

20/10/2001
LISI – Futuroscope – France.

PLIBEditor Manual

PLIBEditor is a standalone Java application for browsing, modifying or creating a parts library according to the PLIB specification based on the **ISO13584_25 library implicit schema**. This free version supports conformance class 0 (compatibility with IEC 61360-2 and -5) and 2 (complete dictionary). Currently, it does not support functional models and functional views. But, it support all others constraints including external files and Case-of relationship.

Remarks :

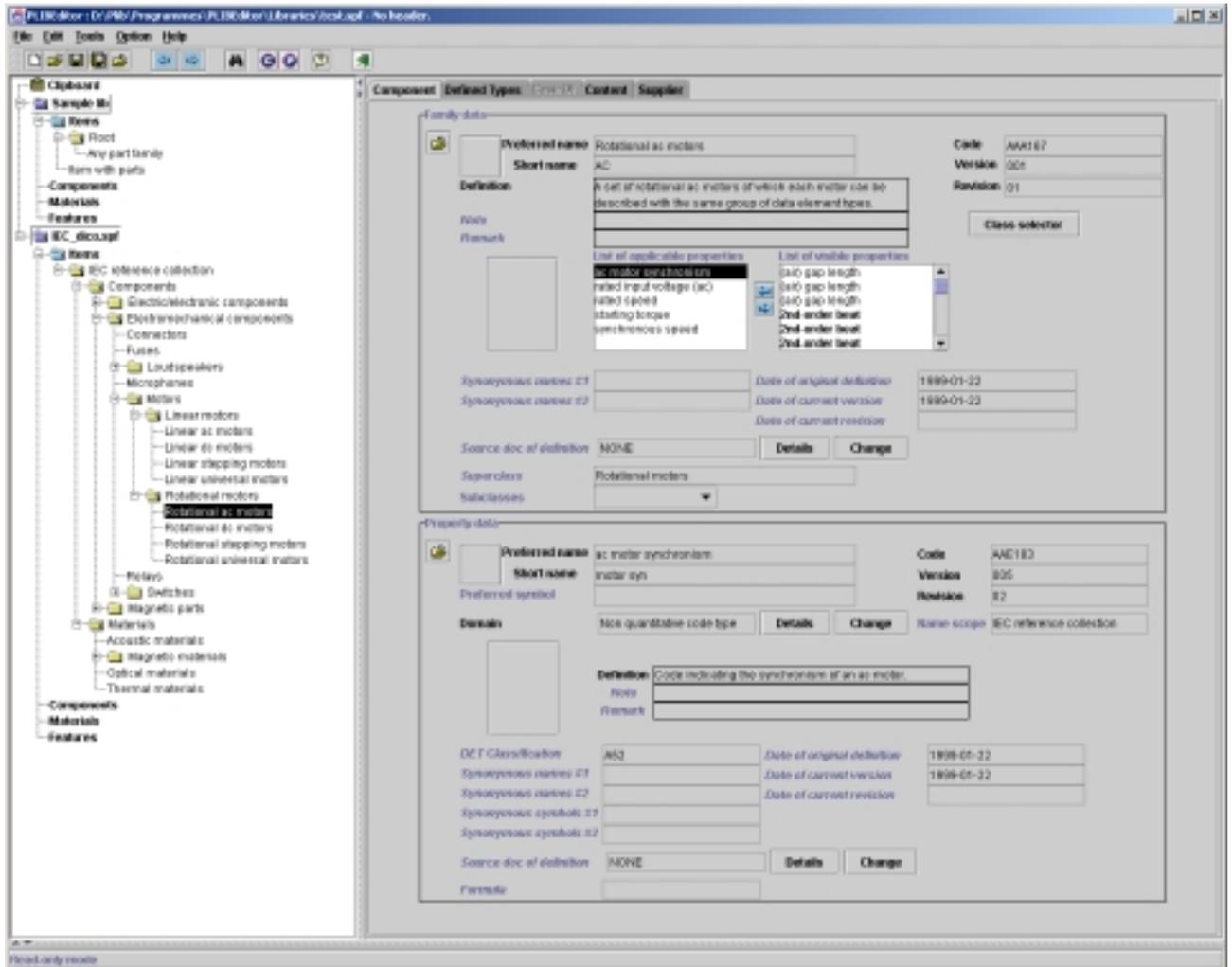
If you read a physical file based on an old version of the ISO13584-25 library implicit schema or on other PLIB standardized schemas (i.e., LIIM24-1, LIIM24-2, LIIM24-3), the system will try to read all the entities which are effectively used in the ISO13584-25 library implicit schema.

An error report may be got just by activating "Show Java virtual machine output" in the "Option" menu. Nevertheless, the reported errors may only be corrected in a text editor or in a PLIB-based EXPRESS tool. Note that "warnings" do not need to be taken into account.

Application overview:

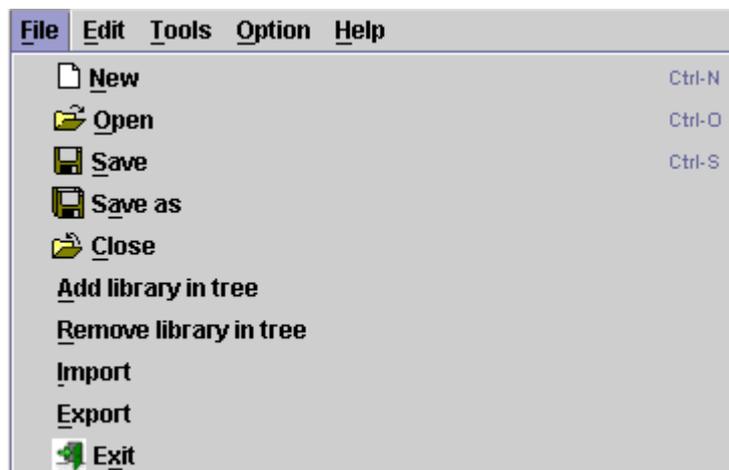
MAIN WINDOW:

The graphical user interface is divided into three distinct areas : on the top, menu bar and tool bar, on the left hand side, a tree representing the parts families hierarchy and, on the right hand side, characteristics of families and properties are displayed (fig.1). Note that you can switch from parts families and properties descriptions to supplier description by using the tab.



MENUS:

FILE MENU:



Common menus with mnemonics and keyboard shortcuts.

New creates a new part library and opens it in PLIBEditor. This library is the primary library i.e. the only one that can be edited.

Open opens a STEP file into PLIBEditor. The library is the primary one as previously described.

Add library in tree lets user insert secondary libraries in order to perform importations between libraries ('case of' concept in PLIB). The first library (upper library node) is the primary library : this is the only library that can be edited. There is a only primary library that is loaded when opening or creating a new file. Secondary libraries cannot be edited and are only available for 'case of' links.

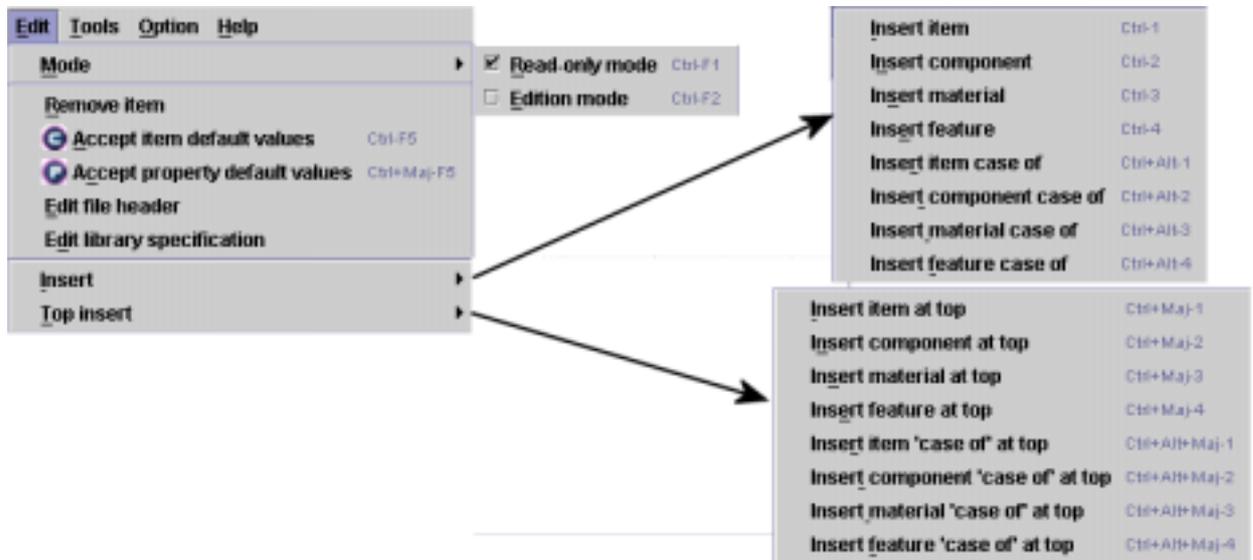
Remove library in tree removes any secondary library from the tree.

Export converts the HTTP directory structure of external files referenced by the library into a flat structure for exchange (cf. Annex C).

Import builds HTTP directory structure from a flat structure in order to browse documents according to HTTP protocol (cf. Annex C).

Lines after separator are the list of most recently used files.

EDIT MENU:



Mode submenus define current mode : read-only or edition.

Remove item delete the currently selected part family.

Accept item default values and *Accept property default values* validate default values that are set when any entity is built (This ensure library integrity at any time). Red labels identify a field with a default value that has not been modified by user.

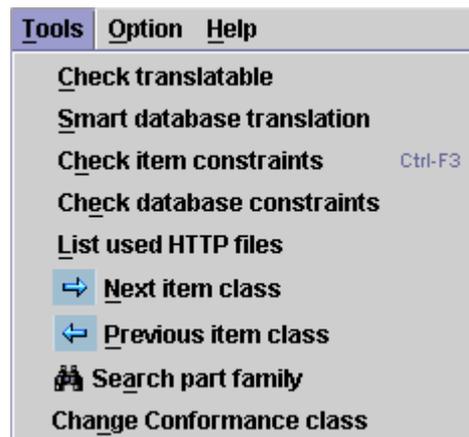
Edit file header can be used to add a description and authors to the STEP file header.

Edit library specification let user enter data relative to the class conformance of the library.

Insert sub menu inserts a new node as child of the currently selected node (nodes are part family representation in a hierarchical structure).

Top insert is the same action but inserts node as root (Warning: it is recommended not to have more than one root per library).

