

# INCOSE - MBSE Initiative

## Model Based Document Generation User Manual

SE2-UM-01 ISSUE 1

2011-12-23

**Owner**                      **Michele Zamparelli** \_\_\_\_\_

**Manager**                    **Michele Zamparelli** \_\_\_\_\_

**Project Manager**        **Robert Karban** \_\_\_\_\_

Name

Date

Signature



## Authors

Name	Affiliation
Michele Zamparelli	ESO
Robert Karban	ESO
Chakajkla Jesdabodi	TUM

## Change Record

Issue	Date	Section / Paragraph affected	Reason / Initiation Documents / Remarks
1	2012-03-21	all	Initial Revision

# Table of Contents

1. Introduction .....	6
1.1. Scope .....	6
1.2. Purpose .....	6
1.3. Glossary, Definitions, and Conventions .....	6
1.4. Abbreviations and Acronyms .....	6
1.5. How this document was made .....	6
1.6. Copyright .....	6
2. Related Documents .....	8
2.1. Applicable Documents .....	8
2.2. Reference Documents .....	8
3. Overview .....	9
3.1. Concepts .....	9
3.2. Implementation .....	11
3.3. Advantages .....	11
3.4. A simple Use Case: adding a chapter .....	11
3.5. Model organization .....	12
4. Instructions .....	13
4.1. A note on the user interface .....	13
4.2. Adding a book to your model .....	15
4.3. Using the Edit Panel .....	16
4.4. Adding a part to the book .....	17
4.5. Adding a Bibliography .....	17
4.6. Adding a Revision History .....	19
4.7. Adding a chapter to a part or book .....	19
4.8. Adding paragraphs .....	20
4.9. Including external images .....	21
4.9.1. Adding comment text example .....	23
4.9.2. Setting Figure Size and Width .....	25
4.10. Adding internal references .....	26
4.10.1. Adding <token> references example .....	27
4.11. Adding Queries .....	31
4.11.1. Requirements .....	35
4.11.2. Operations .....	35
4.11.3. Properties .....	35
4.11.4. Documentation .....	36
4.11.5. Ports .....	36
4.11.6. Constraints .....	36
4.11.7. Flow Properties .....	38
4.11.8. Part Properties .....	39
4.11.9. Value Properties .....	39
4.11.10. Example Usage .....	40
4.12. Using Generic Tables .....	43
4.13. Adding Tables .....	45
4.14. Adding a Table Paragraph .....	45
4.15. Changing the ordering of elements .....	45
4.16. Generating the final Document .....	46
5. Installation .....	47
5.1. MagicDraw Plugin .....	47
5.2. Transformation XSL .....	47
6. Maintenance .....	48
6.1. Extension Procedure .....	48
6.2. Fault identification procedure .....	48
6.3. Diagnostic tools and procedures .....	48
7. Known Problems .....	49

---

8. Future Enhancements .....	50
------------------------------	----

---

# Chapter 1. Introduction

## 1.1. Scope

This document is the user manual for a software plug-in to the MagicDraw UML/SysML modelling tool. It also briefly addresses the Model Based Document Generation technique in general, and the rationale for such a plug-in.

This document assumes some familiarity with generic UML concepts (most notably stereotypes).

## 1.2. Purpose

This document explains how to install and use a software add-on to the *MagicDraw* UML/SysML modeling tool which allows users to extract a printable formatted document from a properly edited model. The parts of the model from which the document is extracted shall in general contain structure, prose and references to various types of system elements also present in the same artifact.

The modeling tool is MagicDraw, and the software is installed as a plug-in.

This documents explains how to use the document generation plugin for MagicDraw.

The plug in is used to transform a document, which is "stored" in a SysML/UML model into an XML file conforming with DocBook, for further usage. Most typically the document will be converted to PDF.

## 1.3. Glossary, Definitions, and Conventions

For a definition of book, chapter, section, part, paragraph etc. please refer to the documents in Section 2.2, "Reference Documents"

## 1.4. Abbreviations and Acronyms

MBDG     Model Based Document Generation

## 1.5. How this document was made

This document was generated from a SysML model archived in the Teamwork Server (RD3). The model is essentially an augmented version of the example model for a Hybrid Sport Utility Vehicle which the MagicDraw manufacturer ships with its product.

Notice that the SysML model does not need to be archived in a Teamwork Server for this software to work.

The HSUV model was chosen since it is fairly complete and contains elements which allow to show all features of the MBDG plug-in. Since in most cases only the rendering is shown in this document and no detailed explanation can be easily given, the reader is encouraged to look at the model directly to fully understand what is happening "behind the scenes".

## 1.6. Copyright

The software described in this document is subject to the following copyright statement:

\*

\*

---

\* (c) INCOSE SE2 Challenge Team for Telescope Modeling 2011

\* Copyright by ESO, HOOD, TUM, oose, GfSE

\* All rights reserved

\*

\* This library is free software; you can redistribute it and/or

\* modify it under the terms of the GNU Lesser General Public

\* License as published by the Free Software Foundation; either

\* version 2.1 of the License, or (at your option) any later version.

\*

\* This library is distributed in the hope that it will be useful,

\* but WITHOUT ANY WARRANTY; without even the implied warranty of

\* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

\* Lesser General Public License for more details.

\*

\* You should have received a copy of the GNU Lesser General Public

\* License along with this library; if not, write to the Free Software

\* Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

\*

\*

\*

## Chapter 2. Related Documents

### 2.1. Applicable Documents

#### Bibliography

[AD1] *SEWG glossary*. GEN-SPE-ESO-50000-0023 . Issue 1. 2010.

### 2.2. Reference Documents

#### Bibliography

[RD1] *DocBook 5: The Definitive Guide*. by Norman Walsh for DocBook V5.0+. . Version 1.1. Updated: 20 May, 2010.

[RD2] <http://www.docbook.org/tdg5/en/html/docbook.html>. . 5. .

[RD3] *ESO SDD Teamwork Server installation for MagicDraw*. . . .

[RD4] *Magic Draw User Manual*. . 17.0. .

---



## Chapter 3. Overview

In current (systems) engineering developments printed documentation is culturally the primary artifact for information interchange, including contractual purposes.

Recently though, system models have shown their usefulness for several system engineering activities and engineers have started maintaining descriptive models of systems using SysML, gradually enhancing them with details and enriching them with different points of views and perspectives.

Such models live therefore a separate, autonomous life with respect to printed documentation, and authors often find themselves having to copy & paste diagrams, or other information into documents.

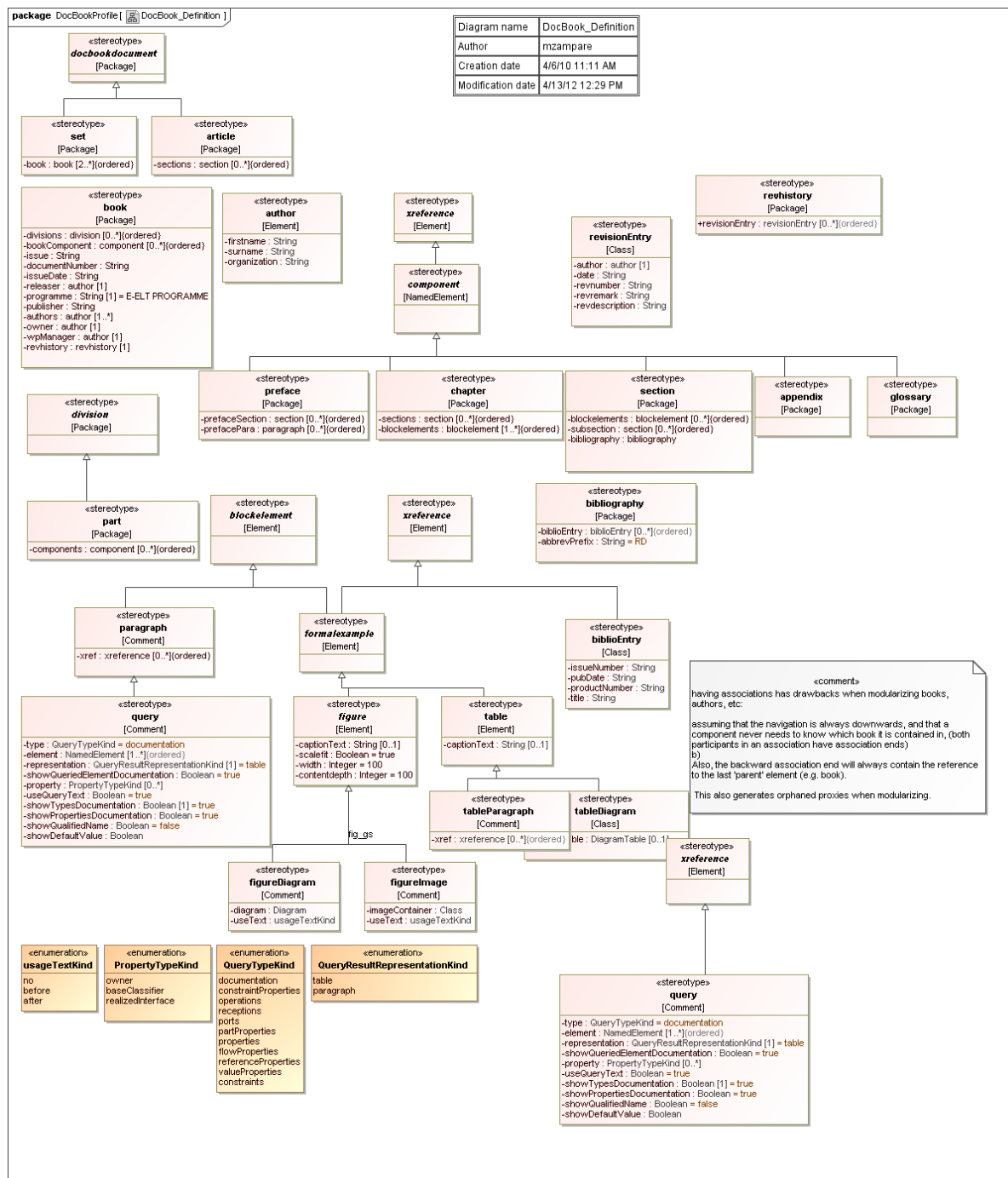
In other words, two sources of information (Model and Document) have to be maintained and get out of sync very easily.

Document generation engines (velocity template) have typically hard-coded references to model elements and diagrams, whereas the solution suggested here makes relocation, renaming, changes and even deletions in the model transparent to the final document artefact.

### 3.1. Concepts

A DocBook "book" is essentially a set of containers containing other containers or terminal elements which can be primarily textual paragraphs or queries to system elements (see Section 4.11, "Adding Queries" ). The ordering obviously matters for the final artifact, and is obtained from the ordering of the tags in the module.

---



### Figure 3.1. DocBook Definitions

The solution to the described situation consists in creating a model of the document, possibly, though not necessarily, in the same artefact (.mdzip file) where the system model is stored. Existing system model elements have then unique identifiers.

A subset of the DocBook ontology is mapped to model elements using a dedicated SysML profile. Figure 3.1, “DocBook Definitions” shows a class diagram of such a profile, where the stereotypes correspond to those elements which normally appear in a DocBook conform XML document. Stereotypes were created for a minimal subset of DocBook elements.

## 3.2. Implementation

Once the DocBook standard, maintained by OASIS, ( <http://www.oasis-open.org/home/index.php> ) is chosen as a basis for development, the functionality for code generation is implemented by means of a software plug-in to the modeling tool, using the tool's own customization facilities (open API). In addition, a SysML profile for DocBook was created, which is available for usage at ESO's Teamwork Server.

The suggested architecture is largely tool independent, the software presented here is a specialization for the MagicDraw case.

Stereotype tags are used to reference to system element models, text is entered in comments (which can be visualized in diagrams) and references to model element or diagrams can be selected using the tool navigation facilities.

## 3.3. Advantages

1. consistent integration of system model and system documentation
2. direct linking to model elements (also diagrams) from the document
3. changes of diagram names are automatically reflected in the document
4. using proper definition of the stereotype associations only compatible elements can be selected to compose the document, e.g. a figure references diagrams, a chapter references paragraphs.
5. the documentation is at the same time navigable in the model and printable
6. documents are modeled in a tool independent way.
7. A plugin is needed to generate DocBook XML.

## 3.4. A simple Use Case: adding a chapter

In order to add a chapter B to a book A, the following modifications are needed in the model:

1. Create a package B for the chapter, give it a name
2. Stereotype package B by <<chapter>>
3. Modify properties of <<book>> element A, by adding a reference to B, .e.g. modifying the tagged value to include B in its list

In order to increase the usability of the product, all these steps have been incorporated into actions, in this case the "SE2: Add Chapter" action, which is available through the plugin.

Users familiar with modeling tools will probably be acquainted with the issue that the containment tree (the "tree explorer" on the left of the main window) normally orders items according to a lexicographic ordering, and not in temporal one.

In other words, repeating the above steps for multiple chapters would soon lead to difficulties in figuring out which their respective ordering in the final document will be.

This problem is addressed in the Section 4.3, "Using the Edit Panel" , where a pseudo WYSIWYG display is presented.

