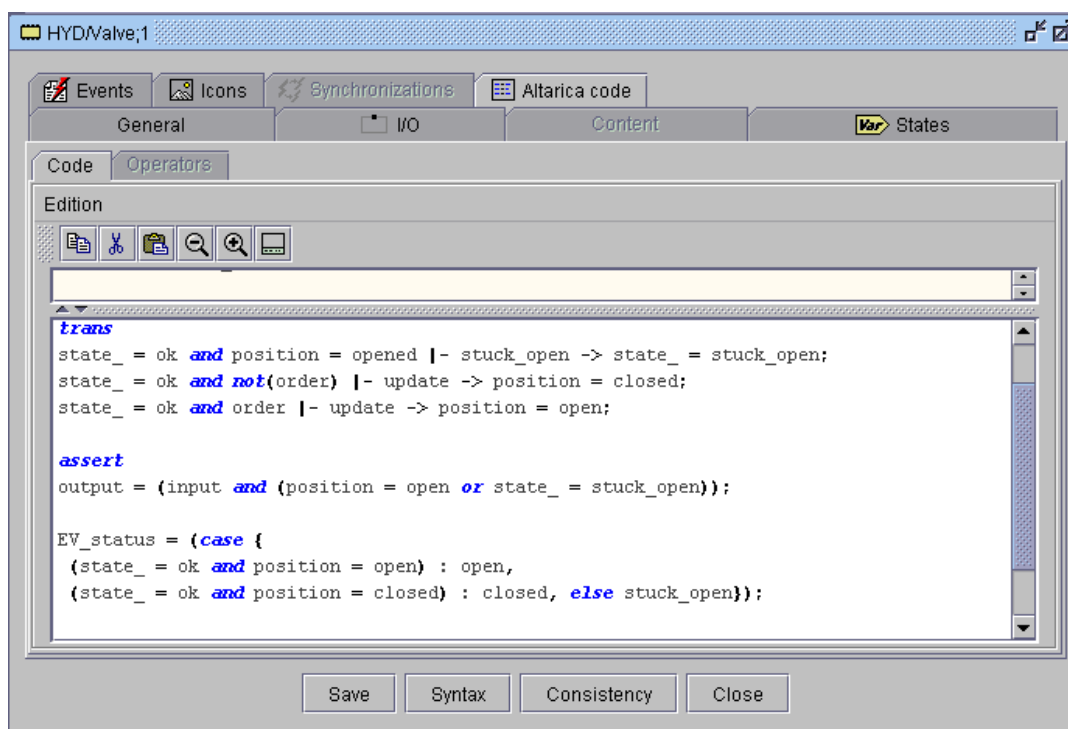


Figure 62 : Altarica code tab

- The short cut **Ctrl+<SPACE>** gives access to the variables selector for the search and selection of the state variables and the flow variables.

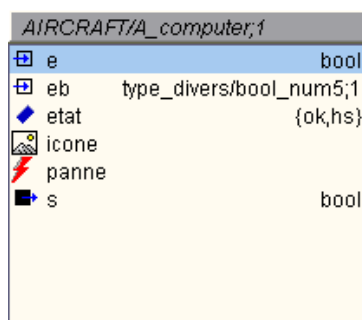
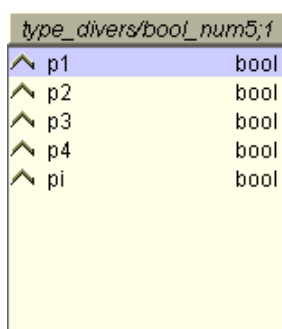


Figure 63 : Variables selector

In order to select a variable, proceed as follows:

- Click in the filling zone of the Altarica code,
- Use the shortcut **Ctrl+<SPACE>** to activate the variables selector and obtain the list of component variables,
- Double-click on the desired variable, the name **<Variable_Name>** will then be inserted in the text editor at the current position of the cursor,
- For a flow variable with a Record type, the selection of the flow parameters is carried out in the following way :
- Type the character **"^"** (**Alt gr+<^>**) following the name of the variable **<I/O_Name>^**, (eg. eb^), to reveal the list of the associated flow parameters:



- Double-click on the desired parameter, the name of the parameter will then be inserted in the text editor at the current position of the cursor : **<I/O_Name>^<Flow_Name>** (ex. eb^p1).
- Click on this selection using the right button of the mouse ; a vertical window appears indicating the two possible values of "state_" : **ok** or **stuck_open**.

Note : The colour code for the Altarica code is the following:

- Keyword (for example, trans, assert, and, or, if, then et else...) : blue,
- Numerical values : purple,
- Operators name : green,
- Icons : red,
- Punctuation in bold.

Remark: If operators are used in Altarica code, Operators tab is accessible (Figure 64). It allows selecting operator family and operator version, if different operators have the same name. If there is no ambiguity, the only family containing the operator is selected with the last version.

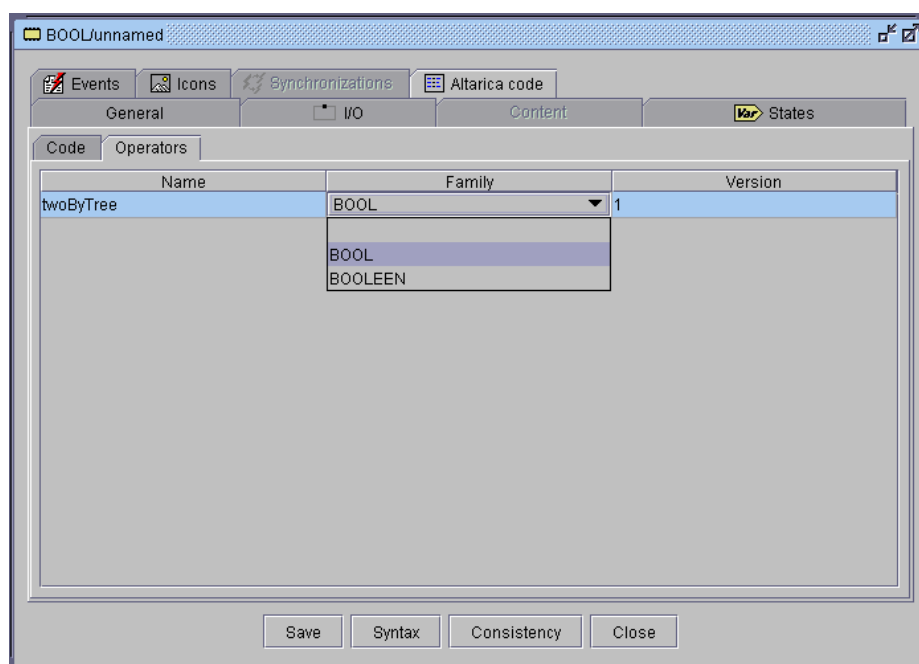


Figure 64 : Altarica Code - Operator Tab

Finally click on the button **Syntax** (Figure 62), to do the syntax control of the written Altarica script. In the event of a syntactic error, an error window appears describing precisely the kind of error and indicating the line of the script.

The most common errors are :

- A typing error in the Altarica code induces an error in the spelling and therefore the correct name in the heading doesn't correspond.
- The declaration syntax for the transitions is not respected (operator = for the conditions, operator: = for the assignments).
- Assertions define affectation with variable from different type (enumerate with Boolean, ...)
- The keywords « **trans** » or « **assert** » were omitted.

The syntax of the expressions between brackets is not respected. It must be the following form:

(if expression1 **then** expression2 **else** expression3) ;

or :

```
(if expression1
  then expression2
  else (if expression3
        then expression4
        else expression5
      )
);
```


Since DAS V3, “**if then else**” can be replaced by a “**case**” statement which as the following syntax:

```
flow-var = ( case { bool-exp1 : expression1 ,
                    |
                    bool-expn      :      expressionk
                    ,
                    else expression-else } )
```


A good indentation of the code in the expressions **if then else** can generally avoid these problems. Each open bracket must correspond to a closed bracket.

We can now click on the button **Consistency** (Figure 62), in order to do the consistency control of the Altarica script.

- In the event of a consistency error, an error window appears describing precisely the kind of consistency error of the Altarica code.
- The consistency problems detected are :
 - An exit flow variable is not defined whatever the internal state.
 - There is an internal state and a combination of input flow variables in which an exit flow variable is not defined.
 - There is an internal state and a combination of input flow variables in which an exit flow variable has 2 different evaluations.

Edit a component model

In order to edit a component model proceed as follows:

- Click on the **Components** tab in the left part of the screen to access components family list,
- Select the chosen model (click left),
- Click  to access the tree structure versions of the selected model (Figure 65),

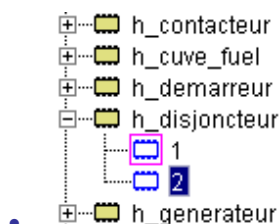


Figure 65 : Tree structure versions

- The edition of the version is carried out in 2 ways :
- Click twice (left click) on the number of the selected version
- Select a model and double click on it with the mouse or use the **Library – Edit model** command or use the **Edit model** command in the contextual menu displayed by clicking with the mouse right button on the selected model.
- A *Reading model* bar graph (Figure 66) is displayed.

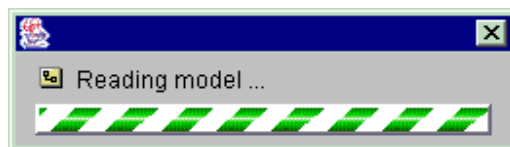

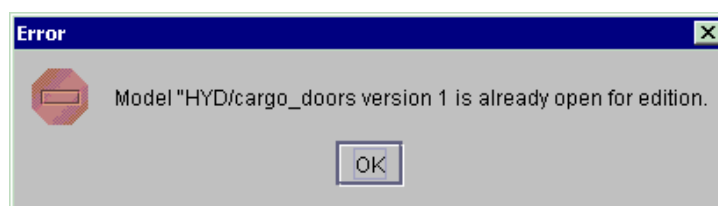


Figure 66 : Reading model bar graph

The model edition locks the possibility of launching again the edition from the tree structure versions (or from the System architecture). In multi user mode, this mechanism allows management of the concurrent accesses. In the same session, a locked model can be open in consultation only by another user.

The symbol  2 in the tree structure versions shows that the component model is in the course of edition. Any attempt of edition will cause the display of the following message:



Remark : In order to restore the possibility of edition (for example following a problem system), the locked attribute can be withdrawn manually via the properties editor of a model (Figure 67). It can be obtained by the **Properties** command of the contextual menu (Figure 40) or from the general menu **File**.

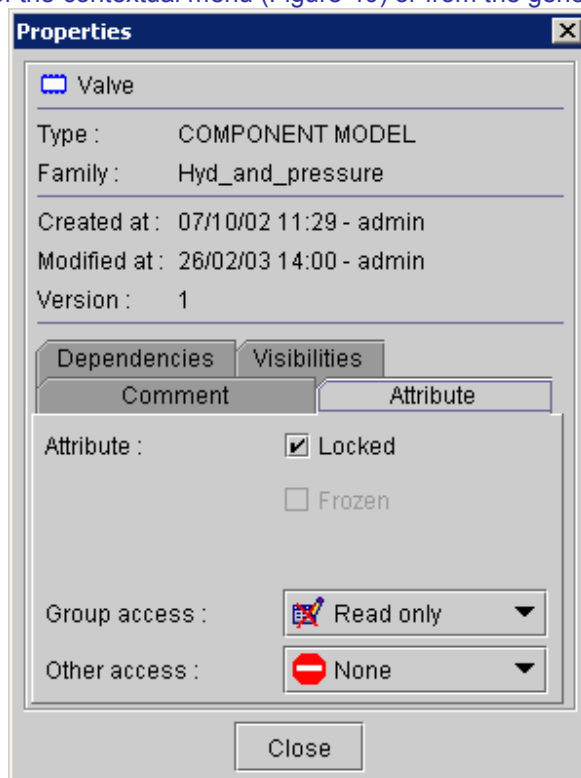


Figure 67 : Properties edition – locked attribute

Rename a component model

CAUTION: This operation is **very risky** and must be performed with extreme care, because it entails deletion of the component and its associated links in all equipment and architectures which use it.

Open all equipment and architectures containing the component **before** renaming it. Save all equipment and architectures **after** renaming.

If a component is renamed when the architecture using it is closed, the component becomes an unknown component for this architecture.

Cf. § 9.5.3 for the *Removed links* and § 9.5.4 for the *Renamed components*.

- ♦ In the **Components** tab, click on the component whose name is to be modified.
- ♦ In the contextual menu (Figure 40), click on the **Rename model** command; the name of the component is available and the cursor flashes.
- ♦ Type the new component name, and then validate with the Enter key.

N.B: If an existing name is typed, an *Error* message indicates that the component already exists. Click on the **OK** button and type a new name, then validate with the **Enter** key.

Duplicate a component model

In order to duplicate a component model, proceed as follows:

- ♦ Click on the component tab in the left part of the screen to access to the component family list.
- ♦ Select the model *Component_C* with the mouse left click,
- ♦ Access to the contextual menu (Figure 30) by clicking on the component name using the mouse right click,
- ♦ Select the command **Model ... → Duplicate**
- ♦ A bar graph (Figure 68) indicates that the component is being duplicated; *Component_C* is copied with the *Component_C_copy* name in the relevant family.

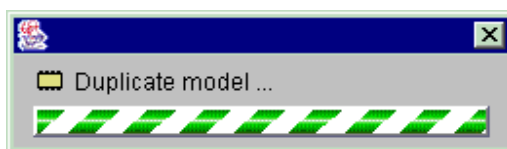


Figure 68 : Duplicate model bar graph

- Type the desired component name.

Caution : If the model has several versions, the duplication is carried out only in the last version. To duplicate another version of the component, it is necessary to select the version in the tree structure version model (Figure 65) before activating the command **Model ... → Duplicate**.

Remark : To move a component model into another model, proceed as follows:

- Copy the component : command **Model ... → Copy**,
- Paste the component in the family of destination : command **Model ... → Paste**,
- Restore the original name of the component (remove the extension « _copy ») in the family of destination.
- Validate the procedure with the Enter key.

Caution : If the model has several versions, the copy is carried out only in the last version. To copy another version of the component, it is necessary to select the version in the tree structure version model (Figure 65) before activating the command **Model ... → Copy**.

Create a new version of a component model

In order to create a new version of a component model, proceed as follows:

- Click on the **Components** tab in the left part of the screen to access the component family list,
- Select the model *Component_C* using a mouse left click,
- Click on the component name with the mouse right click to access the Pop-up menu (Figure 40).
- Select the **Model ... → New version** command.
- A dialog box appears (Figure 69) asking for the reference version from which the new version will be created (the possible versions are already listed in the combo box).

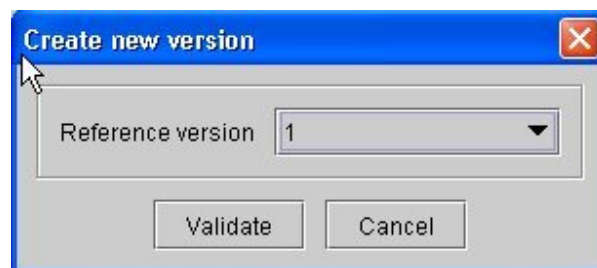


Figure 69 : Selection of the reference version

- Click on "Validate"
- A bar graph (Figure 70) indicates that the new version of the *Component_C* is being created.

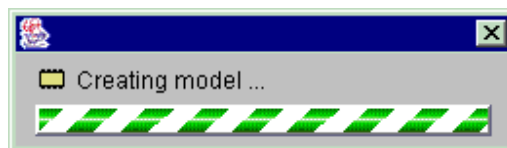



Figure 70 : Creation of a model new version


When the new component version has been created, it is inserted in the tree structure versions (Figure 65).

Freeze a component model

CAUTION: This operation is very risky and must be performed with extreme care, because it doesn't allow later modifications on the frozen component model and its associated types.

In order to freeze a component model, proceed as follows:

- Click on the **Components** tab in the left part of the screen to access the component family list,
- Select the component model (left click on the mouse),
- Click  to access the versions tree structure of the selected model,
- Select the version number (left click on the mouse),
- Access the Popup menu (Figure 40) by clicking on the mouse right click,
- Select the **Model ... → Freeze** command.

Remark: The frozen versions are represented in the versions tree structure by the symbol: . This **Frozen** attribute can be also visualized via the model properties editor (Figure 71). It can be obtained by the **Properties** command of the contextual menu (Figure 40) or from the general menu **File**.

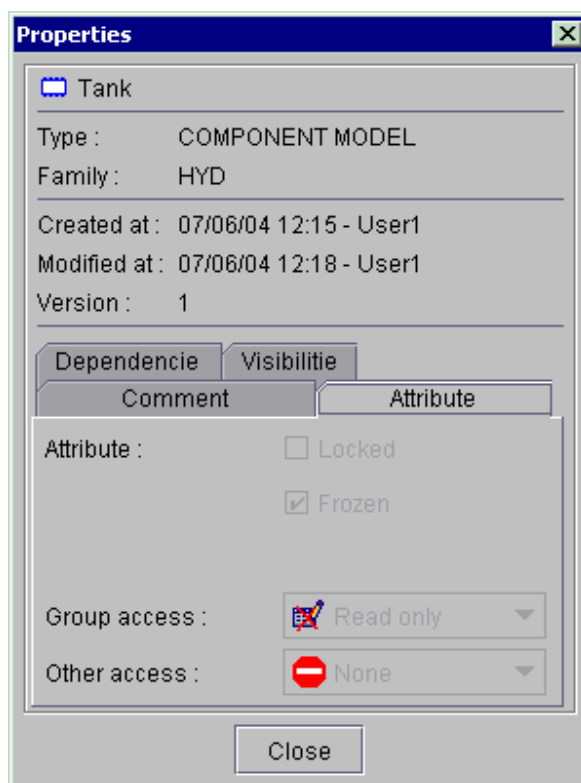



Figure 71 : Model properties

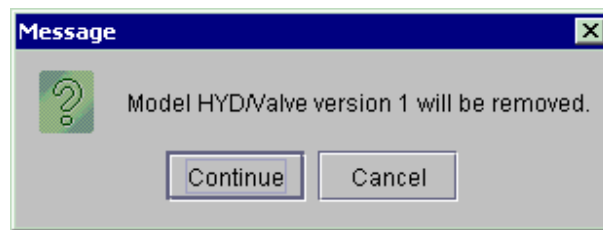
Delete a component model

CAUTION:

This operation is **very risky** and must be performed with extreme care, because it entails deletion of the component and its associated links in all equipment and architectures which use it.

In order to delete a component model, proceed as follows:

- ◆ Click on the **Components** tab in the left part of the screen, to access the components family list.
- ◆ Select the model (left click on the mouse),
- ◆ Click  to access the versions tree structure of the selected model (Figure 49)
- ◆ Select the version number (left click on the mouse)
- ◆ Access the contextual menu (Figure 40) by clicking on the mouse right click,
- ◆ Select the **Delete model** command; a *Message* window indicates that the component will be removed.



- Click on the **Continue** button to continue deletion, or on the **Cancel** button to abort deletion.

EDITION ON EQUIPMENT

Create equipments family

Create a new equipment family as follows:

- Click on the **Equipment** tab to gain access to the list of equipment families in the left part of the screen.
- Use the mouse right button to display the following contextual menu (Figure 72) or the **Library** menu:

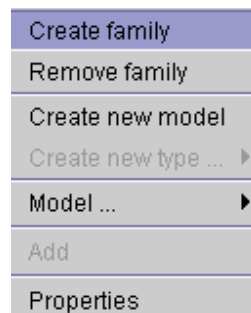


Figure 72 : Contextual menu – Create equipment family

- Click on the **Create family** command.
- The *Create equipment family* window is displayed (Figure 73)

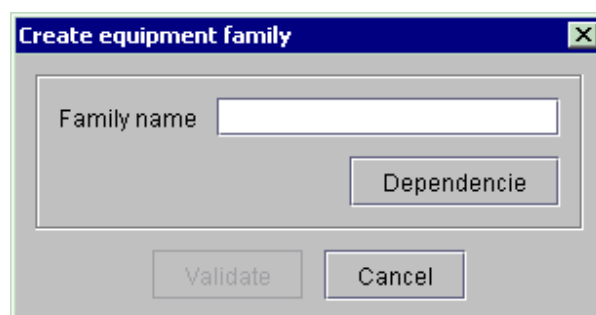
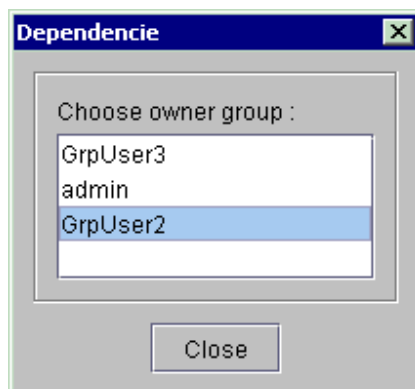


Figure 73 : Create equipment family

- In the *Family name* field, type the name.

- ♦ If the user creating family belongs to more than one group, he must click on Dependencies to select the group that will be the family owner. Validate by clicking on Close.



- ♦ Click on the **Validate** button, otherwise on the **Cancel** button; the new family appears in the **Equipment** tab.

Remove equipment family

Remove an equipment family as follows:

- ♦ Click on the **Equipment** tab to access to the list of equipment families in the left part of the screen.
- ♦ Select the family to be removed.
- ♦ Use the mouse right button to display the contextual menu (Figure 72) or use the **Library** menu.
- ♦ Click on the **Remove family** command.
- ♦ If the family is not empty, an *Error* message is displayed indicating that the family cannot be removed.
- ♦ Click on the **OK** button.
- ♦ Remove each equipment in the family and then remove the family.

Create a new equipment model

Create a new equipment model as follows:

- ♦ Click on the **Equipments** tab in the left part of the screen to gain access to the list of equipment families.
- ♦ Select an equipment family.
- ♦ Use the mouse right button to display the contextual menu (Figure 72) or use the **Library** menu.
- ♦ Click on the **Create new model** command; the *Creating model* bar graph (Figure 74) is displayed

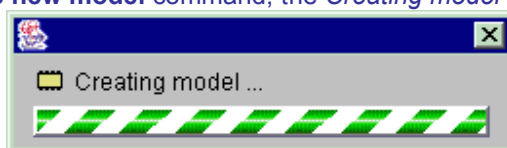


Figure 74 : Creating model bar graph

- The model editor (Figure 75) is displayed
- Fill the right information in the different tabs.

The Editor model window has the following tabs :


- General,
- I/O,
- Content,
- Icons,
- Synchronizations,
- Altarica code.

When the new model has been created, it is automatically inserted in the equipment manager (in alphabetical order).

Remark : Every new equipment model created is in version 1.0 by default.

General tab

The **General** tab (Figure 75) allows the following information to be typed in:

- ♦ **Name**: type the model name,
- ♦ **Note**: In order to save the model, the name is the only obligatory field.
- ♦ **Width**: ww (icon width default value),
- ♦ **Height**: hh (icon height default value),
- ♦ **Icon file**: the  button enables selection of the icon to be assigned to the model.

Remark: Icons are image files in “*.gif” or “*.jpg” format.

- ♦ Tick **Draw border** to display or not the border associated to the icon.
- ♦ Click on the **Resize** button to validate, if necessary, the icon larger dimensions.
- ♦ **Comment** : fill the comment describing the created component.

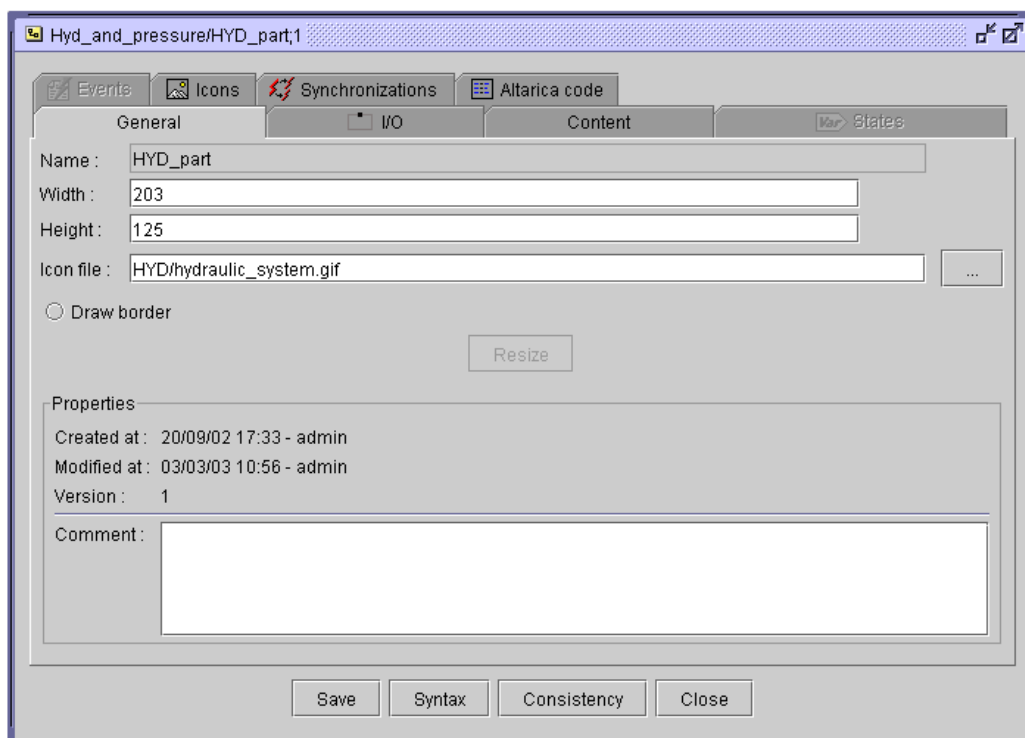


Figure 75 : Equipment - General tab

The equipment model editor tabs contain four common buttons:

- ♦ **Save**: allows storing of all typed information,
- ♦ **Syntax**: allows a syntax control on the Altarica code,
- ♦ **Consistency**: allows a consistency control on the overall components contained in an equipment.
- ♦ **Close**: allows closure of the model editor.

Two icons on the top right hand side allow window size modification:


- ♦ Reduce,
- ♦ Extend.

I/O tab

The **I/O** (Inputs/Outputs) tab (Figure 76) allows configuration of the input or output flows within an equipment.

- ♦ In the *Name* field, type the I/O name.
- ♦ Use the **Type** menu and choose the variable type:
 - bool**: boolean variable,
 - enum**: enumerated variable; values **must** be separated by commas without spaces,
e.g.: **null,low,high**,
 - bound**: variable whose value varies between a minimum value and a maximum value,
 - predefine**: predefined type in library,

- Cf. § 8.2.3.2 for more detail on the enumerate, bound and predefined variables.

- ♦ Click on the  icon or the **Enter** key; the I/O appears on the equipment border (“*bool*” is the default type).
- ♦ The line corresponding to the typed I/O is added in the I/O table with the following information (*Name, Type, Orientation, X, Y*); X and Y indicate the coordinates of the input or output port location on the icon with respect to an origin at the top left-hand side; the default orientation is “in”.

- ♦ Position the I/O on the model by clicking on the corresponding icon:





- ♦ top:  icon,

- ♦ down:  icon,

- ♦ left:  icon,

- ♦ right:  icon

N.B: An I/O can be positioned anywhere on the equipment border by means of the mouse.

- ♦ Repeat the same procedure for any additional I/O.
- ♦ If necessary, select the I/O and click on the  icon to remove the selected I/O.
- ♦ Three other icons are available:
- ♦ Edit type: use the  icon or double click with the mouse left button in the *Type* cell of the selected I/O line,
- ♦ Move up: ,
- ♦ Move down: ,

N.B: An input is identified by a white rectangle and an output by a black one.

Remark: If an I/O is selected, the added I/O (with the  icon) is positioned before the selected I/O. If no I/O is selected, the added I/O is positioned at the end of the list.

- ♦ To modify the type of an existing I/O, select the I/O ; choose the type and click on the **Assign** button; the type corresponding to the selected I/O is updated.
- ♦ To modify the orientation of an I/O, click with the mouse left button and then the right button on the *Orientation* cell; in the pull-down menu, assign the flow orientation (in or out).

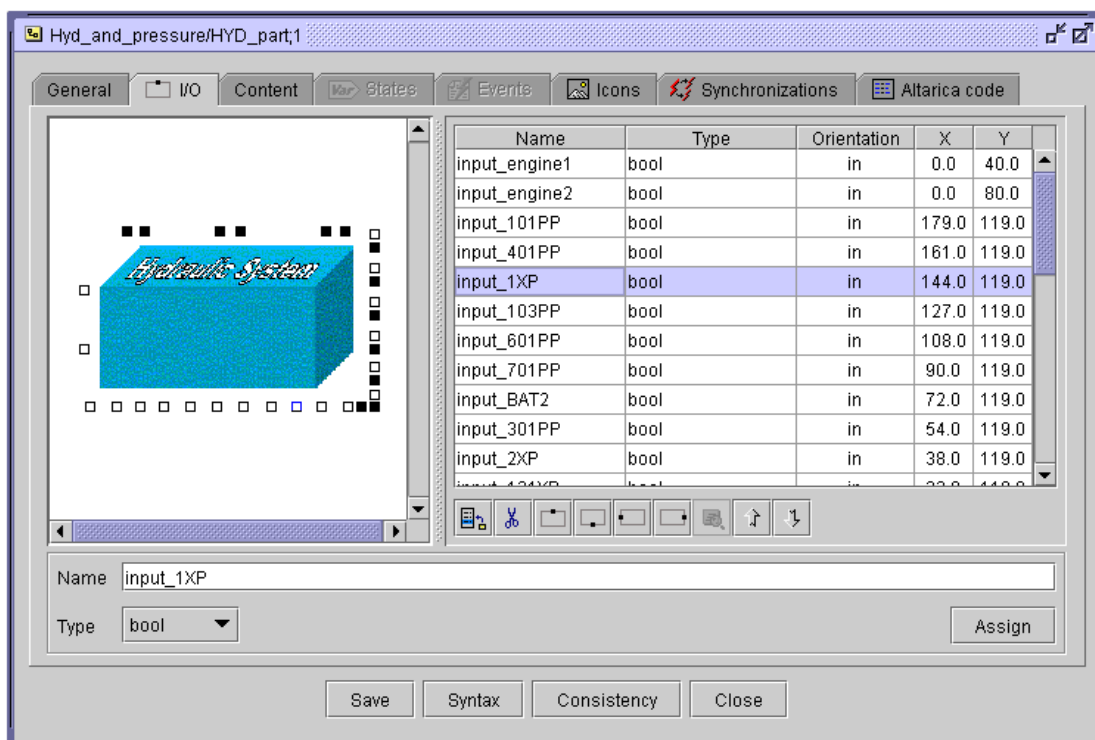


Figure 76 : Equipment - I/O tab

Content tab

N.B: When an equipment is created, it contains only the Input/Output nodes; the latter can be positioned anywhere inside the equipment.

The **Content** tab (Figure 77) indicates the equipment internal architecture; modelling an equipment is identical to modelling a system architecture.

Connections between the Inputs/Output nodes of the equipment and the I/O components it contains are identical to the connections between components.

Remark: when equipment is indexed in architecture and has:

1. A non empty content, it cannot be modified in the architecture,
2. An empty content, it can be modified in the architecture.

- The **Layers** button allows users to control the display of different graphical elements.
- The **Display** button allows users to control the display of different architecture type.
- None: no name is displayed,
- Node Labels : all the instance names are displayed above the components and equipments,
- Link labels: the type transported by the link is displayed at the center of the link for all links
- All labels: Display "Node labels" and "Link labels"

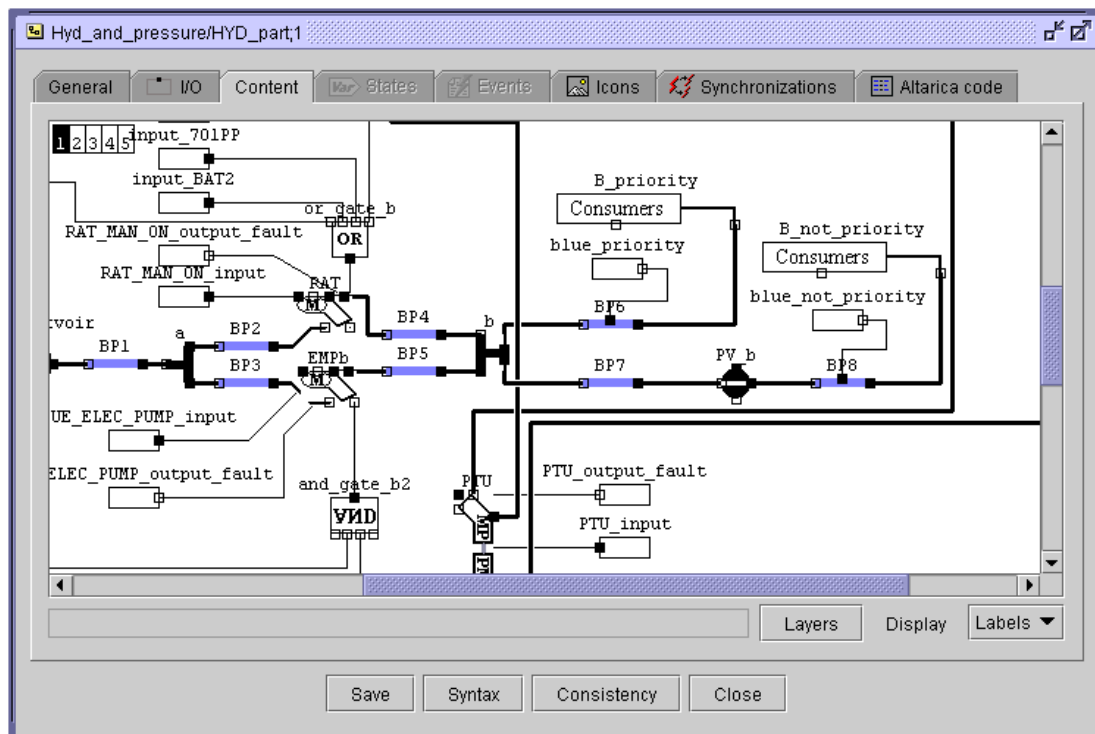


Figure 77 : Equipment – Content tab

- ◆ Since 3.2 version, system assertions are defined in Altarica **code** tab.

Synchronisation tab

The **Synchronization** tab (Figure 78) allows specifying synchronizations whose. The activation of these synchronizations during the simulation phases will allow synchronizing the occurrence of certain number of component events defined in the equipment architecture (**Content** tab).