TOOLS MENU:



Check translatable verifies all translated texts in any translated library to ensure that translations exist in every language defined in the library.

Smart database translation builds a translated library from a not translated library. All translatable entities are converted into translated entities and their current values are set as translation into the selected language. If a global language assignment exists then library is translated according to this language.

Check item and *database* generate a log of constraint violations.

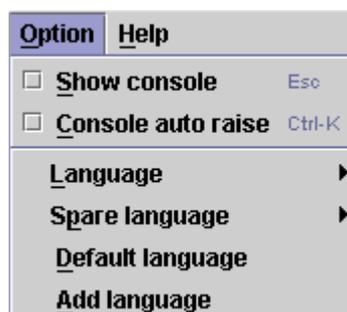
List used HTTP files presents the list of absolute file paths that are referenced as external content in library.

Next Item Class and *Previous Item Class* go forward and backward in tree node selection history.

Search part family performs a search in library for any part/property according to some criteria (preferred name, code, etc.) with direct links to matching part/property.

Change conformance class lets user convert the library into another conformance class. Available conformance classes are 0, 2 and 4 (cf. Annex D).

OPTION MENU:



Show console in order to easily hide or show console (ESC shortcut).

Console auto raise sets or unsets the possibility to make the console automatically appears when any integrity constraints are violated.

Language lets user select language to use for data description.

Spare language is the language to use when current language is not available for some data; this language is selected from the list of available languages in library.

Default language is the language of all data when library is not translated ; a translated library does not have a default language as all data have to be translated. When a library is translated there is no default language and this cannot be changed.

Add language opens a dialog window for language selection; language is added and set for presentation. When adding a first language to a library, a smart translation is performed : all translatable texts (none of them were translated) are considered to be the translations in the global language assignment, when it exists, or in the selected language, when global language assignment does not exist.

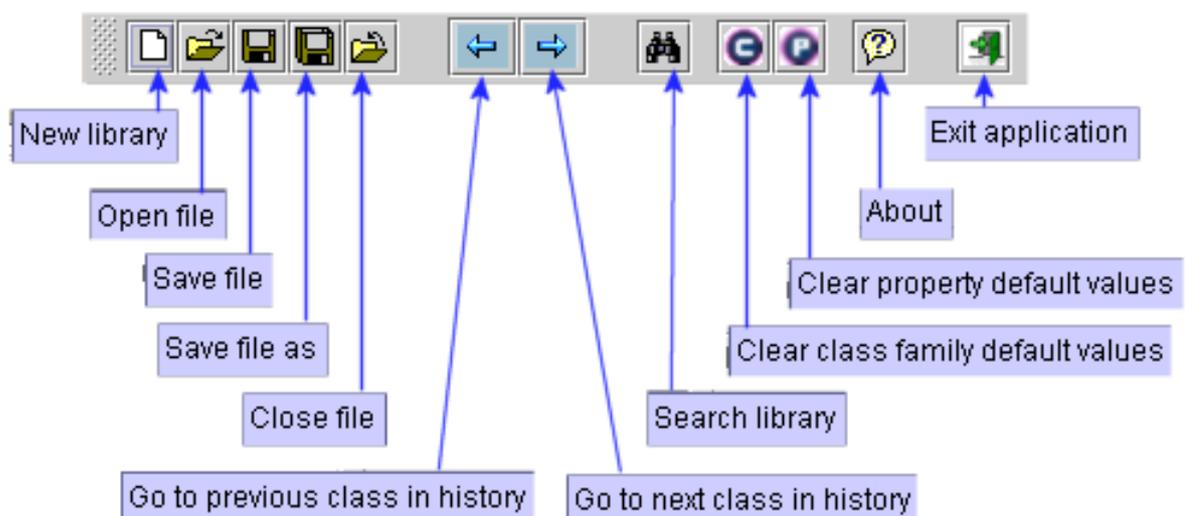
NB: A color code identifies in which language data are translated (cf. Annex A).

HELP MENU:



Help is not available and only *About* exists to show application version and contacts.

TOOLBAR:



INTERFACE

TREE

The tree represents the hierarchical structure of the part families.

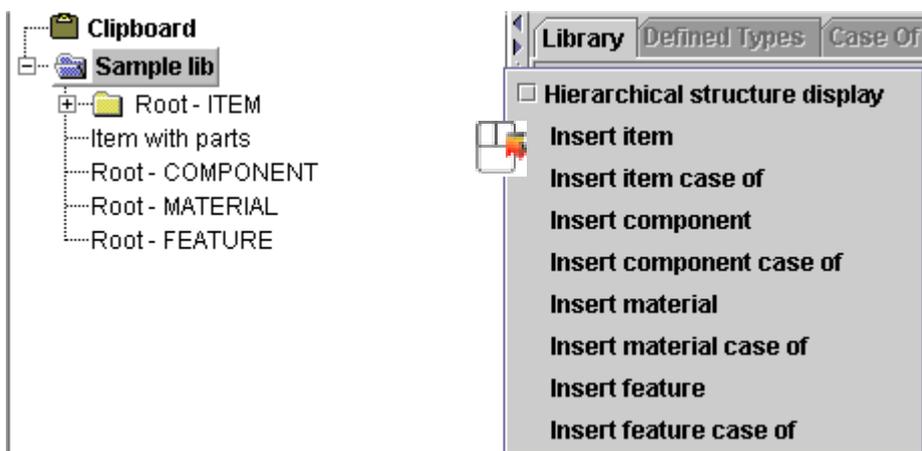
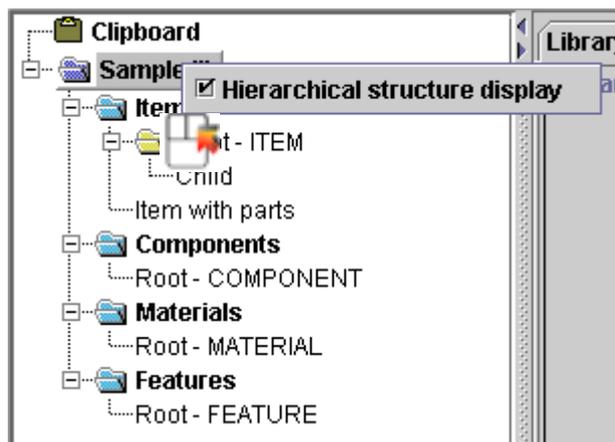
There are two types of nodes :

- blue nodes are logical nodes;
- yellow nodes are part families.

There are two types of logical nodes :

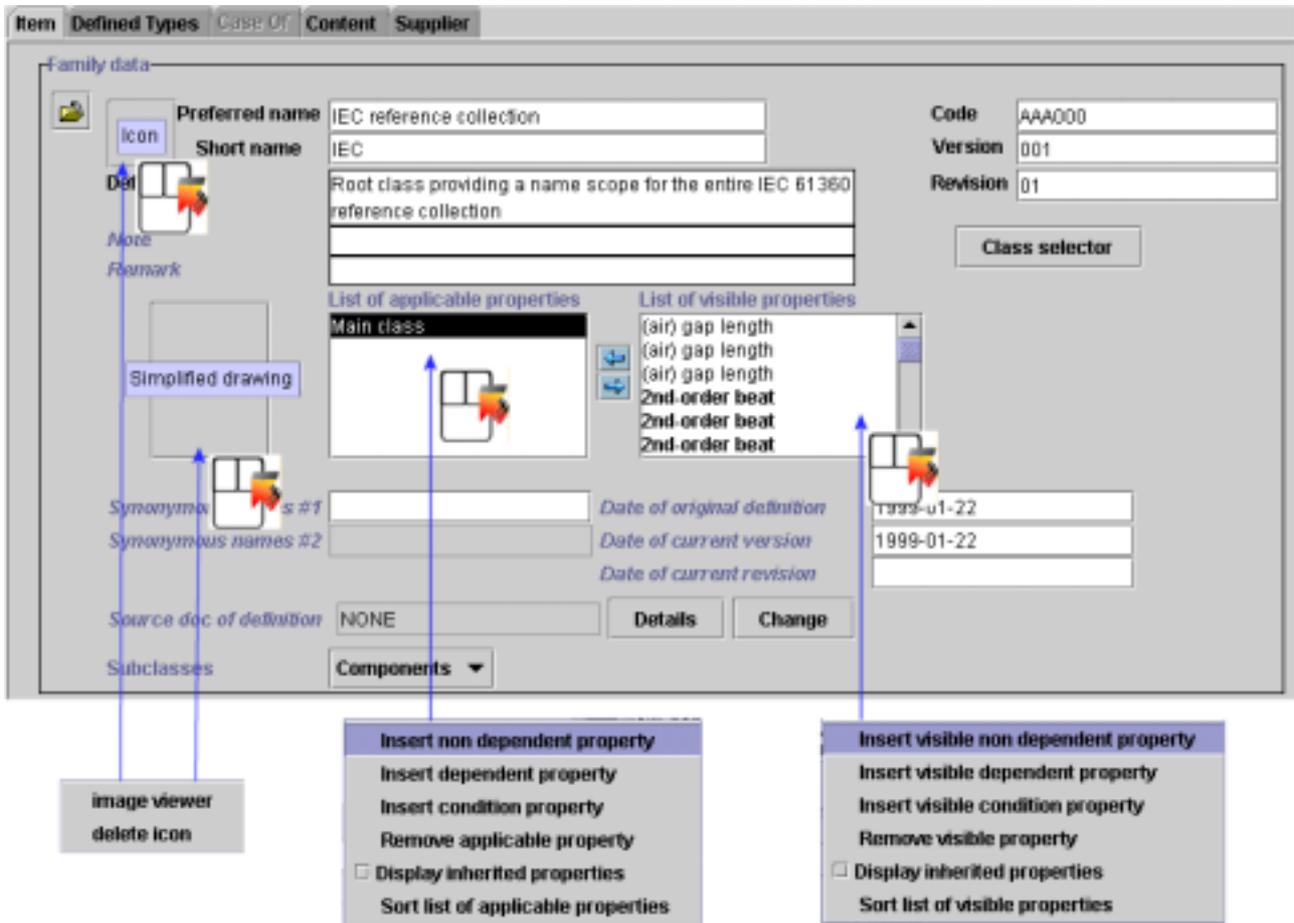
- deep blue nodes with border represent libraries (bordered up and right for the primary library and bordered down and left for secondary libraries);
- light blue nodes are roots for part family of a given type (item, component, material, feature). If a material family inherits an item family that has no super class, then the material family can only be found in the sub tree of its super class that does appear under the logical node 'item'. Only part families that have no super class are taken into account to populate the logical nodes 'items', 'components', 'materials' and 'features'.