---

### **3.5. Model organization**

- The document should be a MagicDraw project or module
- Create per document a single module
- Use the modules (describing you system) you need for the documentation
- Do NOT make a document part of your system module
- The document module shall depend on the system module but not the other way round.

## Chapter 4. Instructions

The following table clarifies which elements can be added and where, specifying the native SysML elements plus the stereotypes, and the tagged values which need to be modified. The **Addendum** is the type of docbook element which needs to be added, the **Container Stereotype** represents the stereotype which should be applied to the container for an addendum of that type to be added. In turn, the addendum is nothing but a native UML element of type **Addendum Type**. The Container Stereotype has a **Tag** where the Addendum is added to, by means of the **Action** specified in the last column.

Addendum	Container Stereotype	Addendum Type	Tag	Action
chapter	Book	Package	bookComponent	SE add Chapter
preface	Book	Package	bookComponent	SE add Preface
paragraph	Chapter	Comment	blockElements	SE add Paragraph
paragraph	Section	Comment	blockElements	SE add Paragraph
section	Chapter	Package	Sections	SE add Section
Revision History	Book	Package	revhistory	SE add Revision History
Revision Entry	revhistory	Class	revision	SE create Revision Entry
Biblio Entry	bibliography	Class	biblioEntry	SE create Biblio Entry

Table 4.1.

### 4.1. A note on the user interface

All the functionality supplied by the MBSE plugin is implemented in "Actions" which are triggered from the context menu from MagicDraw containment tree. Actions are available depending upon the type and permissions (read-write or read-only) on the currently selected element.

For example, the action to create a Book element will not be available when a chapter is selected, whereas the vice versa (creating a chapter) is true, if the book is currently editable.

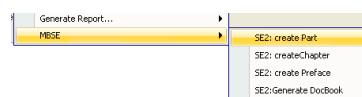


Figure 4.1. contextMenu

Each element when created can be customized in its specification dialog. The specification dialog will have a sub menu called "MBDG", under this menu the user can edit different tags of those element, for e.g. for Book user can specify its name, components and authors etc. Also "MBDG" sub menu will appear in the quick properties menu (under the containment tree).

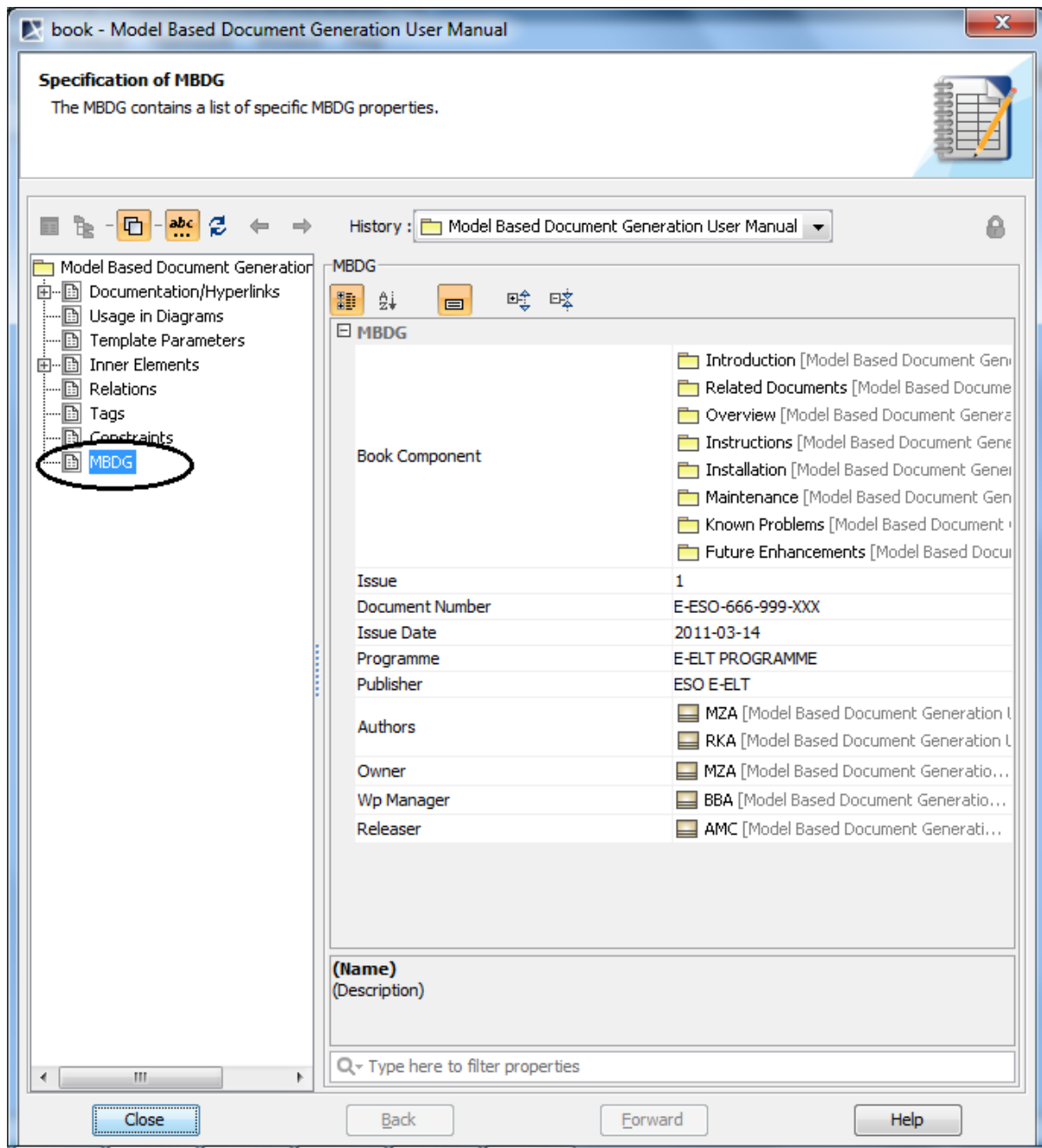


Figure 4.2. specificationDialog

The following example shows how Authors components can be added to a Book. The specification dialog will show matches of possible components to be added e.g. (23 matches). The user can toggle between multiple selection or single selection. Recursive add of elements in a folder can also be done by selecting "Add recursively" option.

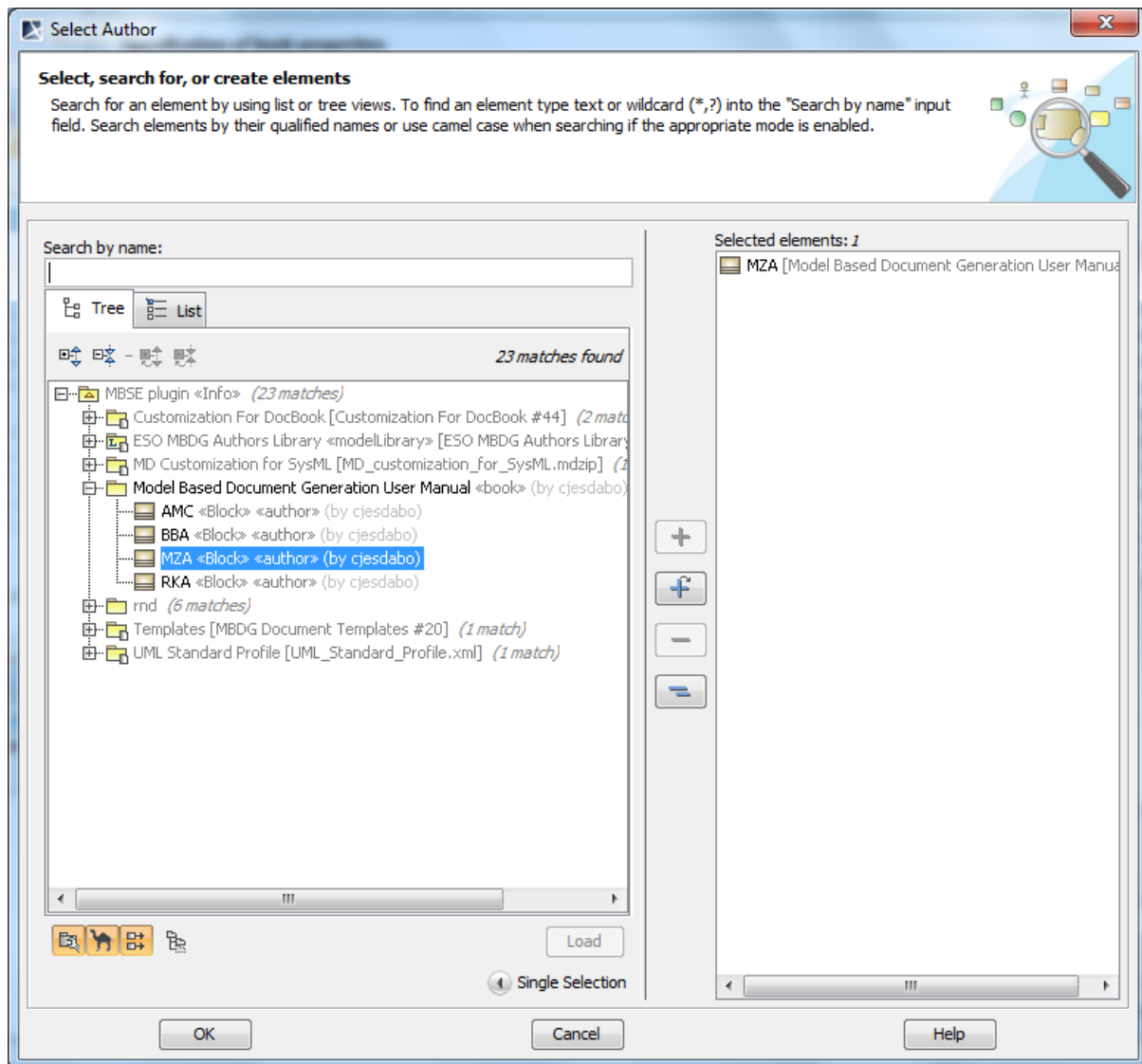


Figure 4.3. exampleMulti

## 4.2. Adding a book to your model

In order to add a book to your model, simply look for a package where you have write permission (e.g. any package, if you're not using teamwork server for configuration control) and right-click in the context menu. The action "SE add book" should appear in the list in the MBSE submenu.

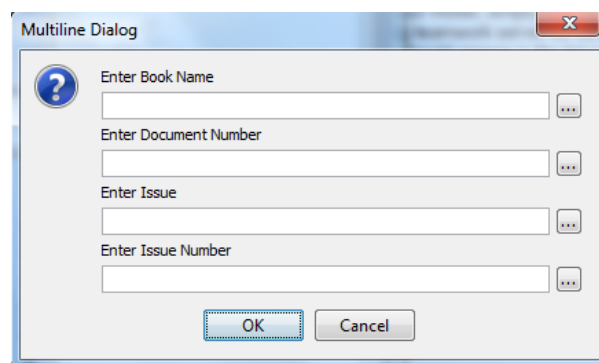
The book stereotype in the DocBook profile has a variety of tags which get rendered primarily on the front page of the resulting PDF document.

These are:

- Document Number: self explanatory.
- Issue Date: the issue date for the document. Issue: a string containing the number of the issue for this document/book
- owner: the document owner, rendered on the front page.

- Work Package Manager (WPManger) : a reference to an element stereotyped by <<Author>> which represents the Work Package Manager.
- releaser: this is rendered with the corresponding field after Owner and Work Package Manager. In the ELT xsl template it is rendered to Head of Project Office, in the SE2 xsl template it is rendered to Releaser.
- Authors: the complete list of authors rendered on the second page. These tag contains references to elements stereotyped by <<Author>>.
- publisher: this string is used by the rendering for the header and footer for any page but the first one. (in this model it is set to "SE2 Challenge Team")
- Programme: a string which appears on the front page, above the title of the document. (in this model it is set to "INCOSE - MBSE Initiative")

Notice that the title of the document is the name of the element of type <<book>>. The following diagram shows the dialog that will appear when the user click SE add book option. Additional information like Programme, WP Manager or Authors can be added from the "MBDG" sub menu in specification dialog of the book



**Figure 4.4. bookDialog**

### 4.3. Using the Edit Panel

The *Edit Panel* is an extended feature of the MBSE plugin which provides a user interface elements to preview the document before generating the final PDF file. The Edit panel allows operations such as re-arranging the order of the elements (e.g. chapters, sections, paragraphs) under the same parent or different parent; movements may be performed via drag and drop actions within the limitations provided by the DocBook profile (e.g. a paragraph cannot be moved under a bibliography element). The preview shows how the document will look like when it is generated as a PDF, and also includes scaled diagrams and tables.

The Edit Panel for a Book element can be by invoking "SE2: Show Edit Panel" command in the MBSE plugin for Book Element. The generated preview allows the user to navigate through the documents by a document tree; sublevel contents (chapters, sections) can be opened hierarchically. Th Edit Panel also provides a content pane (right side) which illustrate each element information in more detail. The preview information in the content pane is updated automatically when the user updates the contents in the model for e.g., changing a name of a chapter or section, or changing the content of a paragraph etc. Also delete and undo operation is supported by the content pane. Elements like figureImage, figureDiagram and figureTables etc. are rendered in the content pane in a scaled form with its description, this is just to provide the user with an idea of how the PDF file will look like.