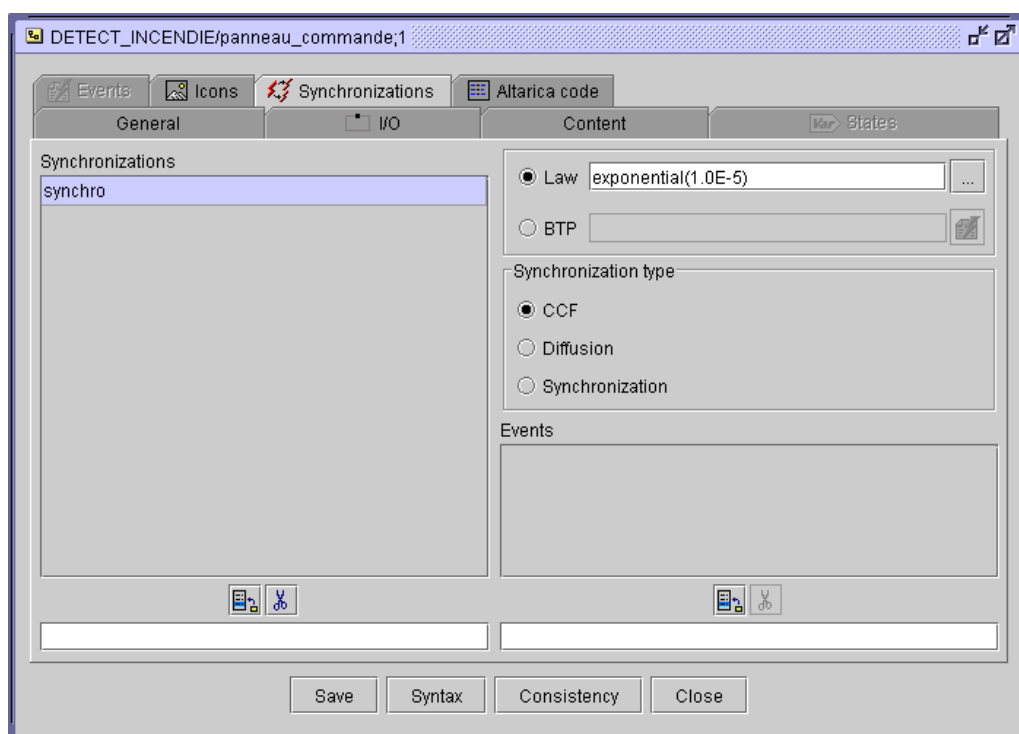



Figure 78 : Equipment Synchronization tab

In order to create synchronization, proceed as follows:

- Fill the synchronization name in the line editor of the **Synchronizations** area.
- Click on , to transfer the name in the synchronizations list defined for the equipment.
- Specify the synchronization law. There is two ways to do it:
 1. Select the Law radio button and fill the field in or alternatively, click on the right button to access to the law editor assistant window (See Figure 56). The law entered in the field must be In AltaRica format (cf. Annexe 2). The associated syntax is the following:

<Law_Name> (<Parameters_List>); parameters are separated with comma.

Currently the available probability laws are:

- **constant** (<probability>) ; example : constant(0.5) ;
- **exponential** (<lambda>) ; example : exponential(1e-6) ;
- **GLM** (<gamma> ,<lambda> ,<mu>) ; example : GLM(0.5,1e-6,1e-3) ;
- **Weibull** (<alpha>,<beta>) ; example : Weibull(0.5,1.5) ;
- **Periodic_test** (<lambda>,<tau>,<t0>) ; example : periodic_test(1e-3,10,0) ;
- **GLM_asymptotic** (<lambda>,<mu>) ; example : GLM_asymptotic(1e-6,1e-3) ;
- **CMT** (<lambda> ,<duration> ,<probability>) ; example : CMT (1e-3, 72, 0);
- **Dirac** (<delay>); example : Dirac(10.0);
- **dormant** (<lambda> ,<MTTR> ,<period>) ; example : dormant (1e-3, 10, 25) ;
- **periodic** (<Delay1>,<Delay2>) ; example : periodic(100.0,10.0);
- **unif** (<Delay1>,<Delay2>) ; example : unif(10.0,20.0);

2. Select the FRB radio button, then fill the field in or alternately click on the right button to access to the FRB (Failure Rate Bank) tree manager that shows the existing event models (See Figure 58). The path entered in the field must be an event model from the FRB (Failure rate Bank) defined in FaultTreeAnalysisSystem (See User's Guide - FaultTreeAnalysisSystem Module: Failure Rate Bank).

- Chose the synchronisation type : CCF (Common Cause Failure), diffusion, synchronisation (« Rendez-Vous »)
- Fill the events list having to be synchronized, using the line editor associated to the Event zone. In order to facilitate the filling of this assertion, the short cut **Ctrl+<SPACE>** gives access the hierarchical navigator (Figure 79) for the search and selection of component events.

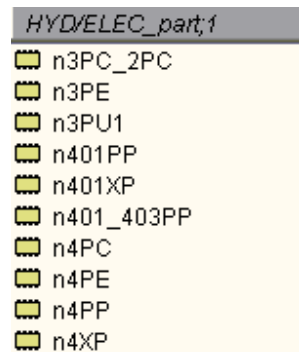


Figure 79 : Synchronization – Component selection

In order to select an event, proceed as follows:

- Click in the line editor,
- Activate the variables selector using the short cut **Ctrl+<SPACE>**,
- Browse to visualize the component or equipment required,
- Double-click on this component, its name **<Component_Name>** will then be inserted in the line editor,
- Type **<.>** (point) following the name **<Component_Name >**., to display the constituents list for an equipment, and to pursue the hierarchical search or the events list for a component (Figure 80),
- Double-click on the selected event; it will then be inserted in the events list having to be synchronized.

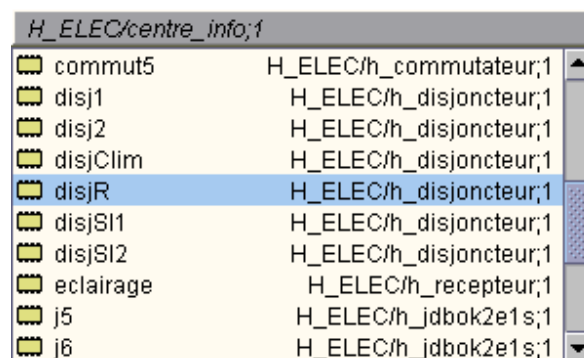


Figure 80 : Synchronization – Event selection

To complete the events list (Figure 81) will be synchronized, repeat the same procedure.

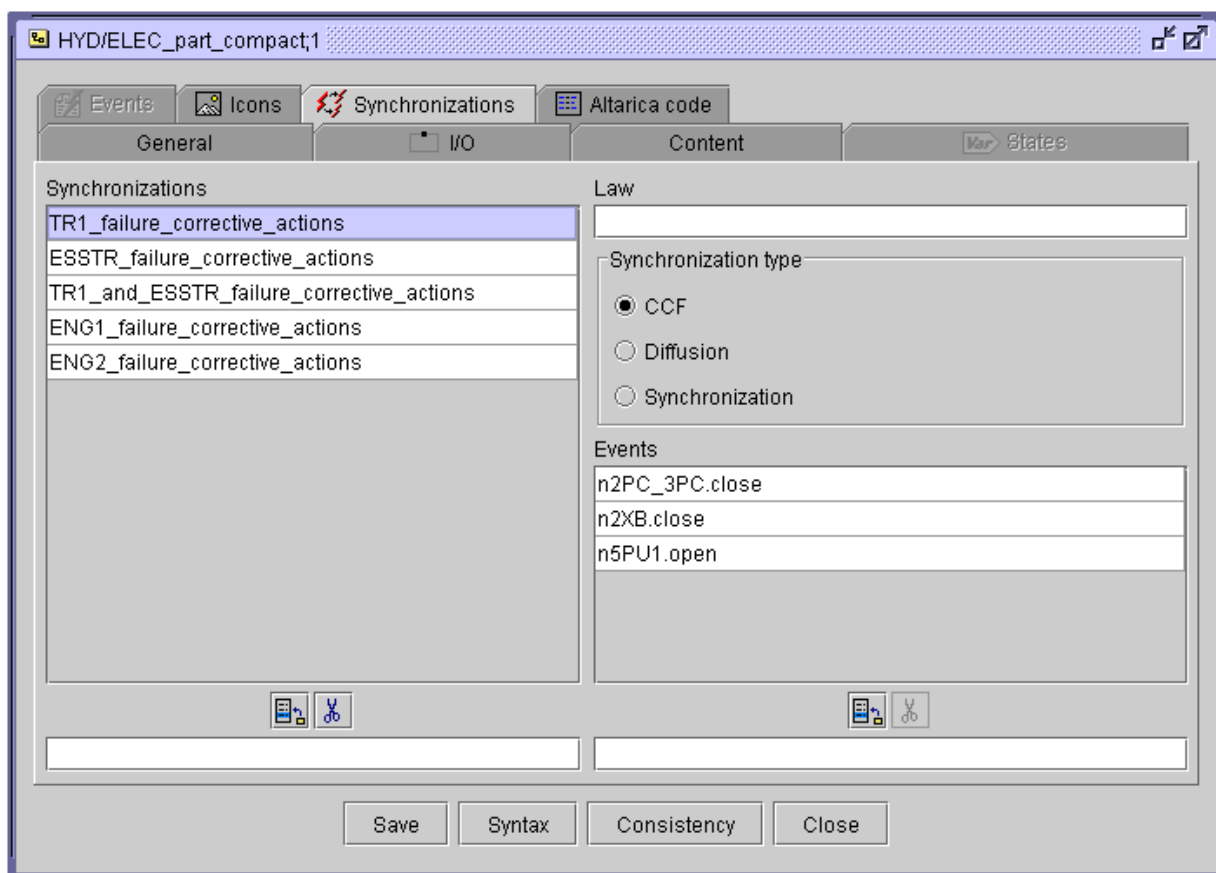



Figure 81 : Synchronization – Events to be synchronized

Remark: during the system simulation phases, synchronizations will be added at the list of events.

Code Altarica tab

The **Code** Altarica tab (Figure 82) allows filling of the Altarica code, specifying the change of the equipment icon (cf. Annexe 1 : Altarica code syntax).

The **Edition** area includes from top to bottom:

- An icon bar (copy, cut, paste, zoom in, zoom out, display of initial states).
- The initial states display area can not be edited and the icon  is used for display/mask this zone,
- A data summary area, concerning the equipment currently in edition node, input and output flow variables and the icon variable [1,3],
- The script of the Altarica code (Figure 82) specifies how the change of icon for the equipment is done. The syntax of the Altarica code associated is identical to the component assertions (see Annexe 1: Syntax of Altarica code). The assertion filled must specify the assignment of output flow variable "icon" (key word Altarica) of the equipment according to the equipment flow variables or the state variables associated to components within the equipment architecture.

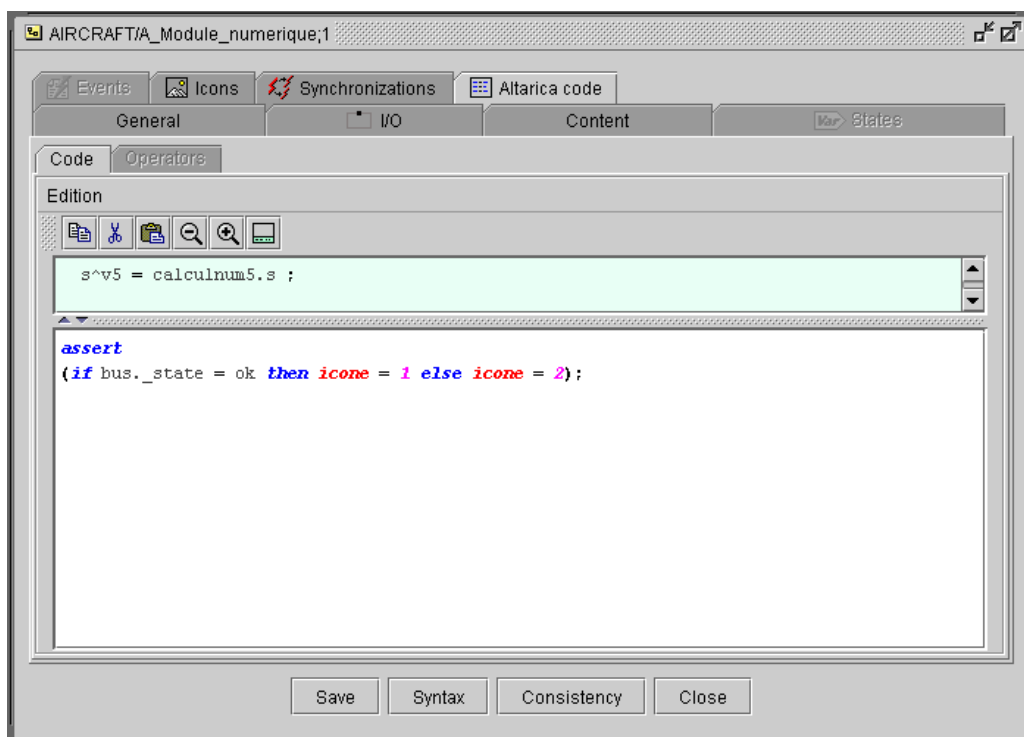


Figure 82 : Equipment - Code Altarica code

- Since 3.2 version, system assertions must be written in this area. A system assertion enables modelling of a link, without any graphical representation between two components by an Altarica code. This link is independent of the equipment hierarchy (a system assertion can link two components which are located in two different hierarchical levels). This system assertion concerns only the flow variables of components located within the equipment hierarchy whatever its hierarchical level

The syntax for the system assertions is:

1. Simple affectation between "in flow" and "out flow" with same type.

```
<component_i>.<In_Name> = < component j>.<OutName>;
```

2. Simple affectation between "in flow" and a specific value from its definition domain.

```
<component_i>.<In_Name> = true ;
```

```
<component_i>.<In_Name> = faible ;
```

3. Affectation using operators:

```
<component_i>.<In_Name> = (<component j>.<OutName> or  
                           <component k>.<OutName>);
```

```
<component_i>.<In_Name> = my_operator (<component j>.< OutName > ,  
                                       < component_k>.<OutName S>);
```

- The written of the system assertions of equipment is simplified since the copy of a port name is made possible. In the system architecture edited in the Content tab, after having pointed the mouse on the port of a component or equipment, we can reach the "Copy port name" function via the contextual menu. The name of this port can be then pasted in the equipment assertions editor.
- In order to facilitate the assertion filling, the short cut **Ctrl+<SPACE>** gives access to the hierarchical menu (Figure 83) for the search and selection of equipment flow variables, state variables, or equipment component flow.

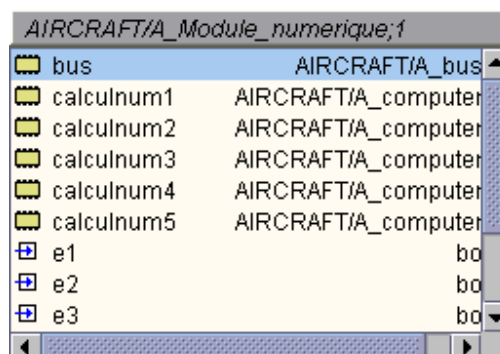


Figure 83 : Altarica Code – Variables selector

- In order to select a variable proceed as follows:
Click in the assertion filling zone,
Activate the variables selector using the short cut **Ctrl+<SPACE>**,
Browse to visualize the component or equipment required,
Click twice on this component, his name **<Component_Name>** will then be inserted in the line editor,
Type a . (point) following the name **<Component_Name>**., to display the constituents list associated and the components of the same hierarchical level. Double-click on the requested variable, it will then be inserted in the text editor at the cursor current position of .
- When the selection is completed, click on the button **Syntax** (Figure 82) in order to carry out the syntax control of the Altarica script. In case of syntax errors, a window error appears describing precisely the type of error and showing the script line in question.

Remark: The use of operators to define Altarica Code is allowed, Operators tab is then accessible to help user to precise operator if it has ambiguous name (same name in different family). If operator is defined in only one family, it's this family in its last version which is selected.

The most common syntax errors are:


- A typing error in the Altarica code, creates a false identifier that doesn't correspond to the text in the header dedicated to the display of variables specified in the tabs.
- Assertions define affections on variables of different types (two types enumerated differently, one type enumerated with Boolean ...).

Click now on the button **Consistency** (Figure 82), in order to carry out the consistency control of Altarica script defined for the assignment of the **icon** flow variable.

- In case of detection of consistency errors, an error window appears describing in a precise way the consistency problems detected on the Altarica code.

Edit equipment model

In order to edit an equipment model, proceed as follows:

- Click on the **Equipment** tab in the left part of the screen to access the equipment family list,
- Select the selected model (left click on the mouse),
- Click  to access the versions tree structure of the selected model (Figure 84),

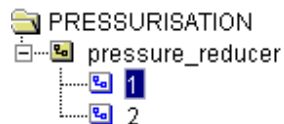
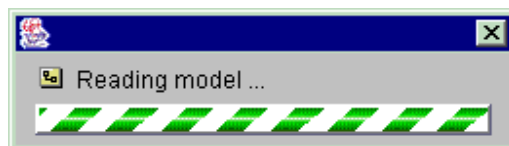


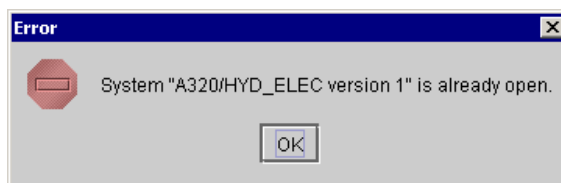
Figure 84 : Tree structure versions

- The edition of the model version is carried out in 2 ways :
- Click twice (left click) on the number of the selected version
- Select the version number (left click) and activate the **Model ... → Edit** command from the **Library** menu or the contextual menu (Figure 72) obtained by a mouse right click on the selected model.
- While reading model a bar graph indicates opening of the equipment.



The model edition locks the possibility of launching again the model edition from the versions tree structure (or the system architecture). In multi-user mode this locking mechanism allows managing the concurrent access.

The symbol  **1** in the tree structure version indicates that the model equipment is currently in edition. Any attempt of edition will cause the display of the following message :



Remark : In order to restore the possibility of edition (following a system problem for example), this locked attribute can be withdrawn manually via the model properties editor (Figure 85) obtained with the **Properties** command from the contextual menu (Figure 72) or the **File** general menu.

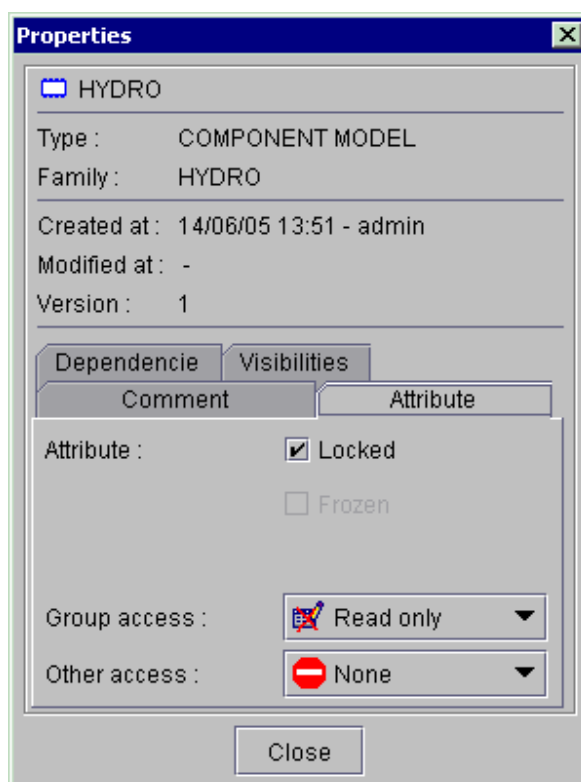


Figure 85 : Model properties - Locked attribute

Rename equipment model

CAUTION: This operation is **very risky** and must be performed with extreme care, because it entails deletion of the equipment and its associated links in all equipment and architectures which use it.

- ♦ Open all equipment and architectures containing the equipment **before** renaming it.
If equipment is renamed when the architecture using it is closed, the equipment becomes an unknown model for this architecture.
- ♦ Save all equipment and architectures **after** renaming.

Cf. § 9.5.3 for the *Removed links*.

- ♦ In the **Equipments** tab, click on the equipment whose name is to be modified.
- ♦ In the contextual menu (Figure 72), click on the **Rename model** command; the name of the equipment is available and the cursor flashes.
- ♦ Type the new equipment name and validate with the **Enter** key.

N.B: If an existing name is typed, an *Error* message indicates that the equipment already exists. Click on the **OK** button and type a new name.

Duplicate equipment model

In order to duplicate equipment model, proceed as follows:

- ♦ Click on the equipment name with the mouse right button to gain access to the contextual menu.
- ♦ Select the model **Equipment_E** (left click on the mouse),
- ♦ Click on the equipment name with the mouse right click to access the contextual menu (Figure 56)
- ♦ Select the **Model ... → Duplicate** command,
- ♦ A *Duplicate model* bar graph (Figure 86) indicates that the equipment is being duplicated; **Equipment_E** is copied with the **Equipment_E_copy** name in the relevant family



Figure 86 : Duplicate model bar graph

- Type the desired equipment name.

Caution : If the model has various versions, the duplication can be carried out only on the last version. In order to duplicate another version of the equipment, it must be selected in the model tree structure versions (Figure 70) before activating the **Model ... → Duplicate** command.

Remark : In order to move the equipment model into another family, proceed as follows:

- copy the equipment : **Model ... → Copy** command,
- paste the equipment in the family of destination : **Model ... → Paste** command,

- restore the equipment original name (by removing the extension « _copy ») in the family of destination.
- validate the procedure by clicking on Entry.

Caution : If the model has several versions, the copy is carried out only in the last version. To copy another version of the equipment, it is necessary to select the version in the tree structure version model (Figure 70) before activating the command **Model ... → Copy**.

Create a new version of equipment model

In order to create a new version of equipment model, proceed as follows:

- Click on the **Equipment** tab in the left part of the screen to access the equipments family list,
- Select the *Equipment_E* model (left click on the mouse),
- Click on the equipment model with the mouse right click to access the contextual menu (Figure 72).
- Select the **Model ... → New version** command.
- A dialog box appears (Figure 69) asking for the reference version from which the new version will be created (the possible versions are already listed in the combo box).
- Click on "Validate"
- A bar graph (Figure 87) indicates that the new *Equipment_E* version is in course of creation

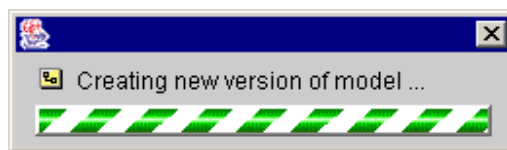


Figure 87 : New model version creation


When the new equipment version has been created, it is inserted in the versions tree structure (Figure 84).

Freeze an equipment version model

CAUTION: This operation is **very risky** and must be performed with extreme care, because it doesn't allow later modifications on the stilled component model and its associated types. In a recursive way the components model versions (or equipment) used in the equipment definition are also frozen.

In order to Freeze a equipment proceed as follows:

- Click on the **Equipments** tab in the left part of the screen to access the equipment family list,
- Click twice on the model *Equipment_E* to access the versions tree structure,
- Select the version number (left click on the mouse),
- Access the contextual menu (Figure 72) by clicking on the version number using the mouse right click,
- Select the **Model ... → Freeze** command.

Remark: The frozen versions are represented in the versions tree structure by the symbol:  1. This **Frozen** attribute can be also visualized via the model properties editor (Figure 88). It can be obtained by the **Properties** command of the contextual menu (Figure 40) or from the general menu **File**.

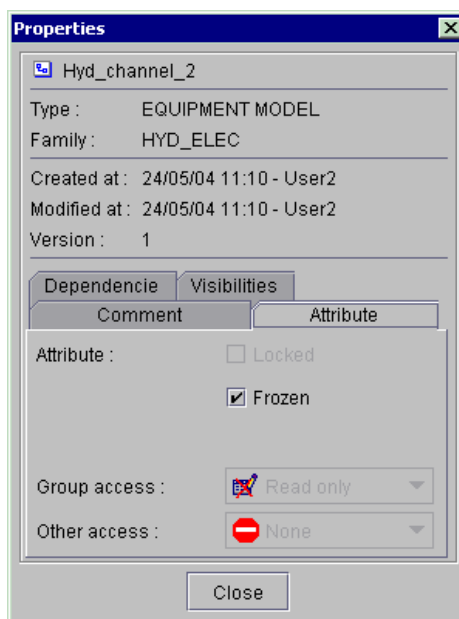



Figure 88 : Model properties – Frozen attribute

Delete equipment model

CAUTION: This operation is **very risky** and must be performed with extreme care, because it entails deletion of the equipment and its associated links in all equipment and architectures which use it.

- To delete an equipment model, proceed as follows:
- - ♦ Click on the **Equipment** tab in the left part of the screen to access the equipments family list.
 - ♦ Select the model **Equipment_E** (left click on the mouse)
 - ♦ Click  to access tree structure of model versions (Figure 70),
 - ♦ Select the version number (left click on the mouse),
- Click on the equipment name with the right click of the mouse to access the contextual menu (Figure 56).
- Select the **Model ... → Remove** command; a *Message* window (Figure 89) is then displayed indicating that the equipment is going to be destroyed;

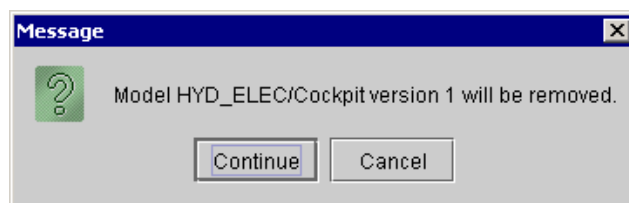


Figure 89 : Equipment destruction message

- Click on the **Continue** button to remove this version or on the **Cancel** to abort destruction.

EDITION ON OPERATOR

Create operators family

In order to create a new operators family, proceed as follows:

- ♦ Click on the **Operators** tab on the left part of the screen to gain access to the list of operators.
- ♦ Display the contextual menu with the mouse right button (Figure 72) or use the **Library** menu.
- ♦ Select the **Create family** command; a *Create operators family* window (Figure 90) is displayed
- ♦ In the *Family name* field, type the name.

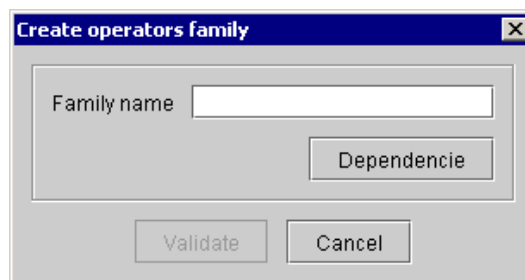


Figure 90 : Create operators family

- In the **Family name** field, type the name of the new family.
- If the family's owner belongs to several workgroups, he must specify the group to which must be reattached the family created.
- Click on the button **Dependencies** and select in the groups list which belongs the user, the owner group on which must belong the family created. Click on the **Close** button to validate the selection.

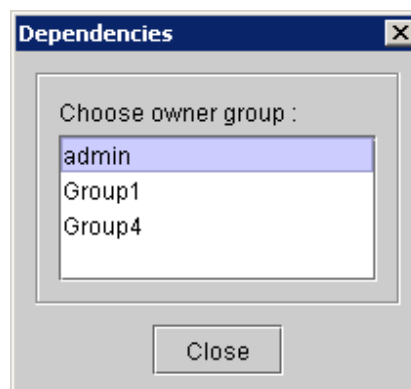


Figure 91 : Selection of the owner group

- ♦ Click on the **Validate** button to create the family, or on the **Cancel** button to abort creation.

Remove operators family

Remove an operator's family as follows:

- ◆ Click on the **Operators** tab on the left part of the screen to gain access to the list of operators.
- ◆ Click on the equipment name with the mouse right button to gain access to the contextual menu (Figure 72) or use the **Library** menu.
- ◆ Select the **Remove family** command; if the operator's family is not empty, the *Error* message (Figure 92) is displayed indicating that the family cannot be removed.

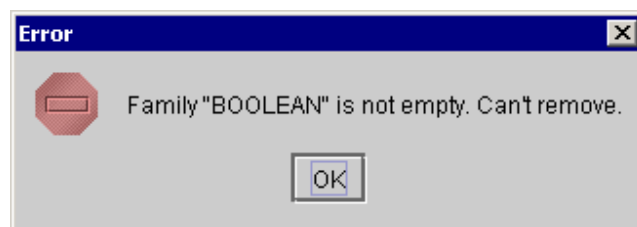


Figure 92 : Error window – Family not empty

- ◆ Click on the **OK** button.
- ◆ Remove all the operators in the family, and then remove the family.

Create new operator model

Definition

The operators allow simplifying the writing of the assertions for the components within the Altarica code. The description of the transfer function is very close to the description of a component model, with the close difference that this function doesn't describe behaviour. An operator is thus a component model which has an output flow and a certain number of input flows, the assertions enable to define the output flow according to input flows.

The flow type variables associated with the definition of an operator are unspecified and can be structured. In this case, the structured flow parameters should not be reversed. Moreover, equality between structured flows, takes systematically into account the crossings defined for the structured field.

Remark: the operators can be used only within the components (or equipments) models assertion. The use of operators in the transitions definition (Altarica code) is currently forbidden.

In order to create a new operator model, proceed as follows:

- Use the mouse right-click to display the contextual menu (Figure 72).
- Select the **Create new model** command, the following bar graph is displayed (Figure 93) :



Figure 93 : Operator creation bar graph


- The window *Operator editor* (Figure 94) is displayed; it is composed of 3 tabs :
 - * **General**,
 - * **Properties**,
 - * **Altatica code**.

General tab

In the **General** tab (Figure 94) carry out the following operations:

- ♦ In the **Name** field (at the top left in the tab), type the operator name, e.g. o2by3.

N.B: In the **Operands** area, the first line is dedicated to the operator result and the first operand has always the same name as the operator ; this first line cannot be removed.

- ♦ In the **Name** field at the bottom left of the **Operands** area, type the operand name.
- ♦ Use the **Type** menu and define the operand type.
- ♦ Click on the  icon or use the **Enter** key.
- ♦ To modify the type of an existing operand, select it and click on the **Assign** button.

Remark: The type of an operator or of one of its operands may be a record-type if the parameters of this type are neither crossed over nor inverted.

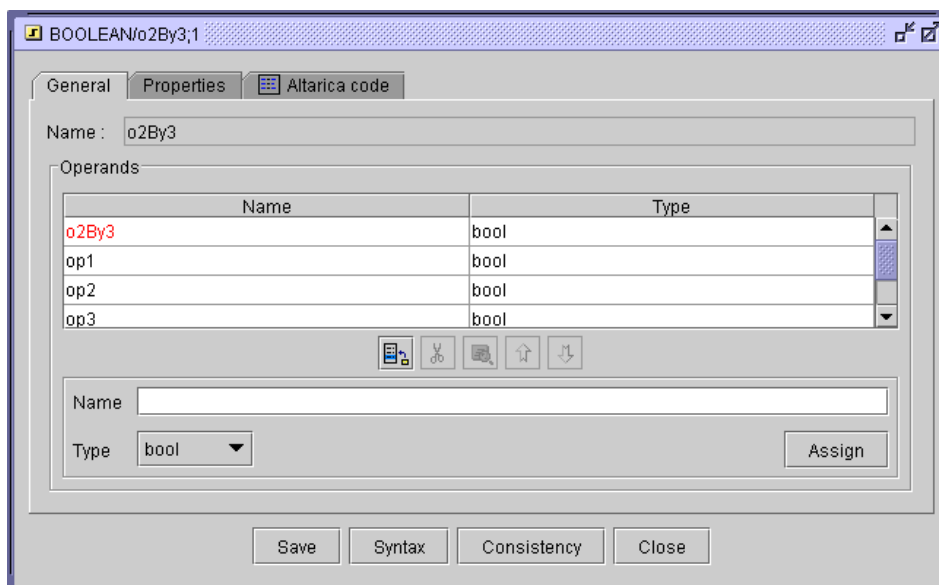


Figure 94 : Operator editor - General tab

Properties tab

- The Properties tab (Figure 95) allows visualization of the model properties:
 - Creation date,
 - Last modification date,
 - Version.
- The **Comment** field allows the filling of a comment describing the operator function.



Figure 95 : Operator – Properties tab

Altarcia code tab

The Altarcia **code** tab contains:

- ♦ An icon bar (Copy, Cut, Paste, Backward zoom, Forward zoom, Display initial states).
- ♦ A display zone (non editable) for the operands and their types (Figure 96). The Altarcia code displayed in this zone has been automatically generated from the information filled in the **General** tab.

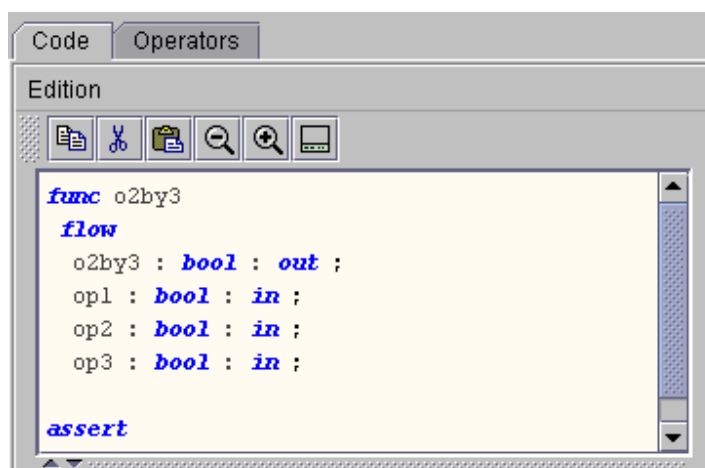


Figure 96 : Operator – Altarica code – Operands

This generation is done as following:

- Creation of an output flow variable carrying the operator name,
- Creation of input flow variable for every operator operand,
- This end of definition is indicated by the key word **assert**.
 - ♦ A zone for typing the operator Altarica code (Figure 97). The Altarica code syntax of an operator is identical to the component assertions (see Annexe 1: Syntax of Altarica code).

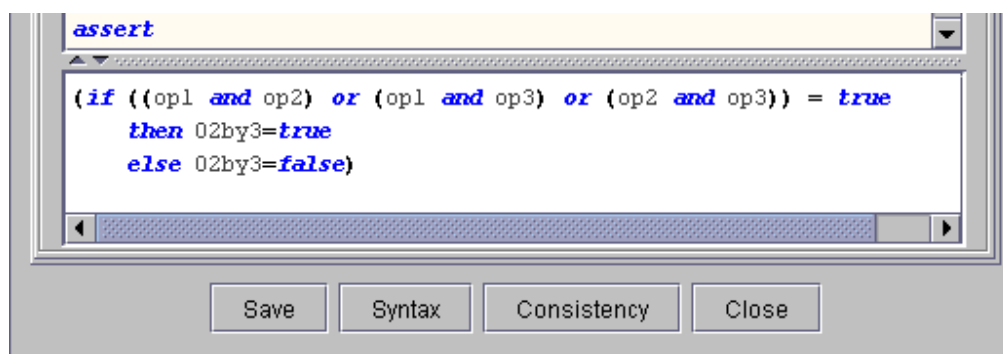
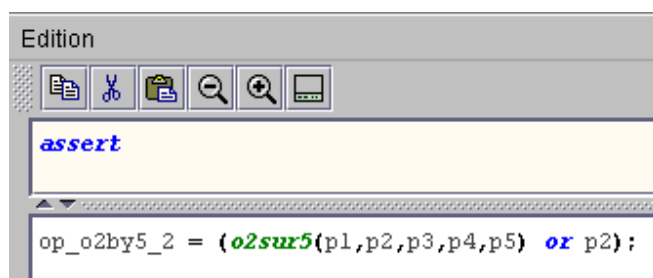


Figure 97 : Operator editor - Altarica code tab

The assertion must specify the result of the transferred function, namely the assignment of the exit flow variable associated with the operator (O2by3) according to the entry variables (operands: op1, op2, op3).