A popup menu let user display or not the light blue nodes in order to sort root families according to their type or not (see figure below).

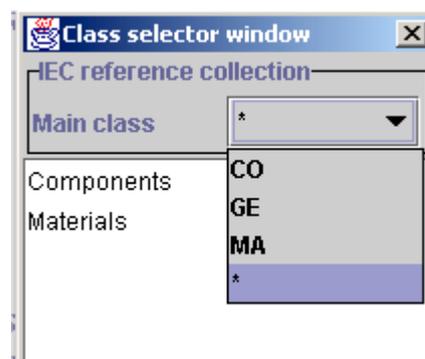
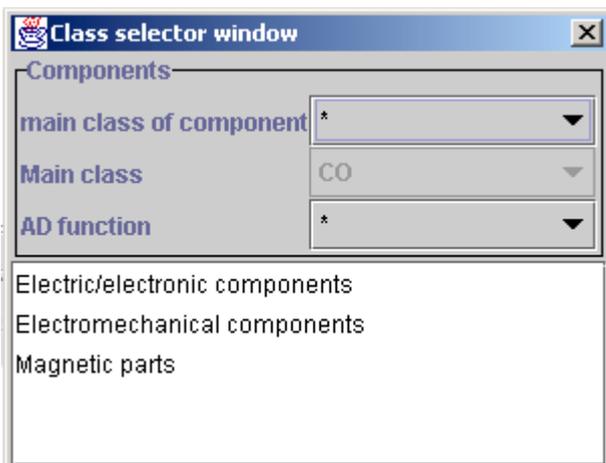


PART CHARACTERISTICS

Property lists and graphic buttons (icon and simplified drawing) have popup menus.

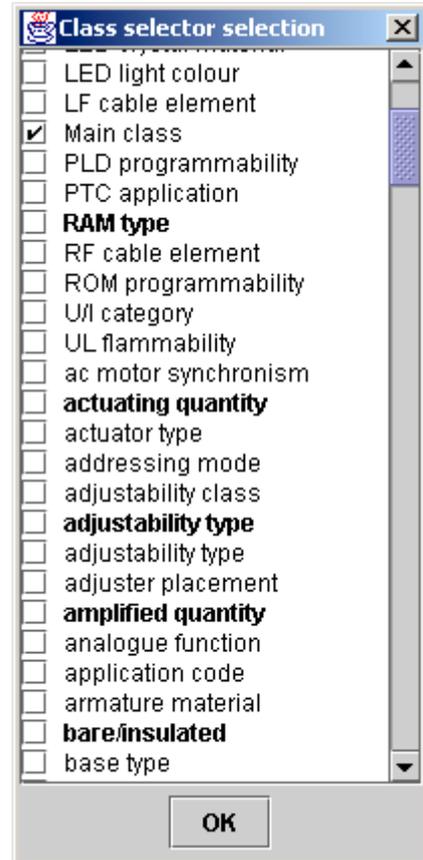
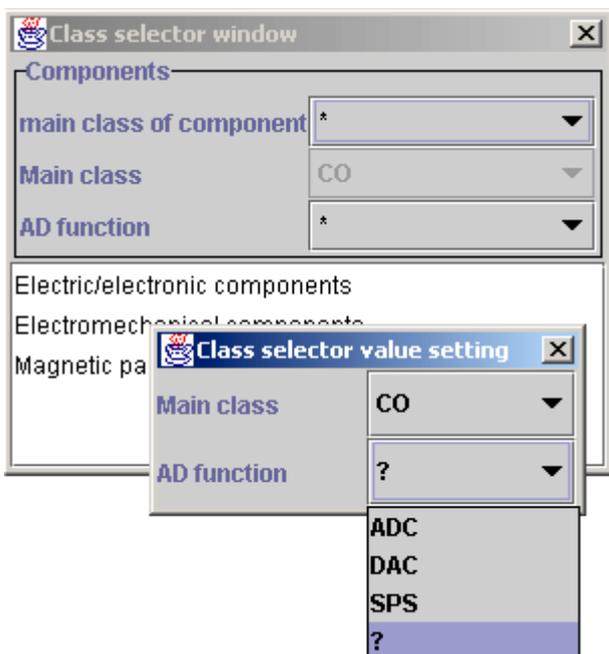
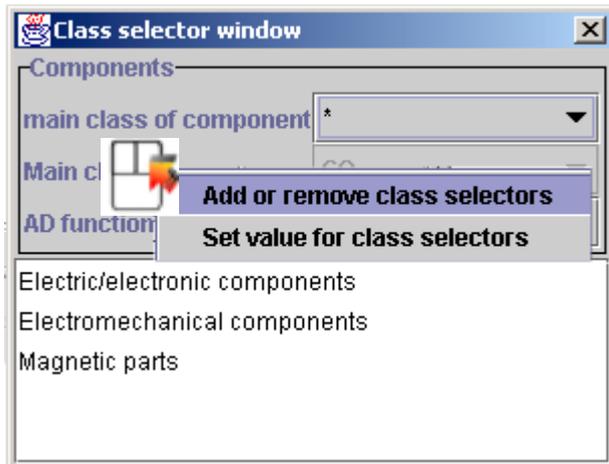


Class selector button opens a frame with available class selectors and combo-boxes to select values. Matching subclasses are displayed and a click onto a family name send you to the part family.



Class selector frame has a popup : you can add/remove a class selector (i.e. a property of non quantitative type) or modify a class selector value. A list presents all available properties for class selectors and check boxes indicate if property is already used as class selector. A bold font is used

for properties used as class selector in a subclass and a bold italic font for properties used as class selector in a super class. Class selector are valued by selection in combo-boxes.



Documents can be added as source document of definition. Identified document are represented with an identifier. Referenced document are more complex. An example of referenced document is explained in the chapter 'Your first library'.

A selection in a property list makes the property characteristics displayed. Property panel is similar with the previously presented panel : graphics are icon and figure and the setting of the domain of property is described in the chapter 'Your first library'.

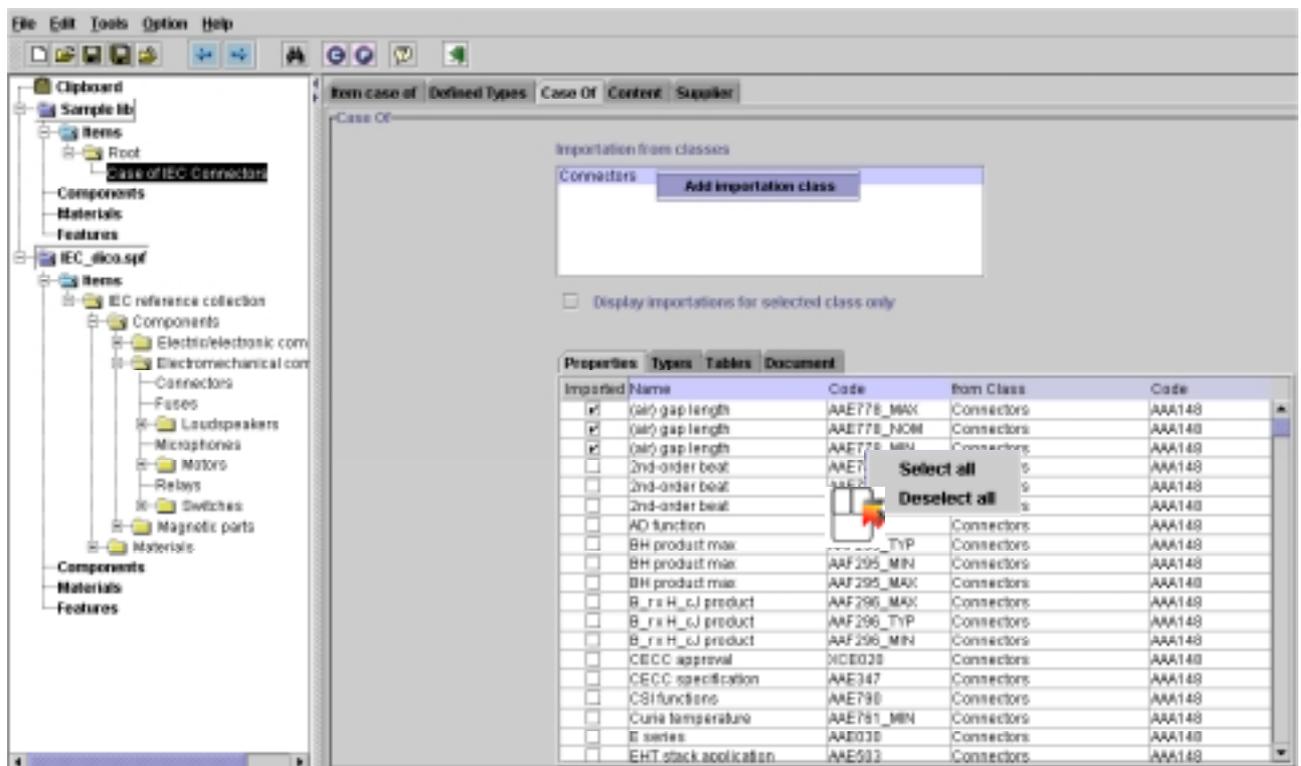
DEFINED TYPES

The mechanism used for defined types is identical with the one used for properties.

CASE OF (REFERENCES BETWEEN LIBRARIES)

N.B: When referencing entities of secondary libraries, you must save your library before removing these secondary libraries from the tree.

A part family can be a case of another one. This way you can import properties of any part family from any library. When creating a 'case of', you have to select a part family that your new family will be case of. Your family can be case of more than one part family : this is the action of the popup menu of the list. All available properties for import are displayed in the table under the list. Checking a property means importing this property. Table can be ordered according to any column by Shift+click onto the column header.