User can perform a right click action on each element to invoke the MBSE actions (context menu), for e.g. right clicking on a Section will give you an option for creating paragraph as show in the following figure image, query etc., this applies for other elements as well. Double-clicking on composite element like Chapter and Sections in the content pane will cause the element to expand/collapse.

The Edit Panel also provides a search functionality with case sensitive option, search can be done by typing into the search box and browsing through search results by next and previous button.

For the first version of the Edit Panel, a "refresh" button is provided and should be used when a user tries to reorder the block elements in the model which does not get reflected in the Edit Panel.

The Edit panel does not take into account the information generated by the queries for e.g. some queries that generate a table of paragraph are not shown in the Edit Panel and can only be observed in the final PDF file.

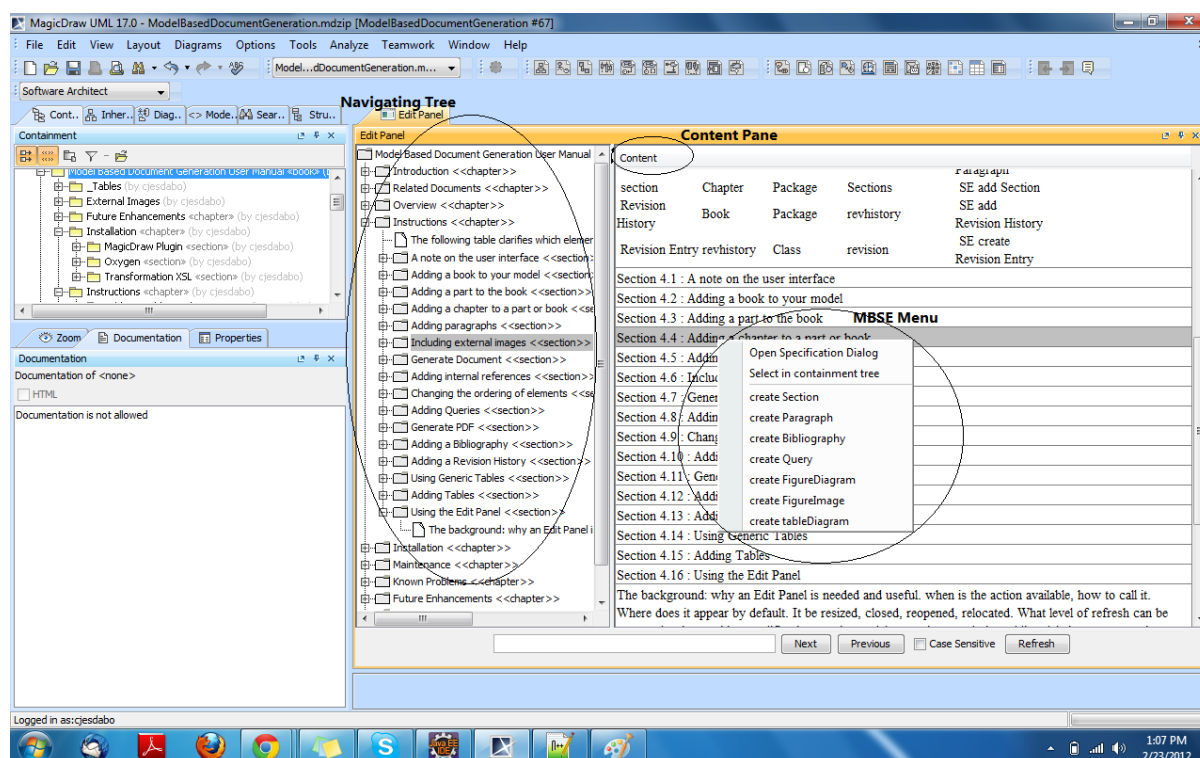


Figure 4.5. editPanel1

## 4.4. Adding a part to the book

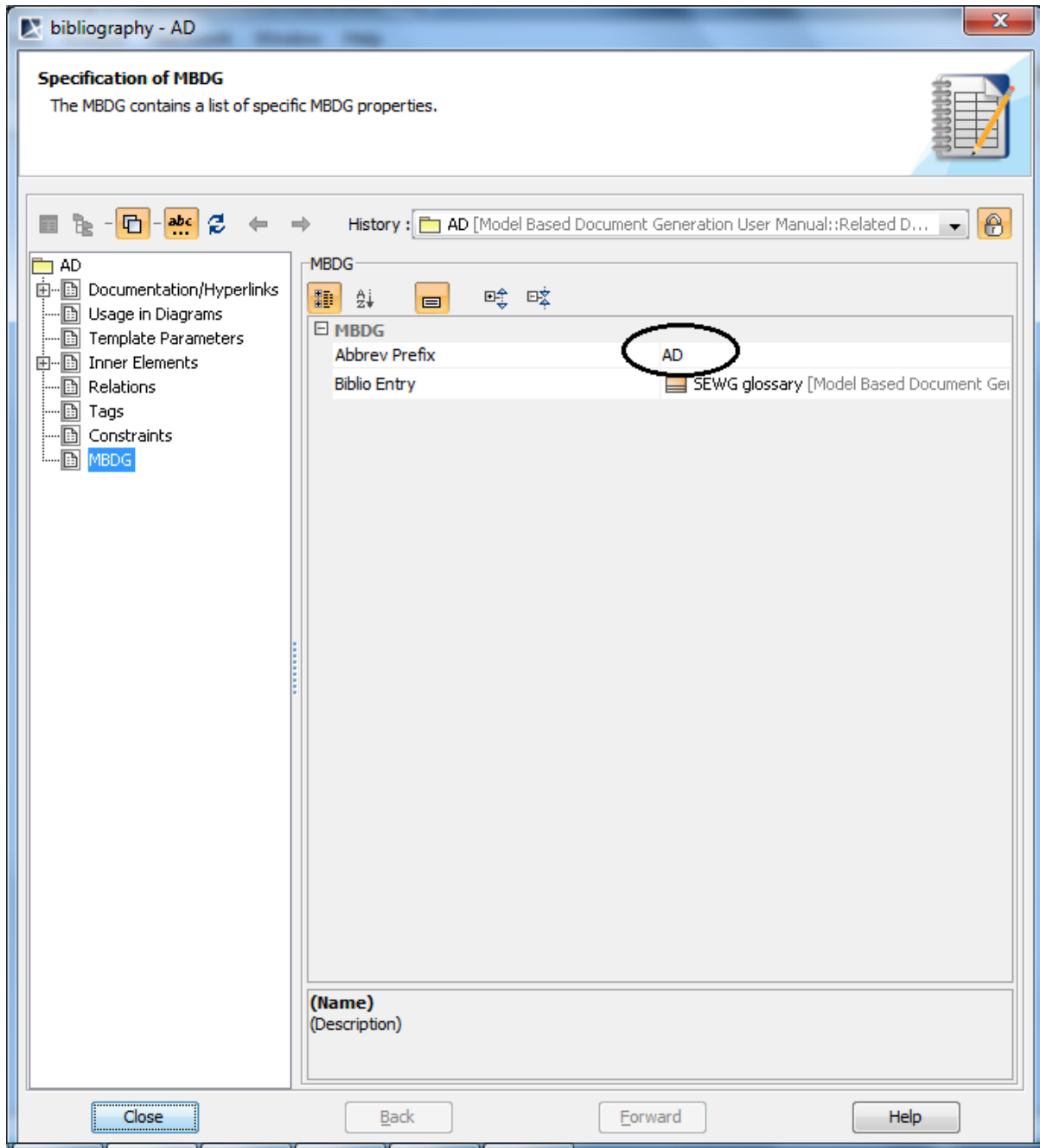
Part can be added under a Book element by selecting SE2: create Part option under MBSE menu (see Figure 3.1, "DocBook Definitions" ).

Part can contain subsections and paragraphs. User can specify part name and its components.

## 4.5. Adding a Bibliography

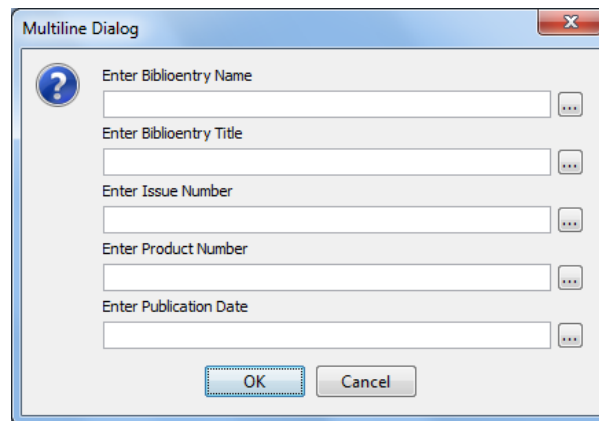
A Bibliography specify the referenced documents for a book. A bibliography element can be added under a Section element by selecting "SE2:createBibliography". A dialog will appear where the user

can specify the bibliography name. A Bibliography can contain many biblioentry elements. The user can specify the Abbreviation of each biblioentry under the bibliography by editing the Abbrev Prefix tag under MBDG sub menu in the specification dialog. In the following image, AD is set as the abbreviation,



**Figure 4.6. bibliographyDialog**

A Biblioentry can be added to a Bibliography by selecting "SE2:create Biblio entry" option. The following dialog will appear where the user can specify a biblioentry details.

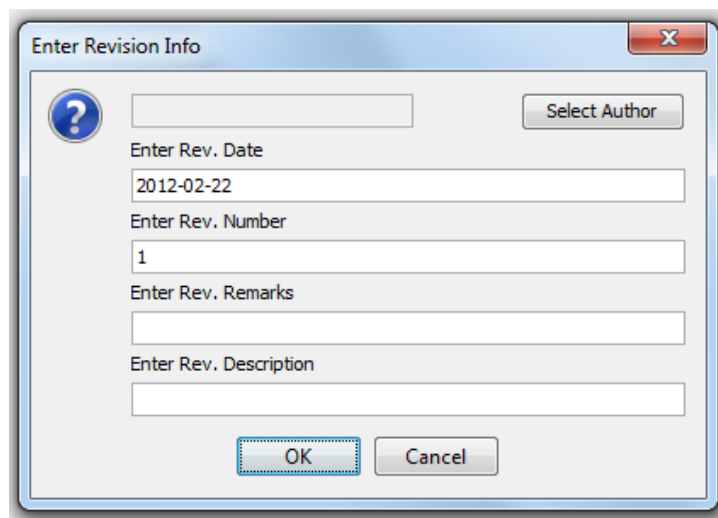
The image shows a 'Multiline Dialog' window with a blue title bar and a red close button. On the left is a vertical toolbar with a question mark icon. The main area contains five text input fields, each with a label and a small '...' button to its right. The labels are: 'Enter Biblientry Name', 'Enter Biblientry Title', 'Enter Issue Number', 'Enter Product Number', and 'Enter Publication Date'. At the bottom are 'OK' and 'Cancel' buttons.

**Figure 4.7. biblientryDialog**

## 4.6. Adding a Revision History

A Revision History is a list of the existing revisions for a book. A Revision History element can be added under a Book element by selecting "SE2: createRevisionHistory". A Revision History can contain many Revision elements. There is only one Revision History per book.

A Revision can be added to a Revision History by selecting "SE2: create Revision Entry" option. The following dialog will appear where the user can specify the Revision details.

The image shows an 'Enter Revision Info' dialog window with a blue title bar and a red close button. On the left is a vertical toolbar with a question mark icon. The main area contains several input fields: a text field at the top, a 'Select Author' button, a date field with '2012-02-22', a text field with '1', a text field, and another text field. Labels for the date, number, and the two text fields are 'Enter Rev. Date', 'Enter Rev. Number', 'Enter Rev. Remarks', and 'Enter Rev. Description' respectively. At the bottom are 'OK' and 'Cancel' buttons.

**Figure 4.8. revisionentryDialog**

By clicking on the "Select Author" button a pop-up window will be opened to choose existing elements in the model which are stereotypes with «author». The name of the selected author will then appear on the left field.

The issue number for the entire book is taken as a default for the Revision Number field.

Revision date, remarks and description may be entered as needed, the latter being used in the rendering to PDF to list the sections/chapters which have been modified in this revision.

## 4.7. Adding a chapter to a part or book

A Chapter is the basic element of a Book. It can contain queries, figures (diagram or image), paragraph and subsections. Chapter can be added by selecting SE2: createChapter option under MBSE menu.

## 4.8. Adding paragraphs

Paragraphs are the primary docbook element for holding prose text

Paragraphs may be added using the "SE2: create Paragraph" action from the Context Menu. Or, as for any other DocBook element, may be added editing the specification of the corresponding containing element, by opening its specification, searching for the *blockElements* tag and using the buttons to create new elements.

For example, adding a Paragraph to a Section would entail:

- creating a Comment element
- Stereotyping it as paragraph
- adding the Comment element to the tag "blockElements" of the Section

As already mentioned. this tedious work can be automated using the Actions from the user interface.

Paragraphs can be edited in plain text or using the HTML editor of MagicDraw.