NB : Another more concise assertion for the operator definition o2sur3 given in example can be : **O2by3=((op1 and op2) or (op1 or op3) (op2 or op3))**.

Remark: Operators can be used to define the script definition of another operator.



If operators are used in Altairica code, Operators tab is accessible (Figure 98 : Altairica Code - Operator Tab). It allows selecting operator family and operator version, if different operators have the same name. If there is no ambiguity, the only family containing the operator is selected with the last version.

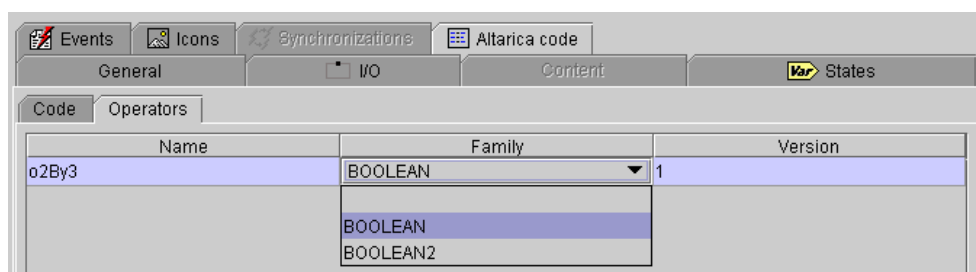


Figure 98 : Altairica Code - Operator Tab

- When the typing is done, click on the **Syntax** button (Figure 97) in order to carry out the syntax control of Altairica script. In the event of detection of syntax error, an error window appears describing the error and indicating the associated script line.

The most common syntax errors are:

- A typing error in the Altairica code, creates a false identifier that doesn't correspond to the text in the header dedicated to the display of variables specified in the tabs.
- Assertions define assignment of variables of different types (two types enumerated differently, one type enumerated with Boolean ...).

Click now on the button **Consistency** (Figure 82) in order to carry out the consistency control of Altairica script defined for the transfer function associated to the operator.

In case of detection of consistency errors, an error window appears describing in a precise way the consistency problems detected on the Altairica code.

The most often consistency errors detected are:

- Existence of a combination of operands values (input flow variables) for which the result of the operator is not defined.
- Existence of a combination of operands values (input flow variables) for which the result of the operator (output flow variable) has 2 different values.


When the new operator is saved, it is saved in 1.0 version and it is attached to the user group of the family it belongs to.

When the new operator is saved, it is automatically inserted in the manager (in alphabetical hierarchy)

Remark : By default, operator is saved in 1.0 version and it is attached to the user group of the family it belongs to.

Edit operator model

In order to edit an operator model, proceed as follows:

- Click on the **Operators** tab in the left part of the screen to access the operators family list,
- Select the model (left click on the mouse),
- Click  to access the versions tree structure of the selected model (Figure 99),

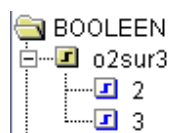


Figure 99 : Operators – Versions tree structure

- The edition of the model version is carried out in 2 ways :
 - 1) Click twice (mouse left click) on the selected version number,
 - 2) Select the version number (mouse left click) and activate the command **Model ... → Edit** via the **Library** menu or the contextual menu (Figure 72) obtained with a mouse right click on the selected model.
- A bar graph *Reading model* (Figure 100) indicates that the operator is in course of reading.

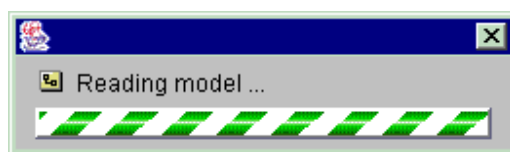


Figure 100 : Operator Bar graph reading

- When the reading is done, the operator edition window is display (Figure 94).

Rename operator

In order to rename an operator, proceed as follows:

- ♦ In the **Operators** tab, click on the operator name to be modified.
- ♦ In the contextual menu (Figure 72) or in the **Library** menu, click on the **Rename model** command; the operator name is accessible and the cursor flashes.
- ♦ Type the new operator name and validate with the **Enter** key.

CAUTION: Renaming of an operator must be followed by the operator Altarica code modification because the name of the operator result is the new operator name.

Modifying the name of an operator can have an impact on the component models using this operator.

N.B: If an existing name is used, an *Error* message indicates that the operator already exists. Click on the **OK** button and type a new name.

Duplicate operator

In order to duplicate an operator model, proceed as follows:

- ♦ Click on the operator name with the mouse right button to gain access to the contextual menu (Figure 72).
- ♦ Select the model *Operator_O* (mouse left click),
- ♦ Access the contextual menu ((Figure 72) by clicking on the operator name with the mouse right button.
- ♦ Select the command **Model ... → Duplicate**
- ♦ A *Duplicate operator* bar graph (Figure 101) indicates that the operator is being duplicated; *Operator_O* is copied with the *Operator_O_copie* name in the relevant family

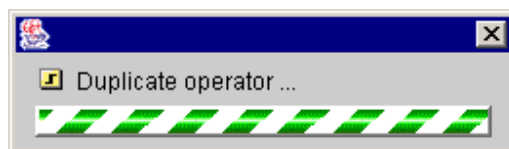


Figure 101 : Duplicate operator bar graph

- Type the desired operator name.

Caution : If the model has various versions, the duplication can be carried out only on the last version. In order to duplicate another version of the operator, it must be selected in the model tree structure versions (Figure 84) before activating the **Model ... → Duplicate** command.

Remark : In order to move the component model into another family, proceed as follows:

- copy the operator : **Model ... → Copy** command,
- paste the operator in the family of destination : **Model ... → Paste** command,
- restore the operator original name (by removing the extension « *_copie* ») in the family of destination.

Caution : If the model has several versions, the copy is carried out only in the last version. To copy another version of the operator, it is necessary to select the version in the tree structure version model (Figure 84) before activating the command **Model ... → Copy**.

Create a new version of operator model

In order to create a new version of operator model, proceed as follows:

- Click on the **Operators** tab in the left part of the screen to access the operators family list,
- Select the *Operator_O* model (left click on the mouse),
- Click on the operator name with the mouse right click to access the contextual menu (Figure 72).
- Select the **Model ... → New version** command.
- A dialog box appears (Figure 69) asking for the reference version from which the new version will be created (the possible versions are already listed in the combo box).
- Click on “Validate”
- A bar graph (Figure 87) indicates that the new *Operator_O* version is in course of creation.

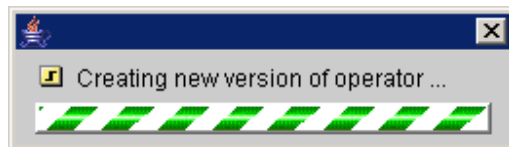


Figure 102 : New operator version creation

When the new operator version has been created, it is inserted in the versions tree structure (Figure 84).

Freeze an operator version model

CAUTION:

This operation is **very risky** and must be performed with extreme care, because it doesn't allow later modifications on the stilled operator model.

In order to Freeze operator models proceed as follows:

- Click on the **Operators** tab in the left part of the screen to access the equipment family list,
- Select the model *Operateur_O* with the mouse left click,
- Access the contextual menu (Figure 72) by clicking on the operator name using the mouse right click,
- Select the **Model ... → Freeze** command.


Remark: The frozen versions are represented in the versions tree structure by the symbol: . This **Frozen** attribute can be also visualized via the model properties editor (Figure 71). It can be obtained by the **Properties** command of the contextual menu (Figure 40) or from the general menu **File**.




Figure 103 : Operator - Model properties - Frozen

Delete operator model

CAUTION:

This operation is **very risky** and must be performed with extreme care, because it entails deletion of the operator and its associated links in all components which use it.

- To delete an operator model, proceed as follows:
 - ♦ Click on the **Operators** tab in the left part of the screen to access the equipments family list.
 - ♦ Select the model *Operator_O* (left click on the mouse)
 - ♦ Click  to access tree structure versions model (Figure 99),
 - ♦ Select the version number (left click on the mouse),
 - ♦ Click on the operator name with the right click of the mouse to access the contextual menu.

- ♦ Select the **Model ... → Remove** command; a *Message* window (Figure 104) is then displayed indicating that the operator is going to be destroyed and precising which version

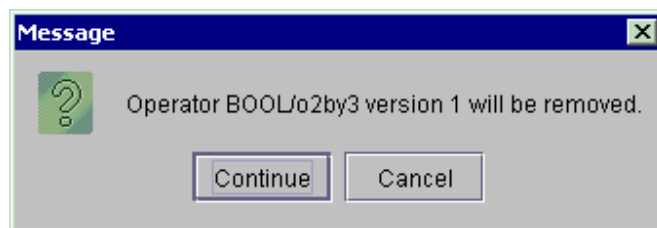


Figure 104 : Operator deleting message

- Click on the **Continue** button to pursue destruction or on the **Cancel** to stop destruction.

EDITION ON TYPE

Create a new type family

In order to create a new type family, proceed as follows:

- ♦ Click on the **Types** tab to access the types list on the left part of the screen.
- ♦ Display the contextual menu with the mouse right button (Figure 105) or use the **Library** menu.

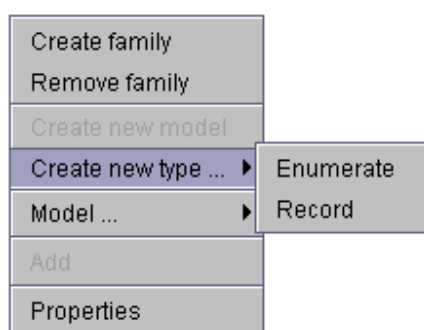


Figure 105 : Contextual menu – Create Family

- Click on the **Create famille** command.
- The window *Création d'une famille de types* appears (Figure 106) :

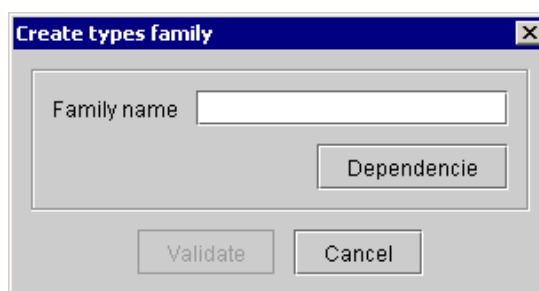


Figure 106 : Types family creation

- In the field *Nom de la famille*, type the name.
- If the owner belongs to several workgroups, he must specify the group to which must be reattached the family created.
- Click on the button **Dependencies** and select in the groups list which belongs the user (Figure 107), the owner group on which must belong the family created. Click on the **Close** button to validate the selection.

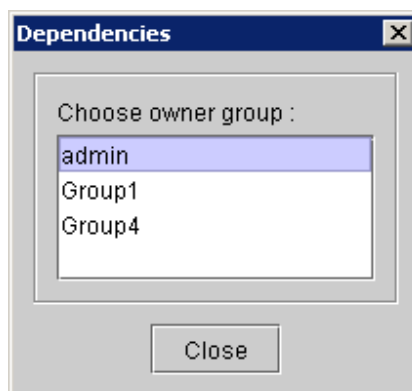


Figure 107 : Selection of the owner group

- Click on the button **Validate**, the new family appears in the **Types** tab, or click on the button **Cancel**.

Remove family

Remove a family as follows:

- ♦ Click on the **Types** tab to gain access to the list of types in the left part of the screen.
- ♦ Select the family to be removed.
- ♦ Click with the mouse right button to gain access to the contextual menu (Figure 105) or use the **Library** menu.
- ♦ Click on the **Remove family** command.
- ♦ An *Error* message is displayed if the family to be removed is not empty; in this case, removal is impossible.
- ♦ Click on the **OK** button
- ♦ Remove all the types in the family and then remove the family.


Create new type

Create new enumerate type

In order to avoid having to type variables values of « enumerated » type in the **I/O** and **Etat** editor component model tabs, it is possible to define enumerated types in Library.

Create a new enumerate type as follows:

- ♦ Use the Library – Create new type... Enumerate command or,
- ♦ Click on the **Types** tab to gain access to the list of types in the left part of the screen.
- ♦ Click with the mouse right button to gain access to the contextual menu (Figure 105) or use the **Library** menu.
- ♦ Click on the Create new type...- Enumerate command.

- ◆ In the *Type editor* window (Figure 108) and in the *Type name* field, type the name.
- ◆ In the *Name* field at the bottom of the window, type a value name.
- ◆ Click on the  icon or use the **Enter** key, the variable is displayed in the table.
- ◆ Click on the **Save** and **Close** buttons.

Remark: by default, every new type is created in 1.0 Version, it is attached to the family user group.

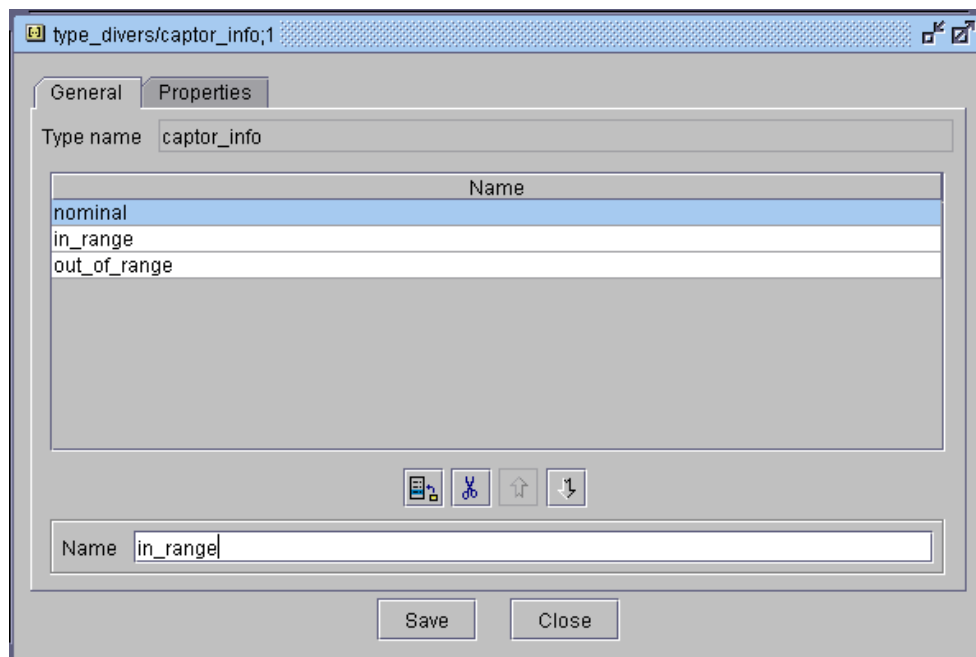



Figure 108 : Create enumerate type

N.B : In the **Types** tab in the left part of the screen, the enumerate type is indicated by the  symbol.


Create new structured type

The structured type allows the transit of several flow information variables through the same link. In the Altarica model, the enumerate type is a list of values for a variable (of state or flow) whereas a structured type is a list of flow variables.

The use of structured types allows the regrouping of several flows (information flow variables) on one bus and thus to reduce the number of graphic links to create in order to connect the components (equipments) during the edition of the architecture system. Only the flow variables (*in* or *out*) can be declared with this type of structured field (the state or local variables don't have any access).

In order to create a new structured type, proceed as follows :

- ◆ Use the Library – Create new type... Structured or,
- ◆ Click on the **Types** tab to gain access to the list of types in the left part of the screen.
- ◆ Select a family
- ◆ Click with the mouse right button to gain access to the contextual menu (Figure 105) or use the **Library** menu.
- ◆ Click on the Create new type...- Structured command.

- ◆ In the *Type editor* window (Figure 109) and in the *Type name* field, type the name.
- ◆ In the *Name* field at the bottom of the window, type a parameter name.
- ◆ Use the **Type** menu and choose the new type.
- ◆ Click on the  icon or use the **Enter** key, the variable is displayed in the table.
- ◆ In the **Type** menu, select the new type.
- ◆ Click on the **Assign** button to modify the parameter type, if necessary; the type is displayed in the *Type* column.
- ◆ Click in the *Orientation* column; select the orientation: *normal* or *inverse*.
- ◆ Click in the *Crossfield* column which is used to show a bidirectional flow (Figure 110)
- ◆ Type the desired cross field and validate with the **Enter** key.

Remark: by default, every new type is created in 1.0 Version, it is attached to the family user group.

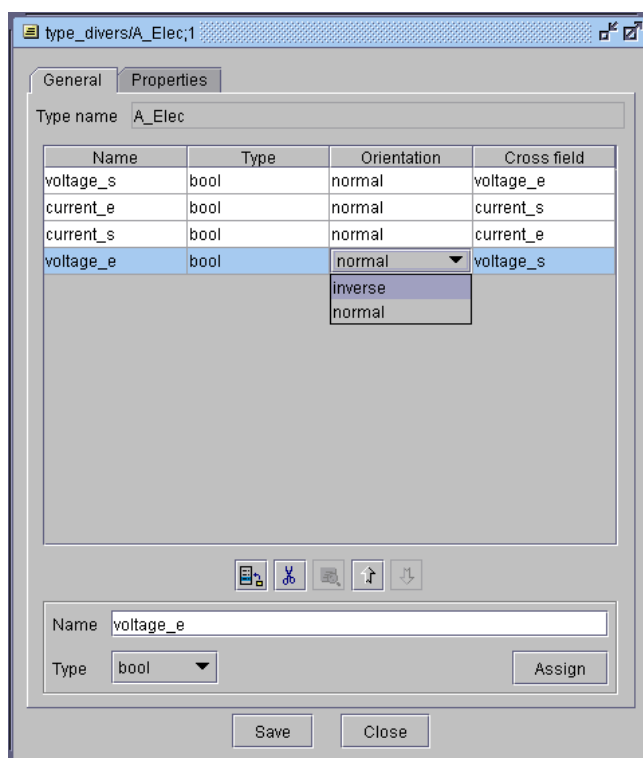


Figure 109 : Create flow structured type

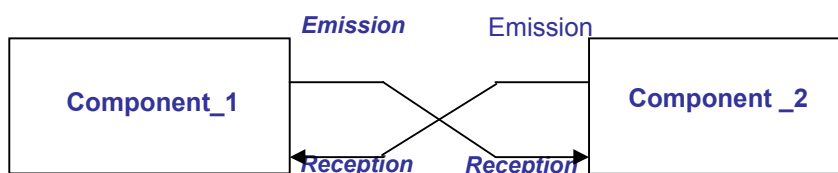


Figure 110 : Cross field for Inputs/Outputs

Transcription in Altarica code :

The definition of the structured type flow generates two connection studs:

1. a port `in`,
2. a port `out`.

Each connection stud is composed of `n` fields (flow parameters).

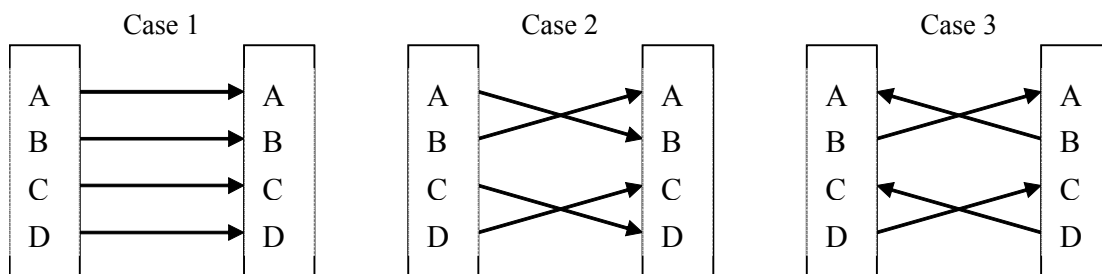
In general the field has the same orientation as its reference stud. With the clause `inverse` we can specify if the fields of the stud `in` or stud `out` have an inversed orientation.

The figure below specifies the various relations of possible equality between the fields of the stud `out` and stud `in` according to the crossfields and orientations defined for the flow parameters (A, B, C, D).

Case 1 : A data structure with direct assignments,

Case 2 : A data structure with cross field assignments,

Case 3 : A data structure with inversed and cross field assignments.



In order to simplify the Altarica code interpretation presented below, the equality relations are oriented and thus defined using the assignments.





To access the type field, the character is '^'

```
link
flow A,B,C,D : int ;
assert
  in^A := out^A;
  in^B := out^B;
  in^C := out^C;
  in^D := out^D;
knil
```

```
link
flow A,B,C,D : int ;
assert
  in^A := out^B;
  in^B := out^A;
  in^C := out^D;
  in^D := out^C;
knil
```

```
link
flow A,B,C,D : int ;
inverse out^A; out^C;
      in^B; in^D;
assert
  in^A := out^B;
  out^A := in^B;
  in^C := out^D;
  out^C := in^D;
knil
```

- ♦ The following icons are used for creating a type:

-  : to delete an Input/Output,
-  : to edit a parameter type if it is predefined (or double click with the mouse right button in the *Type* cell of the selected line),
-  : to move up in the list of Inputs/Outputs,
-  : to move down in the list of Inputs/Outputs.


These commands can also be activated via a contextual menu obtained by clicking in a cell with the mouse right button.

N.B 1: In the **Types** tab in the left part of the screen, the record type is indicated by the  symbol.

N.B 2: In the list of “predefined” types, only the list of enumerate types appears (the list of record types is not displayed) because a record type cannot contain another record type.

Edit type

In order to edit a type model, proceed as follows:

- Click on the **Types** tab in the left part of the screen to access operators family list,
- Select the model (using the mouse left click),
- Click  to access the tree structure versions of the selected model (Figure 111),

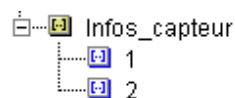


Figure 111 : Types – Tree structure versions

- The edition of the model version is carried out in 2 ways :
 - 1) Double click (left click on the mouse) on the number of the selected version,
 - 2) Select the number of the selected version (left click on the mouse) and activate the command **Model ... → Edit** using the **Bibliothèque** menu or the contextual menu (Figure 72) obtained with the right click of the mouse on the selected model.
- A bar graph *Lecture du type* (Figure 112) shows that the type is in course of reading.



Figure 112 : Type reading bar graph

When the reading is done, the type edition window is displayed (Figure 108).

Rename type

CAUTION:

This operation is **very risky** and must be performed with extreme care, because it entails deletion of the type and resulting consequences on the behaviour and validity of the Altarica laws of components whose variables are of this type (Cf. § 9.5.3 for *Suppressed links*).

- ♦ In the **Types** tab, click on the type whose name is to be modified.
- ♦ In the contextual menu (Figure 105) or in the **Library** menu, click on the **Rename model** command.
- ♦ The type name is accessible and the cursor flashes.
- ♦ Type the new type name and validate with the **Enter** key.

N.B: If an existing name is typed, an *Error* message indicates that the type already exists.

- Click on the **OK** button and type a new name.

Duplicate type

In order to duplicate a type, proceed as follows:

- ♦ Click on the type name with the mouse right button to gain access to the contextual menu (Figure 105) or use the **Library** menu.
- ♦ Select the **Duplicate model** command.
- ♦ A *Duplicate type* bar graph (Figure 113) indicates that the type is being duplicated; *Type_T* is copied with the *Type_T_copy* name in the relevant family.



Figure 113 : Duplicate type bar graph

- ♦ Key in the desired type name and validate with the **Enter** key.

Caution : If the type has several versions, the duplication is possible only in the last version. In order to duplicate another type version, we must select the version in the tree structure type versions (Figure 111) before activating the command **Model ... → Duplicate**.

Remark: In order to move a type into another family, proceed as follows:

- copy the operator : command **Model ... → Copy**,
- paste the operator in the family of destination : **Model ... → Paste** command,
- restore the operator original name (by removing the extension « _copie ») in the family of destination.

Caution : If the type has several versions, the copy is carried out only in the last version. To copy another version of the type, it is necessary to select the version in the tree structure version model (Figure 111) before activating the command **Model ... → Copy**.

Create new type version

In order to create a new type version, proceed as follows:

- Click on the **Types** tab in the left part of the screen to access the types family list,
- Select the type *Type_T* (using the mouse left click),

- Click on the type name with the mouse right click to access the contextual menu (Figure 56)
- Select the command Model ... → Nouvelle version.
- A dialog box appears (Figure 69) asking for the reference version from which the new version will be created (the possible versions are already listed in the combo box).
- Click on “Validate”
- A bar graph (Figure 114) shows that the new *Type_T* version is in course of creation.



Figure 114 : New version creation bar graph

When the new type version has been created, it is inserted in the versions tree structure (Figure 111).

Copy type

In order to copy a type, proceed as follows:

- ♦ Click on the type name with the mouse right button to gain access to the contextual menu (Figure 105) or use the **Library** menu.
- ♦ Select the **Copy** command.
- ♦ A bar graph (Figure 113) indicates that the type is being duplicated; *Type_T* is copied with the *Type_T_copie* name in the initial family.
- ♦ Select the new target family for the copy.
- ♦ Click on the family name with the mouse right button to gain access to the contextual menu.
- ♦ Select the **Paste** command.
- ♦ Key in the desired type name.

Caution : If the type has several versions, the copy is possible only in the last version. In order to copy another type version, we must first select the version.


Freeze a type version

CAUTION:

This operation is **very risky** and must be performed with extreme care, because it doesn't allow later modifications on the stilled type version.

In order to Freeze an type model proceed as follows:

- Click on the **Type** tab in the left part of the screen to access the equipment family list,
- Select the type *Type_T* with the mouse left click,
- Access the contextual menu (Figure 40) using the mouse right click,
- Select the **Model ... → Freeze** command.

Remark: The frozen versions are represented in the versions tree structure by the symbol:  2. This **Frozen** attribute can be also visualized via the model properties editor (Figure 71). It can be obtained by the **Properties** command of the contextual menu (Figure 40) or from the general menu **File**.

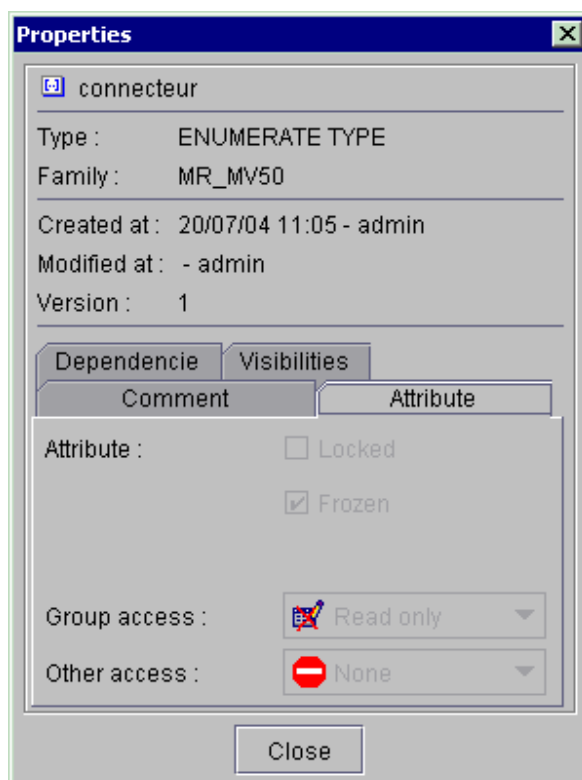


Figure 115 : Type – Model properties

Delete type

CAUTION:

This operation is **very risky** and must be performed with extreme care, because it entails deletion of the type and resulting consequences on the behaviour and validity of the Altarica laws of components whose variables are of this type (Cf. § 9.5.3 for *Suppressed links*).

- ◆ Click on the operator name with the mouse right button to gain access to the contextual menu (Figure 105) or use the **Library** menu.
- ◆ Select the **Delete model** command; a *Message* window (Figure 116) is displayed indicating that the type will be deleted.

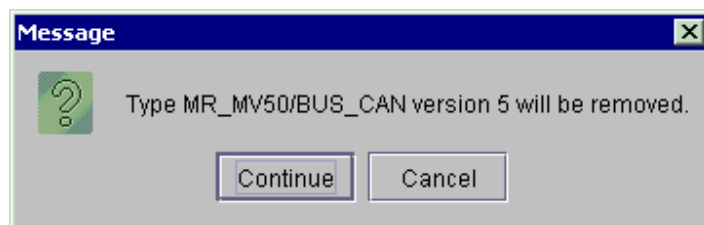



Figure 116 : Delete type message

- ◆ Click on the **Continue** button to delete the type, or the **Cancel** button to abort deletion.

SEARCH MODEL

Search for a model as follows:

- ◆ Use the **Edition – Search** menu or the **CTRL + B** command on the keyboard or the  icon in the standard tool bar; the *Search* window is displayed (Figure 117).
- ◆ Click on the **Model** tab and in the *Filter* field, type the model name (use the “*” character to fill a model name which is partially known).
- ◆ **NB:** the character « * » can be used to decrease the search space (example: **H*/val***) or to complete a partially-known-name (examples: **HYD*/val***, ***/valve**) .

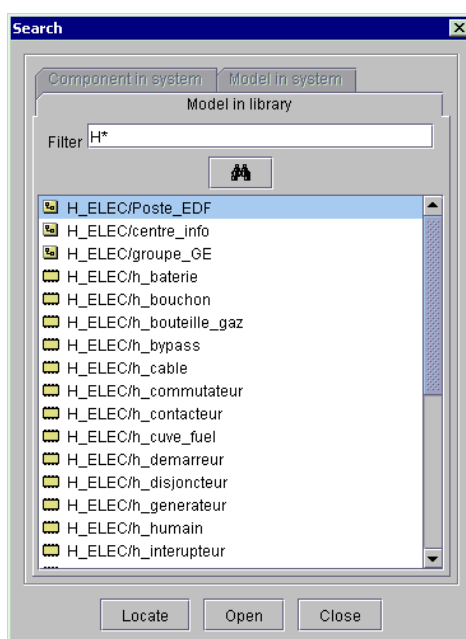



Figure 117 : Models search

- ◆ Click on the  icon located in the tab; the searched model (or the models having a name corresponding to the filter used) is displayed in the lower part of the window.
- ◆ Click on the **Locate** button.
- ◆ The DAS tool selects the tab in the left part of the screen, opens the family and displays the model looked for.
- ◆ If necessary, click on the **Open** button to edit the model or double click with the mouse right button on the selected model in the window lower part.

MANAGEMENT OF PROJECT

EDITION ON PROJECT

Create project

Create a project as follows:

- ◆ Click on the **Systems** tab in the left part of the work area.
- ◆ Use the **File – New...- Project** menu or click on a project name with the mouse right button to gain access to the contextual menu (Figure 118).
- ◆ Select the **Create...- Project** command.

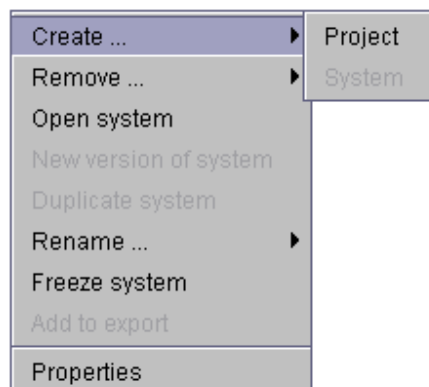


Figure 118 : Contextual menu – Create project

- ◆ In the *Create project* window (Figure 119) and in the *Project name* field, type the name.

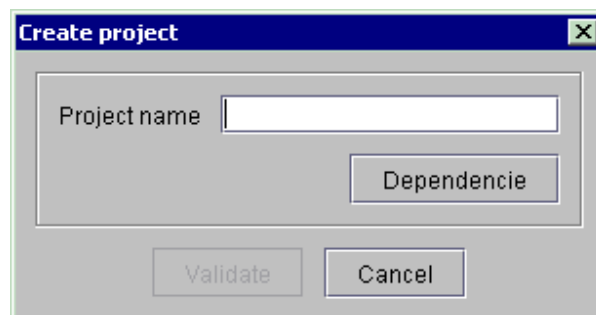
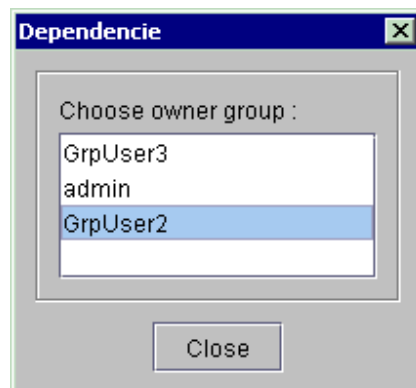


Figure 119 : Create project window

- ◆ If the user creating family belongs to more than one group, he must click on Dependencies to select the group that will be the family owner. Validate by clicking on Close.



- ◆ Click on the **Validate** button; the new project appears in the **Systems** tab in the left part of the screen, otherwise click on the **Cancel** button.

N.B: The **Systems** tab displays the projects and new systems are created within these projects.

Remove project

Remove a project as follows:

- ◆ Select the project to be removed with the mouse.
- ◆ Use the **Remove...- Project** command in the contextual menu (Figure 118).
- ◆ A *Message* window (Figure 104) is displayed and indicates that the project will be removed.

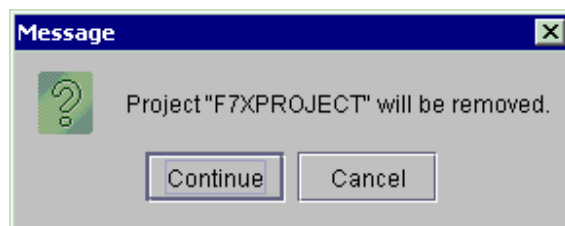


Figure 120 : Remove project message


- ◆ Click on the **Continue** button to remove the project, or on the **Cancel** button to abort the operation.

N.B: When the project contains systems, an *Error* message indicates that removal is impossible. In this case, remove all systems inside the project and then remove the project.

EDITION ON SYSTEM

Create new system

Create a new system as follows:

- ♦ Click on the **Systems** tab.
- ♦ Use the mouse and select the project inside which the system is to be created.
- ♦ Activate the command **Create...System**
- ♦ Use the File – New... - System menu
- ♦ Click on the  icon (Figure 14).
- ♦ Click on the system name with the mouse right button to gain access to the contextual menu (Figure 118).
- ♦ Select the **Create...System** command; the *Create system* window (Figure 121) is displayed.

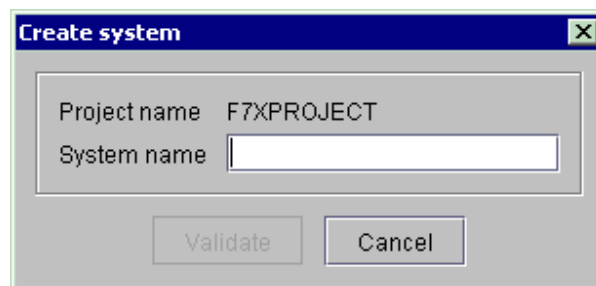


Figure 121 : Create system window

- In the *System name* field, type the name.
- Click on the **Validate** button; the new system appears in the **Systems** tab in the left part of the screen, otherwise click on the **Cancel** button.
- When the new system is created, it is automatically inserted in the systems management by alphabetical order (Figure 122) , and it is automatically edited.

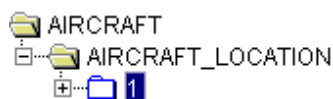


Figure 122 : System – Tree structure versions

Remark : Every new system is created in the version 1.0

Open system

Open a system as follows:

- Click on the **Systems** tab to gain access to the list of projects.