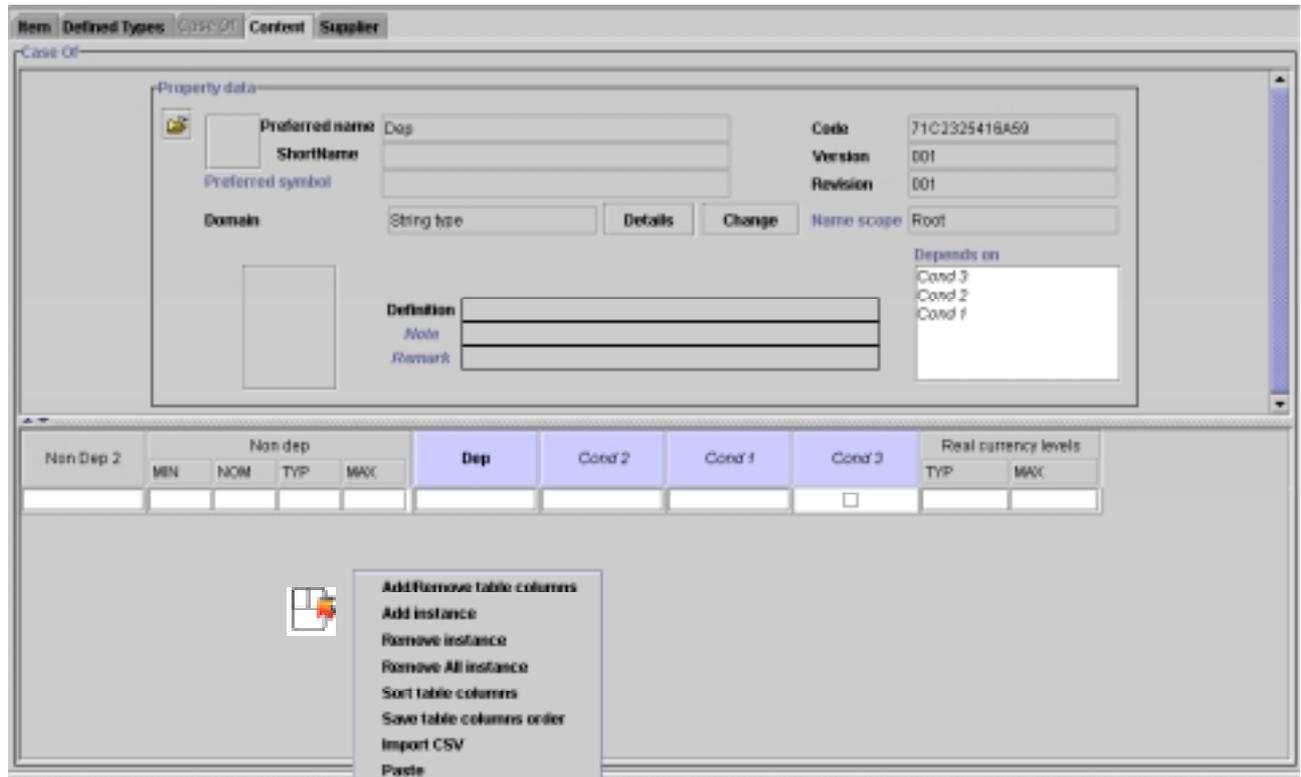
Definitions of imported property are not saved with your library : only the universal identifiers are. So, if the library from which you have made imports does not appear as a secondary library, you need to add it (via 'File-Add library in tree') to access their definition (for instance to see the name of properties in the table in 'case of' tab or 'content' tab).

CONTENT

The content tab presents two panels.

The lower panel presents a table : columns are properties and rows instances of part family.

The upper panel shows the characteristics of a property : a click onto the header of a column results in the display of the characteristics of this property.

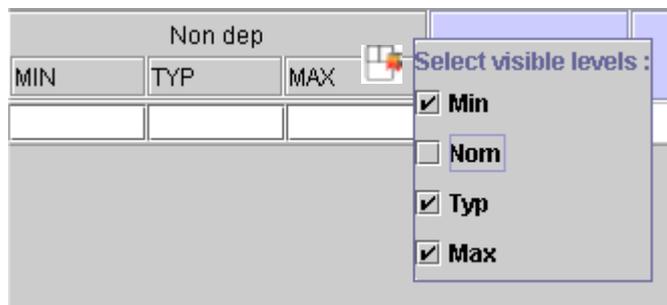
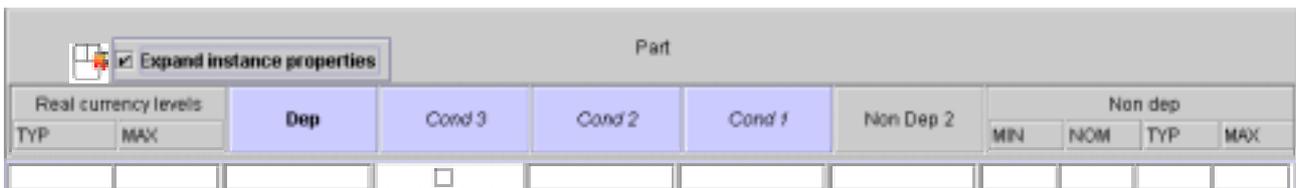


Two popup menus are available.

A table popup lets user define the table, add/remove instance etc. 'Add/Remove table columns' menu defines the table according to available applicable properties ; check boxes indicate if the key of the table contains the property. Columns can be sorted by drag and drop : 'save table columns order' can save the order of the view into the model (context property will actually always be just after the dependent property in the correct order).



A table header popup helps user in improving the display of the table : some levels can be hidden for level type columns and class instance columns can be collapsed or expanded to see all the characteristics of the referenced instance.



PLIBEditor features:

OPENING A FILE:

There are three ways : select *Open - File*, open icon in tool bar and shortcut CTRL-O.

BROWSING :

Select a node in the tree, expand the right hand side panel for advanced characteristics of the selected family.

Select a property in any list, expand the right hand side panel for advanced characteristics of the selected property.

There are two lists of properties : a list of visible properties and another one for applicable properties. You can select from both of them. Lists have popup menu.

MODIFYING AND CREATING A PART LIBRARY:

When opening a file, a read only mode is set: you can switch between «read only» and «edition mode» by selecting menu item in menu *Edit - Mode* (or using shortcuts CTRL-F1 (RO) CTRL-F2 (RW)). A new file can be created with CTRL-N or *File - New*.

Items, components and material are inserted in the parts family hierarchy with *Edit - Insert* or *Edit - Insert at top*.

Till you have not modified default values of mandatory attributes (labels in red), the tree is disabled : it becomes enabled when all values are entered or you use automatic clear (menus *Accept * default values*). If it is the first item in library, you have to fill in supplier mandatory fields too. Creating a property is based on the same mechanism but both property lists and tree are disabled.

Pop-up menus in tree and property lists present the fastest way to insert items, properties, etc. Two arrows let you change a visible property into an applicable property and vice-versa; control is done to ensure integrity (cf. Annex B).

You can add translations to your library. Color and style codes, described in annex A, allows to identify the language of the texts.

Remarks :

Drag and drop in the tree is allowed but without any control.

When editing a file with more than one PRESENT TRANSLATIONS entity, you may be asked for selecting one (it never happens with file created with PLIBEditor since it can constrain a unique PRESENT TRANSLATIONS entity use).

SAVING YOUR WORK:

«Save» and «Save as » commands are provided. The physical file is in STEP format. SPF file extension is added automatically if none is provided.

SOME TOOLS:

Some tools are available :

Check translatable – All translated data are verified : a translation must exist for each language of the library.

Smart translation - You can copy all values from a language into another one (to help translations but you can use a spare language too).

Check item constraints - verify local constraints (not final version).

Check database constraints - check entire library (not final version).

List used HTTP files – Listing of external files.

Next/Previous item class – History move.

Search part family – Look for items and property matching some criteria (preferred name, code, etc).

Change conformance class – Conformance class of library can be changed at any time.

LIMITATIONS :

Some regression involves a bad behaviour when removing a part family : error messages are displayed on standard out but it does work correctly.

When synchronizing universal identifiers of different libraries, error messages are sent on standard out but does not have any effect on PLIBEditor.

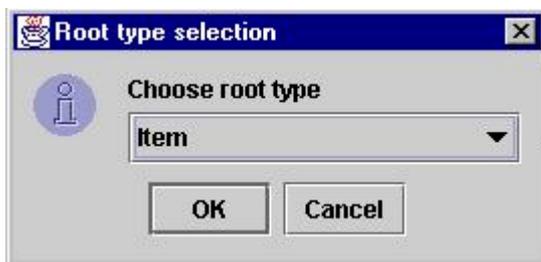
Your first library:

NB: This example uses a previous version of PLIBEditor but it does work the same way with the last version – some menus can have different names or different locations.

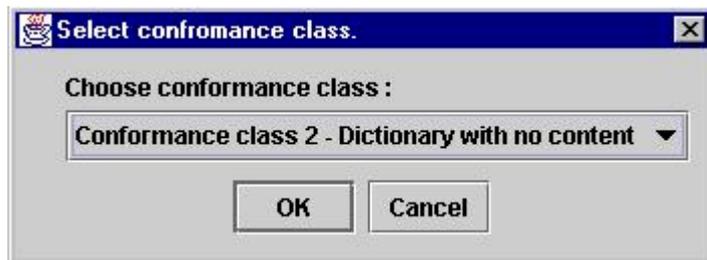
CREATE A NEW LIBRARY:

Select *File-New*.

Select root type from the four alternatives.



Select conformance class (0 or 2 as no content can be added).



Enter library specification if you have chosen a conformance class 2.

DEFINE A NEW PART FAMILY:

Fill in mandatory fields (their labels are red and turn black when modified).
 Use *Edit-Accept item default values* to clear mandatory fields you do not want to fill

The screenshot shows the 'Family data' window with the 'Supplier' tab active. On the left, there are labels for 'Preferred name', 'Short name', 'Definition', 'Note', and 'Remark', each followed by a text input field containing 'Not defined...'. On the right, there are labels for 'Code', 'Version', and 'Revision', each followed by a text input field containing '71C0AB2B99556', '001', and '001' respectively. Below these are two empty text areas labeled 'List of applicable properties' and 'List of visible properties', with two blue arrows between them pointing in opposite directions. A 'Class selector' button is located to the right of the text areas.

As it is the first part family, you need to fill in supplier data two by selecting the second tab.

NB:

Tree exploration is disabled until all mandatory fields are validated.