The HTML editor provides more text formatting capabilities like:

- **Bold text**
- *Italic Text*
- Underlined

It also allows nested ordered lists in a paragraph:

1. Item 1
2. Item 2
3. Item 3

Or even nested bullets and ordered lists:

- b1
  - b1.1
  - b1.2
- b2
  - b2.1
  - b2.1

1. item 1
-

- a. item 1.1
  - b. item 1.2
2. item 2

Simple tables can be created with the HTML editor within a paragraph.

However, they cannot be referenced with a x-reference. For x-reference a separate table stereotype element must be used as described in Section 4.13, “Adding Tables” .

element 1.1	element 1.2
element 2.1	element 2.2
element 3.1	element 3.2
<ul style="list-style-type: none"> <li>• item1</li> </ul>	1. item 1
<ul style="list-style-type: none"> <li>• item2</li> </ul>	2. item 2

**Table 4.2.**

## 4.9. Including external images

The addition of external images involves multiple steps.

A Class or Block element needs to be created. An external image may be then assigned to it, simply by dragging and dropping it from the desktop or from a web browser, or using the selector from the specification panel. In both cases, the image gets added to the 'image' property of the Class.

In order then for the external picture to appear in the final outcome, a figureImage elements needs to be created in the chapter or section, using the available UI commands for section.

The SE2: create Figure Image first asks for a figure caption, and then pops up a selector for the Class itself. The figure caption is used for the title.

The figureImage element is rendered in XML and PDF with the image it refers to.

It is recommended to put all external images in a dedicated package.

## Example of a Hybrid SUV

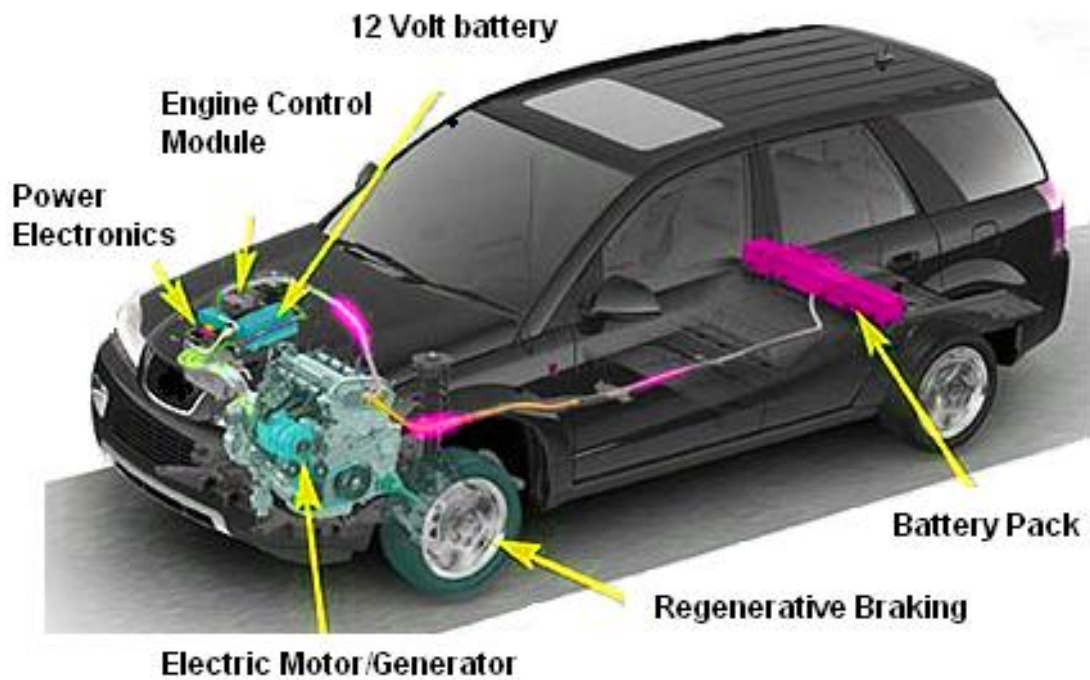


Figure 4.9. HSUV img

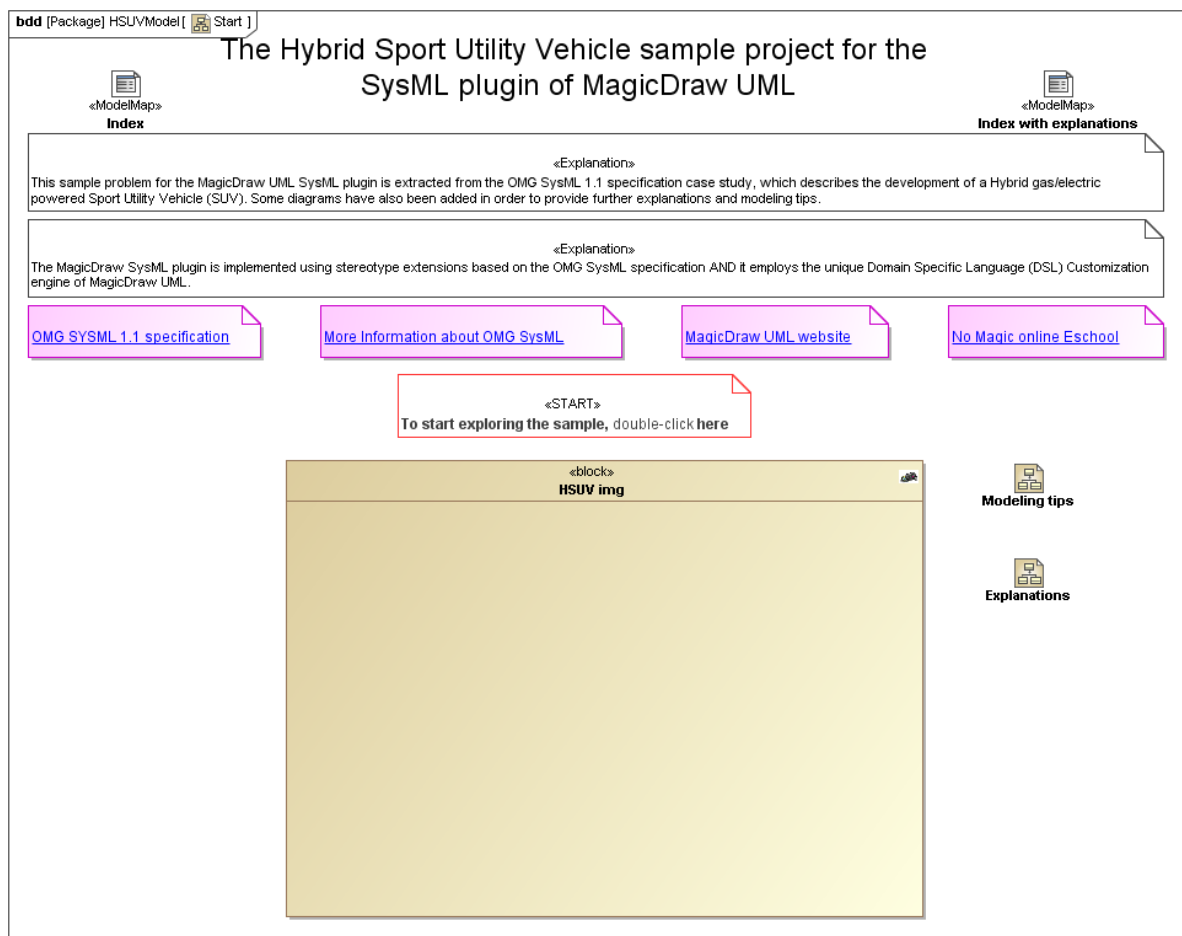
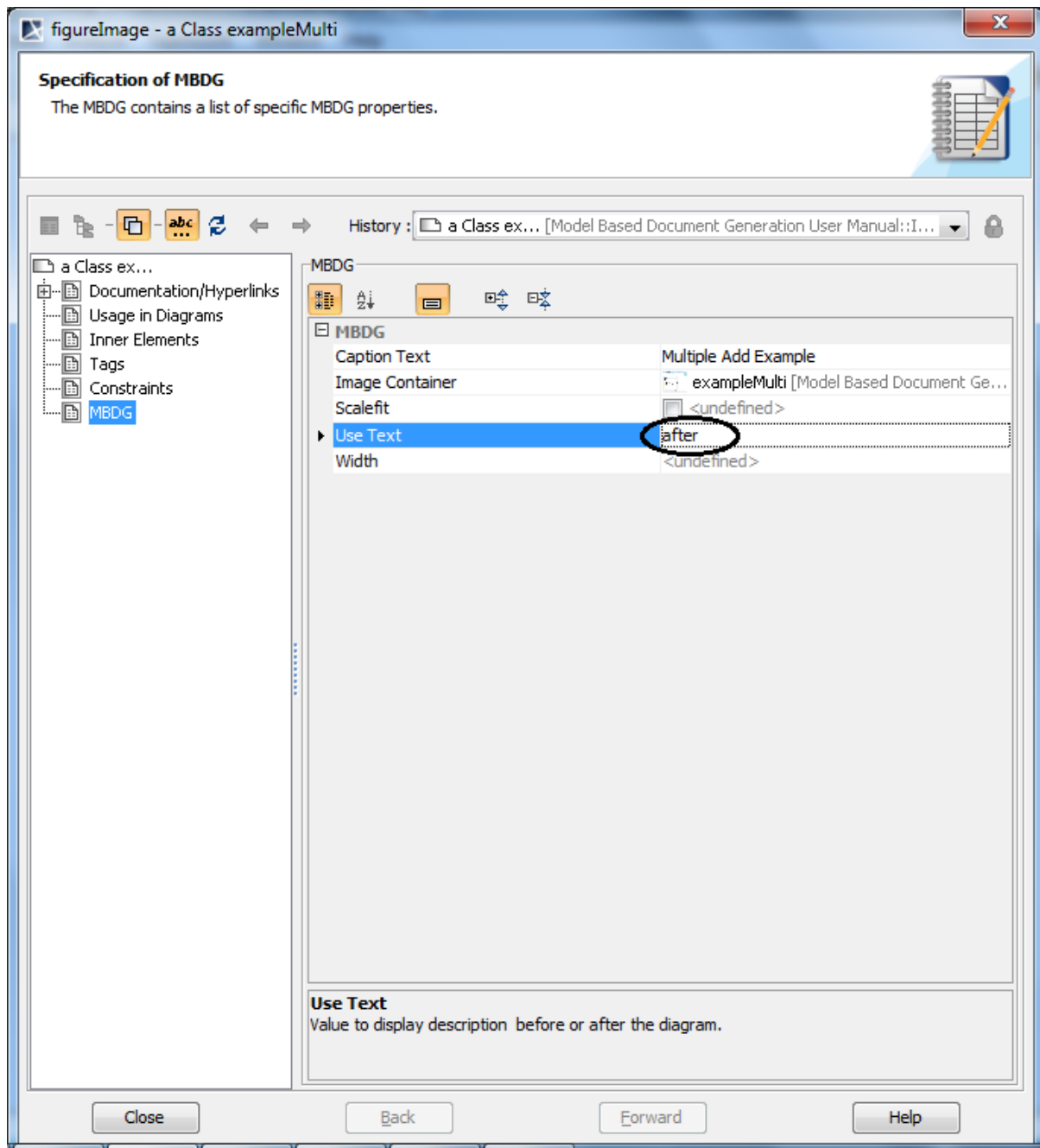