- Double click on a project to display the list of its systems.
- Double click on the system name to gain access to the tree structure system versions (Figure 118).
- Use the **File – Open system** menu or **CTRL + O** command on the keyboard or click on the  icon. Click on the system version using the mouse right click to access the contextual menu (Figure 102).
- Select the **Open system** command.
- If the system is already opened, an error message *The system is already open* is displayed (Figure 107); click on the **OK** button.



Figure 123 : System locked message

- If the system is locked, a message indicates that system opening is authorized in the read mode only.
- In this case, click on the **OK** button; unlock the system if it has been abnormally locked after an “untimely stop”; it is normally locked when it is opened by another user (Cf. §9.2.3).
- The system is displayed in the left bottom part of the screen; select the system with the mouse (Figure 124).

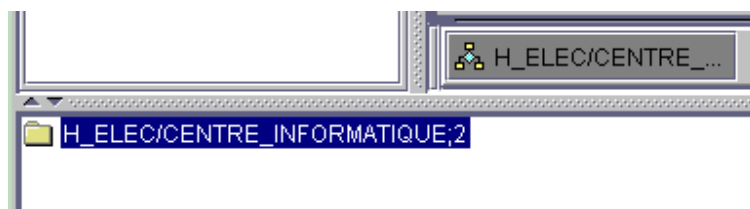


Figure 124 : Systems edition management

- Click on the system with the mouse right button to gain access to the contextual menu (Figure 125):

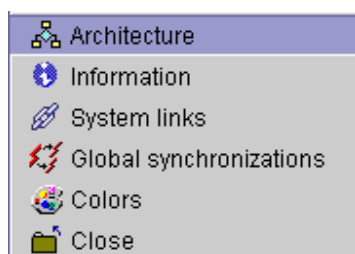


Figure 125 : System contextual menu

- Use also the **Views** menu or the icons of the *Simulation* tool bar.

N.B: A system when opened, can be simulated or edited.

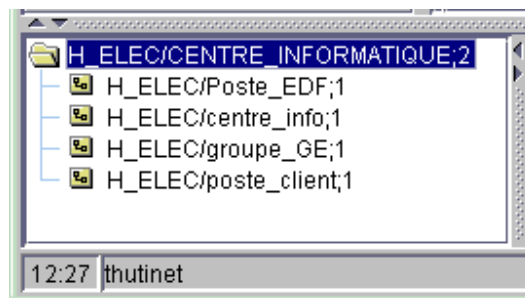


Figure 126 : System equipments

Unlock system

N.B: A system is automatically locked when it is opened to avoid that two users modify it simultaneously. It is unlocked when the user closes a system or quits the DAS tool. If the computer is stopped in an untimely mode, it may remain locked. The system **must** be unlocked before renaming or duplicating it.

The icon  **Local** indicates that the system has been locked. When a system is being renamed or duplicated, an *Error* window (Figure 127) is displayed when the system is locked.

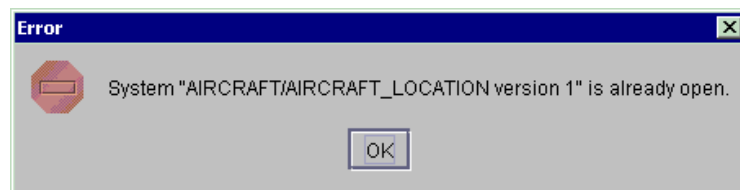


Figure 127 : Locked system message

If the session of the system edition is stopped in an inopportune way (following a problem system for example), the system can remain locked. It is **essential** in this precise case to unlock the system to be able to edit it, rename it or duplicate it again.

To unlock the system, the locked attribute can be withdrawn manually via the properties editor of a system (Figure 128). This can be obtained by the **Properties** command of the contextual menu or the **File** general menu.

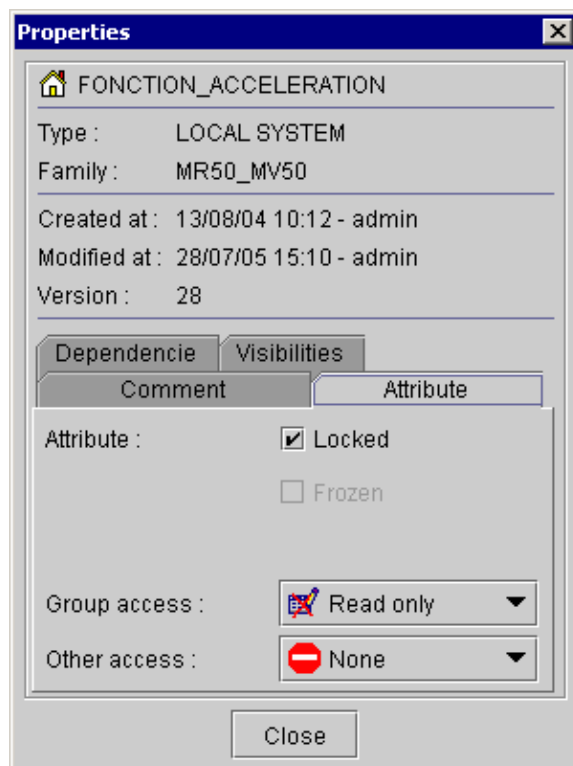



Figure 128 : Properties edition – Locked attribute

Close system

- Use the **File – Close** menu or type **CTRL + F** on the keyboard or click on the  icon or click on the mouse right button on the open system to gain access to the contextual menu.
- If the system has not been modified, a *Closing system* bar graph (Figure 129) indicates its closure.

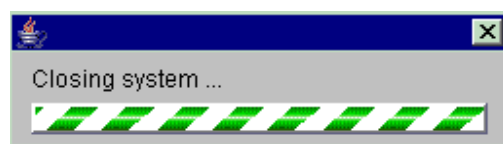


Figure 129 : Closing system bar graph

- If the system has been modified, a message (Figure 130) indicates that it has been modified and requests if it must be saved.

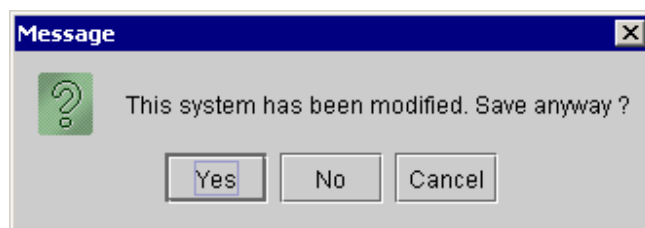


Figure 130 : Save system message

- Click on the **Yes** or **No** button as necessary, or otherwise click on the **Cancel** button.

Rename system

- Click on the system name with the mouse right button to gain access to the popup menu (Figure 118).
- Select the **Rename system** command.
- Unlock the system in case the tool has been abnormally locked following an untimely stop.
- Repeat the renaming procedure.

Duplicate system

- Click on the system name with the mouse right button to gain access to the contextual menu (Figure 118).
- Select the **Duplicate system** command.
- A *Duplicate system* bar graph (Figure 131) indicates that the system is being duplicated; *System_S* is copied with the *System_S_copie* name in the relevant project

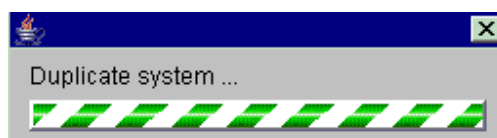


Figure 131 : Duplicate system bar graph

- Type the desired system name and validate with the **Enter** key.

Caution : If the type has several versions, the duplication is possible only in the last version. In order to duplicate another type version, we must select the version in the tree structure type versions (Figure 111) before activating the command **Duplicate system**.

Create new system version

In order to create a new system version, proceed as follows:

- Select the system *System_S* (using the mouse left click),

- Click on the system name with the mouse right click to access the contextual menu (Figure 102)
- Select the command **New system version**.
- A dialog box appears (Figure 69) asking for the reference version from which the new version will be created (the possible versions are already listed in the combo box).
- Click on "Validate"
- A bar graph shows that the new *System_S* version is in course of creation



Figure 132 : New system version message creation

When the new system version has been created, it is inserted in the versions tree structure (Figure 111).


Freeze system

CAUTION:

This operation is **very risky** and must be performed with extreme care, because it doesn't allow later modifications on the stilled type version.

In order to Freeze a system model, proceed as follows:

- Click on the **Systems** tab in the left part of the screen to access the projects list,
- Double click on the project *Projet_P* to access the systems tree structure,
- Double click on the system *System_S* to access the tree structure versions system,
- Select the number version to be frozen using the mouse left click,
- Access the contextual menu (Figure 4056) using the mouse right click,
- Select the **System Freeze** command.

Remark: The frozen versions are represented in the versions tree structure by the symbol  Local. This "Frozen" attribute can be also visualized via the model properties editor (Figure 71112). It can be obtained by the **Properties** command of the contextual menu (Figure 40) or from the general menu **File**.

Remove system

In order to remove a system, proceed as follows:

- Click on the **Systems** tab in the left part of the screen to access the projects list,

- Double click on the project *Projet_P* to access the systems tree structure,
- Double click on the system *System_S* to access the system versions tree structure,
- Select the version number to remove (mouse left click).
- Click on the version number using the mouse right click (Figure 72) to access the contextual menu.
- Select the command **Remove...** → **System**, a *Message* window is displayed indicating that the selected version system is going to be removed (Figure 133).

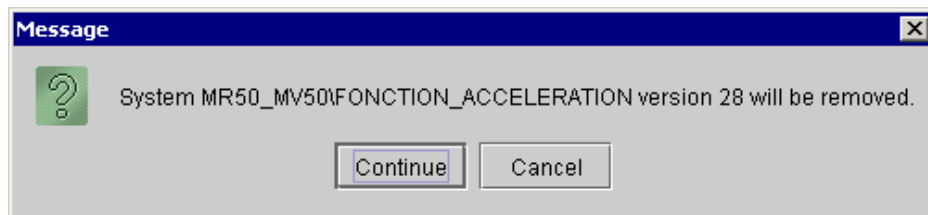



Figure 133 : System removal message

- Click on the **Continue** button to pursue the removal or the button **Cancel** to cancel the operation.

Remark: A system is definitely removed from the tree structure project if all its versions have been removed.

MODELLING A SYSTEM ARCHITECTURE

In order to create a system architecture by using components and equipment in library, proceed as follows:

- ♦ In the Systems tab, create a project “project_pp” and then create a system, e.g. “system_ss” inside this project.
- ♦ Use the Views – Architecture menu or in the system contextual menu, click on the Architecture command or in the tool bar, click on the  icon; the blank system “system_ss” is displayed in the work area.

Insertion of a component in the architecture

In order to insert a component in the architecture, proceed as follows:

- ♦ Click on the Components tab and select the component, e.g. “component_c” in order to build up the system architecture.
- ♦ Double click on the component “Component_C” to access the versions tree structure.
- ♦ Select the selected component version using the mouse left click (Figure 134).

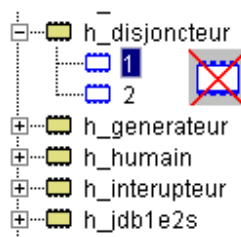




Figure 134 : Component selection

- ♦ In order to insert the component in the graphical page of the system architecture, use one of these 2 procedures :
- ♦ Maintain selection and move the mouse cursor positioned above the icon  towards the system architecture view, above the graphical page dedicated to the edition of the graphical architecture.

The moved icon becomes  indicating the possibility of inserting the authority within the architecture,

- ♦ Release the left button; the component is dropped with a default name in the architecture,
- ♦ In the contextual menu, use the Add command (the component or equipment at the top left of the architecture view).

Remark: Proceed as this same way for equipment.

When they are inserted, components and equipments are created with a default name :

- * **<component model name>xx:** for a component name, where xx is an increasing number,
- * **<equipment model name>yy:** for an equipment name, where yy is an increasing number.

The option « Don't add automatic number at the end of default instance name » found in menu “Options -> Preferences -> Edition -> Graphic options” allows to not adding automatically a number for instance component or equipment at the end of the instance name.

In order to modify a component name :

- ♦ Double Click on the component or equipment name using the mouse right click to access the contextual menu (Figure 119) concerning the edition, design and navigation commands; if necessary, rename the component.
- ♦ Right-click display Contextual-Menu (Figure 135) for Edition and Navigation :

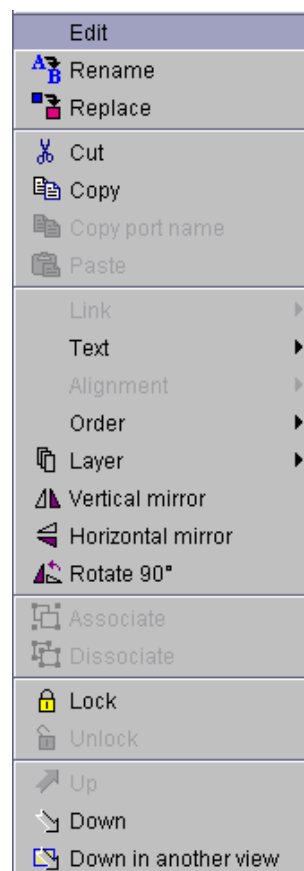







Figure 135 : Architecture – Contextual menu

- ♦ The Edit model command starts reading of the model and enables access to the various tabs of the Model editor concerning the selected component or equipment.
- ♦ Remark: A double click on a component or equipment allows the editing of its model without using the contextual menu.
- ♦ The **Rename** command () or a double click with the mouse left button on the component or equipment name when the *Labels* option of the *Display* menu is selected, enable renaming a component or an equipment.

- ♦ Use the Display menu (Figure 136) to define the display type :
 - ♦ *None*: no name is displayed,
 - ♦ *Node Labels* : all the instance names are displayed above the components and equipments,
 - ♦ *Link labels*: the type transported by the link is displayed at the center of the link for all links
 - ♦ *All labels*: Display “Node labels” and “Link labels”



Figure 136 : Architecture – Display menu

- ♦ The Replace command () or a double click with the mouse left button on the component or equipment name when the Models option of the Display menu is selected, enable replacement of a model by a model already existing in the library.
- ♦ Remark: If the new model has the same interfaces (inputs/outputs with same names and types) as the previous model, the links eventually present in the hardware architecture will be saved, otherwise they are deleted.
- ♦ The Cut, Copy and Paste commands allow the corresponding operations on equipment, a component or a link. For components and equipments, the instance name of the newly created object follows the rules of the action of adding a component or equipment.
- ♦ The Lock command enables locking of a component or of a link in architecture to avoid its displacement; the selection edges of a component which were in red colour appear grey.
- ♦ If in an architecture only one component is locked, unlock it and then lock the overall architecture.
- ♦ In an architecture, the equipment name appears in bold characters which enable the use of the following icons or commands:
 - * : move up in an architecture to display a higher level architecture,
 - * : move down in an architecture to display a lower level architecture,
 - * : move down to another view to display a lower level architecture in a new window.
- ♦ Carry out the same transfer procedure for all the components or equipment making up the system architecture (Figure 137).

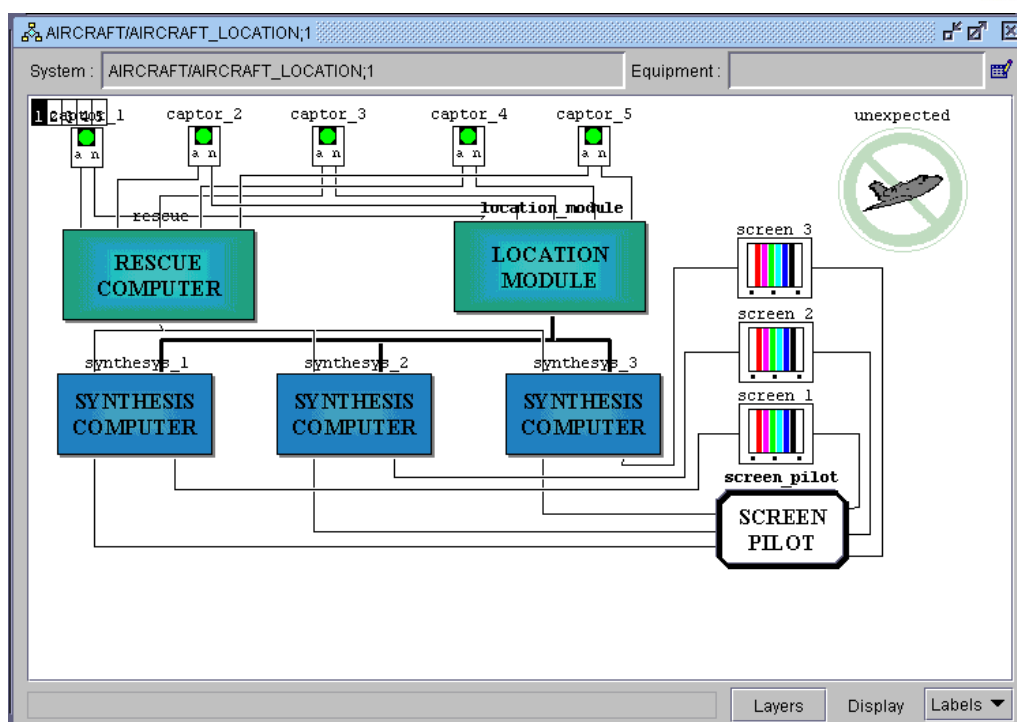





Figure 137 : Architecture without links

- ◆ Use the following design commands of the contextual menu (with the mouse right button) or the icons of the Design tool bar to optimize the layout of components or equipment in the work area:

- * Vertical mirror: 
- * Horizontal mirror: 
- * 90° rotation: 

- ◆ Maintaining Left-Click on an element allows moving it into the workspace. It can also be made with the keyboard arrows (dot par dot).

*

Link two components

In order to link two components via their communication ports:

- ◆ Click on the output black square of one and establish a link to the input white square of the other one. When the link is established, if necessary, click on the link and move it; this entails appearance of a breakpoint (white colour: selection and black colour: non-selection) (Figure 138).

This link procedure checks the compatibility of the connected inputs/outputs (direction: in/out; type: inverse/normal/cross field).

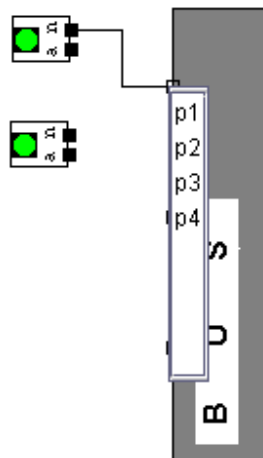

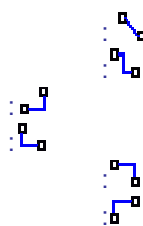


Figure 138 : Structured type port connection

Once link is made, if necessary, click and move the link, it creates a break-point (white : selected, black : not selected). Create break-point to avoid link intersection.

- ♦ Click on a link and use the Link command of the contextual menu (with the mouse right button), or the icons of the Link tool bar, to optimize the path of the links between the components or equipment:

- * Direct
- * Right angle
- * Right angle from right to bottom
- * Right angle from left to bottom
- * Right angle from right to top
- * Right angle from left to top
- * Possibility of positioning an arrow on a link in order to visualize the direction of a flow (2 arrows in the event of bidirectional flow : )



Remark: When a link features two right angles, move the straight line between these two right angles as follows: position the mouse cursor on the straight line and move the cursor while pressing the SHIFT key; the straight line follows the cursor and moves in a parallel direction.

Remark: To create a link from an existing link with its breakpoints, select a link with the mouse while pressing the CTRL key. Then, move the mouse cursor to a port and release it when you have reached the port.

- ♦ Select a group of components with the mouse and use the contextual menu commands to optimize the components or equipment alignment:

- * Left 
- * Right 
- * Top 
- * Down 
- * Vertical 

- ◆ The overall system architecture with its entire links is displayed in the work area (Figure 139).

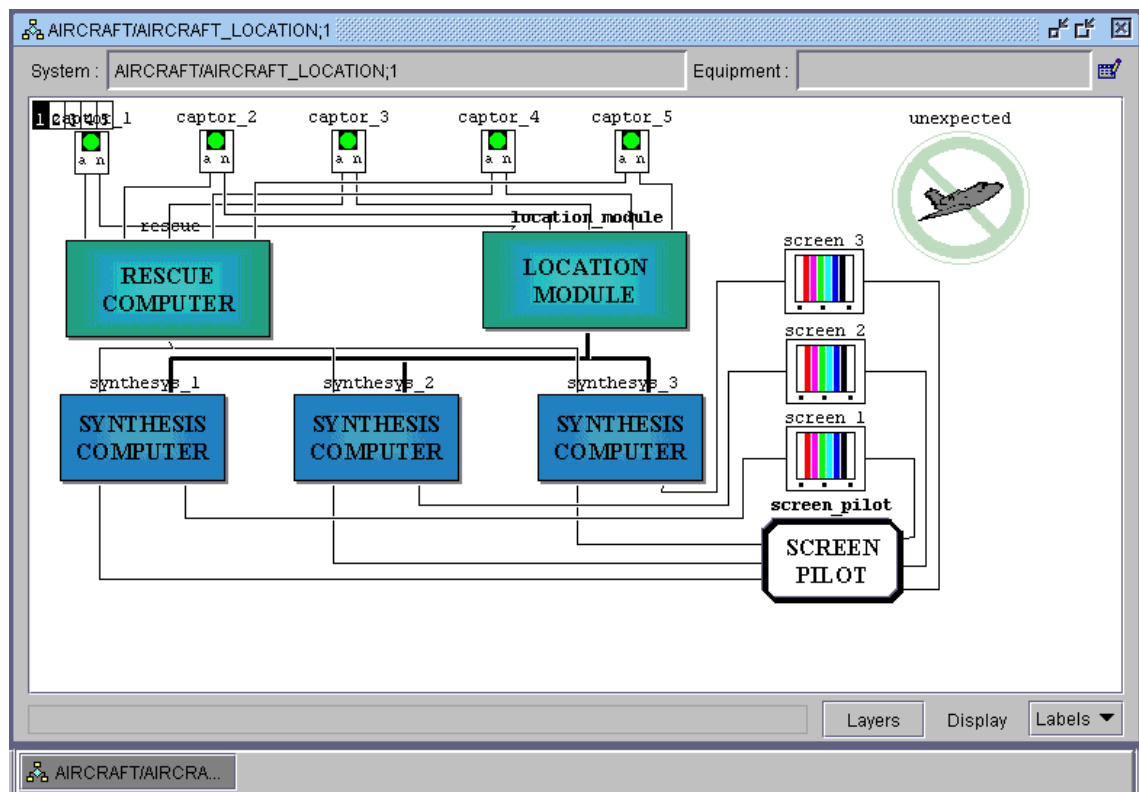


Figure 139 : Architecture example with links

- ◆ Click on an input port (White Square) or output port (Black Square) of a component or equipment (e.g., an input of the “synthesis computer” equipment).
- ◆ The status line located at the bottom of the screen displays the “synthesis_3.e_anna (bool)” information which gives the following indication:
 - * Equipment name: synthesys_3,
 - * Port: e_anna,
 - * Type: bool.

Remark: When the mouse cursor is on a link, the display area located at the left lower part of the architecture view indicates the link description: e.g. a.b <->c.d.

Remark: When the mouse cursor is on a component or equipment, the visualization zone located in the left bottom architecture view shows the name given within the architecture and between brackets the name of the component or equipment.

```
synthesys_3 (AIRCRAFT/A_synthesis_compute2)
```

Remark: When an existing architecture is opened or if modifications are carried out in the library and these modifications concern an opened architecture, the DAS tool executes a consistency control on all links between the components and equipments. If a consistency error is detected (consistency between the architecture and the library: model, type used in architecture), the link is deleted.

- ◆ When a link is deleted in architecture, a message (Figure 140) indicates its deletion.

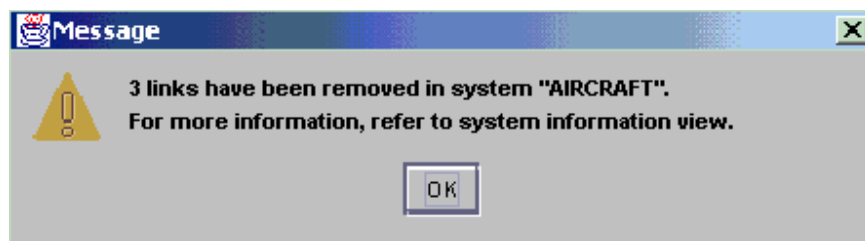


Figure 140 : Link deletion message

- ◆ Click on the OK button.
- ◆ Consult the system Information window
- ◆ Click with the mouse right button on the button corresponding to an opened architecture at the bottom of the work area; the following contextual menu (Figure 141) is displayed:

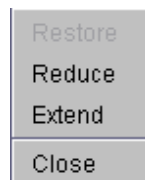


Figure 141 : Contextual menu

Utilization of display layers

The user has 5 display layers. Graphical objects representing components, equipments, system architecture links are put into these layers according to display properties.

In order to put graphical elements in the different display layer, user must proceed as follows:

- ◆ Select the graphical element (component, equipment, link) (Figure 142),
- ◆ In contextual menu (right click) select the layer on which the element must be put.

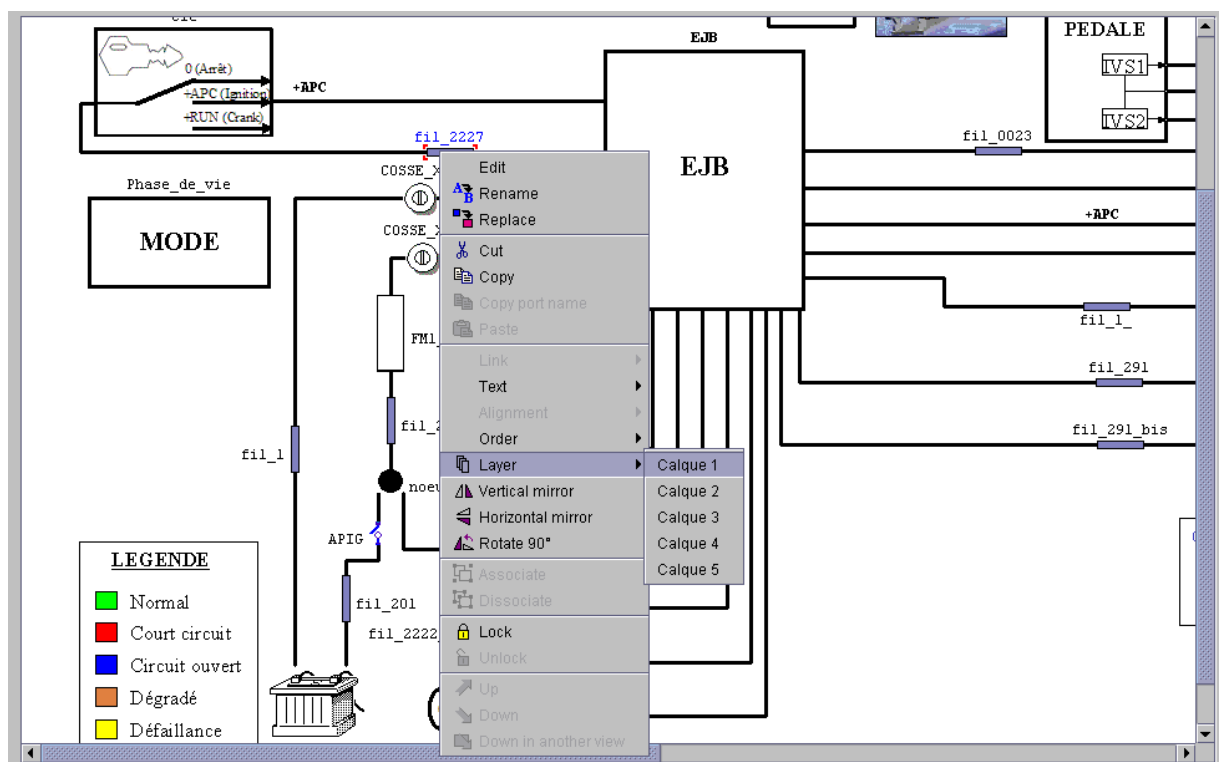


Figure 142 : To put an element on a display layer

In order to control the elements display, the user must proceed as follows:

- ◆ Click on **Layers** button (Figure 143) to display Definition of display Layers window (Figure 144).

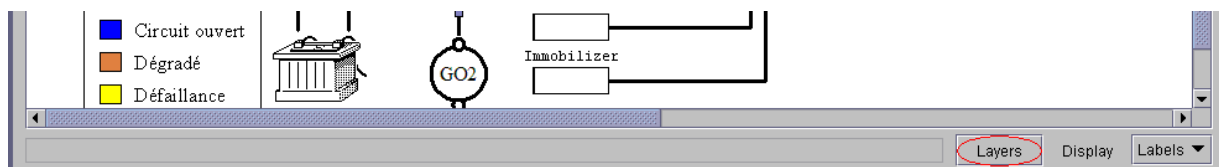


Figure 143 : Layers edition

With the window of layer definition (Figure 144), display priority order can be defined for the 5 layers in which the user has put equipments, components or links.

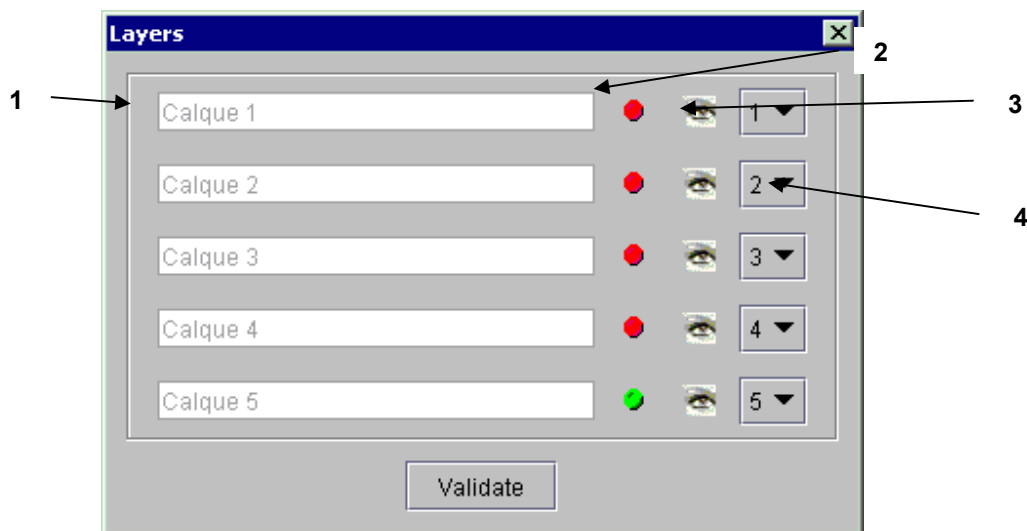






Figure 144 : Window of layers definition

This window is made of the following areas:

- Name Area (1): double-click on the area in order to edit the layer name; the default name is “Calque n”.
- Active layer in architecture edition (2): this area is made of a radio-button  whose selection (became green ) shows the active layer choice in edition of system architecture. Components or equipment inserted in system architecture during edition will be put on this layer.
- Visible layers (3): the user can choose if a layer is visible  or not .
- Layers orders (4): the user can select the layers display priority (1 : foreground layer, to 5 : background layer)

The final system visibility is made by the 5 layers superimposition according to their priority.

For example, the following configuration specifies that elements put on “Calque 2” are not displayed.

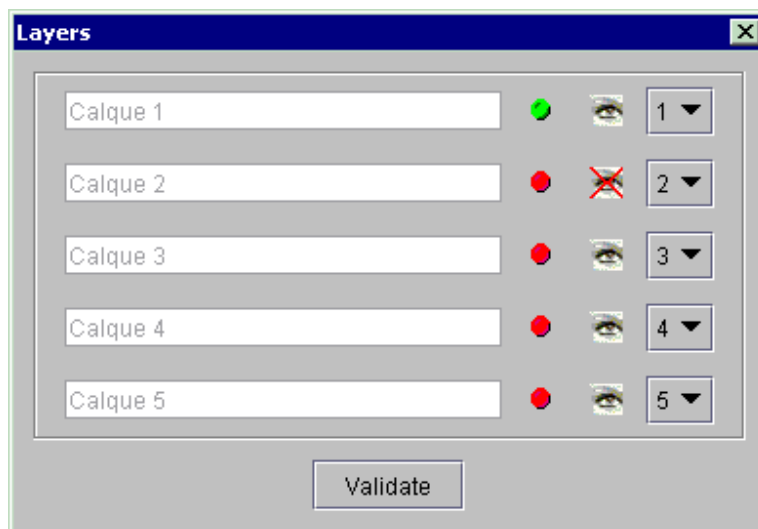




Figure 145: to hide layer 2

SEARCH COMPONENT MODEL IN A SYSTEM

Search for a component as follows:

- Use the **Edition – Search** menu or type the **CTRL + B** command on the keyboard or use the  icon in the *Standard* tool bar; the *Search* window (Figure 146) is displayed.
- Click on the **Component** tab and in the *Filter* field, type the component name as follows : <Family_Name>/<Model_Name>
- **NB:** the character « * » can be used to decrease the search space (example: **H*/val***) or to complete a partially-known-name (examples: **HYD*/val***, ***/valve**).
- Click on the  icon, the component searched for (or the components having a name corresponding to the filter used) is displayed in the middle part of the window

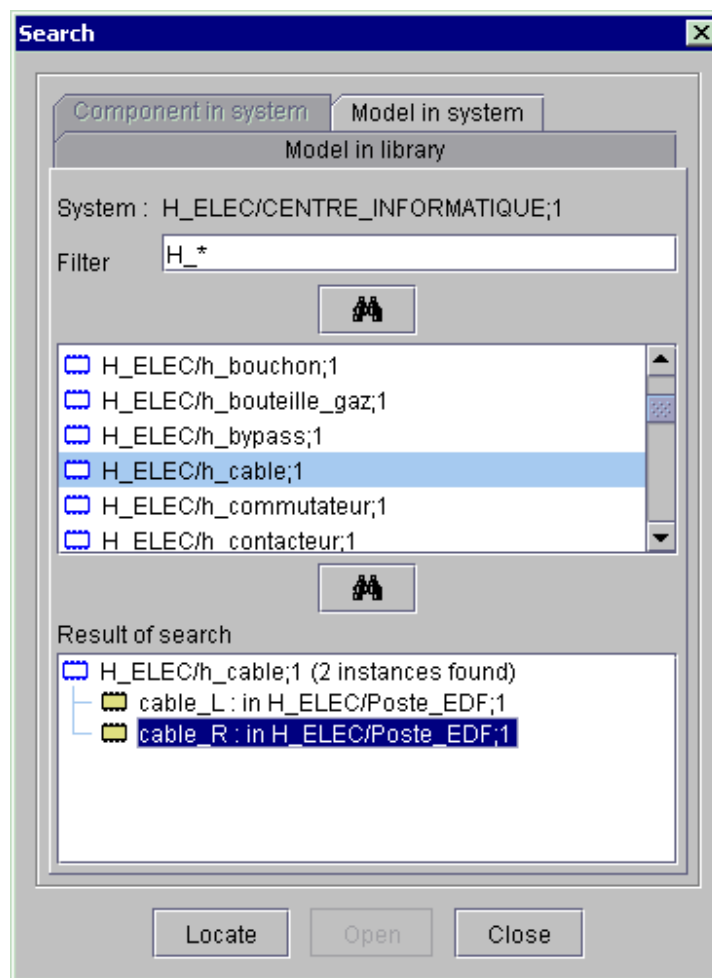



Figure 146 : Search window

- Select the searched model, click on the second icon , the search model and the number of found instances appear at the bottom (result of search).
- Double click on the model to unroll the list of model instances in the system. It provides information's about localisation (in H_ELEC/Poste_EDF; 1).
- Select an instance, click on the **Locate** button in order to move the view on this instance.