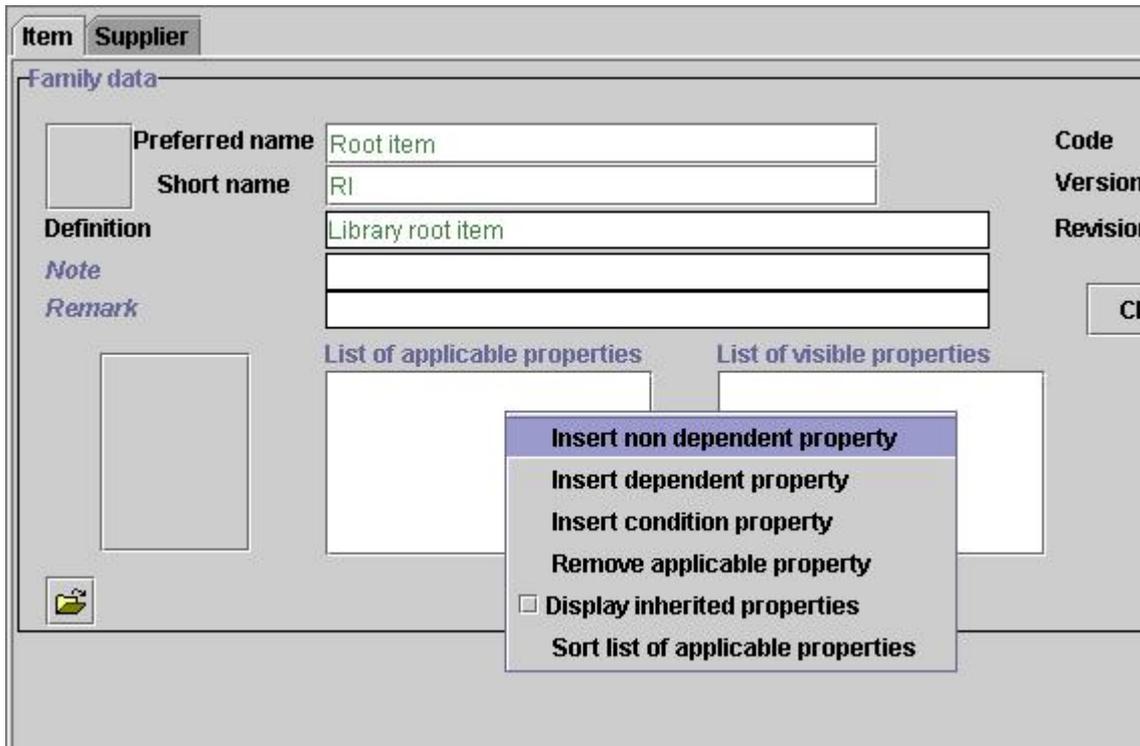
(*Edit-Accept item default values* clears supplier fields too.)

Remark: some supplier mandatory fields are dependent : filling one will validate the dependent ones.

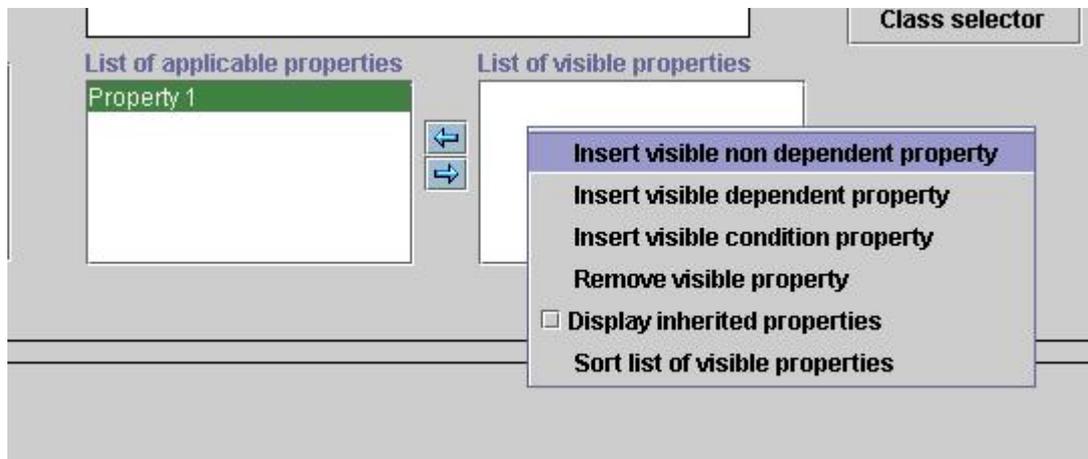
The screenshot shows the 'Supplier data' window. It contains a list of fields with corresponding input boxes. The 'Code' field contains '71C0AB2B99574' and the 'Revision' field contains '001'. The 'Description' field is a wide text area. Below the main fields are three date fields: 'Date of original definition', 'Date of current version', and 'Date of current revision'. At the bottom, there is a vertical list of fields for address information: 'Internal location', 'Street number', 'Street', 'Town', 'Postal box', 'Postal code', 'Region', and 'Country'.

ADD SOME PROPERTIES:

Now, add an applicable property : right click the applicable property list and choose menu according to property type you want.



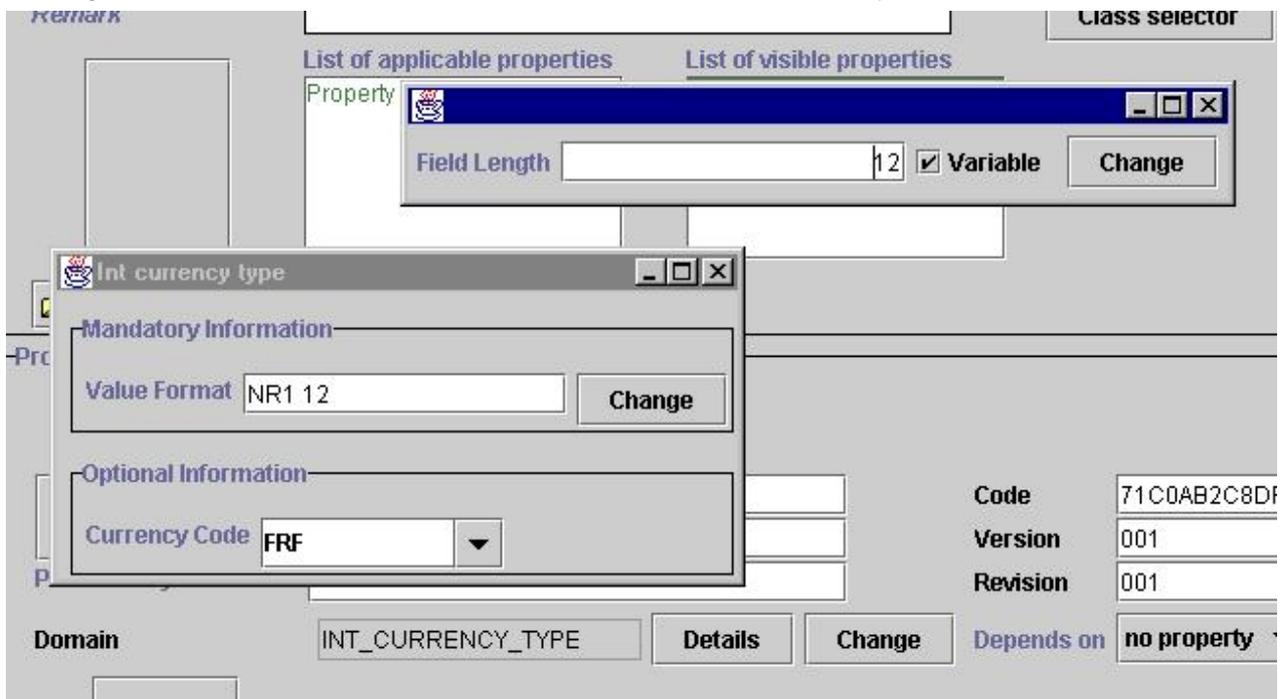
Fill in property fields as for item; as for item tree is locked until mandatory fields are validated. Add a visible property : right click the visible property list and select menu.



Change property domain : click change button at domain line in property panel.
 Select any type : Int currency for instance.



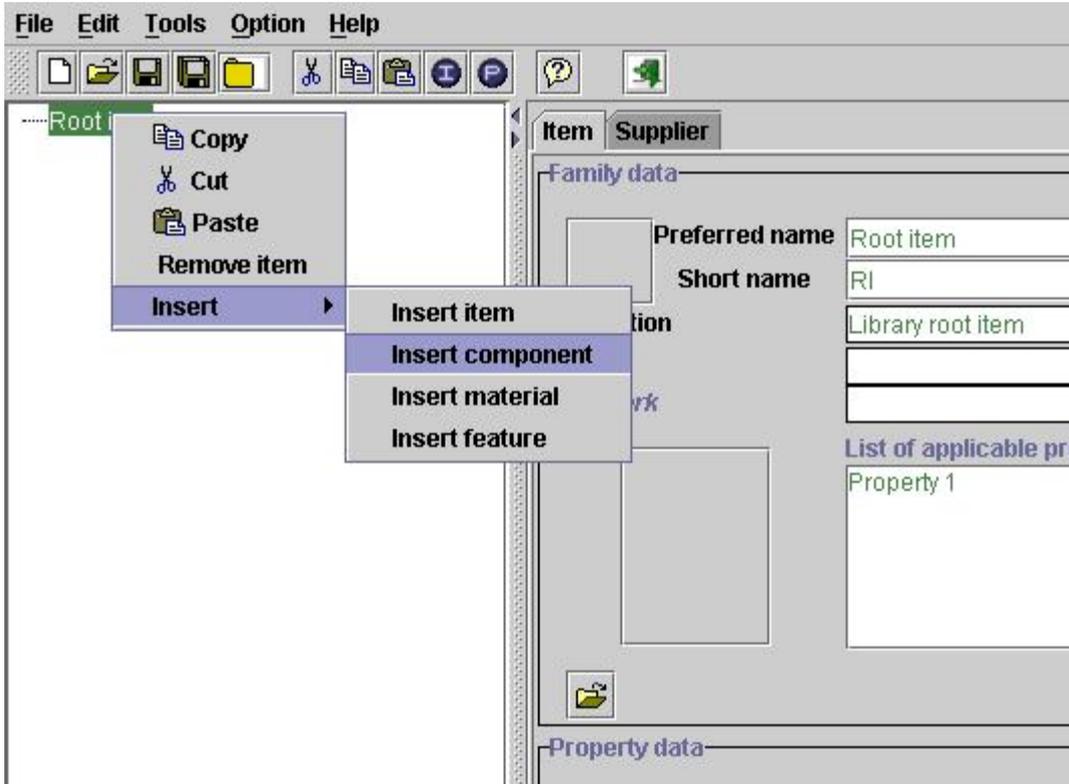
Change domain value : click detail button at domain line in property panel.