Figure 4.10. a figure diagram

#### 4.9.1. Adding comment text example

A comment text can be added to a figureImage or figureDiagram by

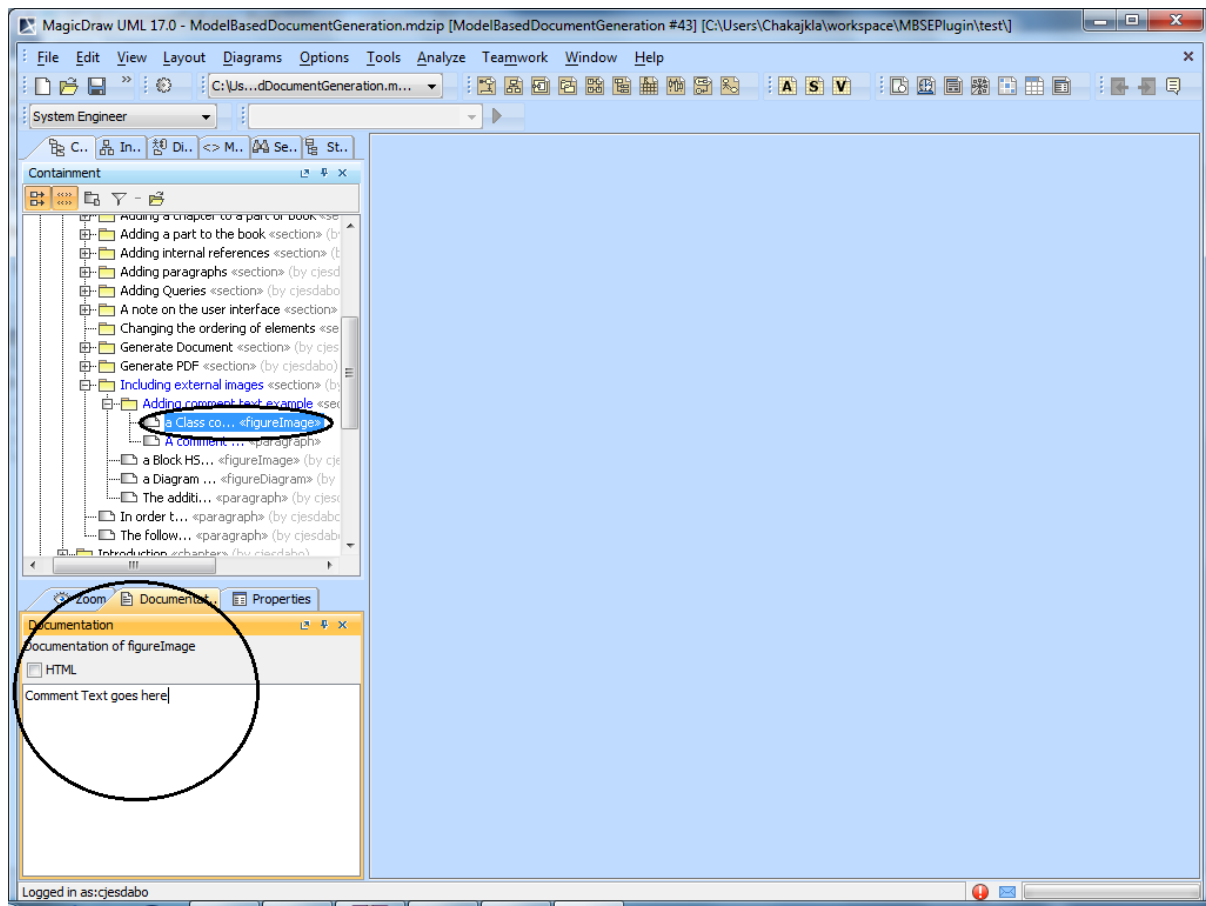
1) specifying the comment value in the figureDiagram / Image documentation. And 2) setting the appearance of this comment text , to make the comment appear before or after the diagram, like the given image below.



**Figure 4.11. commentText**

Use after value set, this image shows how the appearance of the comment text can be customized to appear before or after the figureImage/ Diagram by setting the Use Text tag values accordingly.





**Figure 4.12. commentText2**

Comment text goes here, this image illustrates how a comment text can be added by simply editing the documentation text of the figureImage element.

#### 4.9.2. Setting Figure Size and Width

A figureImage or Diagram can be set to scalefit or increase or decrease its dimension. This can be done by setting the Scalefit tag or Width tag value. For e.g. to reduce the image size by 50 percent we can set the Width tag value to 50. like the image below.

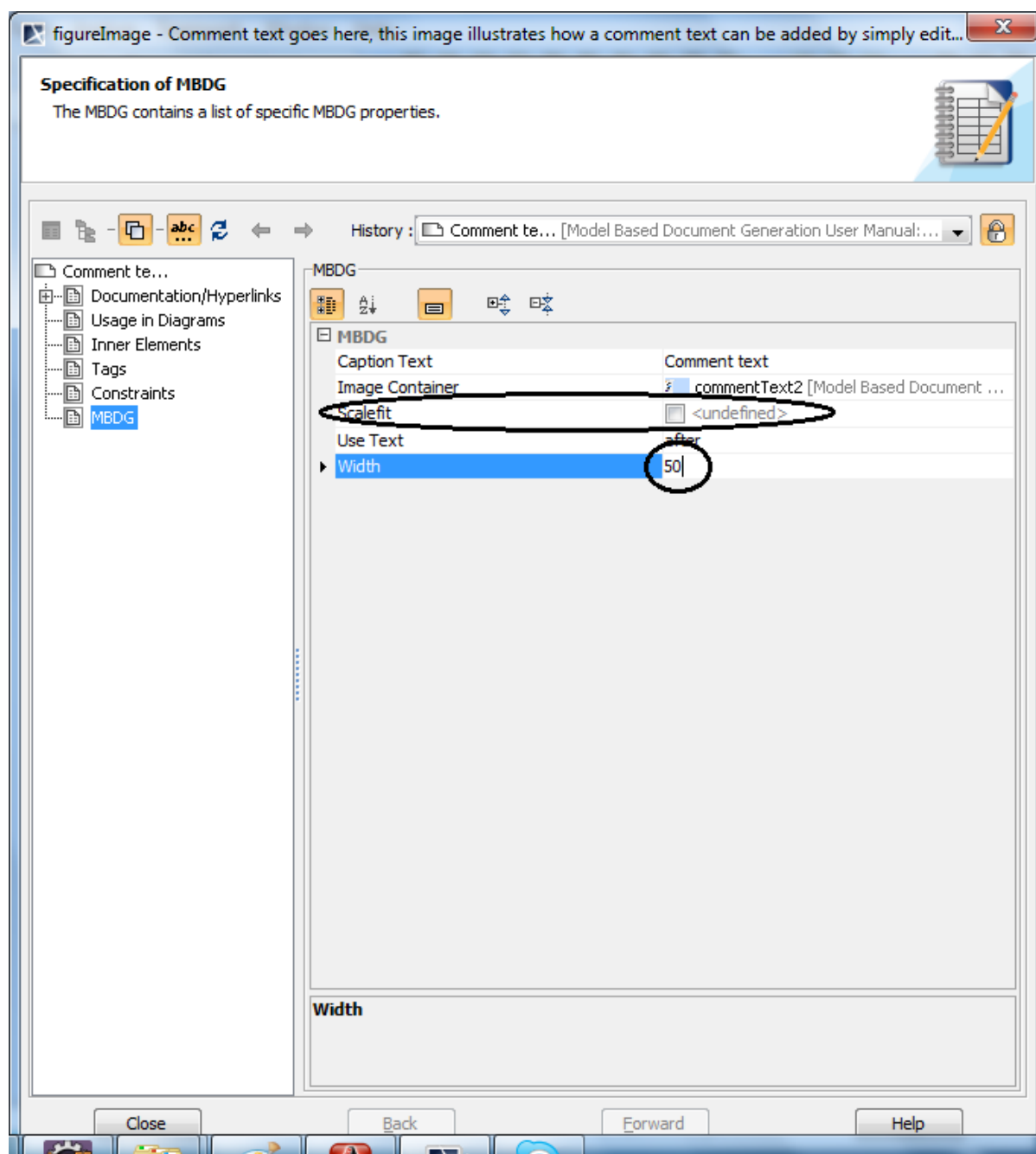


Figure 4.13. widthDialog

## 4.10. Adding internal references

Any model element which is stereotyped with *xreference* can be referred to. During generation, each document element besides paragraphs is given a unique ID which corresponds to its location in the containment tree.

In order to add one or more references from within a paragraph, the keyword `< token >` (hereafter always written with space characters, only necessary in order for it to appear properly in this manual and not to be used otherwise) will have to be added as many times as references are desired. Correspondingly the xref tagged value in the paragraph will need to refer to the desired elements.

The following image shows the Xref tag under MBDG sub menu in the specification dialog of a paragraph. User can edit the value of Xref tag to reference as many elements as possible (user must note that the element to be added must be labeled "matches" in the dialog). For e.g. if the Xref tag has 5 xref values, meaning the user has referenced 5 elements, then in the paragraph text, the user must add the keyword < token > 5 times, in the same order as when the elements were referenced. This way 5 references will be created in that paragraph when the user generates the DocBook.

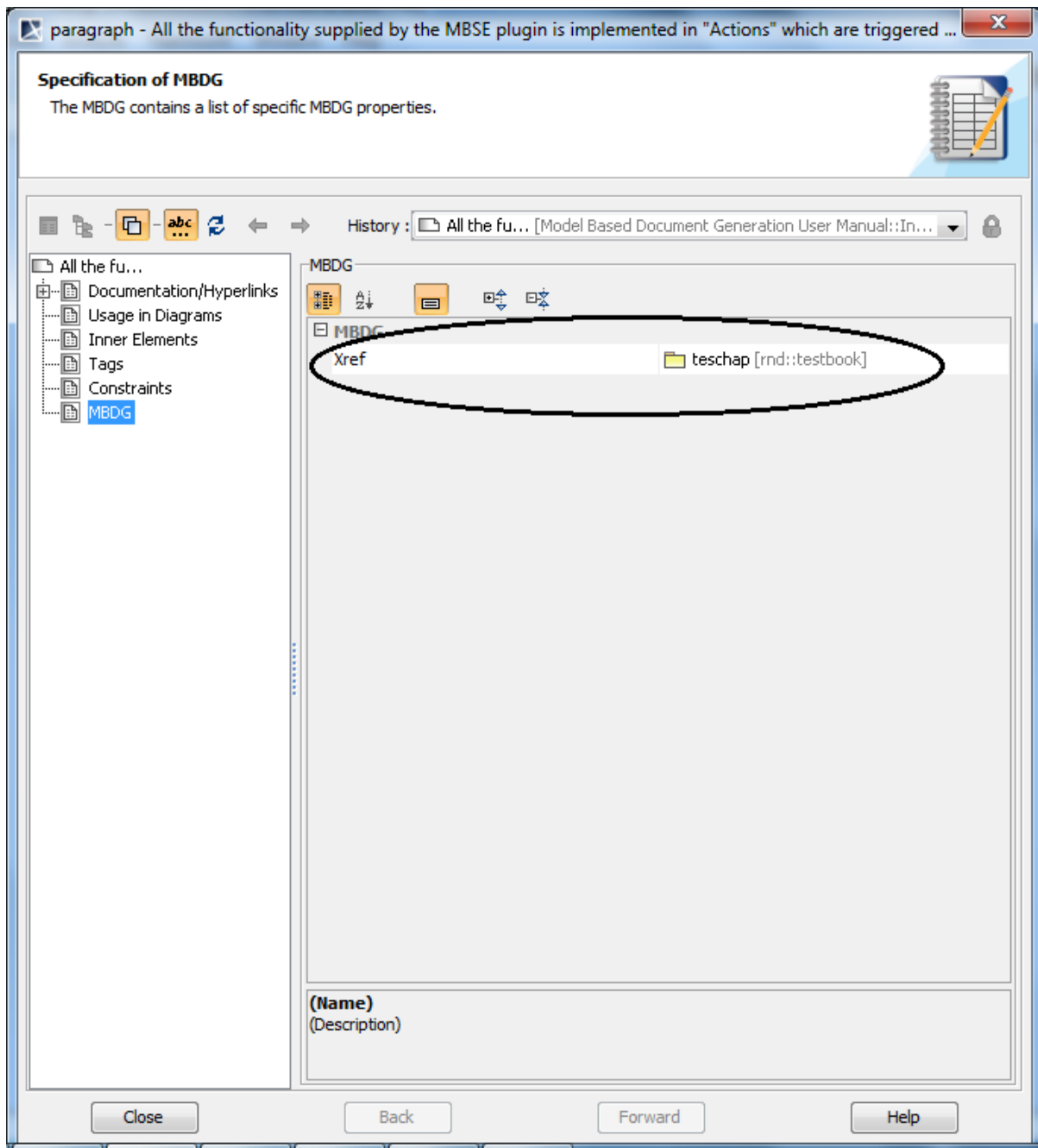


Figure 4.14. xrefDialog

#### 4.10.1. Adding <token> references example

In this example we will illustrate a step by step guide on how a < token > reference can be created inside a paragraph. This reference mechanism can be done against tables, figures or biblioentry etc.

The example will show how a biblioentry can be referenced.

Step 1: Create a paragraph element

This can be done by selecting "SE2: create Paragraph" option under the MSBE menu.

Step 2: Add Xref values

Open the paragraph's specification dialog, and click add values for the Xref tag like the image below.