INFORMATIONS

Remark: The **Information** functionality is used when a consistency problem message is displayed to the user indicating that models are unknown or that links have been deleted.

These access commands check that architecture is consistent with respect to the library objects after renaming or deletion operations.

- Open the system architecture.
- Use the **Views – Information** menu, or in the system contextual menu for an open system, click on the **Information** command to display system information.
- The *Information* window (Figure 148) is displayed: it contains four tabs:
 - * **Unknown models,**
 - * **Suppressed links,**
 - * **Renamed components,**
 - * **Changed models.**

Access commands

- Click on the **Properties** tab (Figure 147) the window displays the following information's :
 - System Creation date
 - Last modification date
 - Version number,
 - A comment on the system, it can be modified by any user with write privileges.

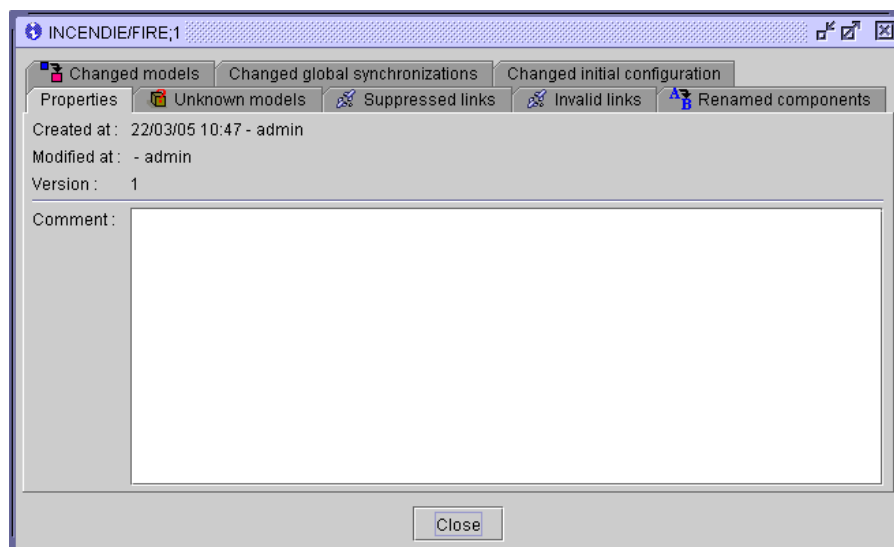


Figure 147 : Information - Properties

Remark: This information can be accessed by **Properties** command in **File** Menu.

Unknown models tab

- Click on the **Unknown models** tab (Figure 148) the window displays the list of unknown models.

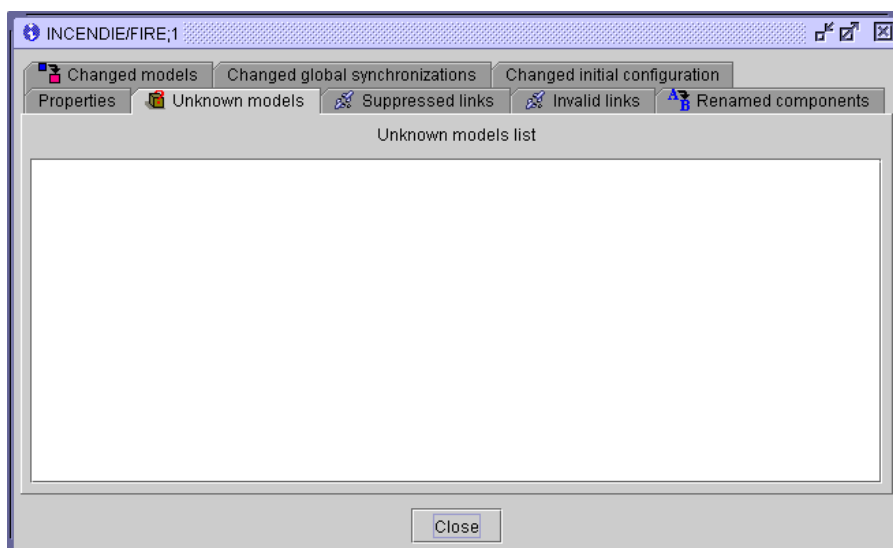


Figure 148 : Information –Unknown models tab

Suppressed links tab

- Click on the Suppressed links tab (Figure 149); it indicates the From – To information for each of the following elements: component, component model, port, port type and field of type.

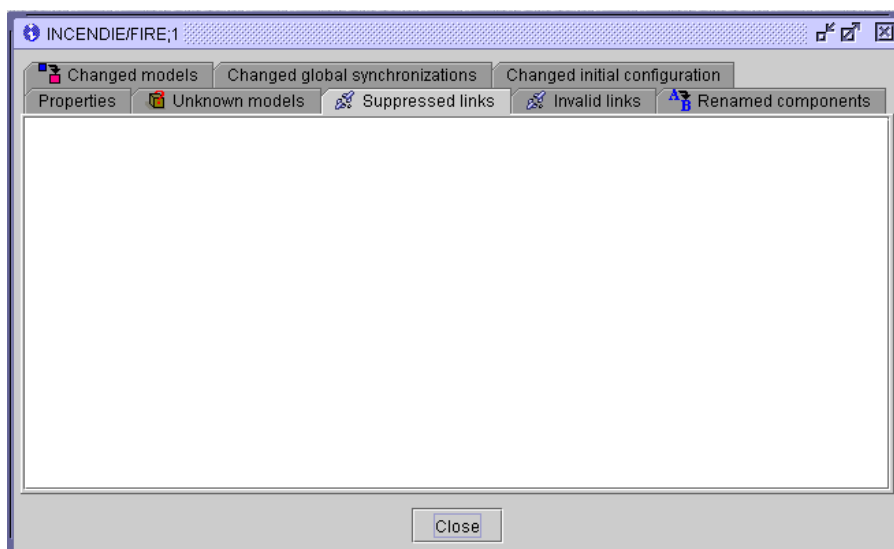


Figure 149 : Information – Suppressed links tab

Invalid links tab

- Click on the Invalid links tab (Figure 149); it indicates the From – To information for each of the following elements: component, component model, port, port type and field of type.

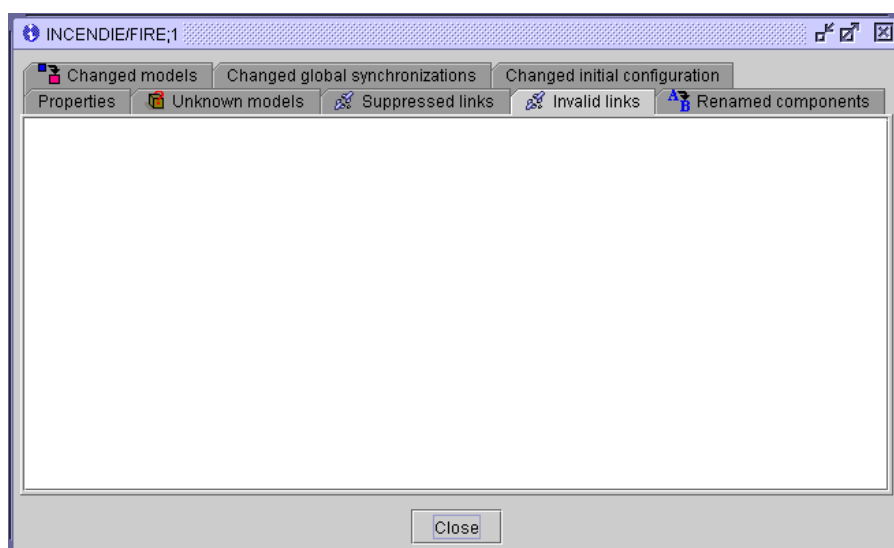


Figure 150 : Information – Invalid links tab

Renamed components tab

- Click on the **Renamed components** tab (Figure 151).

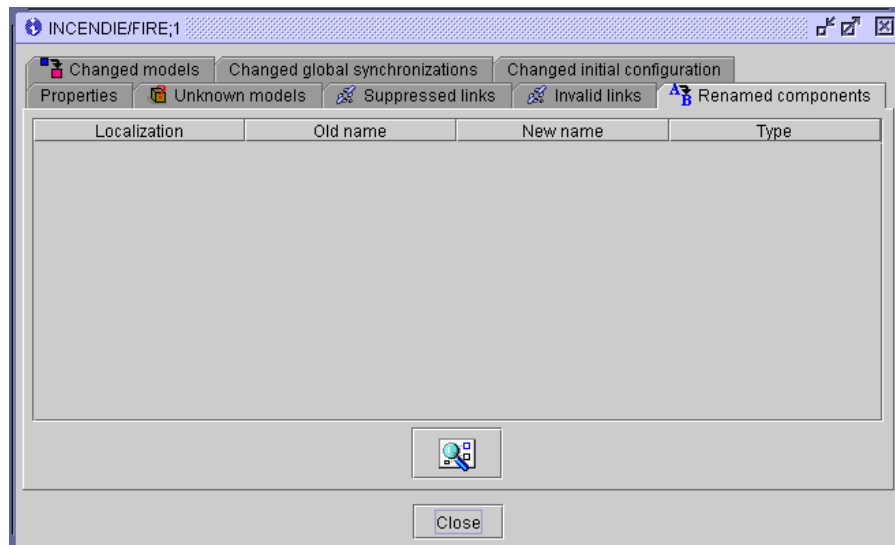


Figure 151 : Information – Renamed components tab

- ♦ The window displays the following information concerning components which have been renamed in an architecture:
 - * Localization,
 - * Old name,
 - * New name,
 - * Type.

Changed models tab

- Click on the **Changed models** tab; the window (Figure 152) indicates the models, the ports and the states, the attributes and the laws.

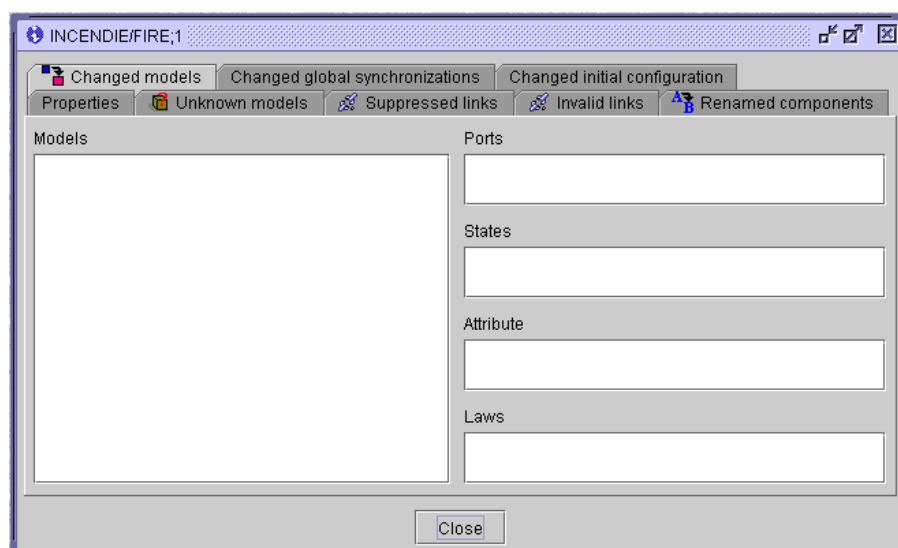


Figure 152 : Information – Changed Models tab

Changed global synchronizations tab

- Click on the **Changed global synchronizations** tab; the window (Figure 152) indicates the invalid laws.

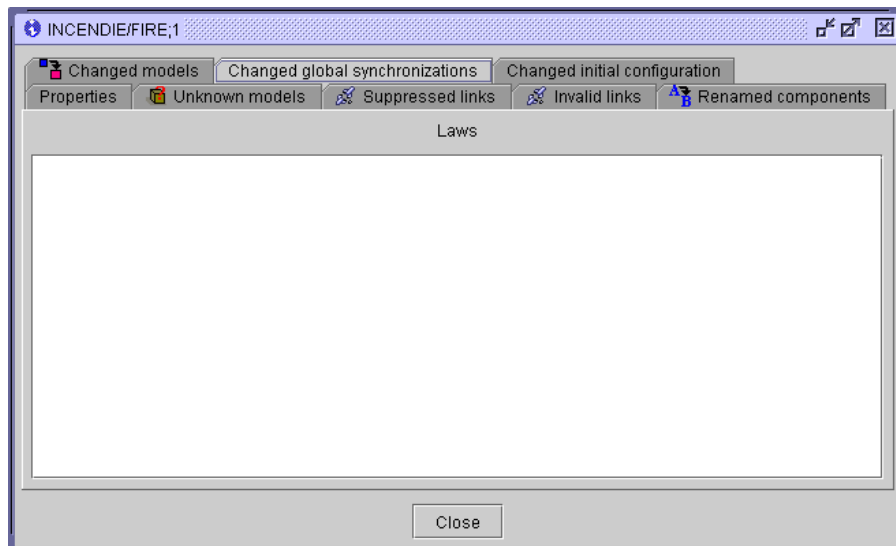


Figure 153 : Information – Changed global synchronizations tab

Changed initial configuration tab

- Click on the **Changed initial configuration** tab; the window (Figure 152) indicates the unknown attributes and the invalid event laws.

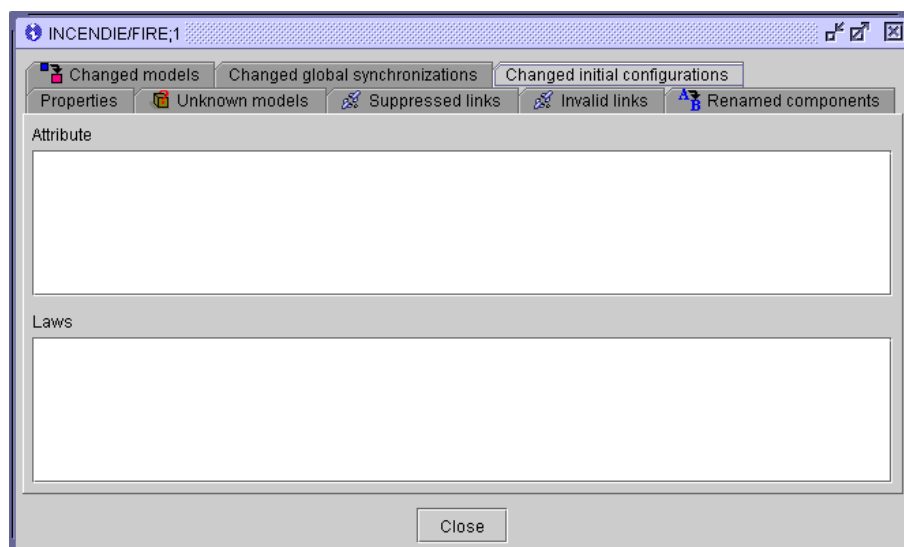


Figure 154 : Information – Changed global synchronizations tab

SYSTEM LINKS

Access commands

- Use the **Views – System links**, or in the system contextual menu, click on the **System links** command to display the corresponding information.
- The *System links* window is displayed; it contains two tabs:
- System assertions,
- Links.

System assertions tab

A system assertion enables modelling of a link which has no graphical representation by an Altarica code.

- Click on the **System assertions** tab; the following window (Figure 155) is displayed.
- In the left part of the window, click with the mouse left button on the “main” hierarchical level, then double click to gain access to the other hierarchical levels; select the desired hierarchical level.
- The *Edition* right area displays the variables assignment in the system assertions.

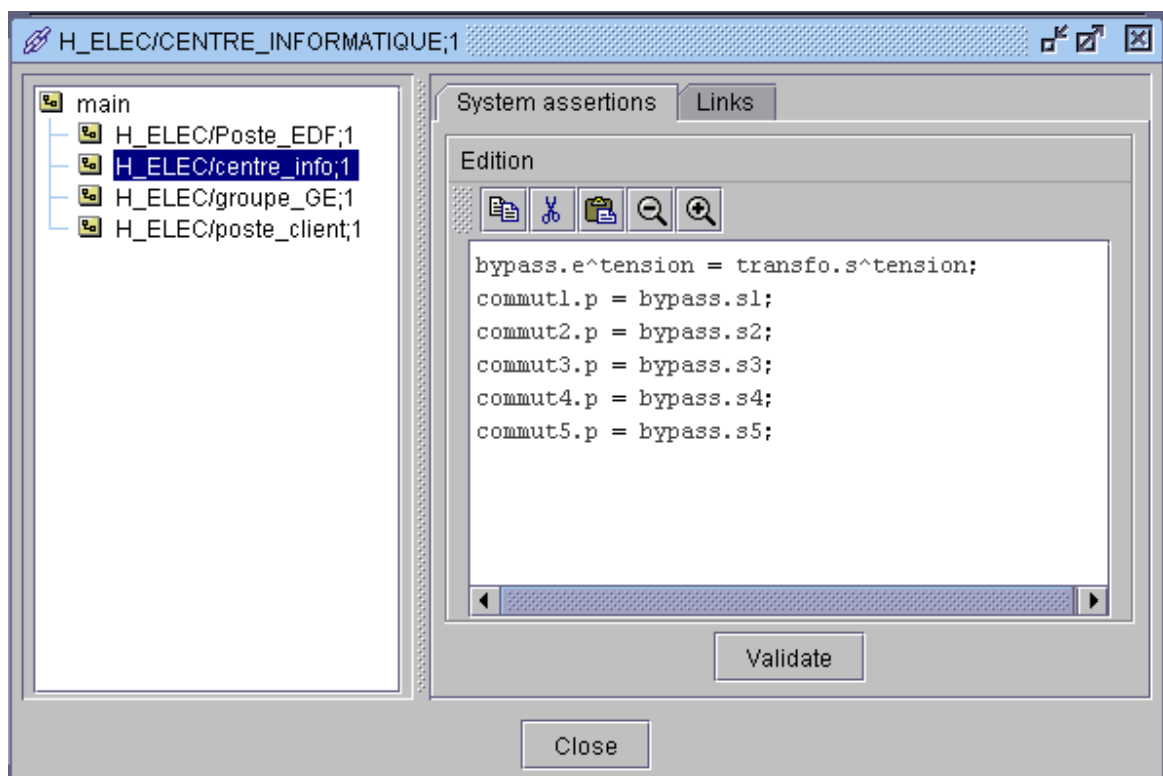


Figure 155 : System assertions tab

Remark: System assertions are links which are not displayed in the hardware architecture; they concern only flow assignments.

The syntax for the system assertions is:

1. Simple affectation between "in flow" and "out flow" with same type.

<component_i>.<In_Name> = < component j>.<OutName>;

2. Simple affectation between "in flow" and a specific value from its definition domain.

<component_i>.<In_Name> = true ;

<component_i>.<In_Name> = faible ;






3. Affectation using operators:

**<component_i>.<In_Name> = (<component j>.<OutName> or
<component k>.<OutName>);**

**<component_i>.<In_Name> = my_operator (<component j>.< OutName >,
< component _k>.<OutName S>);**

- The drafting of the system assertions is simplified since the copy of the port name is made possible. After having pointed the mouse cursor on a component or equipment port, reach the function "Copy le nom du port" via the contextual menu. The port name can then be pasted in the equipment assertions editor.

- Five icons are available to edit system assertions:

- : Copy,
- : Cut,
- : Paste,
- : Backward zoom,
- : Forward zoom.

Remark: the validity of these system assertions will be verified by the system consistency control (activated with the command **Consistency Control** from the **System** menu).

CAUTION: In system architecture, assertions can only be written at the system level (**main** node selected). For hierarchical level associated to equipments, only consultation of assertions defined in library is possible (in equipment model editor).

Links tab

A link defines an interconnection between two objects in architecture.

- Click on the **Links** tab; the following window (Figure 156) displays an exhaustive list of all the links which have a graphical representation in a given hierarchical level.
- In the left part of the window, click with the mouse left button to gain access to the “main” hierarchical level, and then double click to gain access to the other hierarchical levels; select the desired hierarchical level.

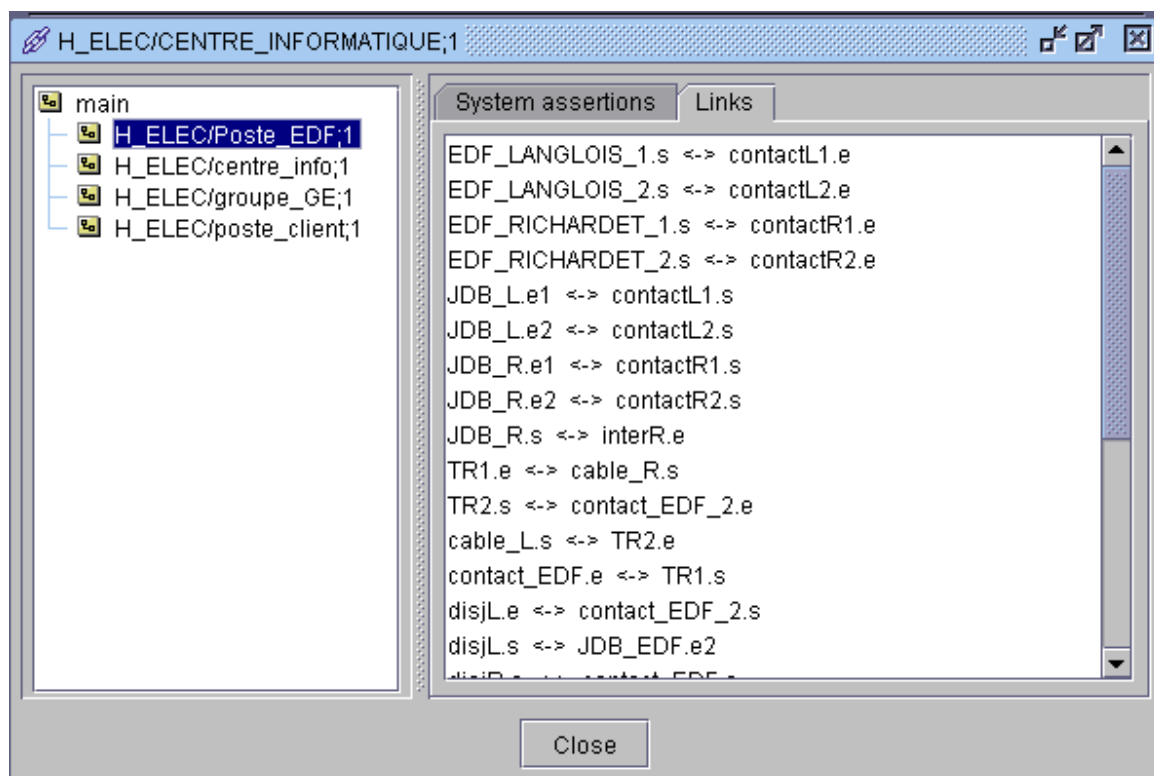


Figure 156 : System assertions – Links tab

LINKS COLOR

- ♦ Open system architecture.
- ♦ Use the Views – Colours menu or the contextual menu obtained by clicking with the mouse right button on an open system, to define the links colour in a system during simulation.
- ♦ The Links colours window (Figure 157) is displayed: it contains in its right part a colour palette, and in its left part, an area displaying the link colours arranged according to their type.
- ♦ When a new architecture is created, the boolean type is the default type with two colours (modifiable) (Figure 157):
 - * False: red,
 - * True: green

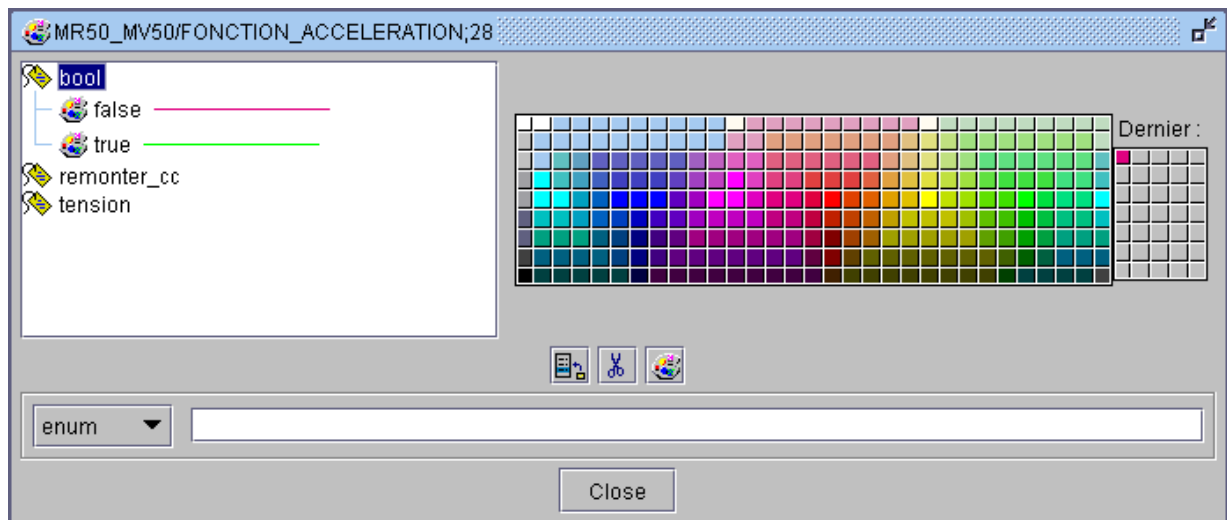



Figure 157 : Boolean variable colours

- ♦ If a new architecture contains links (non-boolean) which must be coloured, carry out the following steps.
- ♦ Use the type selection button associated with the link (enum, bound or predefined). Only the predefined and enumerate types are explained.
- ♦ Click on the  icon or use the Enter key.
- ♦ E.g.: To select a predefined type variable, type pressure in the enum field; the following window is displayed (Figure 158) :

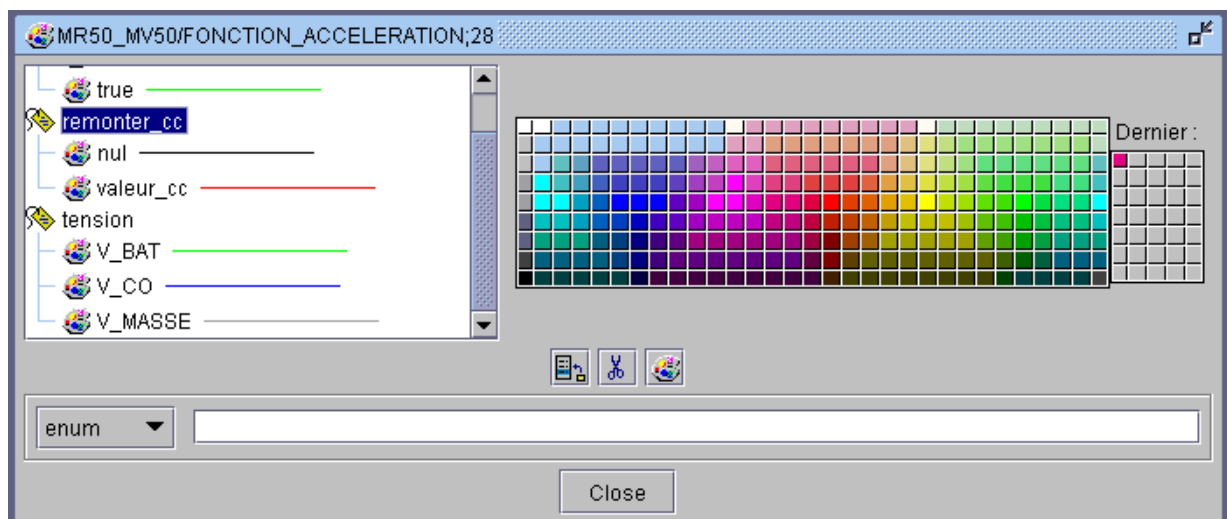



Figure 158 : Links colour

- ♦ Double click on an element, e.g., pressure; the four possible values for pressure links are displayed:
 - * High,
 - * Low,
 - * Normal,
 - * Null,
- ♦ Select a value, e.g. Normal.

- ♦ In the colour palette, select a colour; the recent area on the right indicates the last selected colours.
- ♦ Click on the  icon; the colour of the line next to the Normal value becomes the selected one.
- ♦ Repeat the same procedure to define the colours for the three other values.

N.B: The colour assignment is specific for a system; the colours are saved with the system.

Remark: An existing enumerated type can be deleted by clicking on the corresponding icon.

- ♦ Display the results of this colour assignment during system simulation.

Remark: When a link is of the record type, the colours applicable to the link are those associated to the type of the first parameter in the record

PRINTING

Prepare the printing of a model or system or of the *System links* window as follows:

- Select the File – Printing format menu.
- The Page Setup (*Mise en page*) window (Figure 159) is displayed; in the *Paper* area, select:
 - * The size (A4 or A3),
 - * The source (manual feeding or automatic selection).
- In the *Orientation* area, tick the *Portrait* or *Paysage* (Landscape) mode.
- Each format or orientation modification automatically updates the scale factor of the page displayed.
- In the *Margin* area, type the margins (left, right, up or down).
- Click on the **OK** button to save the new printing format, or click on the **Cancel** button to return to the previous format prior to this window opening.

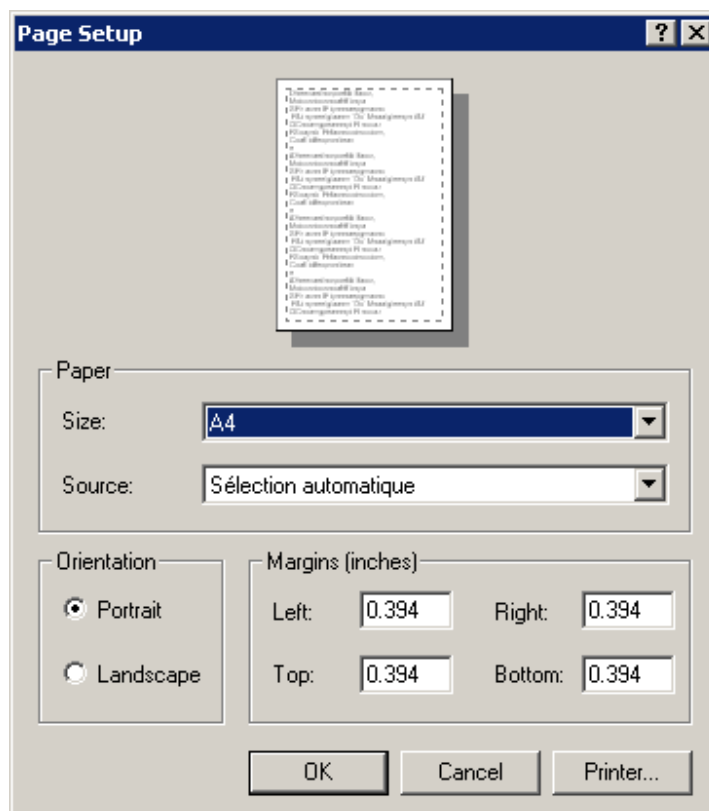
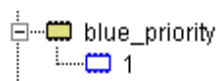



Figure 159 : Page Setup


Print a model as follows:

- Select the model version you want to print, in tree of the considerate model.



- Edit model by right-clicking on the version, select **Model – Edit** in contextual menu
- Select the **File – Print** menu or click on the  icon.

Print a system as follows:

- Select the system version you want to print, in tree of the considerate model.
- Open System (double-click)
- Select the **File – Print** menu or click on the  icon to print selected system. When printing architecture (Figure 160), select the pages to be printed.

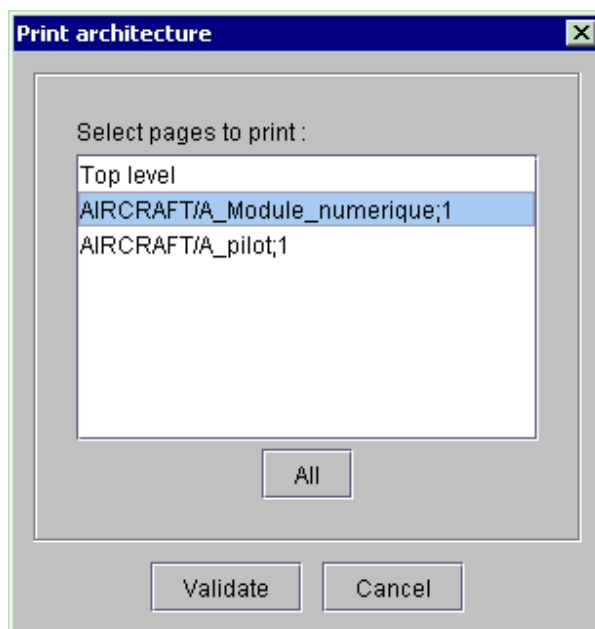


Figure 160 : Print architecture window

- Click on the **Validate** button, or click on the **Cancel** button to abort the printing.
- The *Printing* window is displayed (Figure 161).
- In the *Printing area*, select the area to be printed (all, pages *n* to *m* or selection).

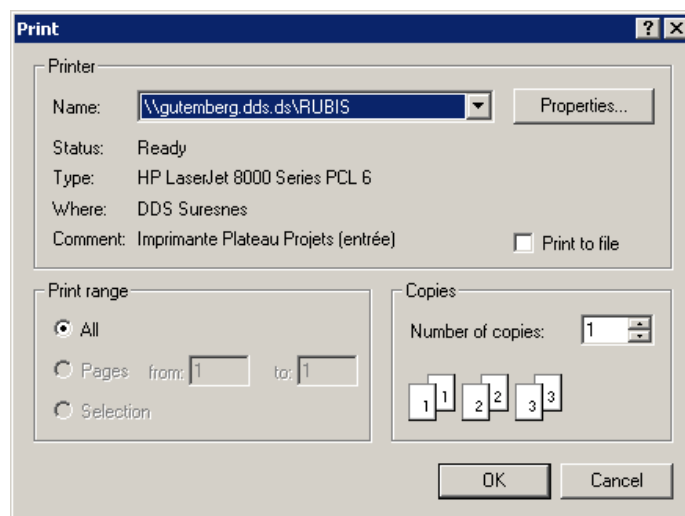


Figure 161 : Printing window

Remark: In the **Preferences** window – **Options** tab, it is possible to select a file containing a logo which will be added when printing.

EXPORT/IMPORT DATA INTO XML FORMAT

In order to facilitate data exchanges, DAS allows export and import of every DAS object (family, project, models, and systems) into XML Format.

EXPORT IN XML FORMAT

To export data, user must proceed as follows:

- Activate the export setup window (Figure 162) with the **Export** command of **File** menu.