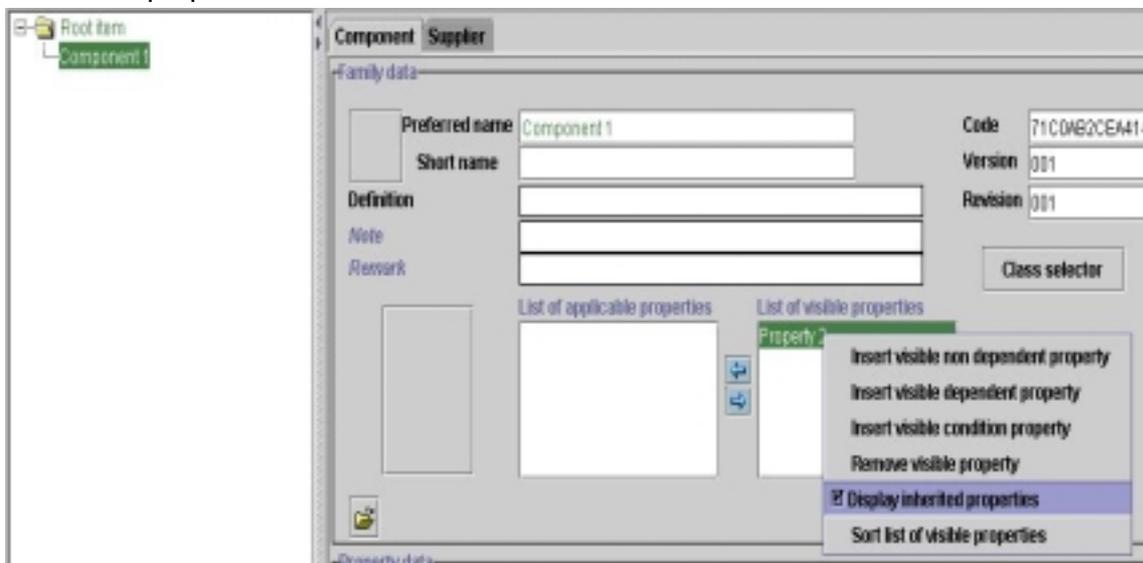
Many value format fields have a change button that opens a dialog to easily create format code.

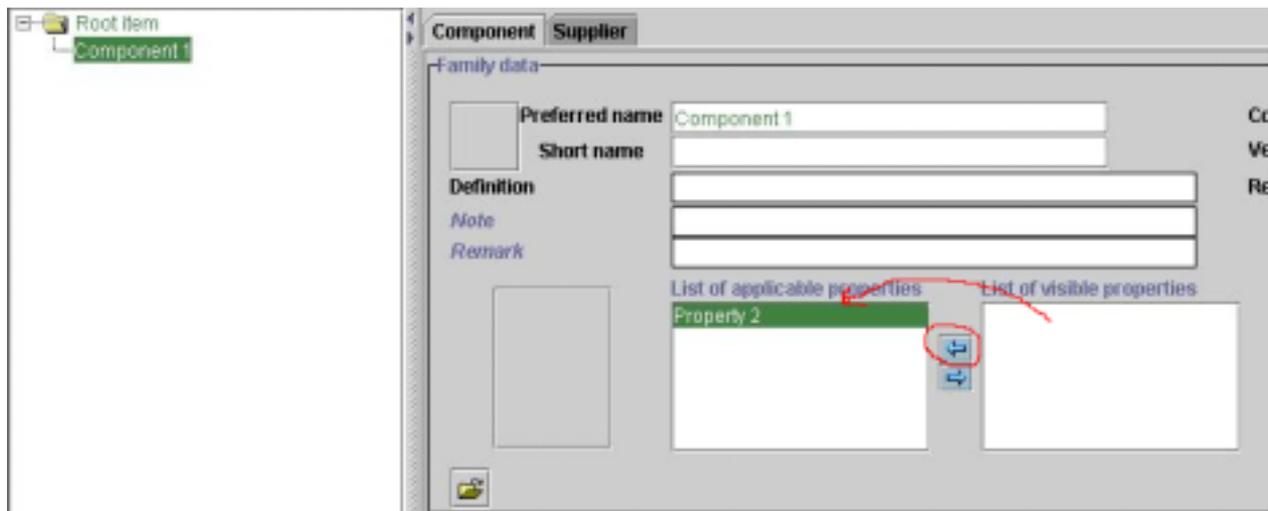
ADD DERIVED PART FAMILY :

Add some derived part family via Edit-Insert or right click the tree node for popup menu.



Display inherited properties : both lists have popup menu and you can select a check box to see all inherited properties.



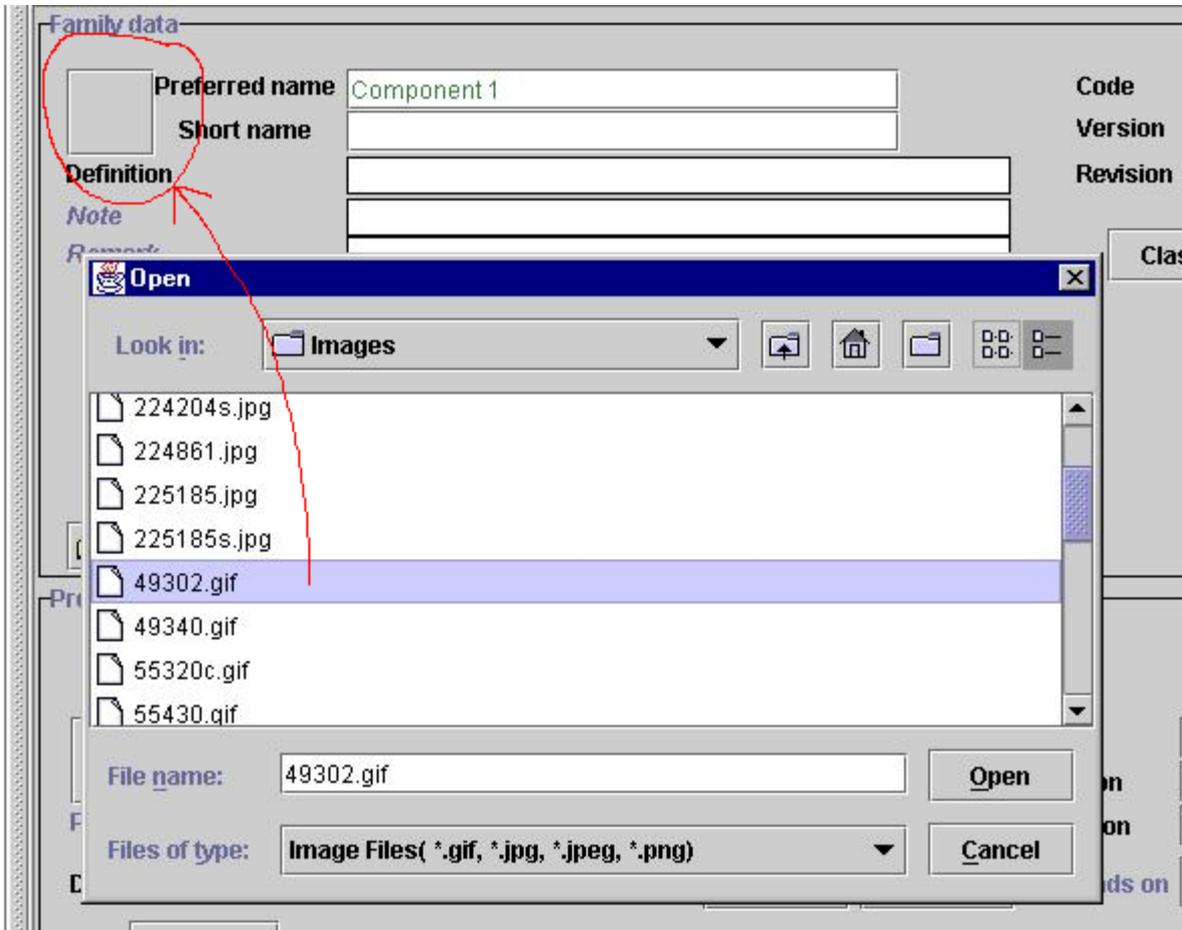


Make applicable a property visible in ancestor family : select a property in the visible property list and use left arrow (located between the two lists). Right arrow makes applicable a visible one. Checking is performed for these operations and some dialogs can appear to point out any trouble to fix before operation can be done. Note that multiple selection is allowed.

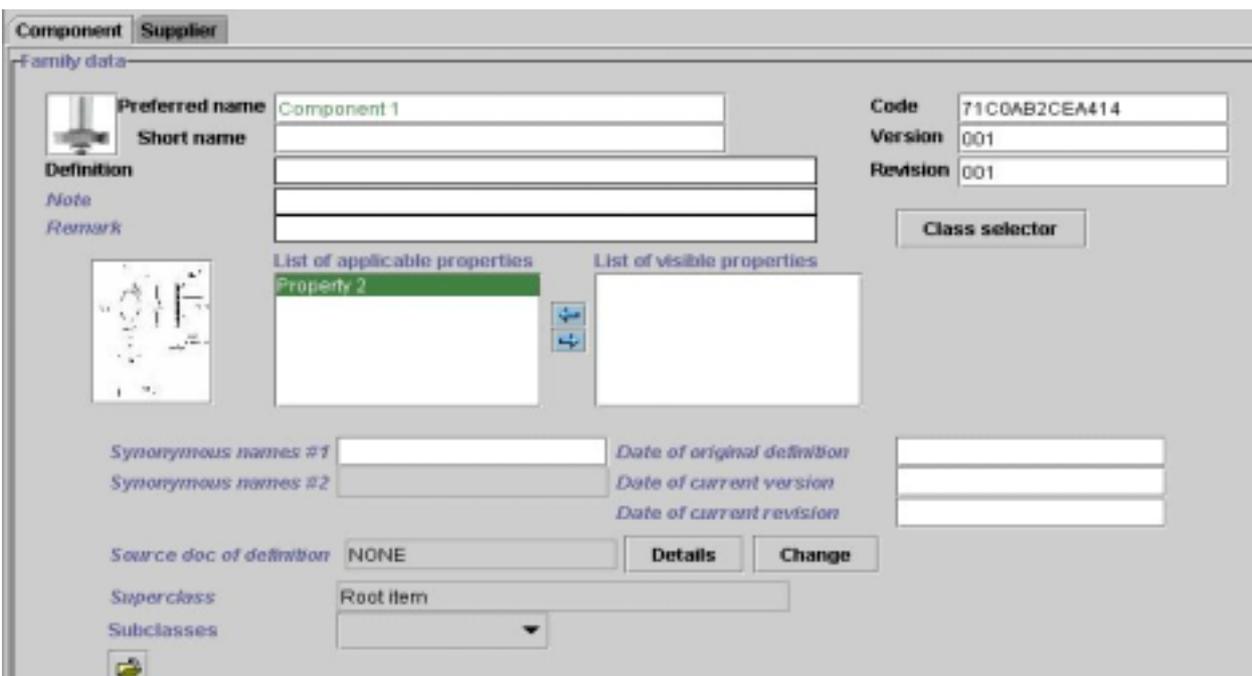
ADD IMAGES INTO LIBRARY:

Now add an icon for your component : left click the box near preferred name and short name (tooltips will tell if it is the icon or the simplified drawing).