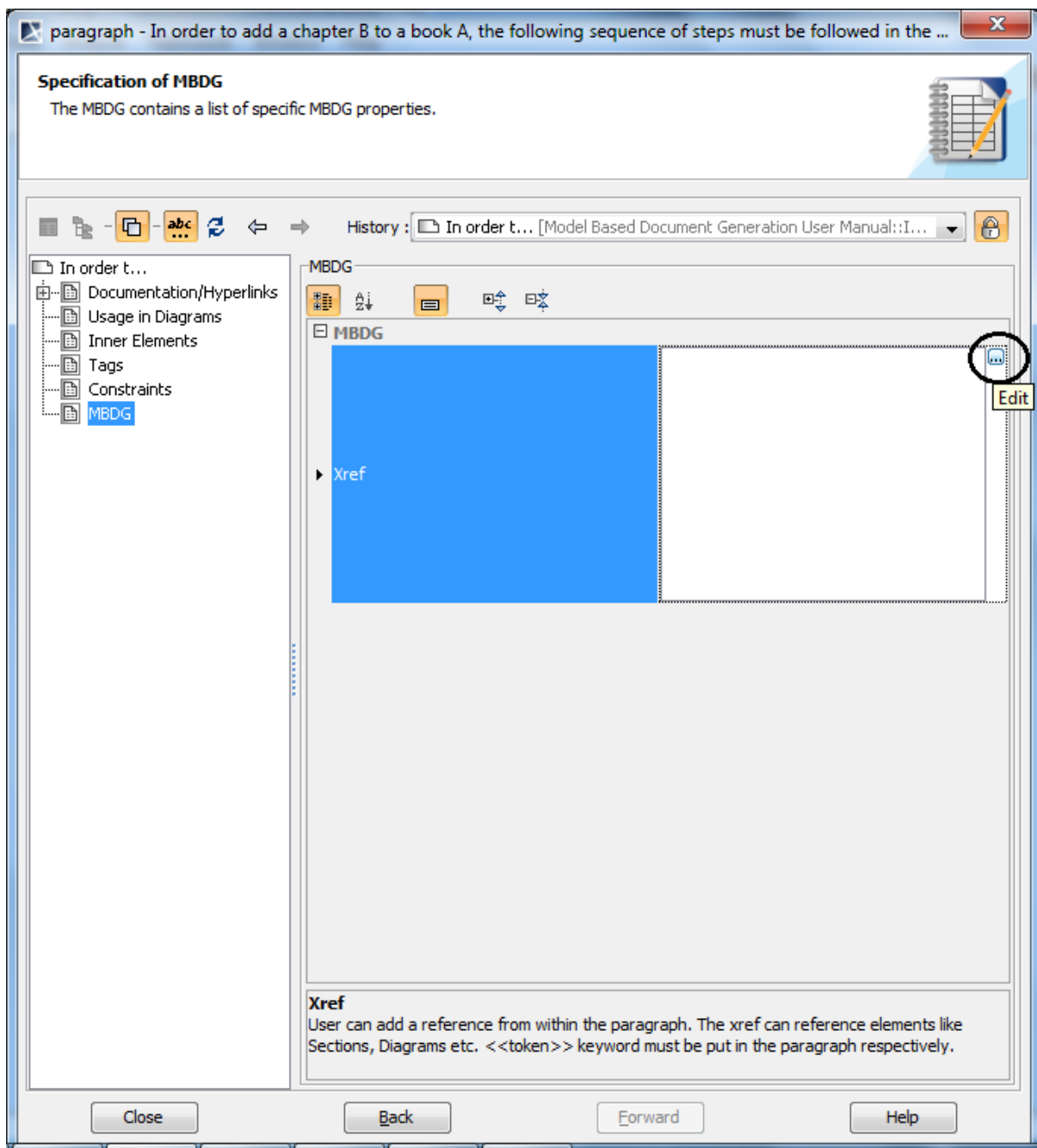
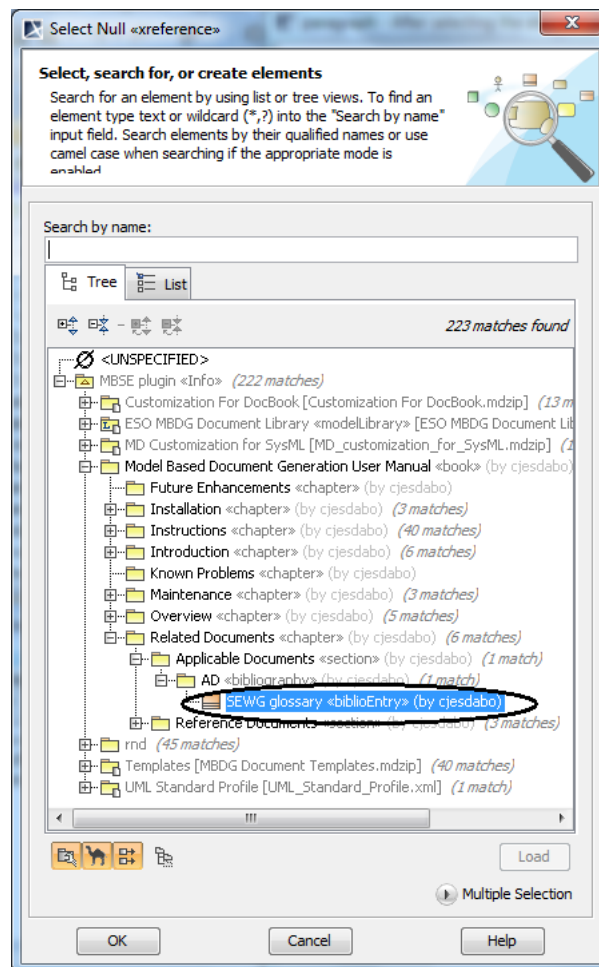


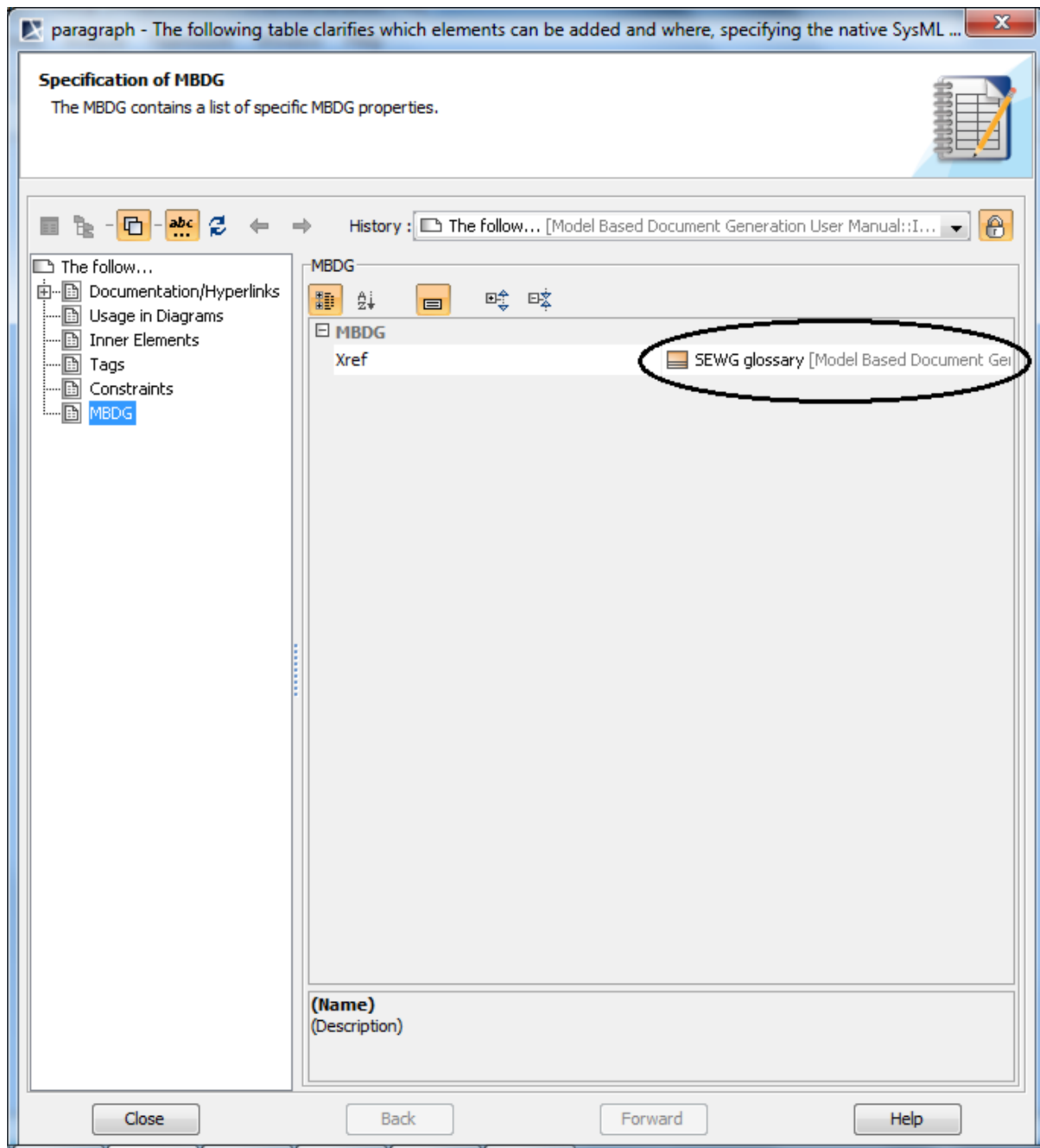
Figure 4.15. Edit option of Xref Tag

After selecting Edit option , a new dialog will appear where the user can browse different elements to be referenced, in our case we will select a Biblioentry element.



**Figure 4.16. Dialog for choosing referenced element**

After selecting the element you will notice that in the paragraph specification dialog, Xref tag will have a new referenced value like the image below.



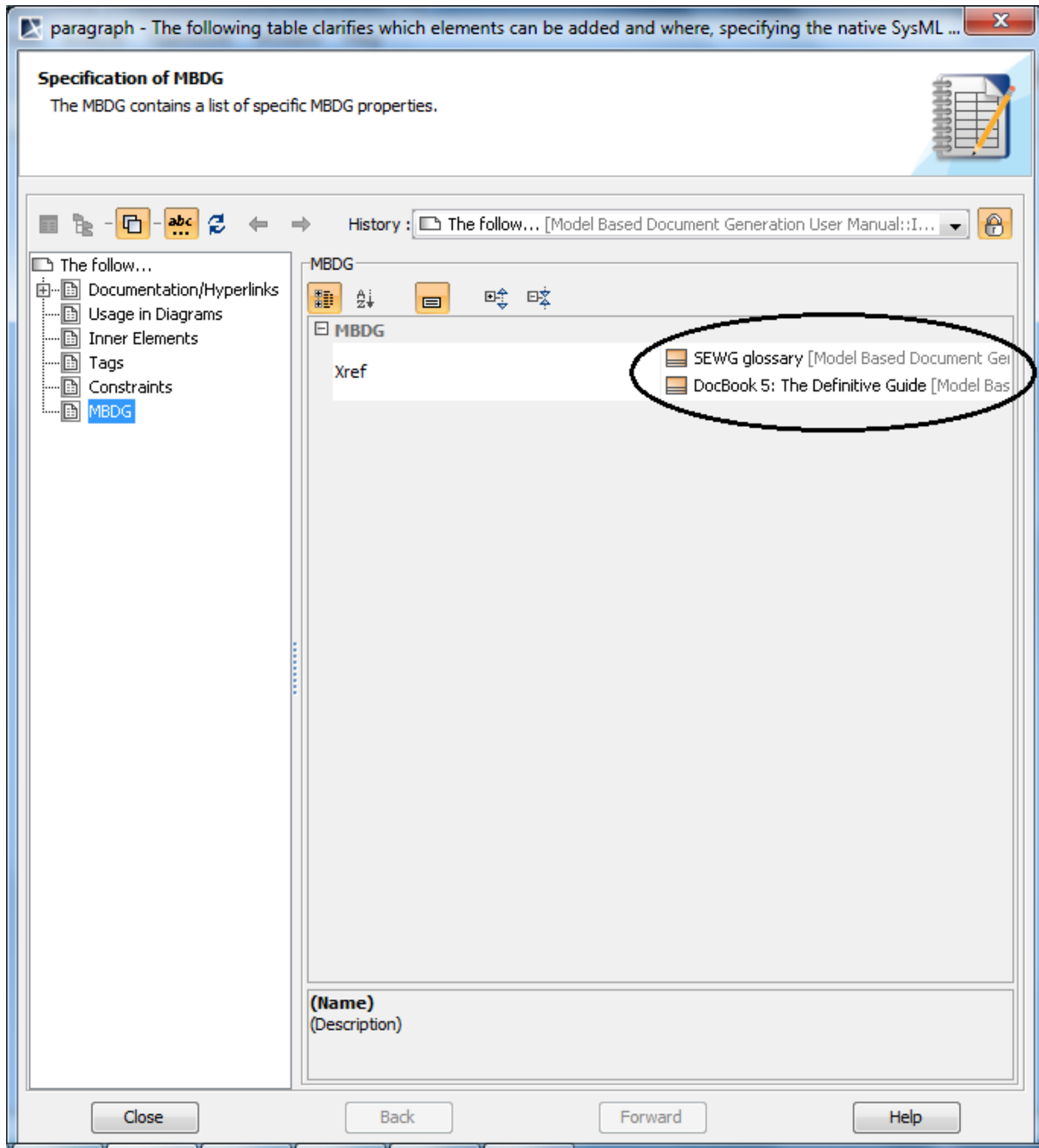
**Figure 4.17. Xref value added**

Step 3: Add < token > tag

In the paragraph where you have just added the Xref tag value. Add a keyword < token > in the paragraph text where you want to reference the biblioentry from.

Step 4: Adding multiple < token > tag

Multiple references from a paragraph can be done by adding the keyword < token > as many times as the number of your referenced values. Example, if you repeat step 1 - 2, and you have added multiple Xref values like so



**Figure 4.18. Multiple Xref Values, two biblioentries to be referenced**

We have added two biblioentry references. Now we must add the keyword `< token >` 2 times in the paragraph text. For e.g. This is a reference `< token >` (this is the 1st reference), this is also a reference `< token >` (this is the second reference).

In this way the user will be able to create internal references, which can be done not only with a biblioentry but also a section, chapter, figures etc.

## 4.11. Adding Queries

A *query* is a mechanism to extract information from several types of system model elements and render it in document's prose.

Support exists for different types of queries (operations, ports, constraints) and visualization styles (table or paragraph), as shown in diagram Figure 3.1, “DocBook Definitions” , where the characteristics of queries are displayed in more detail.