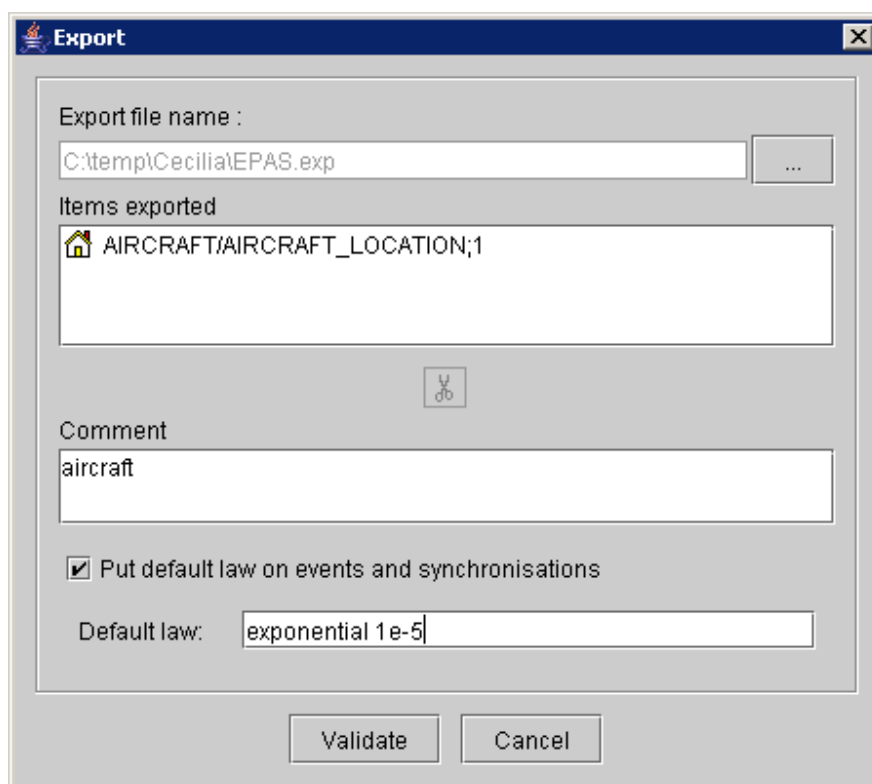



Figure 162 : XML export setup

- Click on  button in *Export File Name Area*,
- Chose the export file,
- The user must now add the elements that have to be exported. Select an element in the object manager and use the **File/Add to Export** menu, or Drag and Drop the object in “Items exported” area.
- The *Comment* area allows exported data description.

- The *Put default law* put a default law on events and synchronizations during export. The default law is only put on events and synchronizations that have no law. Use the *Default law* area to enter the default law
- Click on **Validate** button to export or on **Cancel** to cancel exportation.

WARNING : Freeze Objects can't be exported.

IMPORT IN XML FORMAT

In order to import DAS data from an XML import file (.exp extension) generated by DAS, the user must proceed as follows:

- Activate the import setup window () with the **Import** command of **File** menu.

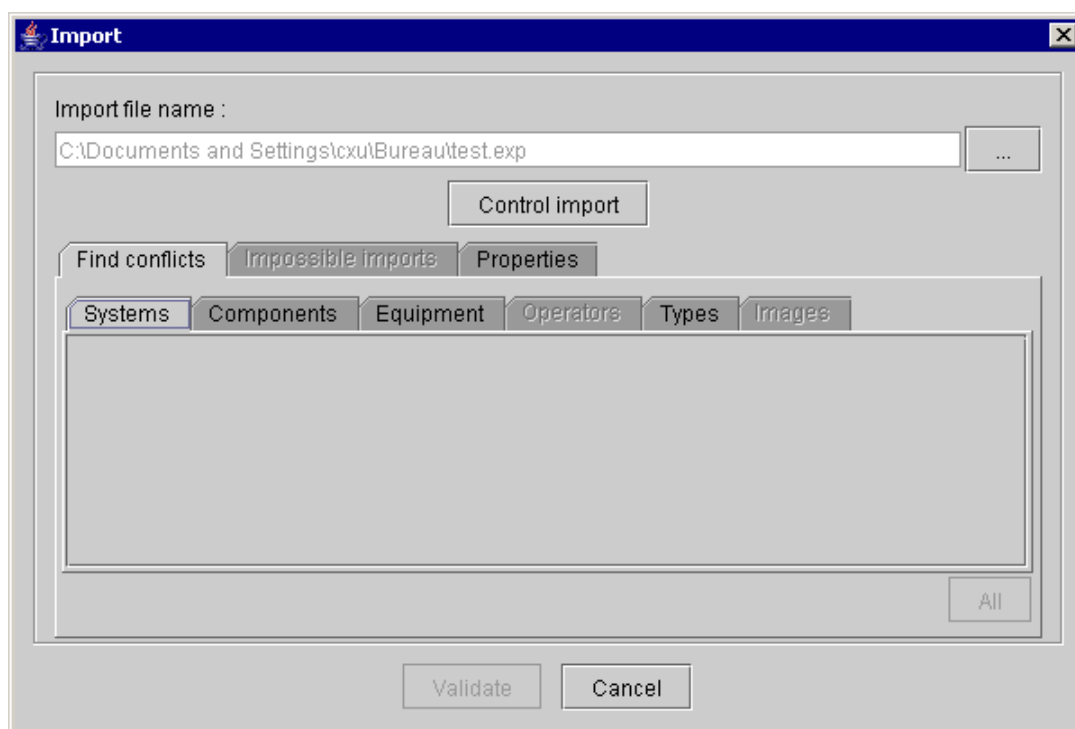



Figure 163 : XML Data Import

- Click on  button in *Import File Name Area*,
- Chose the export file (extension .exp) containing DAS data. The **Properties** tab allows visualization of properties associated to the imported data (creation date, comment, and owner).
- Click on **Check Import** to launch validity check. This check do the following operations:
 - * *XML Format validity Check*. Detected errors are displayed un “impossible imports” ta : data access issues (family or project reachable in read only mode, model or system version unreachable or reachable in read only mode)
Remark: only ascendant compatibility is ensured. (File exported with 3.1 version can’t be imported with previous version).
 - * *Detection of conflict between imported data and data in database*.
Remark: This difference detection is made essentially with the last modification date of each imported element compared to the one in the database.

Detected conflicts are displayed depending on their type in the following tabs: **Systems**, **Components**, **Equipments**, **Operators**, **Types**, and **Images**. By default, data in database are not updated during data importation.

Nevertheless, the user can specify, for each conflict, an update of the database with a click on



. To indicate this update, the lock icon is associated to the imported element (Figure 164). The **All** button allows update of all displayed conflicts in the tab.

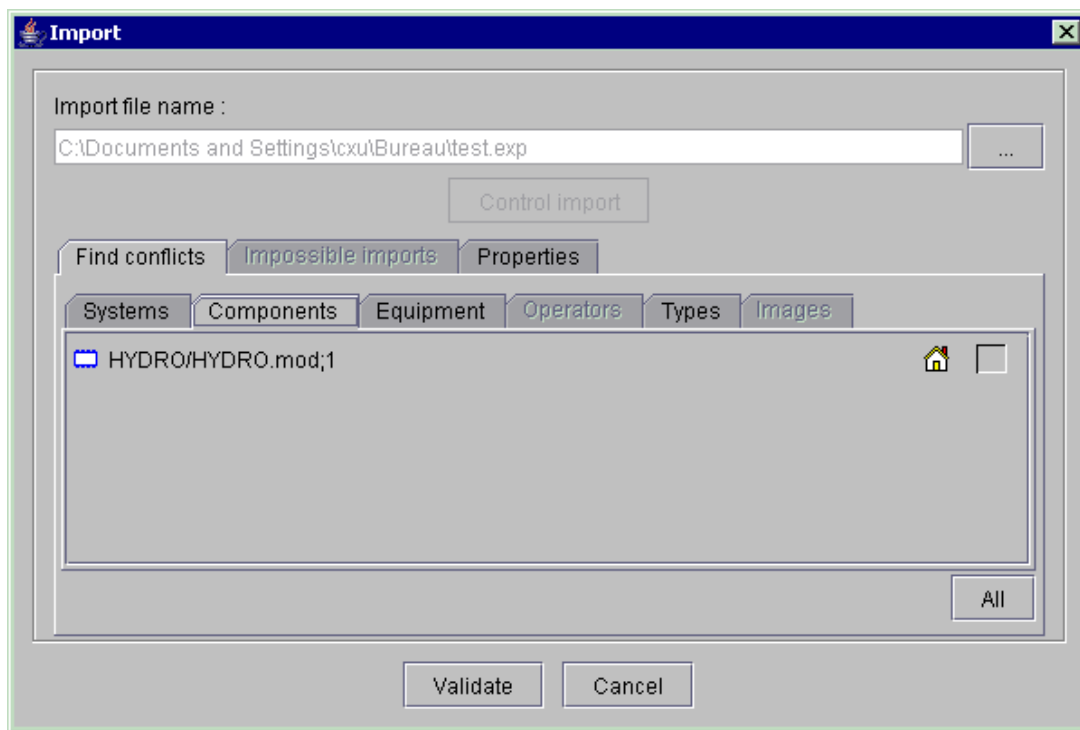


Figure 164 : XML Import – Detected conflict

- Click on **Validate** to import or cancel to cancel importation.

THE PLUGIN MANAGER

INTRODUCTION

The plugin manager allows the user to manage the compute plugins. The compute plugins are processing units taking in input a model generated in Altarica. According to the treatment made, the user will obtain on output event minimal cuts, event sequences, a step by step simulator by inference rules or event a Java step by step simulator.

A user can create new plugins and add them inside DAS application by using the plugin manager. This chapter does not show the plugin creation process. This theme is developed in details in another document.

From their use point of view, the user sees the plugins like treatment units contained in JAR files (in Java). A JAR file can contain several plugins. These files are read by the plugin manager in order to load the plugins into the application during its launch.

GENERAL DESCRIPTION OF THE PLUGIN MANAGER PANEL

The plugin manager window contains three tabs which are the following ones: **Plugins**, **Actions** and **Items** (Figure 165):

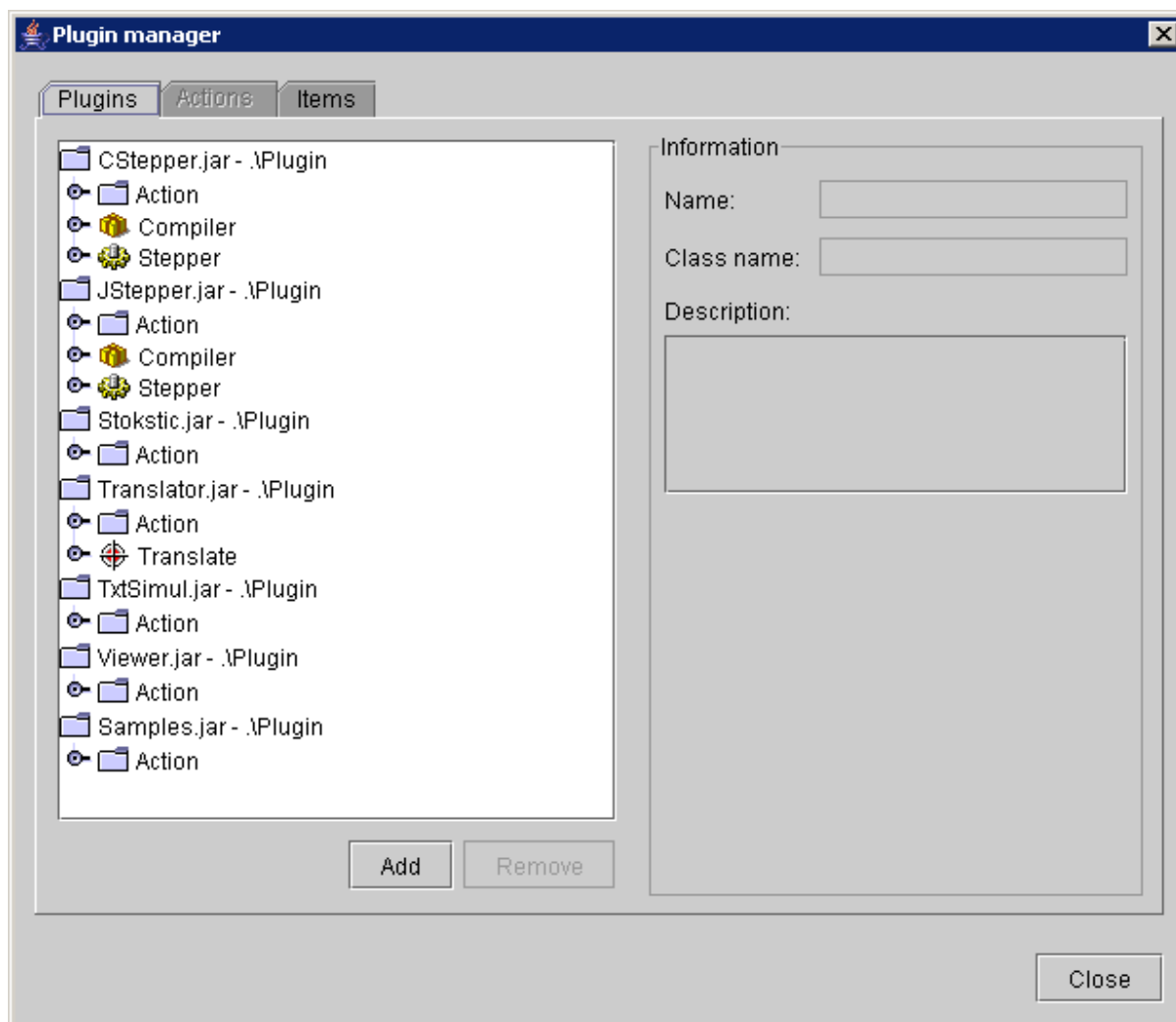


Figure 165 : Plugin manager

The **Plugins** tab allows the user to manage the plugins. The **Actions** tab allows adding actions associated to the plugins and the **Items** tab allows inserting inside the application user interface, plugin linked action launch point.

MANAGE PLUGINS

The **Plugins** tab allows the plugin management inside DAS application. Through this tab the user can add, remove plugin to/from the application and also view the plugin properties.

Plugin organization

Plugins are organized in a tree structure. The first level shows the different JAR files from which the plugins are loaded into the application. The plugin paths are expressed in relative from the location of the file named "plugins.xml" (Figure 166):

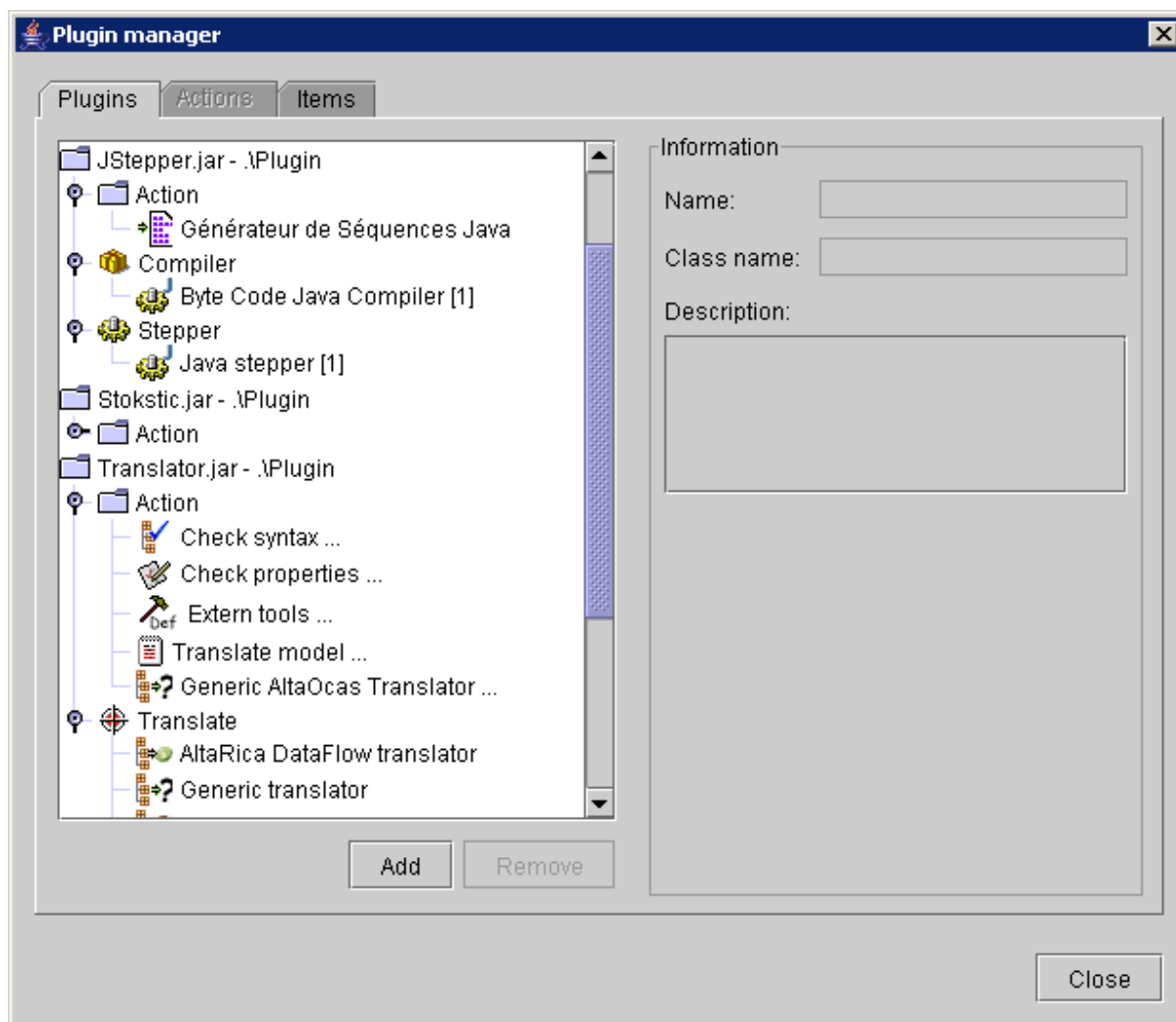


Figure 166 : Plugin manager – plugin tree structure

The second level contains **Action**, **Translate**, **Compiler** and/or **Stepper** nodes. The nodes allow classifying plugins into categories.

The **Action** nodes gather plugins that can be directly executed by the user if associated to an action and a launch point in the application user interface.

The **Translate** nodes gather plugins used in treatment flows and are dedicated to the translation of some languages into other languages.

The **Compiler** nodes gather plugins allowing compilation of models which accelerate its use by the simulation engine.

The **Stepper** nodes gather the plugins that implement simulation engines.

View plugin informations

In order to visualize plugin related information, the user selects a plugin in the tree structure. The information then appears in the right part of the tab (Figure 167):

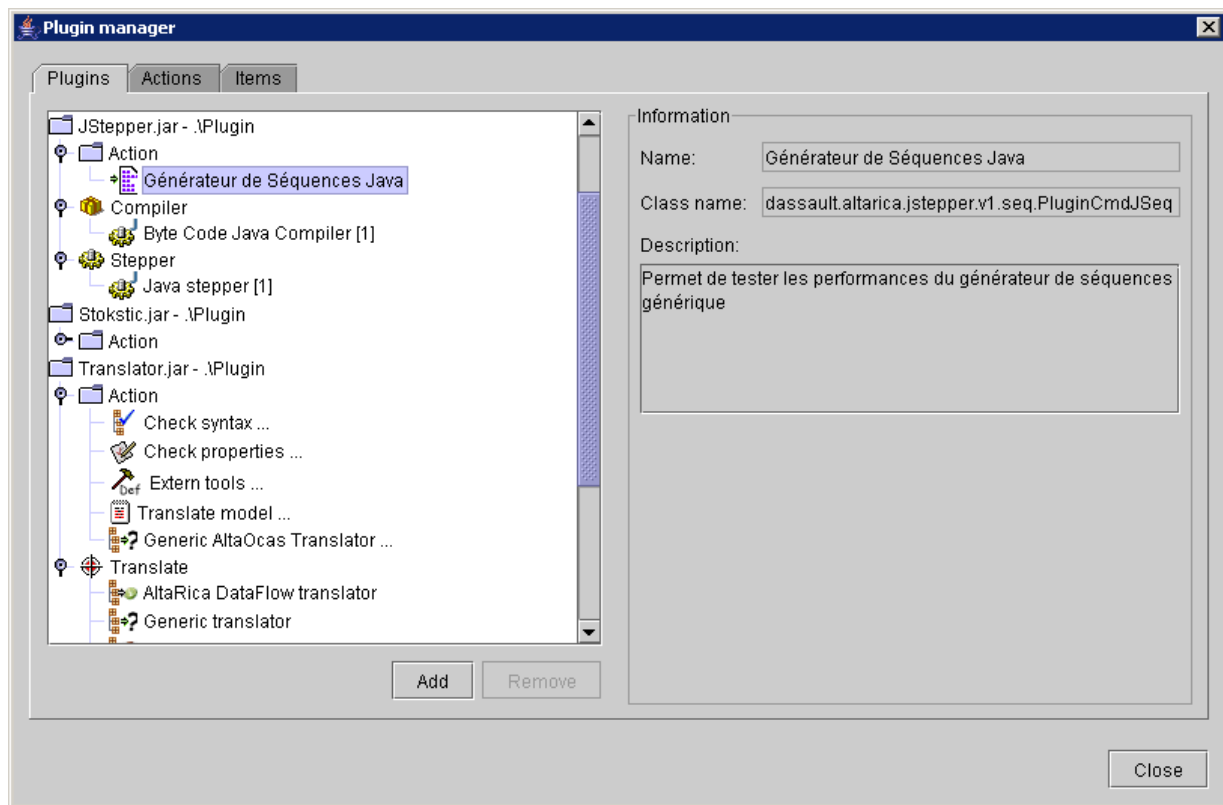


Figure 167 : Plugin manager – Plugin information area

A plugin has a name, a class name and a description. This information is accessible to the user but is not editable for write because it is to the plugin designer responsibility.

The class name is the complete name of the implementation Java class. It is the plugin entry point.

Add JAR files

Plugins are only accessible throughout JAR files. The user load JAR files containing plugins and not plugins directly.

To add JAR files containing plugins into the application, the user must proceed as follows:

- Click on the **Add** button; the following window then appears (Figure 168):

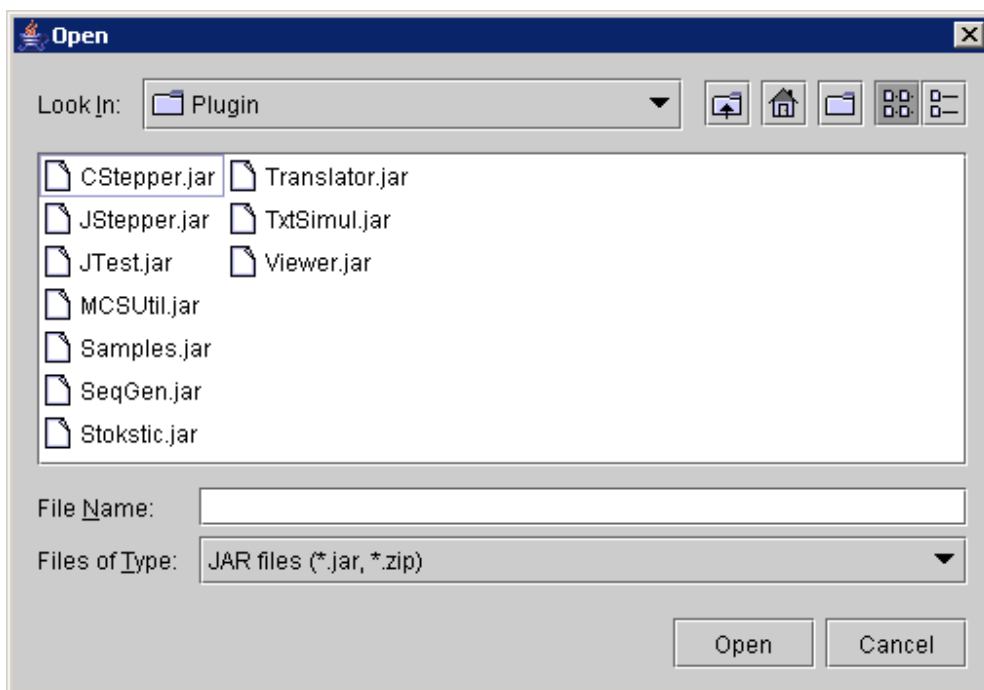


Figure 168 : Add plugins contained in JAR files

- Select the JAR files containing the plugins that are to be integrated into the application,
- Click on the **Open** button.

The plugins are registered in the application and then can be referenced by plugin actions.

Remove JAR files

To remove JAR files from the application, the user proceed as follows:

- Select JAR files nodes that are to be deleted,
- Click on the **Remove** button; the following confirmation message then is shown (Figure 169):

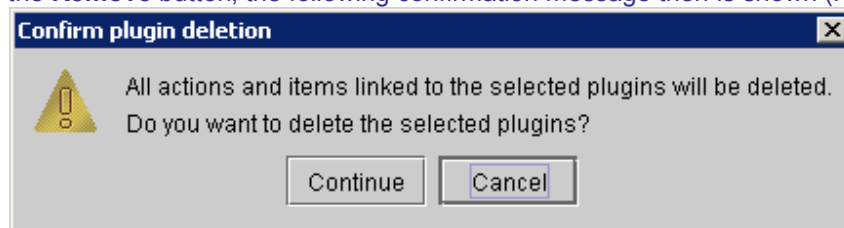


Figure 169 : JAR files deletion confirmation message

- Click on the **Continue** button to confirm the JAR files deletion or click on the **Cancel** button to cancel the action.

If the user clicks on the **Continue** button, the selected JAR files nodes and all their content are removed from the tree structure.

MANAGE ACTIONS

The actions are plugin instantiations in the sense that they allow using the plugins with distinct parameters. The actions only apply to the **Action** plugins and not to the others categories. When the user select a plugin of **Translate**, **Compiler** or **Stepper** type, the **Action** tab become inaccessible.

The **Action** tab appears as follows (Figure 170):

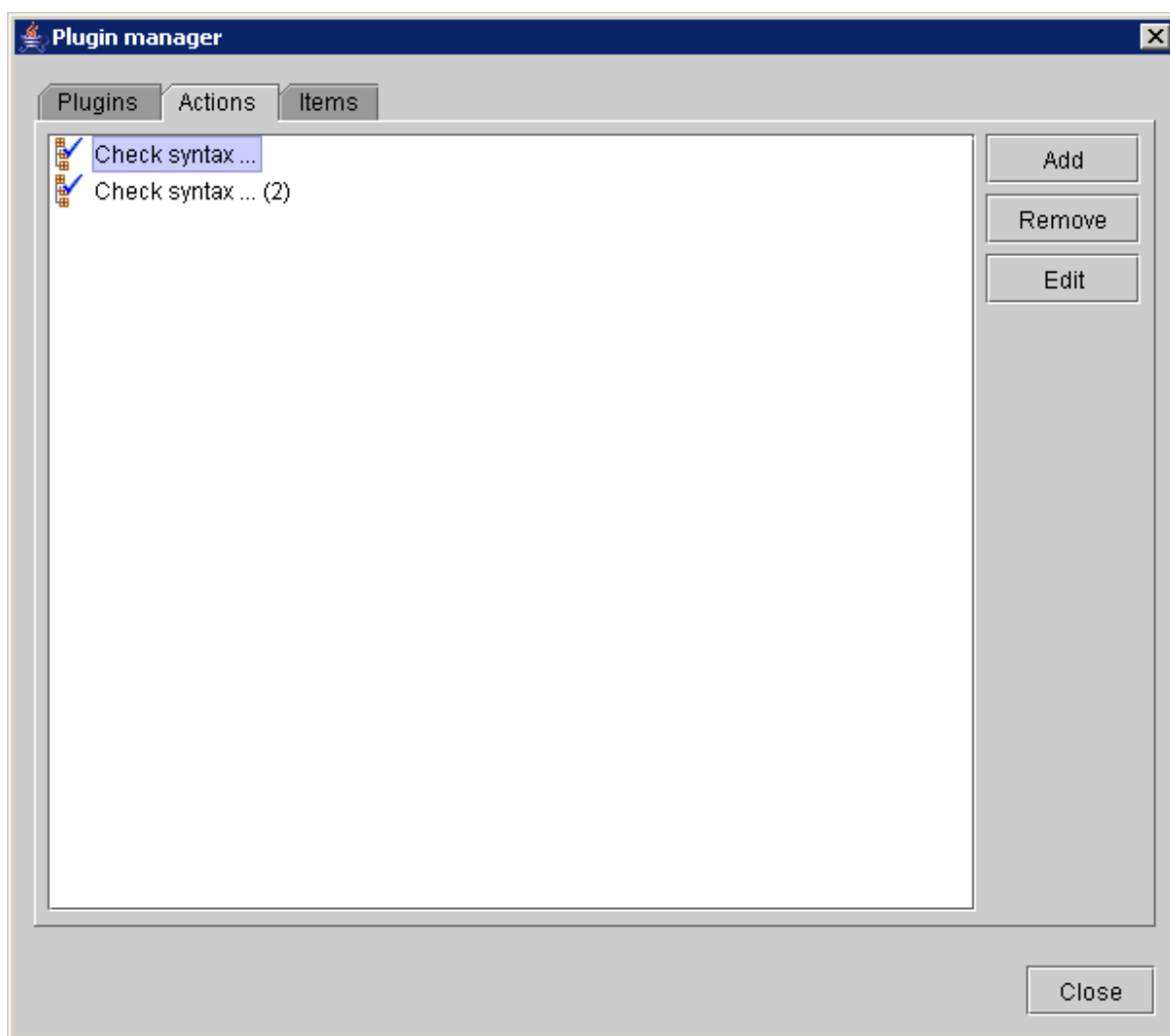


Figure 170 : Plugin manager –Actions tab

Add plugin action

To add a plugin action, the user proceeds as follows:

- Select if necessary the **Plugin** tab and then, in the tree structure, the plugin of **Action** type on which the new action is to be created,
- Select the **Action** tab,
- Click on the **Add** button; the following window then appears (Figure 171):

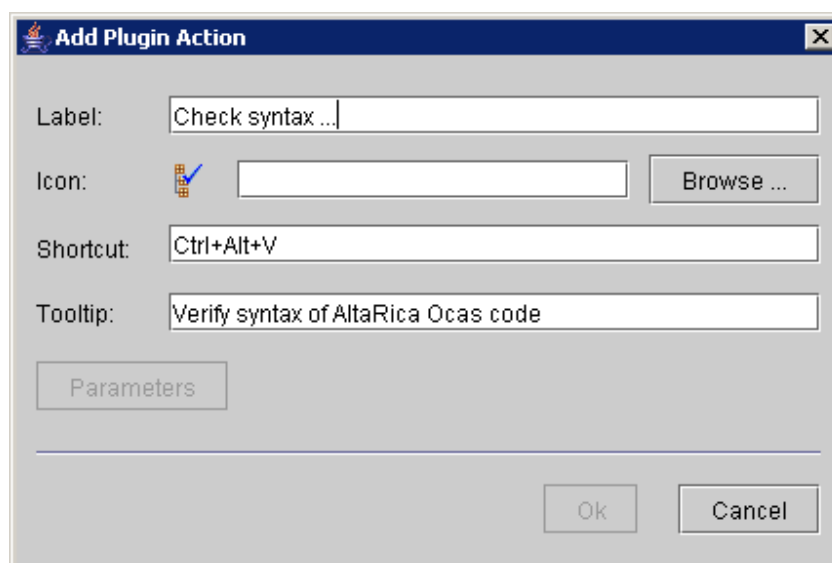


Figure 171 : Add a plugin action

- Fill the dialog box fields in,
- Click on the **Ok** button to confirm the plugin action creation or on the **Cancel** button to do nothing.

If the user clicks on the **Ok** button, the new action is added to the already existing actions list.

N.B.: A default label fill the corresponding field when the **Add a plugin action** dialog box is shown to the user. This label is provided by the plugin. It is the same for the **Icon**, **Shortcut** and **Tooltip text** fields.

Two different actions associated to the same plugin can not have the same label. When an already existing label is entered in the corresponding field, the **Ok** button is then disabled.

The **Parameters** button give write access to the plugin parameter list when the plugin has parameters. These parameters and their viewing are provided by the plugin designer.

Remove plugin action

To remove plugin actions, the user proceed as follows:

- Select the **Actions** tab if needed,
- Select the action nodes ,
- Click on the **Remove** button; the following action deletion confirmation message then appears (Figure 172):

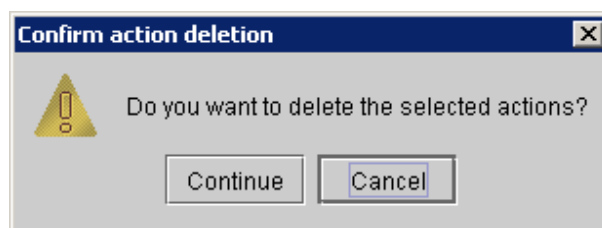


Figure 172 : Plugin deletion confirmation

- Click on the **Continue** button to confirm the actions deletion or on the **Cancel** button to do nothing.

If the user clicks on the **Continue** button, the selected actions then disappear from the action list.

Edit plugin action

To edit a plugin action, the user proceeds as follows:

- Select if necessary the **Action** tab and the action to edit,
- Click on the **Edit** button; the following window then appears (Figure 173):

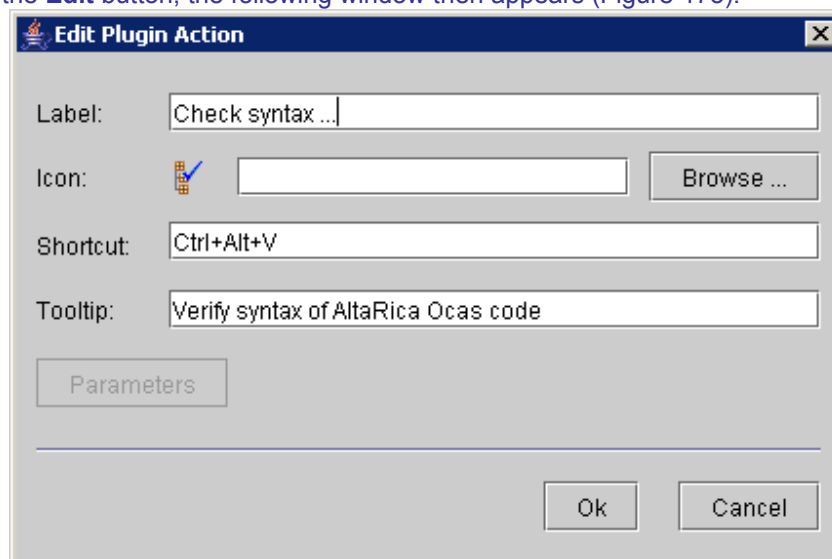


Figure 173 : Edit a plugin action

- Modify the fields in the dialog box,
- Click on the **Ok** button to confirm the modification or on the **Cancel** button to cancel the operation.

MANAGE PLUGIN ITEMS

Plugin items overview

The **Items** tab allows the plugin item management. Note that the plugin items are user interface graphical objects like buttons or menu items that provide a launch point for the **Action** typed plugins. By the other, a plugin item action is an action defined or predefined in the **Action** tab. When created on a tool bar, the plugin item is a button and created on a menu bar, the plugin item is a menu item.

The Items tab is shown bellow (Figure 174). The **Items** tab is divided in two parts: a first part that shows all the application menu items and toolbar buttons in a tree structure view and a second one that gather the commands accessible on the plugin items.

By navigating in the tree structure representation, the user can for instance reach a sub-menu in order to add a plugin launch point. The user has also the possibility to create other type of objects like separators or new menu in existing ones. All the supplementary graphic objects appear in the tree structure with bold text font.