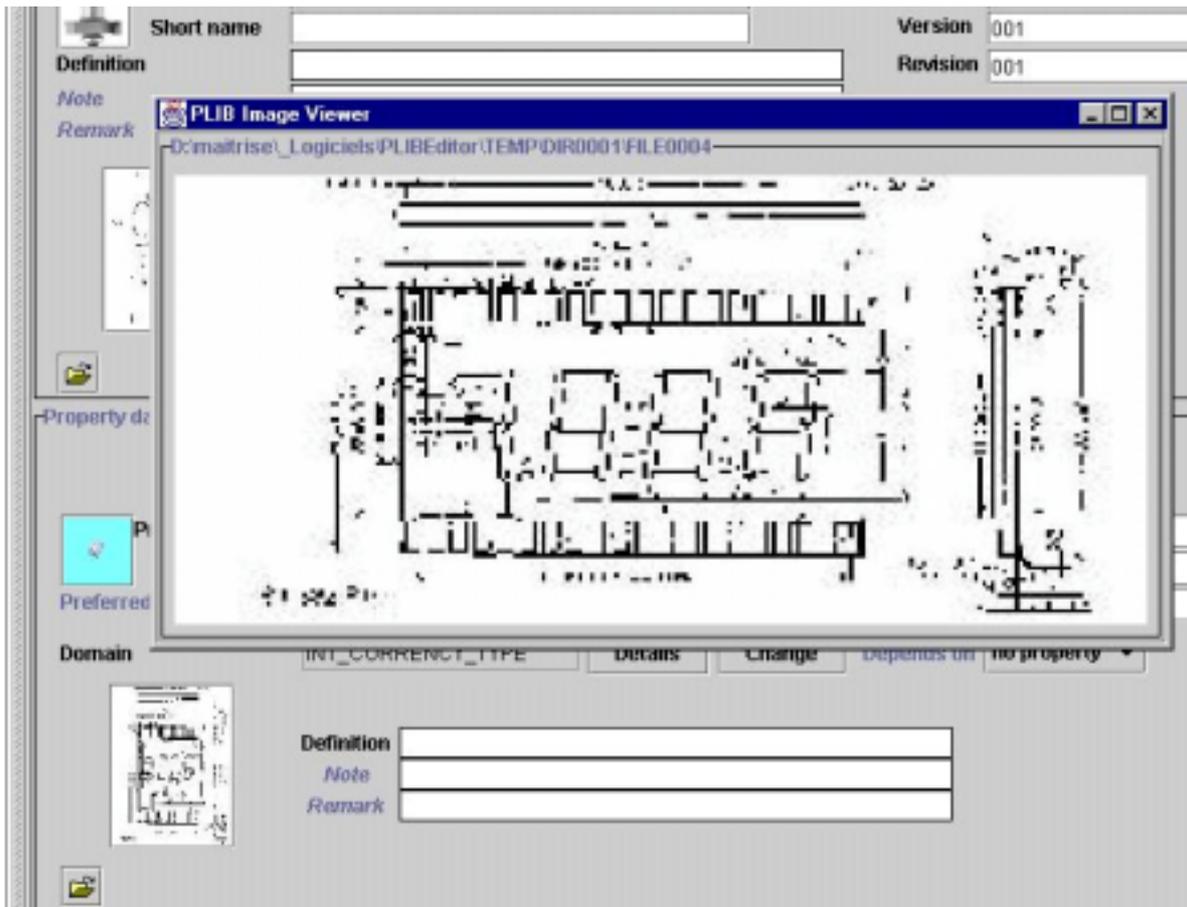
A file chooser appears; select your image file (supported formats are jpg, gif and png).



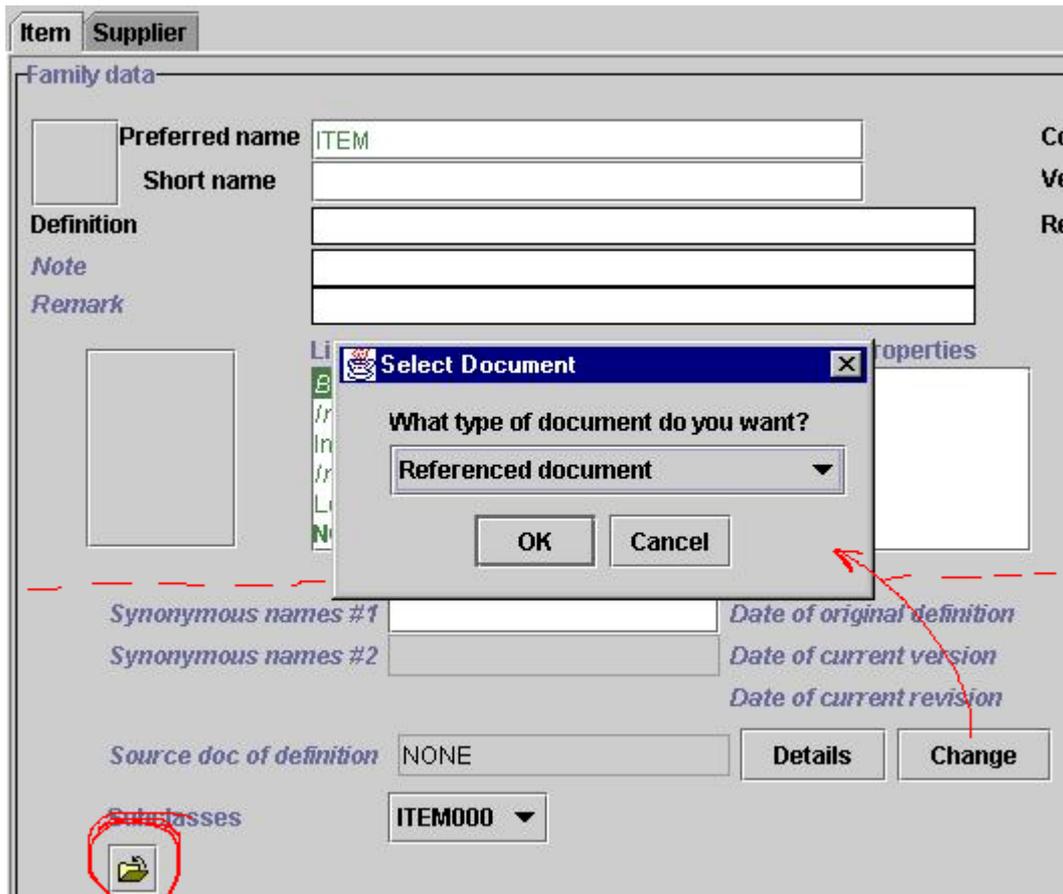
Repeat the same operation with the box below the icon : the simplified drawing of our part family.



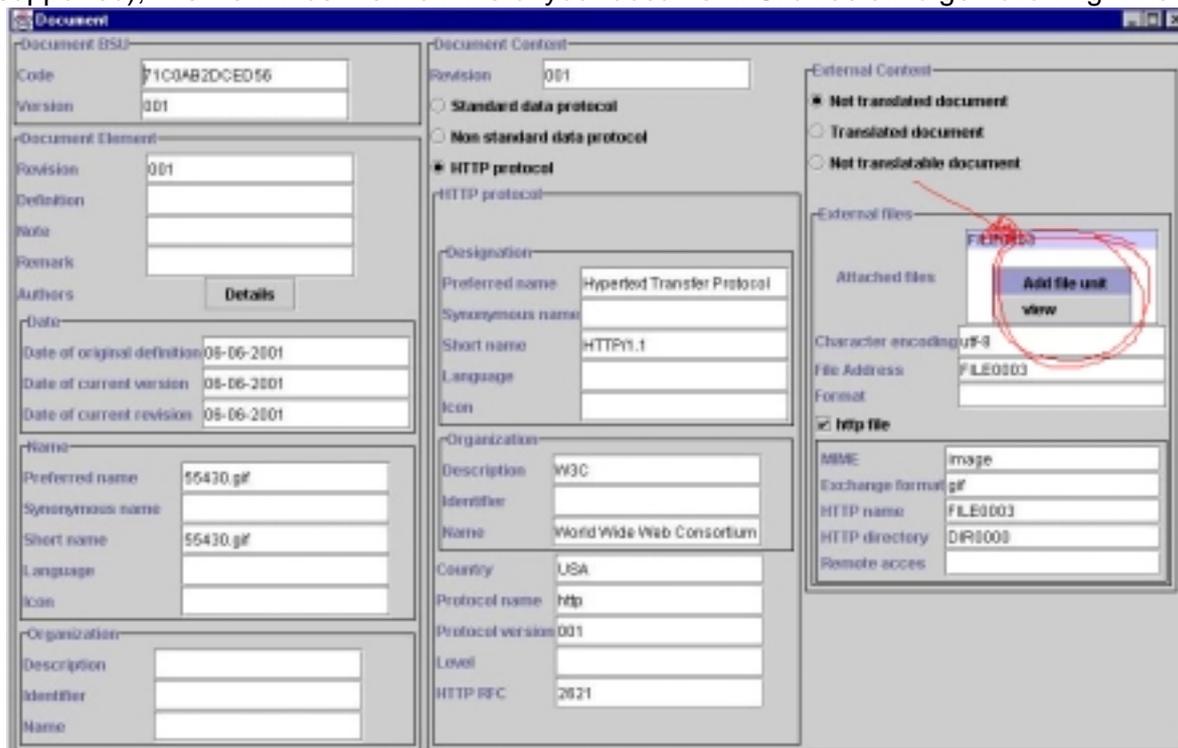
Images are scaled to fit presentation : right click an image to enlarge.



Now, we fill some extra fields : expand the part family panel by clicking the folder on the bottom left of the panel.



As for property domain, *Source doc of definition* attribute has two buttons to change and modify document (identified document = text identifier, referenced document = reference to external files). First, change document to build a new one and then click detail. Select referenced document : a file chooser appears and select your document (only HTTP protocol is supported); this file will be the main file of your document. Click detail to get following window.