The following fields must be specified when adding a query:

element	the system model element whose characteristics the query will report upon.
type	the query type: can take up the values: <b>operations, ports, documentation, constraints, properties</b> .
useQueryText	a boolean specifying whether the text of the query (entered as a body of the corresponding Comment element) shall be used in-line in the document.
representation	this may be either tabular or paragraph, the meaning is self-explanatory.
property	this can take up the values: <b>owner, baseClassifier, realizedInterface, classifierBehavior</b> . This is <u>only</u> used when the type field is set to <b>properties</b>

**Table 4.3.**

If the named referenced element is a requirement, it is handled as such and the value of query type is entirely ignored.

As specified in Figure 3.1, “DocBook Definitions” , queries are a special type of paragraph, and may be added in all places where a paragraph is allowed and may be intermixed with paragraphs with no restrictions. Additional query types (like listing all elements represented in a diagram) are possible but not implemented at this stage.

When queries are created using the Actions from the context menu, a selector pops up to allow the user to navigate the system model and select the element to be queried. The latter may be a Figure, a Requirement or a Block. The other fields of a query must be instead entered using the specification dialog for the query itself.

The elements used in the following examples are the Block WirelessTire Pressure Monitor and PowerControl Unit, from the HSUV Model used as an example.

The following two Figures show two diagrams in which the elements appear respectively.



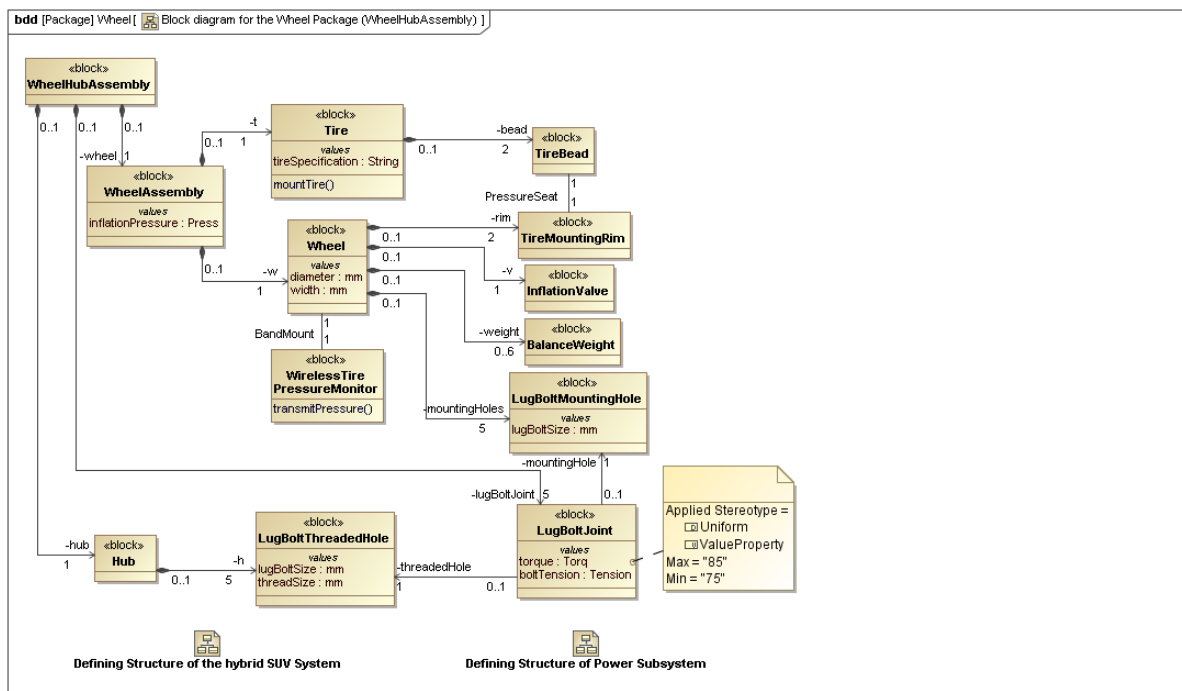


Figure 4.19. This is a figure

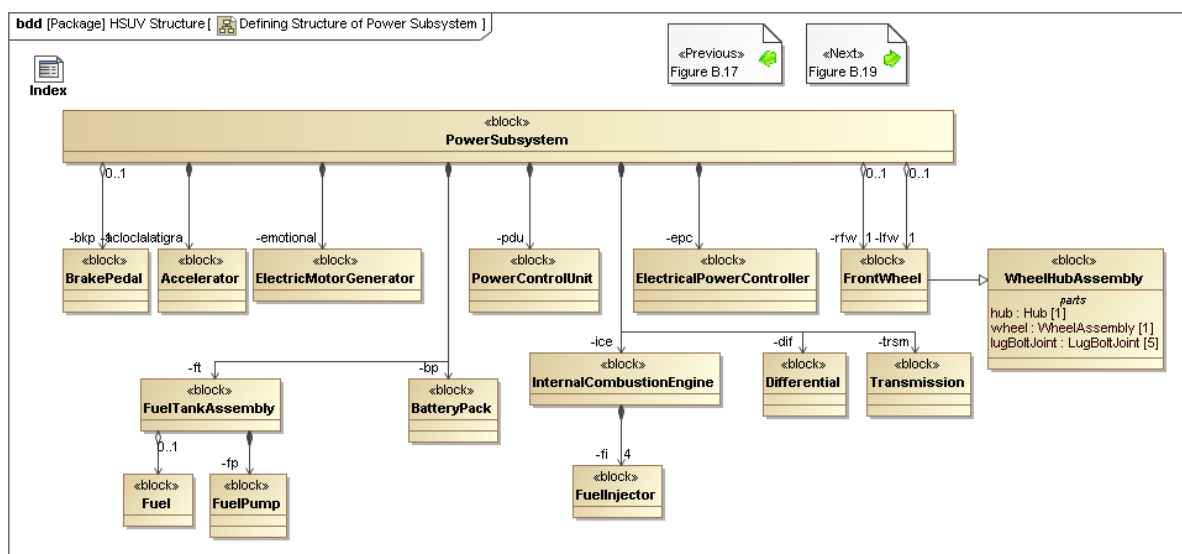
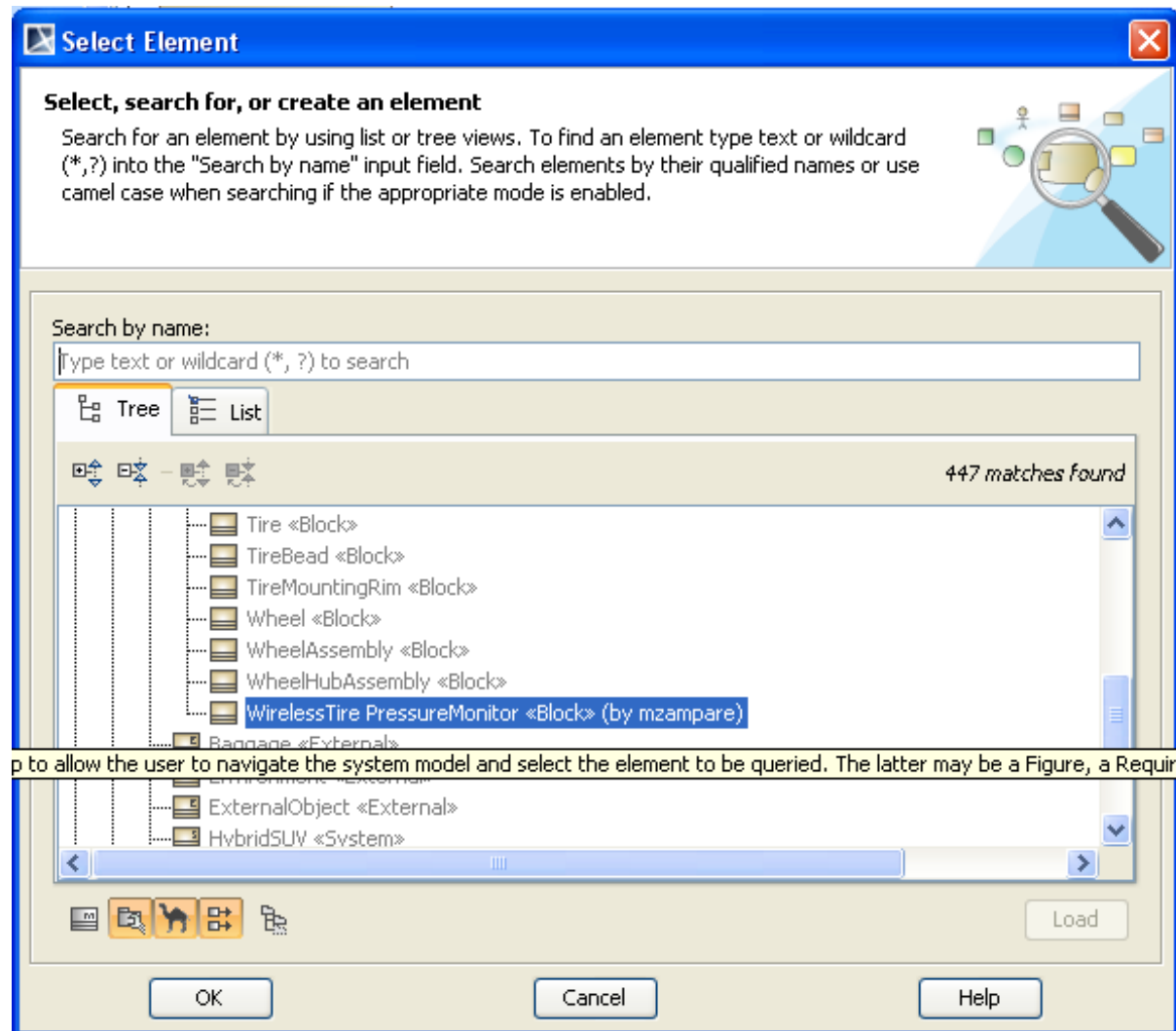


Figure 4.20. Defining structure of power subsystem

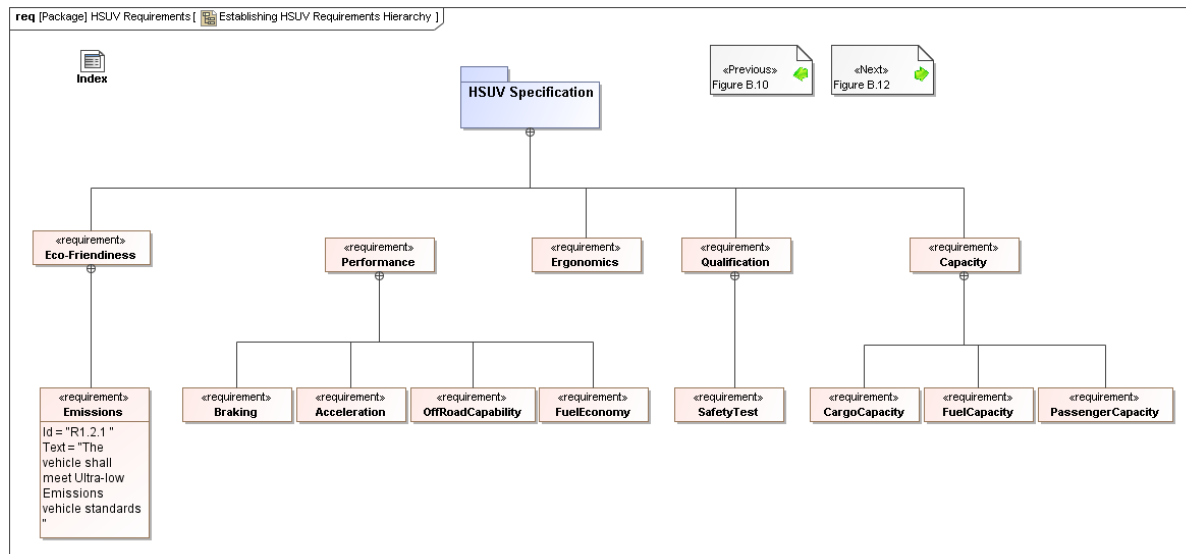
The following subsections provide examples of rendering for various settings of the queries fields, which are printed in boldface when sensible.



**Figure 4.21. mbseSelectionExample**

An example of selection can be see in Figure Figure 4.21, "mbseSelectionExample" .

### 4.11.1. Requirements



**Figure 4.22. Irrelevant since not used.**

When a query is made to an element of type Requirement or derived, query type is ignored, and the requirement ID, name and textual content are displayed, in the customary tabular or paragraph form.

ReqID	ReqName	Text
null	Emissions	null

**Table 4.4. Emissions**

This is a a query on the same **requirement** element as above, but in paragraph format. null,"null"

### 4.11.2. Operations

This is an example of a query where the **operations** of a Block from the HSUV model are displayed in paragraph form. The boolean value useQueryText is set to true. `transmitPressure`

This is just like the query above but in **tabular** representation. Notice the operation descriptions appearing in the table.

Operation	Description
<code>transmitPressure</code>	This operation transmits the pressure.