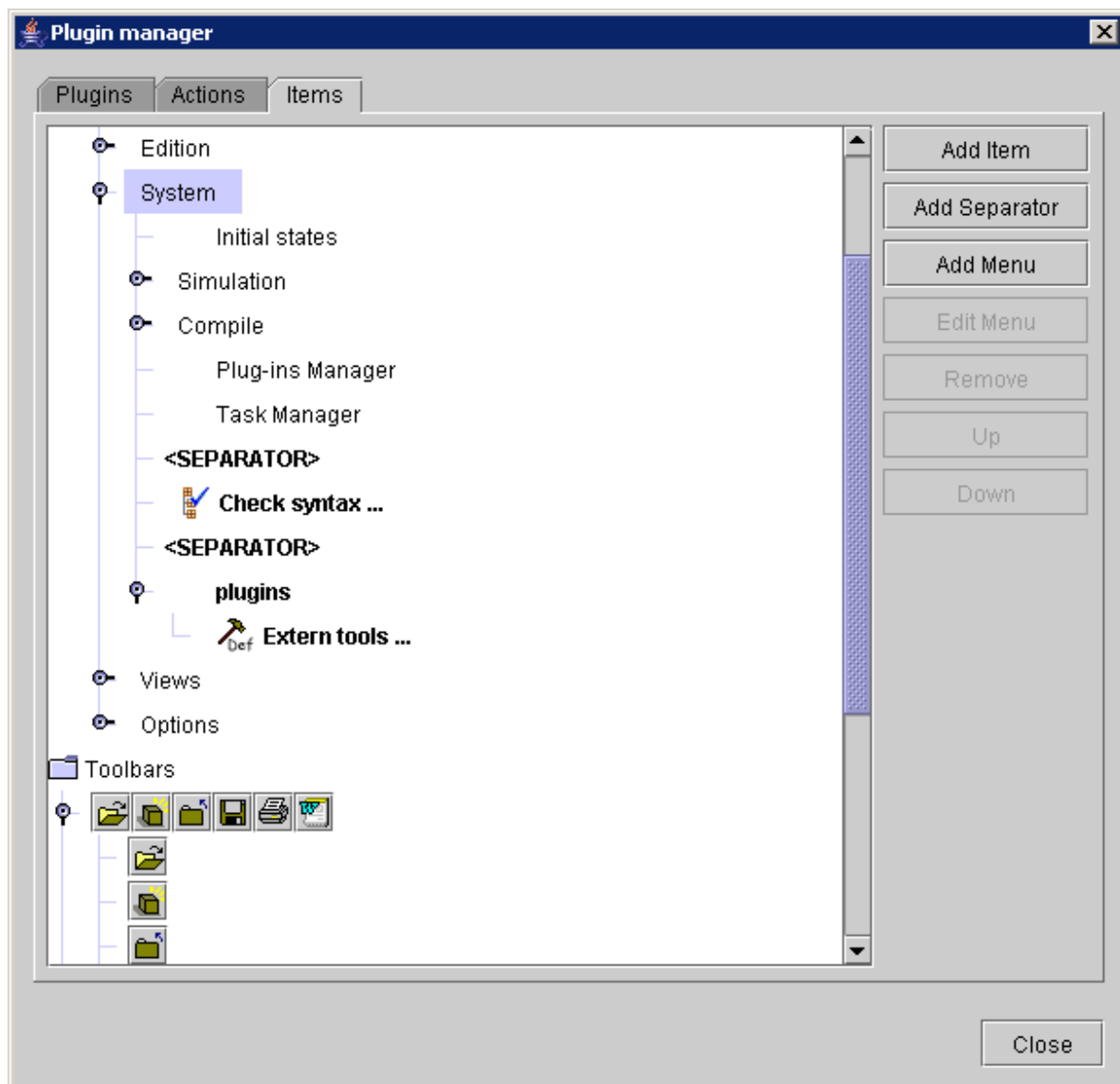


Figure 174 : Plugin manager –Items tab

Add plugin item

To add a plugin item, the user proceeds as follows:

- Select the **Plugins** tab then a plugin of Action type if necessary,
- Select the **Action** tab then an action in the list if necessary,
- Select the **Items** tab if necessary,
- Navigate in the tree structure until the insertion location,
- Click on the **Add Item** button.

The plugin item is then created and added to the tree structure and in the application menubar or toolbars. When the selected object for the item insertion is a menu, the item is inserted at the last position in that menu, when it is a menu item the insertion is done at the preceding position of that menu item.

Add a separator

To add a separator, the user proceeds as follows:

- Select the **Items** tab if necessary,
- Navigate in the tree structure until the insertion location,
- Click on the **Add Separator** button.

The separator is then created and inserted in the selected location. When the selected object for the insertion is a menu, the separator is inserted at the last position in that menu, when it is a menu item the insertion is done at the preceding position of that menu item. Separators created by the user have all the same name:

<SEPARATOR>

Add customized menu

To add a menu, the user proceeds as follows:

- Select the **Items** tab if needed,
- Navigate in the tree structure until the insertion location,
- Click on the **Add Menu** button, the following dialog box then appears (Figure 175):

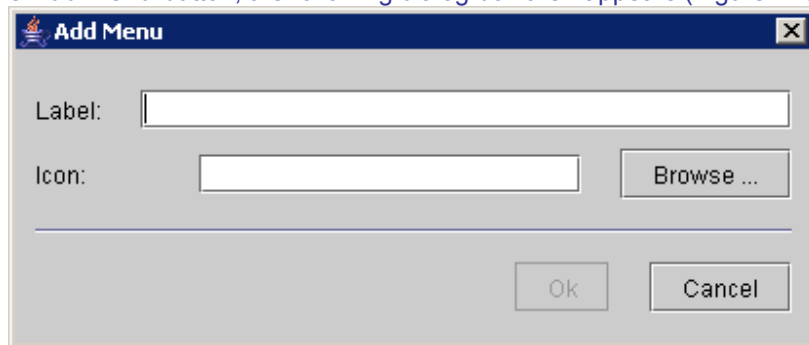


Figure 175 : Add menu

- Fill the label field in and if needed an icon path in the corresponding field,
- Click on the **Ok** button to validate the creation or on the **Cancel** button to cancel the operation.

If the user clicks on the **Ok** button, the newly created menu is added at the selected location. When the selected object for the insertion is a menu, the newly created menu is inserted at the last position in that menu, when it is a menu item the insertion is done at the preceding position of that menu item.

Edit customized menu

To edit a menu, the user proceeds as follows:

- Select the **Items** tab if needed,
- Select the menu to edit in the tree structure,
- Click on the **Edit Menu** button, the following dialog box then appears (Figure 176):

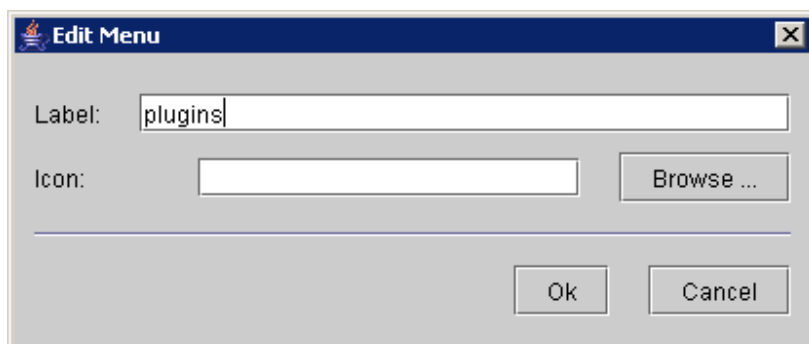


Figure 176 : Edit menu

- Modify the fields,
- Click on the **Ok** button to validate the modification or on the **Cancel** button to cancel the operation.

If the user clicks on the **Ok** button, the menu is then modified.

Remove graphic item

Only the graphic elements added by the user are removable. It can be plugin item, supplementary separator or supplementary menus.

Be careful: when the user remove a menu, all the graphic elements under this menu are also removed.

To remove graphic elements, the user proceeds as follows:

- Select the **Items** tab if needed,
- Select the graphic elements in the tree structure,
- Click on the **Remove** button; the following graphic element deletion confirmation message then appears (Figure 177):

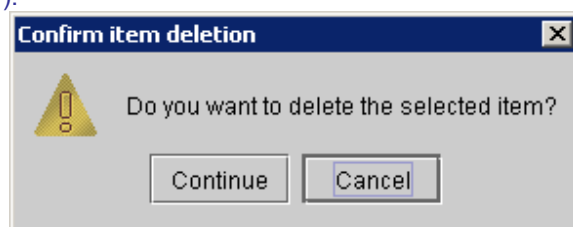


Figure 177 : Graphic element deletion confirmation

- Click on the **Continue** button to confirm the raphic element deletion or on the **Cancel** button to do nothing.

If the user clicks on the **Continue** button, the selected graphic elements then disappear from the tree structure and from the application user interface (menu bar and toolbar).

ADJUSTMENT OF PREFERENCES

GENERAL

A DAS base comprises three directories:

- ♦ Icons: which contains all families and icons used by the components,
- ♦ Models: which contains all objects of the DAS library (component, equipment, operator, type),
- ♦ Projects: which contains all systems.

Each user can configure his work environment. Thus, the DAS tool provides a window for configuring the preferences which are available via the **Options – Preferences** menu.

This menu gives access to a window containing eight rubrics:

- ♦ Environment,
- ♦ Desktop,
- ♦ Tools bars,
- ♦ Edition,
- ♦ Input/Output,
- ♦ Simulator,
- ♦ Fault Tree generation,
- ♦ Sequence generation,

Remark: Only adjustments performed in the **Desktop**, **Tools bars** and **Options** tabs are saved in the “preferences.dat” file. The **Save** button is always inactive if the –DprefPath=« chemin » option is not present in the DAS tool “*.bat” launching file

ENVIRONMENT RUBRIC

N.B: Sub-rubric **Memory** (Figure 178) is for maintenance only.

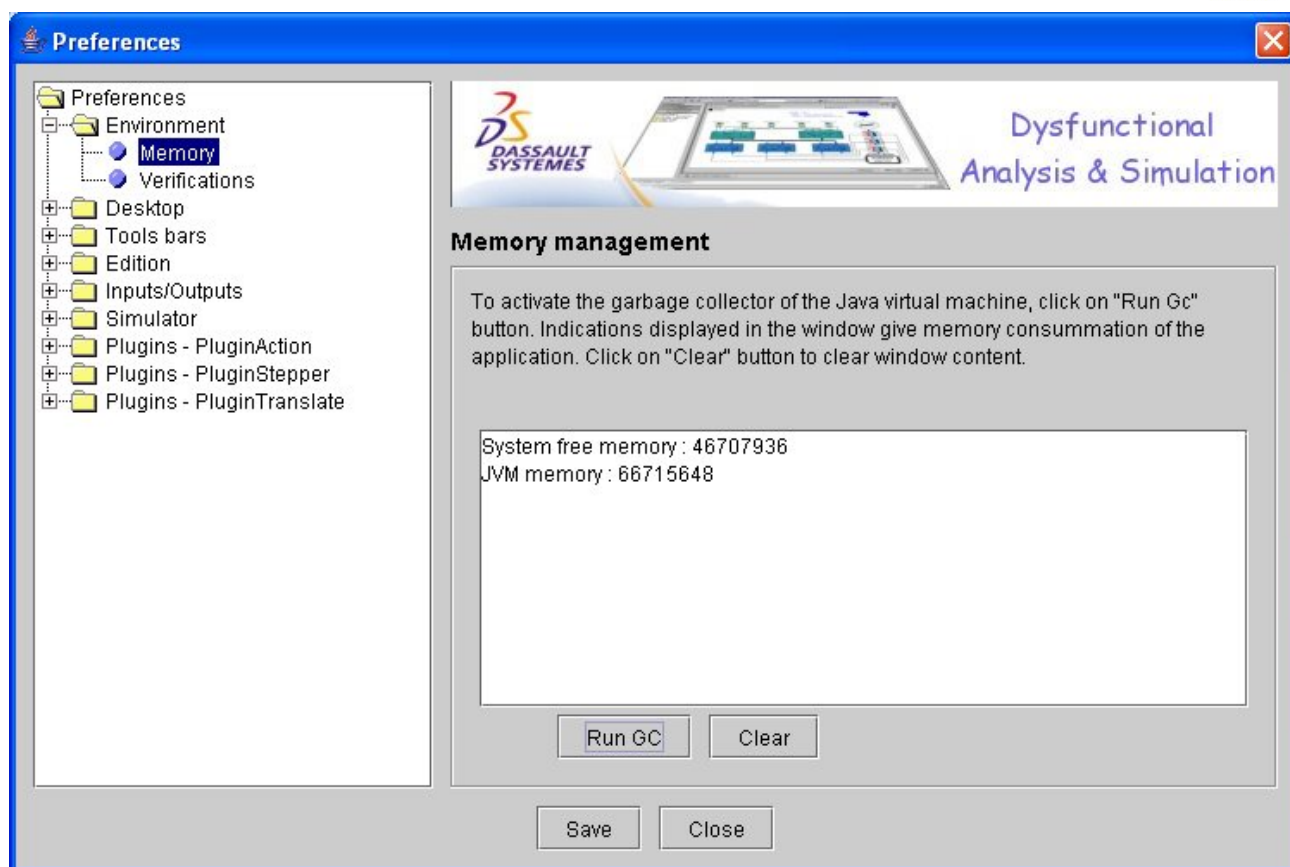


Figure 178 : Memory management

- ◆ Click on **Run GC** to run the Garbage Collector of Java Virtual Machine. Otherwise click on **Clear**.
- ◆ Click on **Close** to close this window.

Remark: The **Save** button as no effect in this rubric.

When the application is run on file data base, this data base can be changed with the File data base modification rubric (Figure 179). For that specify the path to the new data base and click on "Apply" button.

The Verification sub-rubric allows the user to choose the syntactic and consistency checking

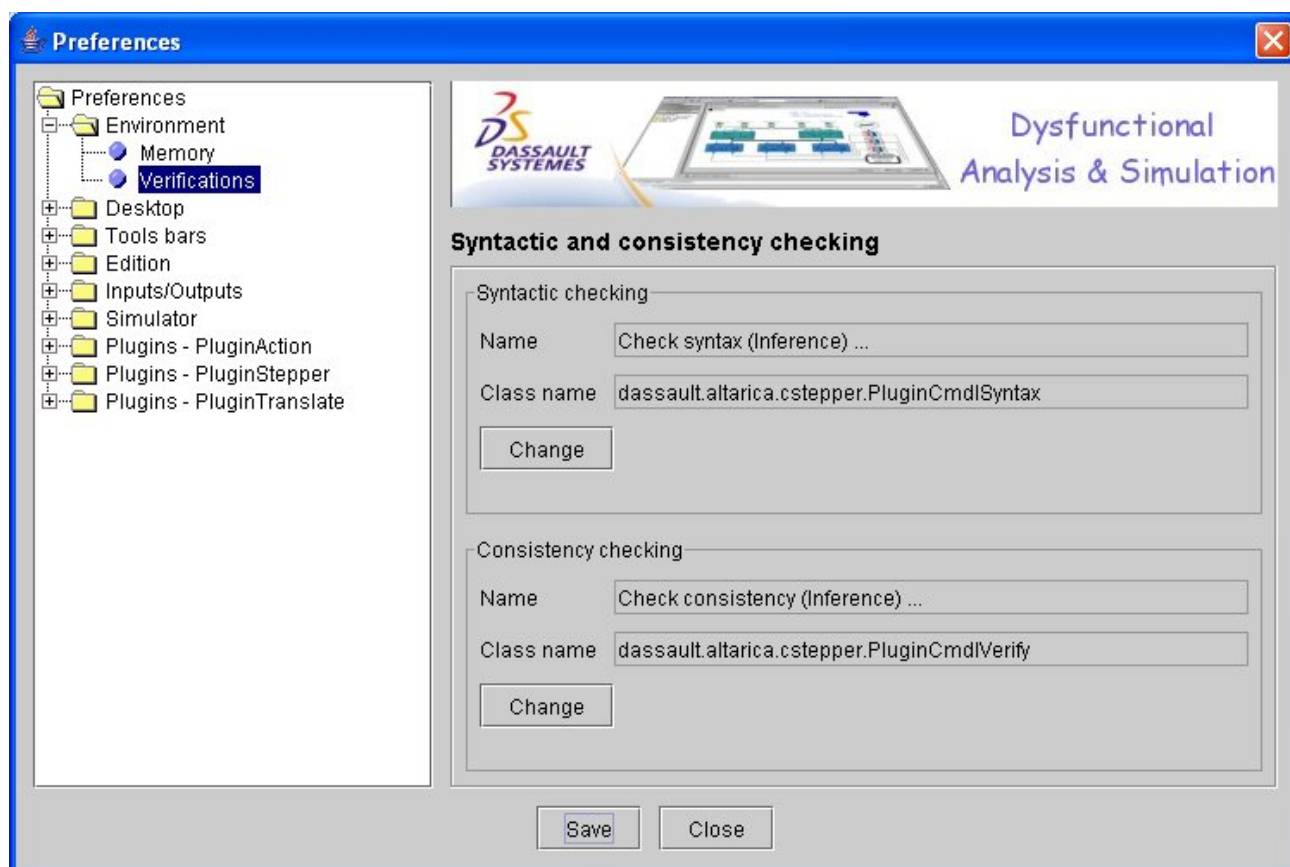


Figure 179 : Syntactic and consistency checking

DESKTOP RUBRIC

The **Screen background** sub-rubric (Figure 180) allows background choice:

- ♦ Selection of Background characteristics:
 - * *Colour* (of desktop): this colour corresponds to the background colour of the work area if the *Image* field is empty (Red, Green and Blue ratios),
 - * *Image*: use the selector to choose the type (gouttes, cuir, crépi, nuages, and bleu). The pathname for a "*.gif" or "*.jpg" file containing another pattern can also be indicated. If the *Image* field is not empty, this selection overrides the *Colour* field selection.
- ♦ Tick Mosaic (for a mosaic display).
- ♦ Click on the Apply button.
- ♦ Click on the Save and Close buttons.

Remark: The **Save** button must be used only if the modifications are to be saved in the "preferences.dat" file.

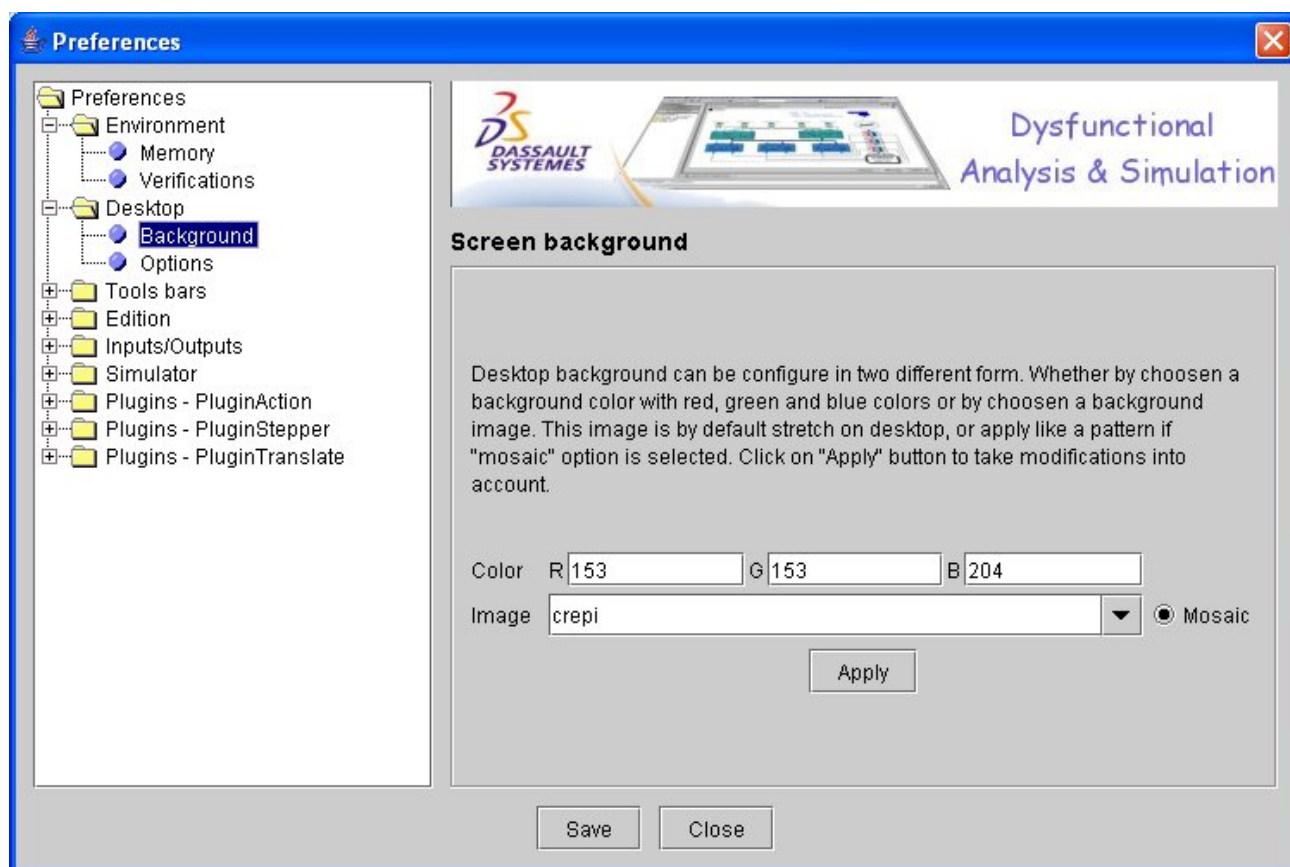


Figure 180 : Screen background

The **Desktop Options** sub-rubric (Figure 181) allows background choice:

- ◆ Display time and date,
- ◆ Display tools bars,
- ◆ Display task bar,

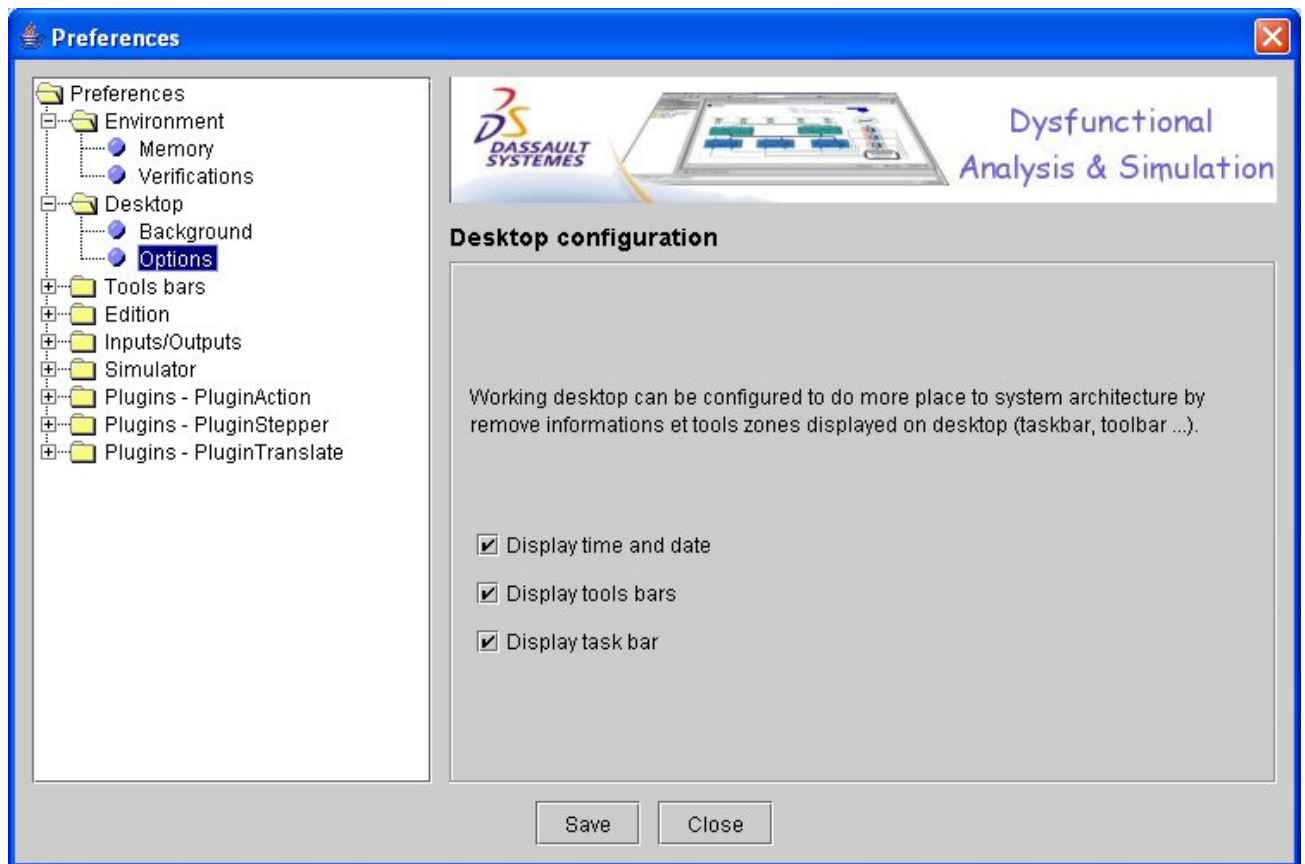


Figure 181 : Desktop options tab

TOOLS BARS RUBRIC

The **Tools bars** tab (Figure 182) allows customization of the tools bars with the following bars:

- ♦ Standard,
- ♦ Edition,
- ♦ Design,
- ♦ Links,
- ♦ Alignment,
- ♦ Navigation,
- ♦ Simulation.

Validate by clicking on the **Save** and then **Close** buttons.

Remark 1: The **Save** button must be used only if the modifications are to be saved in the "preferences.dat" file.

Remark 2: If the number of tools bars exceeds the window width, buttons located at both ends enable access to the tools bars which are not displayed on the screen.

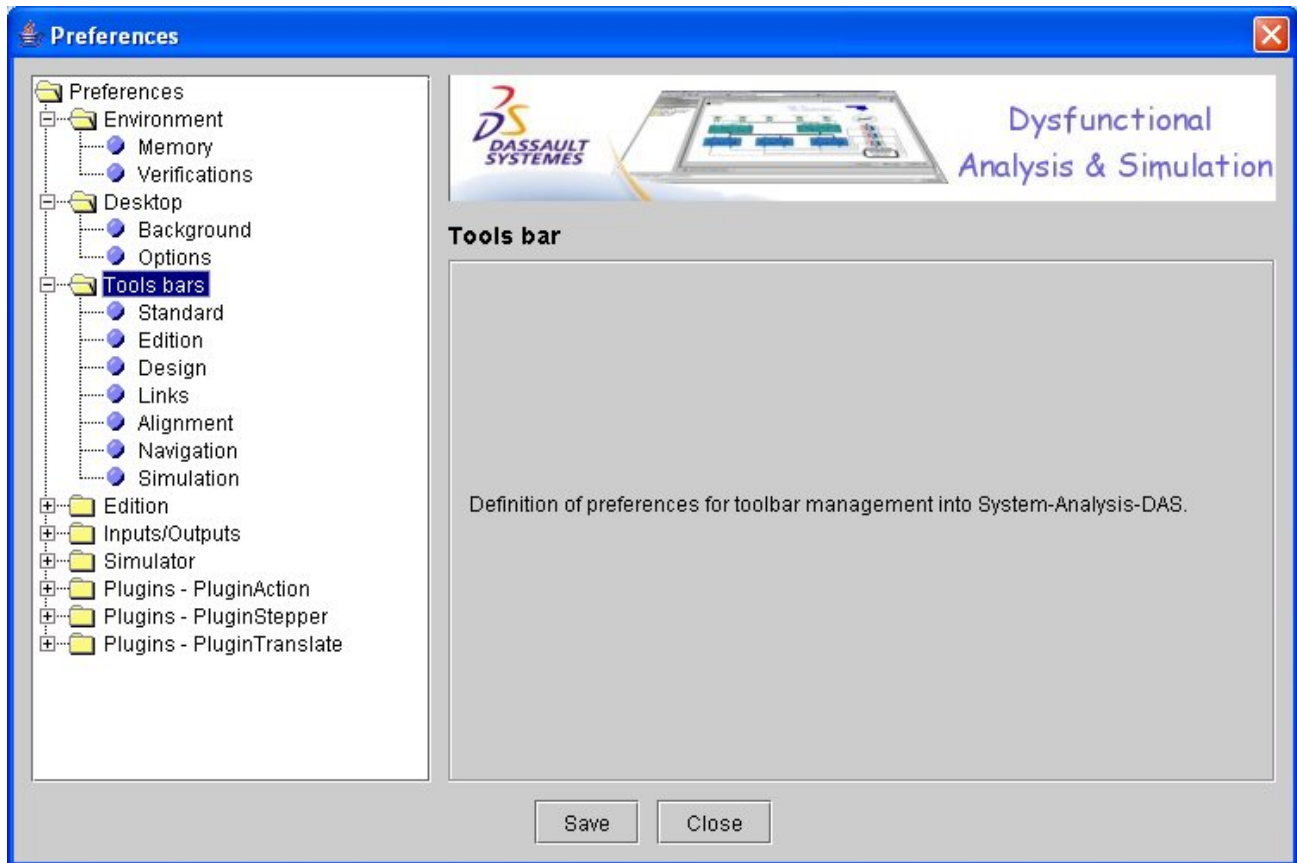


Figure 182 : Tools bars tab

EDITION RUBRIC

Font sub-rubric

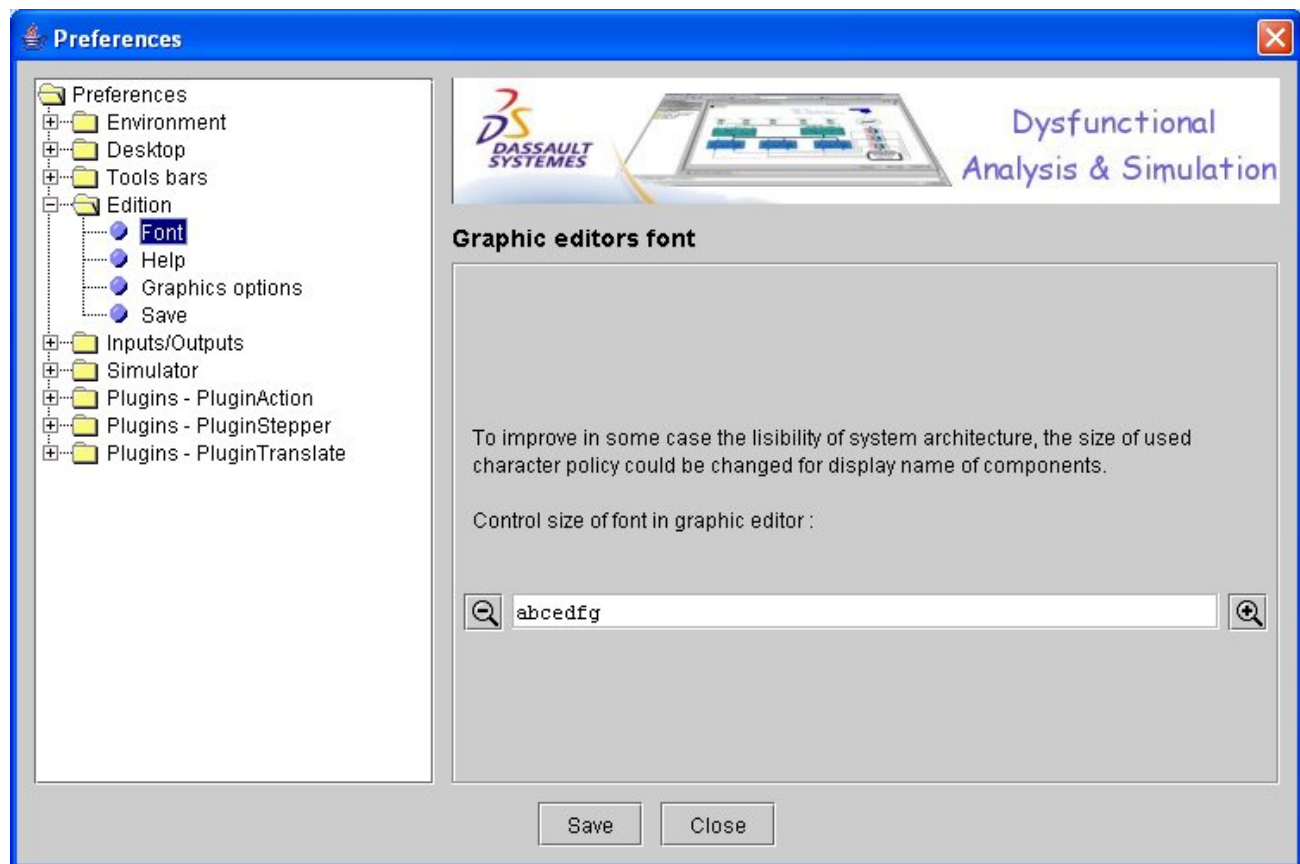


Figure 183 : Graphic editors font

In the Adjustment of font size in architecture view field, type any font size and adjust this size by means of the  and  icons.

Help sub-rubric

DAS workshop have an edition help for writing Altarica code by given, during data acquisition, , the list of constituent which are present at corresponding level.

Check “activate edition helper” to activate it.