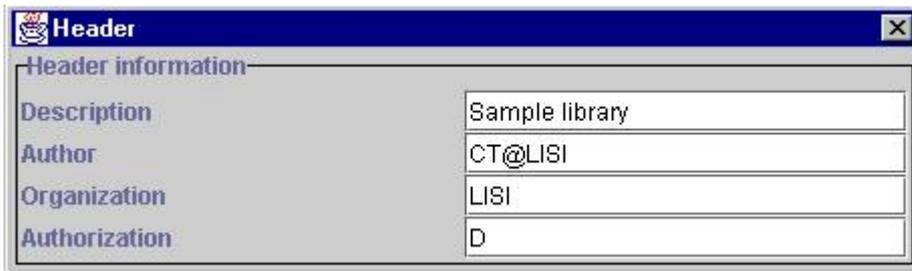


Referenced documents are complex entities; you can keep many default values. Document presentation is divided into three parts : from left to right, document element definition, protocol and external files informations. On the right hand side of the window, the list of external files of this document is available : right click to get a popup menu in order to add files. Operations for property are identical.

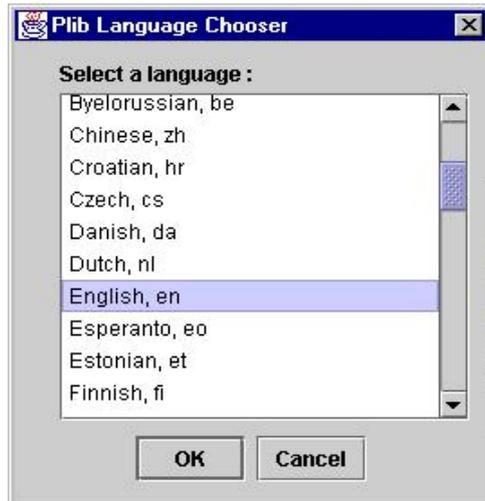
Property data

	Preferred name <input type="text" value="Property 2"/>	Code <input type="text" value="71C0AB2C8DFB2"/>
	Short name <input type="text" value="RP2"/>	Version <input type="text" value="001"/>
	Preferred symbol <input type="text"/>	Revision <input type="text" value="001"/>
Domain	<input type="text" value="INT_CURRENCY_TYPE"/> <input type="button" value="Details"/> <input type="button" value="Change"/>	Depends on <input type="text" value="no property"/>
	Definition <input type="text"/>	
	<i>Note</i> <input type="text"/>	
	<i>Remark</i> <input type="text"/>	
<i>DET Classification</i>	<input type="text"/>	<i>Date of original definition</i> <input type="text"/>
<i>Synonymous names #1</i>	<input type="text"/>	<i>Date of current version</i> <input type="text"/>
<i>Synonymous names #2</i>	<input type="text"/>	<i>Date of current revision</i> <input type="text"/>
<i>Synonymous symbols #1</i>	<input type="text"/>	
<i>Synonymous symbols #2</i>	<input type="text"/>	
<i>Source doc of definition</i>	<input type="text" value="NONE"/> <input type="button" value="Details"/> <input type="button" value="Change"/>	
<i>Formula</i>	<input type="text"/>	
		

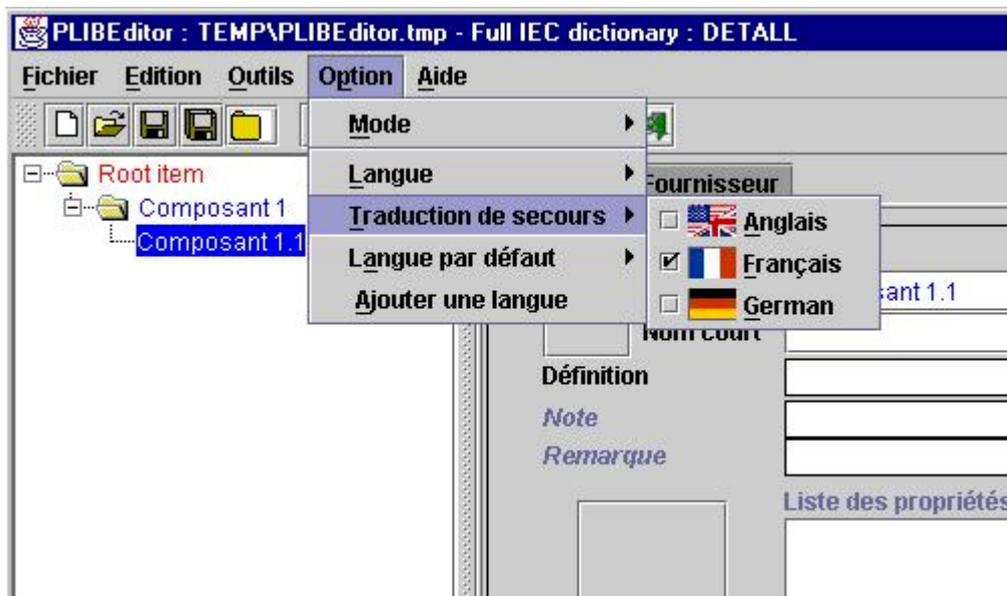
You can customize file header via *Edit – Edit file header*.



To support translations, the easiest way is now to use *Tools-Smart database translation*. You are asked to choose a language and all translatable entities are converted into translated entities with translation in the previously selected language taken from not translated values.



You can add other languages via *Option-Add language*, set the spare language (language to use for display when current language is not available), change the current language etc. (A color code is used and defined in Annex A).



Export function can generate a flat structure of external files for exchange. *Tools-List used HTTP directories* menu will help you in understanding how external files are stored. For more details, see Annex C.



Further development directions and known bugs:

Known bugs :

- When importing properties from secondary libraries, you need to save the primary library before you remove any secondary library used for imports.
- Error messages in standard out with no consequences in many cases.
- When hierarchical display is disabled in tree, you have to select the library node before expanding it. Otherwise the children will be those of the current library as expanding a node do not perform a selection.

This version is a beta version.

We would be pleased if we could get your feedback on the use of this tool.

Annex A: Color codes for translated library

General color code :

Black: no translatable data or current language translation.

Blue: translation not available for current language but the spare language translation exists and so is displayed.

Red: Nor current language, nor spare language are available.

Tree font style code(if tree font code is enabled):

Normal: All elements in subtree are translated in all languages.

Bold: Some elements in subtree are not translated in any language.

Property lists font style code:

Normal: Property that does not depend on another one.

Italic: Property that describes a condition for dependent properties.

Bold: Property depending on another one.

Remarks :

-Translatable data are data that can have translations or not.

-Translated data are translatable data for which translations exist.

-Not translatable data are data that do not have language support.

Example :

We consider an attribute T can be a TEXT or a TRANSLATED TEXT (i.e T is a TRANSLATABLE TEXT) and U an attribute that is not a TRANSLATABLE (such as Code, Revision, etc.).

Case 0 :

If U is not a TRANSLATABLE, it can only be an instance of TEXT. So no confusion can be done in reading this value : it will be always the same in whatever language. For this reason, U always appears black (as translated attributes that have translation for current language ; see below)

Case 1 :

If T is a TRANSLATABLE TEXT and is not a translated ; i.e T is an instance of TEXT and if a current language in application is selected,
Then T appears red.

Case 2 :

If T is a TRANSLATED TEXT ; i.e a TRANSLATABLE that is an instance of TRANSLATED. As TRANSLATED exist in library, current language is automatically sets.

If current language is English and T is translated in English,
Then T appears black.

If current language is English and spare language is French and T is translated in French only,
Then T appears blue in French (English is not available but spare language is).

If current language is English and spare language is French and T is translated in Japanese only,
Then T appears red in the first language added in library or is indeterminate.

Annex B: Property lists control

As list popups let user choose to display or not inherited properties, controls have to be done; some integrity controls are constrained via interface.

Two controls are done:

- You cannot make visible a property that is declared applicable in an ancestor node.
- You cannot make applicable a property that is already applicable in an ancestor node.

Short examples:

B is son of A.

Case 1:

A has P as applicable property : you cannot make P visible in B as applicability is inherited.

Case 2:

A has P as visible property and then B declares P applicable : you cannot make P applicable in A despite you change P as visible in B.

Annex C: External files exchange

A library may reference external files such as schematics, pictures, HTTP documents.

PLIBEditor supports HTTP protocol only : files are identified by their MIME type and are contained in a specific directory structure.

There are two ways to exchange these files.

First, all the directory structure can be sent : references between these files remain valid (till they are relative).

PLIB defines a mechanism to exchange external files with no directory structure : all files are exchanged at a same directory level and to browse HTTP document you need to rebuild HTTP directory structure.

Two procedures are implemented to exchange these files in any way despite PLIBEditor works with files located in the HTTP directory structure. *Export* converts HTTP directory structure into a flat structure : external file unit entities are updated in order to be able to rebuild HTTP directory structure. *Import* reads library and rebuild HTTP directory structure from a flat structure.

So when exchanging library you can send :

STEP file (*.spf) + external files located in the same directory as your SPF file (after *Export*) or,
STEP file (*.spf) + HTTP directories located in the same directory as your SPF file.

N.B: Use *Tools-List used HTTP Files* to get the list of referenced external files (some unnecessary files may exist in some directories as no file is physically deleted in this version).

Remarks :

External files with HTTP protocol are represented with entities that define a file address (flat structure) and a HTTP directory to support both logical and physical structure.

When you select files to include in your library, a copy is performed into the HTTP directory structure – and so no modification of the original file can be performed on the copy.

Annex D: Conformance class

APPLICATION OVERVIEW:1

MAIN WINDOW:.....1

MENUS:2

FILE MENU:2

EDIT MENU:4

TOOLS MENU:.....5

OPTION MENU:5

HELP MENU:6

TOOLBAR:6

INTERFACE7

TREE7

PART CHARACTERISTICS8

DEFINED TYPES10

CASE OF (REFERENCES BETWEEN LIBRARIES)10

CONTENT11

PLIBEDITOR FEATURES:.....13

OPENING A FILE:13

BROWSING :.....13

MODIFYING AND CREATING A PART LIBRARY:13

SAVING YOUR WORK:13

SOME TOOLS:13

LIMITATIONS :14

YOUR FIRST LIBRARY:15

CREATE A NEW LIBRARY:.....15

DEFINE A NEW PART FAMILY:16

ADD SOME PROPERTIES:.....17

ADD DERIVED PART FAMILY :19

ADD IMAGES INTO LIBRARY:.....21

FURTHER DEVELOPMENT DIRECTIONS AND KNOWN BUGS:26

ANNEX A: COLOR CODES FOR TRANSLATED LIBRARY A

ANNEX B: PROPERTY LISTS CONTROL C

ANNEX C: EXTERNAL FILES EXCHANGE D

ANNEX D: CONFORMANCE CLASS A