Graphics options sub-rubric

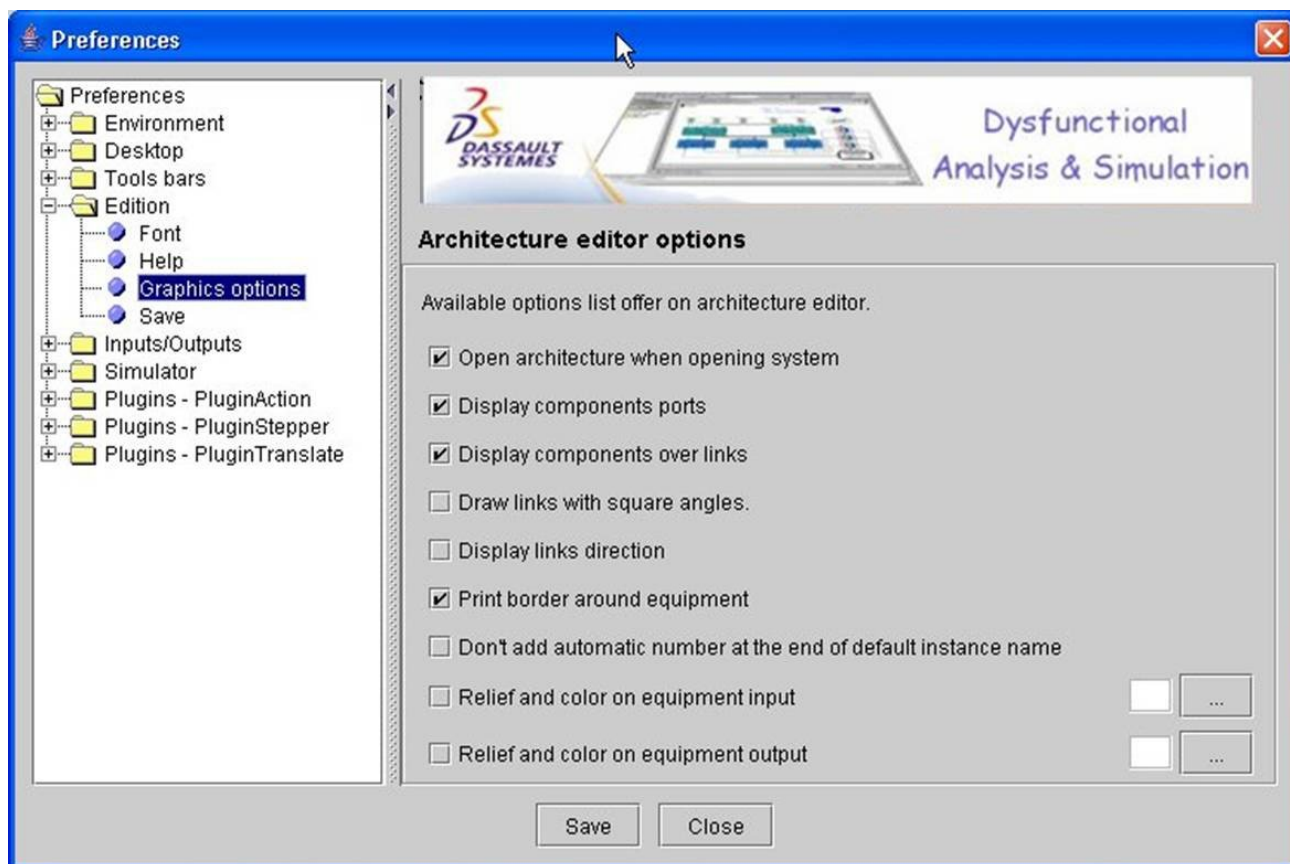


Figure 184: Graphics options sub-rubric

The **Graphics options** sub-rubric (Figure 184) allows customization of the DAS tool with the following options:

- ◆ Open architecture when opening project.
- ◆ Display components ports
- ◆ Display components over links
- ◆ Draw links with square angles
- ◆ Display links direction
- ◆ Print border around equipment.
- ◆ Don't add automatic number at the end of default instance name
- ◆ Relief and color on equipment input
- ◆ Relief and color on equipment output

Save sub-rubric

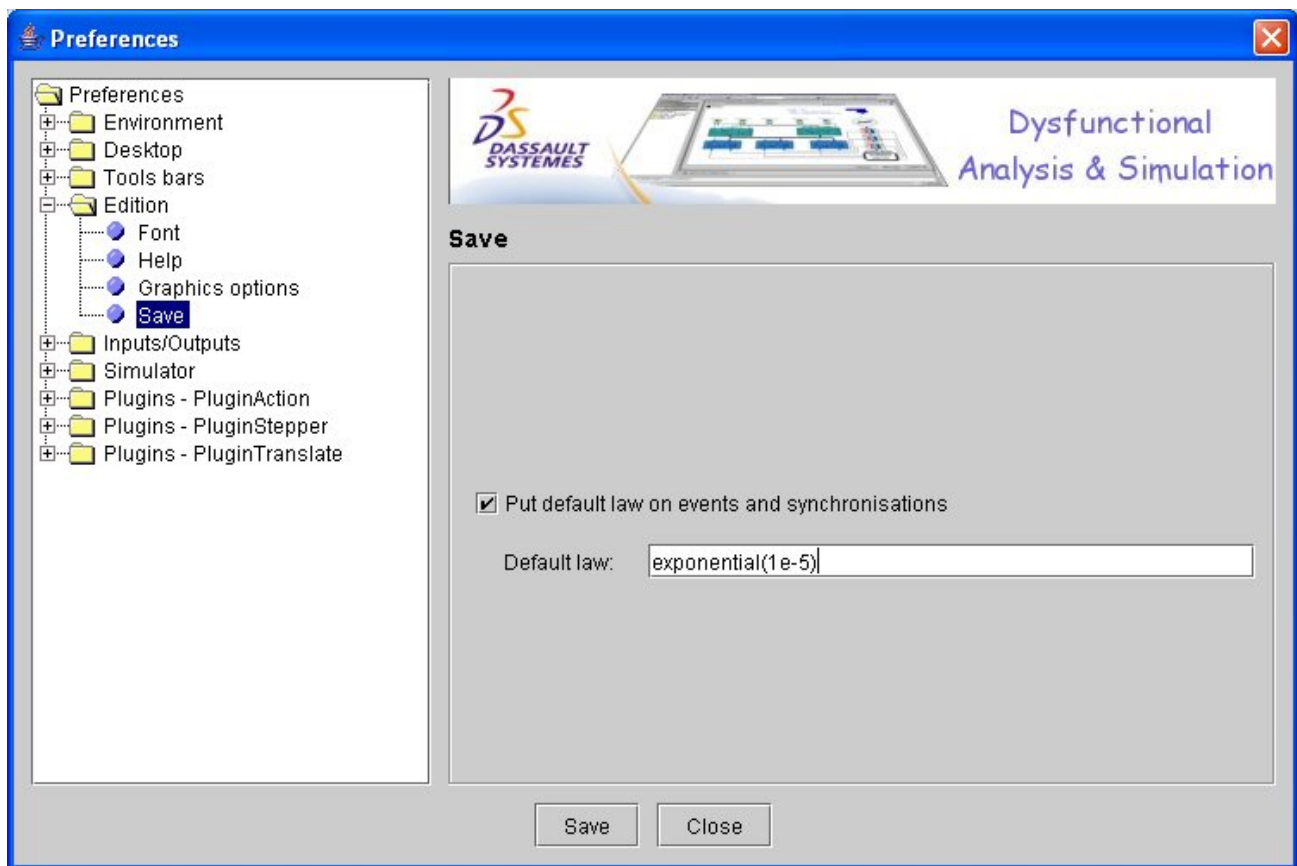


Figure 185 : Save

Check the box "Put default law on events and synchronizations" to save events and synchronizations with a default law. The default law is only put on events and synchronizations that have no law. Use the "default law" area to define this law.

INPUTS/OUTPUTS RUBRIC

The Inputs/Outputs rubric contains 1 sub-rubrics, printing logo (Figure 186)

In the File name field (Figure 186), type the pathname to a logo which will be printed in the page top left corner (the file format containing the logo is of the “*.gif” or “*.jpg” type) or use the Select button to indicate the file containing the logo.

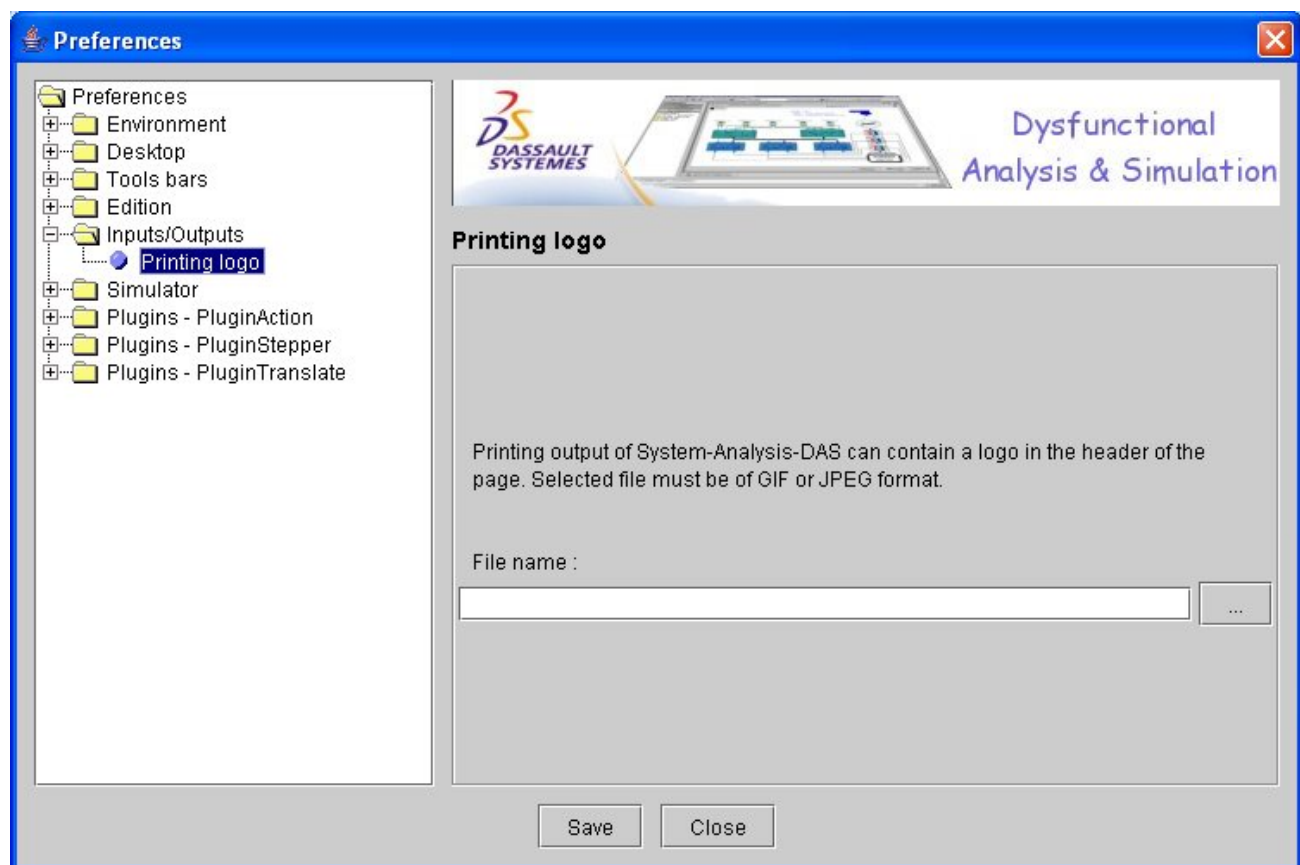


Figure 186 : Printing Logo

To use architectures build with version 3.0 of DAS workshop, it is necessary to do an export of all data. This export must be done with Altarica converter to convert Altarica code. Then make an import of converted data to update data base. If data have already been converted or if they have been build with 3.1 version, it is not necessary to use converter during export.

Validate by clicking on the **Save** and **Close** buttons.

Remark: The **Save** button must be used only if the modifications are to be saved in the “preferences.dat” file.

SIMULATOR RUBRIC

The Simulator rubric allows choosing simulator options.

Simulator behaviour depends on options that can be activated or not (Figure 187):

- Trigger Automatically event and change state off
- Use CCF: If the case is unchecked, common cause failure synchronisations that are defined at the system or equipment level won't be taken into account during fault tree generation and during simulation. If checked CCF will be taken into account.
- Simulate system loop: if unchecked, variable belonging to a loop won't be evaluated during simulation.
- Trigger automatically instantaneous transition off: if checked, instantaneous events having a behaviour law equals to Dirac(0) will be triggered automatically.
- Number of automatic execution: to avoid endless simulation due to automatic triggering of instantaneous transition, user must specify the maximum number of automatic.

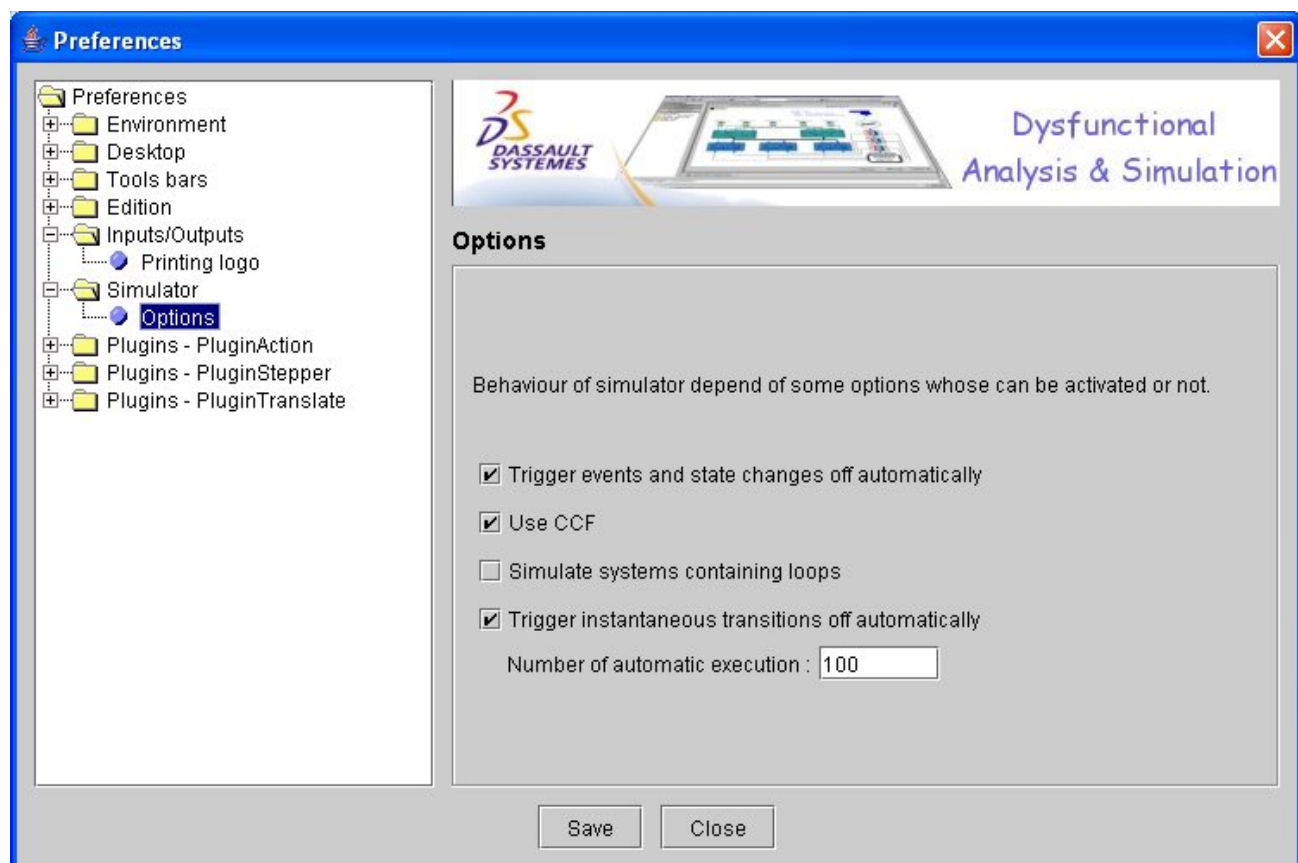


Figure 187 : Simulator Options

PLUGIN OPTIONS

The **Plugins – PluginAction** and **Plugins - PluginTranslate** rubrics (Figure 188) allows to choose options concerning the plugins.

For detailed explanations about these options, please see the plugin document annexes.

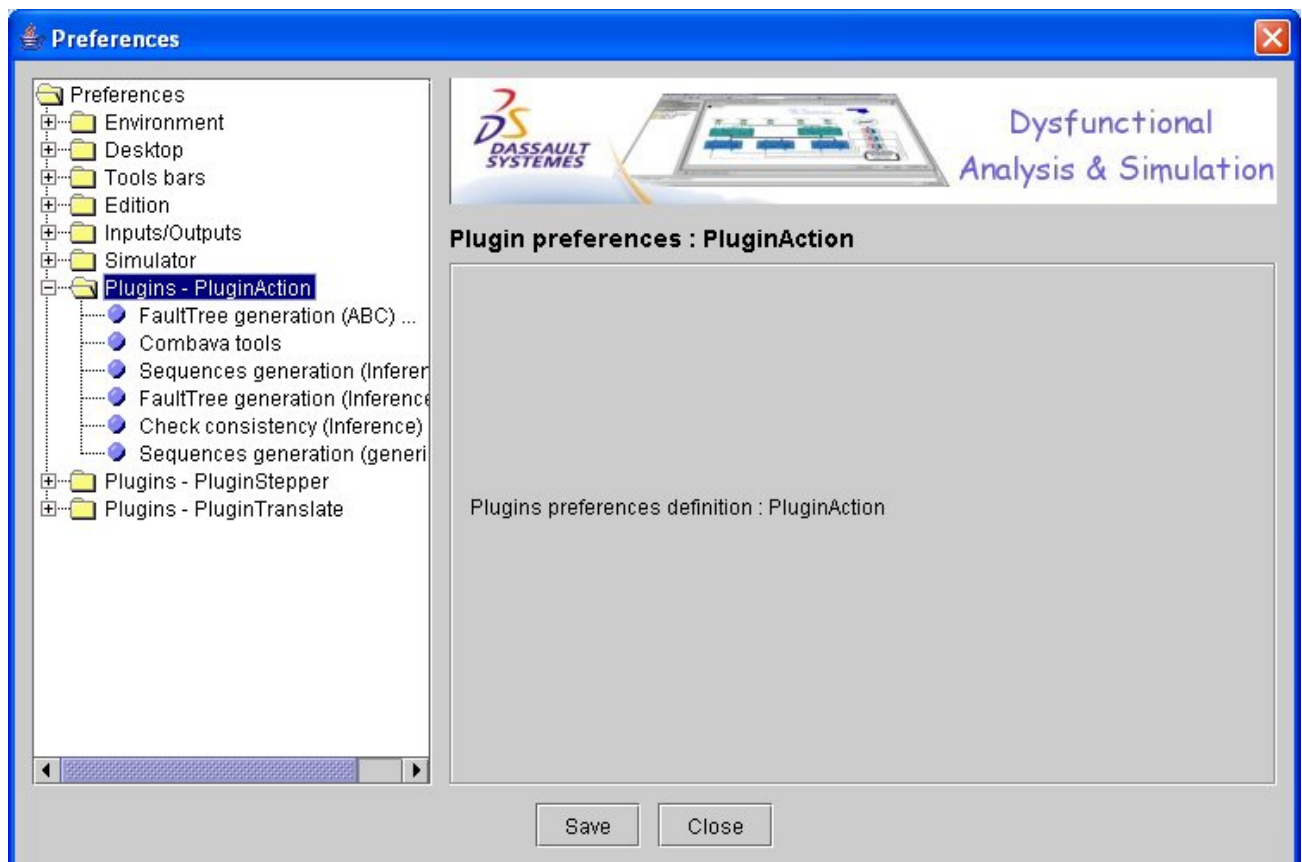


Figure 188 : Plugin option rubrics

QUIT DAS TOOL

Quit the DAS tool as follows:

- Either selects the **File – Quit** menu.
- Or type **CTRL+Q** on the keyboard.

If no system has been modified, the **Quit** window (Figure 189) is displayed.

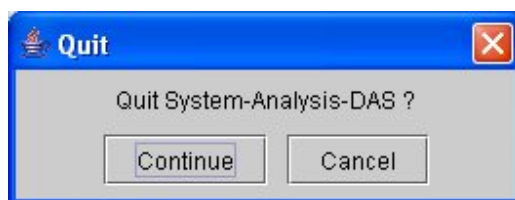


Figure 189 : Quit window

- Click on the **Continue** button, or otherwise on the **Cancel** button.
- The **MSDOS JAVA** window is closed.
- If one (or several) model(s) has (have) been modified, the following window is displayed (Figure 190):

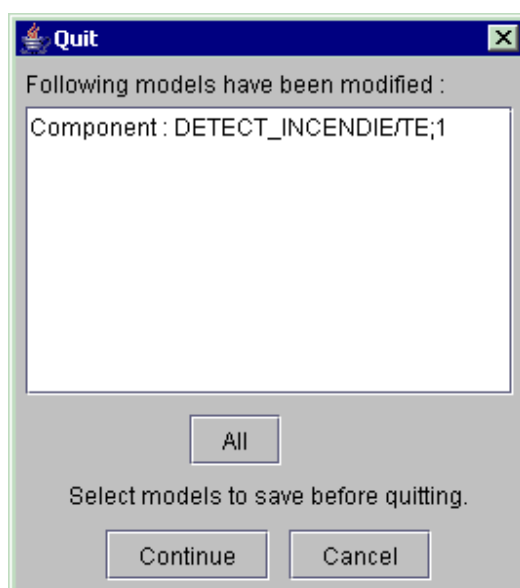


Figure 190 : Saving of modified model (s)

If one (or several) system(s) has (have) been modified, the following window is displayed (Figure 191).

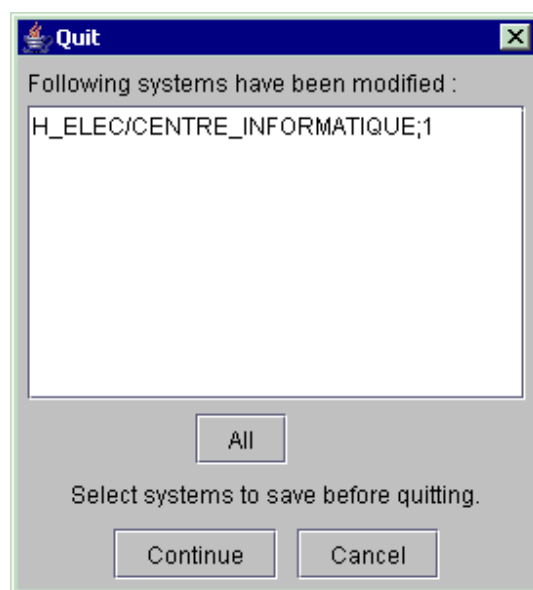


Figure 191 : Saving of systems

- Click on the **All** button; all the modified systems in the list appear in grey colour or select the system(s) which must be saved.
- Click on the **Continue**, otherwise on the **Cancel** button to remain in the DAS tool.
- The **MSDOS JAVA** window is closed.

GLOSSARY OF TERMS

Assertion (assert)	Altarica code allowing variation of the output flow variables of components
Base	<p>It is made up of three directories (icons, models and project) and contains all the systems, equipment, components, operators and types of the DAS tool library.</p> <p>N.B: There are systems which do not belong to the library but which are built up with the use of the library.</p>
Boolean constant	It is one of the two following keywords: true or false.
Component	It is an elementary behavioural object which can be represented graphically. It is a component type model.
Enumerate constant	It is an identifier previously declared in an enumeration field.
Enumerate type	It consists of a single variable which can take several values, e.g. power supply: {null, nominal}
Equipment	It is a hierarchical behavioural object which can be represented graphically. It is an equipment type model.
Event	It is an external action which conditions a component state variable.
Family	It enables classification of models in the library.
Identifier	String of characters containing letters (uppercase or lowercase), numbers or underscore character (_). The string must start with a letter.
Integer constant	It is either zero or a string of numbers which do not start with zero.
Keyword	It is an identifier reserved for exclusive use in the Altarica language, e.g. if, then, etc.
Lexems	They are identifiers, keywords, integer constants, operators and separators in the Altarica code.
Link	It defines a link between two components or equipment in an architecture.
Model	It is a generic term which describes an object in the library.

Object	It is a generic term which denotes a component, an equipment, an operator or a type.
Operator	It is a generic elementary behavioural object without graphical representation which can be used to modelize a component.
Project	It is a generic term representing a set of systems.
System	It is a component of a project which is represented by an architecture.
System assertions	System assertions are links which are not displayed in the hardware architecture; they represent only flow assignments and do not use operators (e.g., <code>cst1.e = true</code>).
Transition (trans)	Altairica code which enables evolution of the state variables of components.
State variable	It is a variable identifying a component internal state (e.g. a variable with the following values "open/blocked", "ok/hs"...).
Structured Type	It is constituted of many variables, and each variable can have many values, for exemple voltage :{ null, nominal} and short-cut :{ yes, no}
Flow variable	It is a component input or output variable.

APPENDIXES

APPENDIX 1: SYNTAX OF Altarica CODE

General

Typing of the behaviour of component models comprises:

- The lexical conventions.
- The [assertions](#).
- The [transitions](#).
- The [comments](#).
- The [identifiers](#).
- The [keywords](#).
- The [constants](#).
- Assertion examples.
- Transition examples.

The lexical conventions

The Altarica language lexems are the identifiers, keywords, integer constants, operators and separators. Spaces, tabulations, end of line characters as well as comments are ignored except when they separate lexems.

The transitions

Transitions allow evolution of the state variables of components.

Transition model:

transition condition | – event -> transition assignments

The declaration of a model transitions starts with the **trans** keyword. This keyword is followed by a non-empty list of transition specifications separated by semi-colons. The transitions declaration may eventually end by a semi-colon.

transitions-definitions

trans transition-list

transition-list

transition ; transition-list

A transition specification always starts with a boolean expression called the transition *condition*. The variables which appear in these expressions are the model flow variables or state variables.

transition
boolean-expression transition-target-list
boolean-expression
[expression](#)

The transition condition is then followed by a non-empty list of actions associated to events.

transition-target-list
transition-target; transition-target-list
transition-target

An action starts by the separator | – followed by a non-empty list of events previously declared in the model.

The list ends with the –> separator followed by an assignment list for the variables.

transition-target
 | – *event-name-list* –> *assignment-list*

Assignments of variables are separated by commas (,).

assignment-list
assignment , assignments-list

An assignment is composed of the identifier of the model state variable followed by the := separator and ends with an expression of the same type as the state variable. The expression assigned to the state variable can contain the model flow variables or state variables

assignment
variable-name := expression
variable-name
[IDENTIFIER](#)

[Transition](#) example

The assertions

Assertions allow modification of the output state variables of components.

The assertions declaration for a component start with the **assert** keyword. This keyword is followed by a non-empty list of boolean expressions separated by semi-colons (;). The declaration may eventually end with one semi-colon.

assertions-definition
assert assertion-list
assertion-list
assertion ; assertion-list
assertion
boolean-expression

The variables which appear in an assertion are the flow variable or state variables of the model in which the assertion is defined.

The boolean expressions authorized at the first level in an assertion are the following:

- The parenthezed expressions such as if then else
- The assignments of the variables:

An assignment is composed of the identifier of an output flow variable (out) for the model followed by the separator = and ends with an expression of the same type as the flow variable. The expression assigned to the flow variable can contain constants and input flow variables (in) or state variables for the model.

Assertion examples

Expression

This section presents the syntax of Altarica expression which may appear in the assertions or the transitions condition.

The expressions below are presented in the decreasing priority order.
(The logic OR is the lowest order priority).

- The parenthezed expressions.
- The variables.
- The atomic expressions.
- The unary operators.
- The relational operators.
- The equality operators.
- The logic AND operator.
- The logic OR operator.
-

The parenthezed expressions

In order to facilitate interpretation of the expressions If-Then-Else and If-Then, the latter must be mandatory between parentheses.

parenthezed-expression
 (expression)
 (expression ? expression : expression)
 (if expression then expression else expression)
 (if expression then expression)

For an If-Then-Else expression, the first sub-expression is mandatory boolean; it is called the operation condition. The two next sub-expressions make up the parts Then and Else of If-Then-Else.

Remark 1: The expressions authorized in the parts Then and Else are the same as those authorized at the first level of an assertion.

Remark 2: (if expression then expression else expression) and (expression ? expression : expression) are equivalent.

The variables

The variables of a description can be specified by a path in the hierarchy of Altarica models. This path is a string of identifiers separated by points (.).

hierarchy-path
identifier
identifier

hierarchy-path

The atomic expressions

Atomic expressions are either pathnames to variables or constants or parenthesized expressions.

Depending on the context, a path can be interpreted as a variable identifier or an enumerate constant identifier or also a declared constant identifier. In these conditions, a same identifier cannot be used for a declared constant, a variable or an enumerate constant.

atomic-expression
parenthesized-expression
hierarchy-path
true
false

The unary operators

The unary operations are evaluated from right to left.

```

unary-expression
operator-not unary-expression
atomic-expression

operator-not
not
~

```

The NOT operator has two identifiers, the not word or the tilde ~. The NOT operand must be of the boolean type and the result of the expression is the inverted value of this operand.

The relational operators

The relational operators are evaluated from left to right. Their operands must be of an arithmetic type because, unlike some programming languages, neither the enumerate constants nor the boolean variables, are assimilated to integers.

The operators < (less than), > (greater than), <= (less or equal to) and >= (greater or equal to) give the value false if the given relation is not satisfied by the operands; otherwise, the value of the expression is true.

```

relational-expression
relational-expression < additive-expression
relational-expression > additive-expression
relational-expression <= additive-expression
relational-expression >= additive-expression
additive-expression

```

The equality operators

These operators are = (equal to) and != (not equal to).

The operands of the = and != operators are boolean, arithmetic or enumerate. The value of these expressions is true if the expressed relation is satisfied by the values of the operands and false in the contrary. In the boolean case, the equality corresponds to the equivalence between the operands and the difference with the Exclusive or (XOR).

```

equality-expression
relational-expression                =                relational-expression
relational-expression                !=               relational-expression
relational-expression

```

The logic AND operator

The logic AND operator has two identifiers: `&` and `and`. Its operands must be of the boolean type. It is evaluated from left to right. It is true if its two operands are true and otherwise it is false.

```
and-expression
and-expression operator-and equality_expression
equality-expression
operator-and
and
&
```

The logic OR operator

The logic OR operator has two identifiers: `|` or `or`. Its operands must be of the boolean type. It is evaluated from left to right. It is false if its two operands are false and otherwise it is true.

```
or-expression
or-expression operator-or and-expression
and-expression
operator-or
or
|
```

The comments

Two types of comments are authorized:

The `/*` characters indicate the start of a comment on several lines and which must end by the `*/` characters. The comments must not be overlap.

Example:

```
/*
* This is a comment on several lines.
*/
```

The `//` characters indicate the start of a comment on one line. The comment ends either by carriage return character or by the end of a data flow.

Example:

```
// This is a comment on one line.
// This is another comment.
```

The identifiers

An *identifier* is a sequence of letters, of numbers or of `'_'`. A single identifier always starts with a letter and can be of any length. Uppercase and lowercase letters are different.

Example : `Motor`, `Motor_9`, `motor_9` (the last two are different identifiers).

Identifiers must belong to the language defined by the following regular expression:

[a-zA-Z]([a-zA-Z0-9_]*)

The keywords

The following identifiers are reserved for use as keywords and cannot be used for other purposes:

and	else	imply	max	sub
assert	event	in	min	sync
bool	extern	init	node	term
case	false	int	not	then
cnuf	float	inverse	or	trans
const	flow	knit	out	true
domain	func	link	private	#
edon	if	local	state	

The constants

There are several types of constants, each one having a particular type: integer constants, enumerated constants and boolean constants.

An *integer constant* is:

Either 0,

Or a sequence of numbers not starting with the 0 character.

Integer constants must belong to the language defined by the following regular expression:

$0 \mid ([1-9][0-9]^*)$

A *boolean constant* is one of the two following keywords: true or false.

An *enumerated constant* is an identifier previously defined in a field of enumerations.

Assertion examples

Example 1

```
assert
(if etat=ok and
(true = (e_v1 and e_v2) or (e_v1 and e_v3) or (e_v1 and e_v4) or (e_v1 and e_v5) or
(e_v2 and e_v3) or (e_v2 and e_v4) or (e_v2 and e_v5) or
(e_v3 and e_v4) or (e_v3 and e_v5) or (e_v4 and e_v5))
then s = true , valide = true);
```

Example 2

```
assert
(if etat = non_repartie or etat = repartie then
(if e1 = normal or e2 = normal then
s = normal
else
(if (e1 = intempestif) or (e2 = intempestif) then
s = intempestif
else
s = absent
)
)
);
```

Example 3

```
assert
(if (panne_moteur_droit and (~auto_drapeau_droit or ~suralimenter_gauche)) or
(panne_moteur_gauche and (~auto_drapeau_gauche or ~suralimenter_droit))
then
s = true , icon = 2
else
s = false , icon = 1);
(if etat =hs then s = false , valide = false);
(if etat=ok then icon=1 else icon=2);
```

Example 4

```
s1 = e1 or e2 ;
s2 = e3 ;
```

Transition examples

Example 1

```
trans
etat=ok |- panne -> etat := hs;
```

Example 2

```
trans
etat = non_repartie |- def_local_panne -> etat := hs;
etat = non_repartie|- def_local_intempestif -> etat := instable;
```

Example 3

```
trans
etat = ouvert and blocage = non |- fermeture_intemp -> etat := ferme , blocage := oui;
etat = ferme and blocage = non |- ouverture_intemp -> etat := ouvert , blocage := oui;
```

PAGE LEFT BLANK INTENTIONALLY

APPENDIX 2: SYNTAX OF PROBABILITY LAWS IN ARALIA FORMAT

Probability laws for Basic Events				
Law	Expression	Par.	Definition	Limits
<i>constant</i>	$A(t) = q$	q	Failure probability	$0 \leq \text{Probability} \leq 1$
<i>exponential</i>	$A(t) = 1 - \exp^{-\lambda \cdot t}$	λ	Failure rate	Rate ≥ 0
<i>GLM</i>	$A(t) = \gamma \exp^{-(\lambda+\mu) \cdot t} + \lambda / (\lambda+\mu) \cdot \exp^{-(\lambda+\mu) \cdot t}$	γ λ μ	Probability of initial start-up refusal Failure rate Repair rate	$0 \leq \text{Probability} \leq 1$ Rate ≥ 0 Rate ≥ 0
<i>GLM-asymptotic</i>	$A(t) = \lambda / (\lambda+\mu)$	λ μ	Failure rate Repair rate	Rate ≥ 0 Rate ≥ 0
<i>Weibull</i>	$A(t) = 1 - \exp^{-\phi(t)}$ avec $\phi(t) = ((t-\tau)/\alpha)^\beta$	α β τ	Scale parameter Form parameter Location parameter ($\alpha \leq 0$ as $t-\alpha$ must always be greater than 0)	Factor > 0 Factor > 0 Time > 0
<i>periodic-test</i>	$A(t) = \begin{cases} 1 - \exp^{-\lambda \cdot t} & \text{si } t \leq \tau \\ 1 - \exp^{-\lambda \cdot ((t-\tau) \% \theta)} & \text{si } t > \tau \end{cases}$	λ τ θ	Failure rate of the component in operation or on stand-by First interval between tests	Rate ≥ 0 Time > 0 Time > 0
<i>periodic-test-2</i>	<i>Numeric integration</i>	λ λ^* μ τ θ γ π X σ ω	Functioning failure rate Failure rate during test Repair rate (once failure is detected) First interval between tests Interval between two tests Failure probability because of test (0 if test can't create failure) Test duration Availability during test (= 0, if unavailable; = 1, if available) Test coverage rate (the probability that test detect failure) reSetUp forgetfulness probability (after test and after repair)	rate ≥ 0 rate ≥ 0 rate ≥ 0 Time ≥ 0 Time ≥ 0 $0 \leq \text{Proba} \leq 1$ Time ≥ 0 0 or 1 $0 \leq \text{Proba} \leq 1$ $0 \leq \text{Proba} \leq 1$
<i>dormant</i>		λ <i>MTTR</i>	Failure rate Medium time to repair	rate ≥ 0 Time ≥ 0

Probability laws for Basic Events				
Law	Expression	Par.	Definition	Limits
		T	interval between two consecutive tests	Time ≥ 0
CMT	$A(t) = Q + 1 - \exp^{-\lambda \cdot T}$	λ	Failure rate	rate ≥ 0
		T	Mission duration	Time ≥ 0
		Q	Probability	$0 \leq \text{Proba} \leq 1$
NRD	$A(t) = \exp^{-\mu \cdot d}$	μ	Repair rate	rate ≥ 0
		d	Delay	Time ≥ 0