**Table 4.5. WirelessTire PressureMonitor**

### 4.11.3. Properties

This is a Query on the **properties** of the referenced elements shown in tabular representation.

<code>owner</code>	Package Wheel
<code>baseClassifier</code>	

**Table 4.6. WirelessTire PressureMonitor**

<code>owner</code>	Package HSUV Structure
--------------------	------------------------

baseClassifier	WheelHubAssembly
----------------	------------------

**Table 4.7. FrontWheel**

This query shows the **property** query type, in tabular format. Notice how since two queryTypes have been selected (property and owner), they both get displayed as rows in a single table (operations and receptions in the SUV model are not particularly well documented in this case..).

realizedInterface	
owner	Package HSUV Interfaces

**Table 4.8. ICECommandInterface\_if**

#### 4.11.4. Documentation

When a query of type **documentation** is used, the element name appears in boldface, followed by its fully qualified name in parenthesis, followed by its documentation, if available, on a new paragraph. This example has the userText field set to true. The representation field value is ignored. **WirelessTire PressureMonitor** ( *HSUVModel::Wheel::WirelessTire PressureMonitor* )

This device monitors the pressure of the Tires using a wireless connection.

Care should be taken that in general the documentation of a Model Element will include the comments attached to such element. Some of these relationships may not always be displayed in diagrams and be therefore difficult to detect.

#### 4.11.5. Ports

This is a query on the **ports** of a selected block. If the representation type is not specified it defaults to *table*. epc, ice, trsm, eepc, etrsm, eice,

Here is a variant in tabular format, other fields unchanged.

Port	Type	Port Doc.	Type Doc.
epc		This is the documentation for the epc port.	
ice	ICEDataInterface	This is the documentation for the ice port.	
trsm		This is the documentation for the trsm port.	
eepc	IFS_EPC	This is the documentatioon for the eepc port.	
etrsm	IFS_TRSM	This is the documentation for the etrsm port.	
eice	IFS_ICE	This is the documentation for the eice port.	This is the documentation for the IFS_ICE

**Table 4.11. PowerControlUnit Ports**

#### 4.11.6. Constraints

A query on the constraints like this one in tabular format show the Constraint Parameters of a given constraint Block, their type and description.

Constraint	Type	Constraint Doc.	Type Doc.
acc	AccelerationEquation	$a(n) = F/M = P \cdot t / m$ This is the documentation of the constraint property for acceleration	This describes the Acceleration Equation.
pwr	PowerEquation	$tp(hp) = \text{wheel power} - \text{drag} - \text{friction}$	This is documenting the PowerEquation ConstrainBlock
vel	VelocityEquation	$v(n+1)(\text{mph}) = v(n) + \text{delta-}v = v(n) + a \cdot \text{delta-}t$	
pos	PositionEquation	$x(n+1)(\text{ft}) = x(n) + \text{delta-}x = x(n) + v \cdot \text{delta-}t$	This is a comment anchored to the Position Equation which will appear in the documentation of the Constraint Block.

Table 4.12. StraightLineVehicle Dynamics Constraint Properties

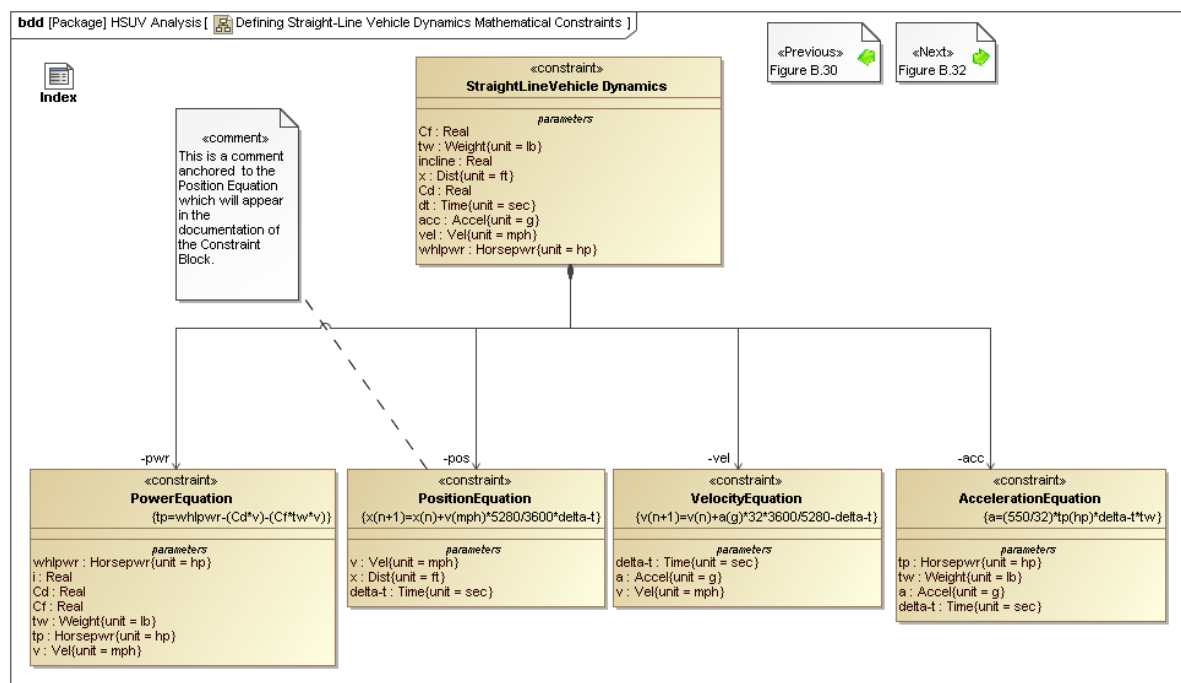


Figure 4.23. This is not used.

The queried Constraint Block appears in the Diagram in Figure Figure 4.23, "This is not used." .

A query of the same type and to the same Constraint Block, but in paragraph format looks like this:  
acc,AccelerationEquation pwr,PowerEquation vel,VelocityEquation pos,PositionEquation

This is documenting the PowerEquation ConstrainBlock

Name	Constraint
unnamed6	$tp = whlpwr - (Cd \cdot v) - (Cf \cdot tw \cdot v)$

Table 4.13. PowerEquation Constraints

This is documenting the PowerEquation ConstrainBlock

Parameter	Type	Description
whlpwr	Horsepwr	
i	Real	
Cd	Real	
Cf	Real	
tw	Weight	
tp	Horsepwr	
v	Vel	

**Table 4.14. PowerEquation Parameters**

#### 4.11.7. Flow Properties

This query differs from the previous one merely in that the types documentation has been switched off.

Property	Type	Dir	Property Doc.
engineData	ICEData	out	
mixture	Real	in	Documentation for the mixture Flow Property.
throttlePosition	Real	in	

**Table 4.15. FS\_ICE Flow Properties**

This flowProperties query has the showDocumentation for the properties themselves switched off.

Property	Type	Dir	Type Doc.
engineData	ICEData	out	
mixture	Real	in	A Real value type represents the mathematical concept of a real number. A Real value type may be used to type values that hold continuous quantities, without committing a specific representation such as a floating point data type with restrictions on precision and scale.
throttlePosition	Real	in	A Real value type represents the mathematical concept of a real number. A Real value type may be used to type values that hold continuous quantities, without committing a specific representation such as a floating point data type with restrictions on precision and scale.

**Table 4.16. FS\_ICE Flow Properties**

Flow Properties may be queried with the corresponding query type. This is one such example in tabular form, where all documentation types have been turned on.

Property	Type	Dir	Property Doc.	Type Doc.
engineData	ICEData	out		
mixture	Real	in	Documentation for the mixture Flow Property.	A Real value type represents the mathematical concept of a real number. A Real value type may be used to type

Property	Type	Dir	Property Doc.	Type Doc.
				values that hold continuous quantities, without committing a specific representation such as a floating point data type with restrictions on precision and scale.
throttlePosition	Real	in		A Real value type represents the mathematical concept of a real number. A Real value type may be used to type values that hold continuous quantities, without committing a specific representation such as a floating point data type with restrictions on precision and scale.

**Table 4.17. FS\_ICE Flow Properties**

#### 4.11.8. Part Properties

Part	Type	Part Doc.	Type Doc.
ice	InternalCombustionEngine		
emotional	ElectricMotorGenerator		
aclocalatigra	Accelerator		
bp	BatteryPack		
pdu	PowerControlUnit		This is the documentation of the unit which controls the power.
epc	ElectricalPowerController		
ft	FuelTankAssembly		
dif	Differential		
trsm	Transmission		
fuelSupply	Fuel		
i1	ElectricCurrent		
i2	ElectricCurrent		
t1	Torque		
t2	Torque		
g1	Torque		
	CAN_Bus		

**Table 4.18. PowerSubsystem Part Properties**

#### 4.11.9. Value Properties

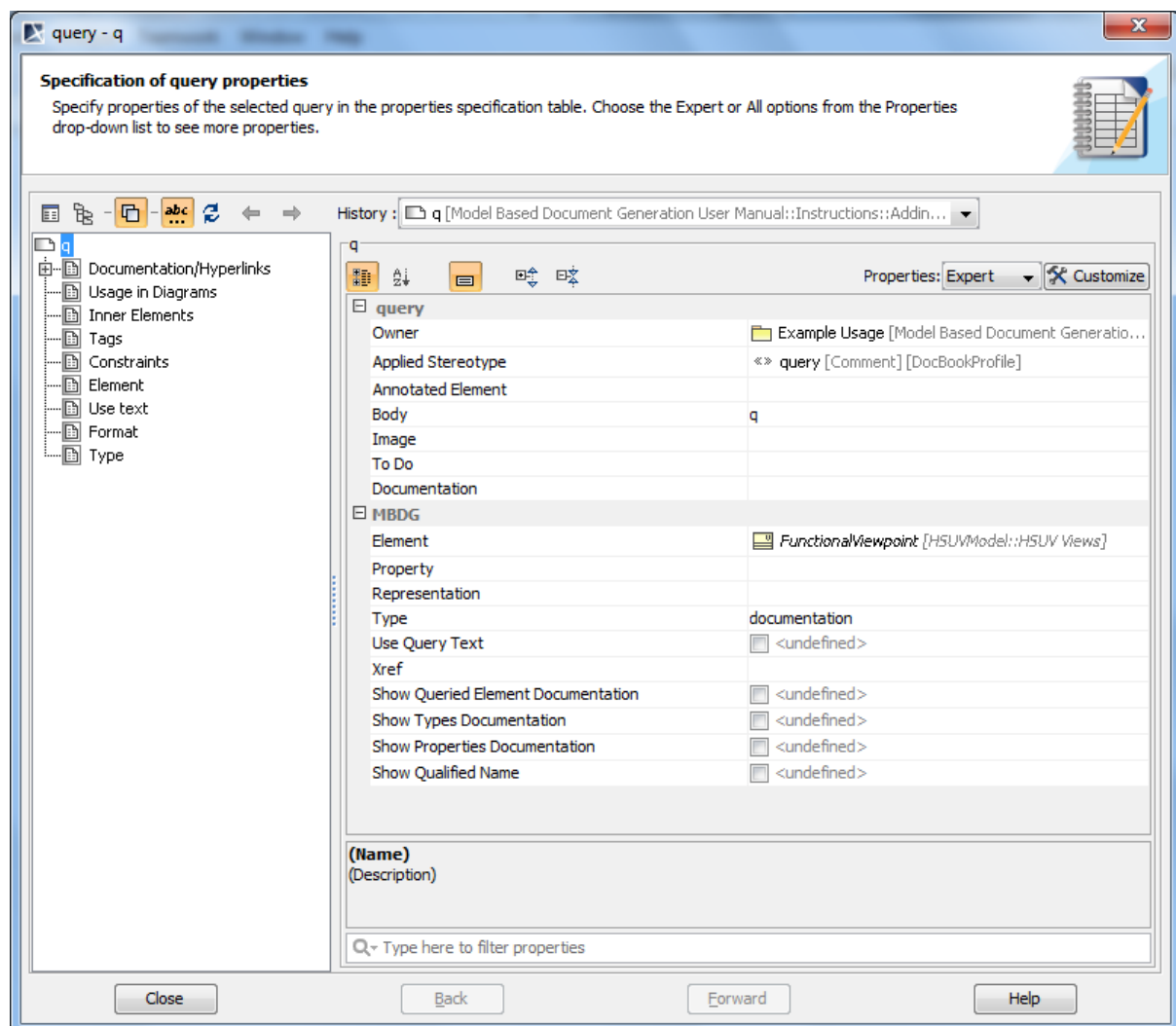
Property	Type	Property Doc.	Type Doc.
temperature	Temp		
pressure	Press		

Property	Type	Property Doc.	Type Doc.
fuelPressure	Real		A Real value type represents the mathematical concept of a real number. A Real value type may be used to type values that hold continuous quantities, without committing a specific representation such as a floating point data type with restrictions on precision and scale.

**Table 4.19. Fuel Value Properties**

#### 4.11.10. Example Usage

A query can be added by selecting "SE2: create Query" option. Query can only be added under a section and chapter. When the add query option is selected the user will be asked to select the element to be queried. Later on, changes to specification of a query can be done by changing the values of query tags in MBDG sub menu in the specification dialog.

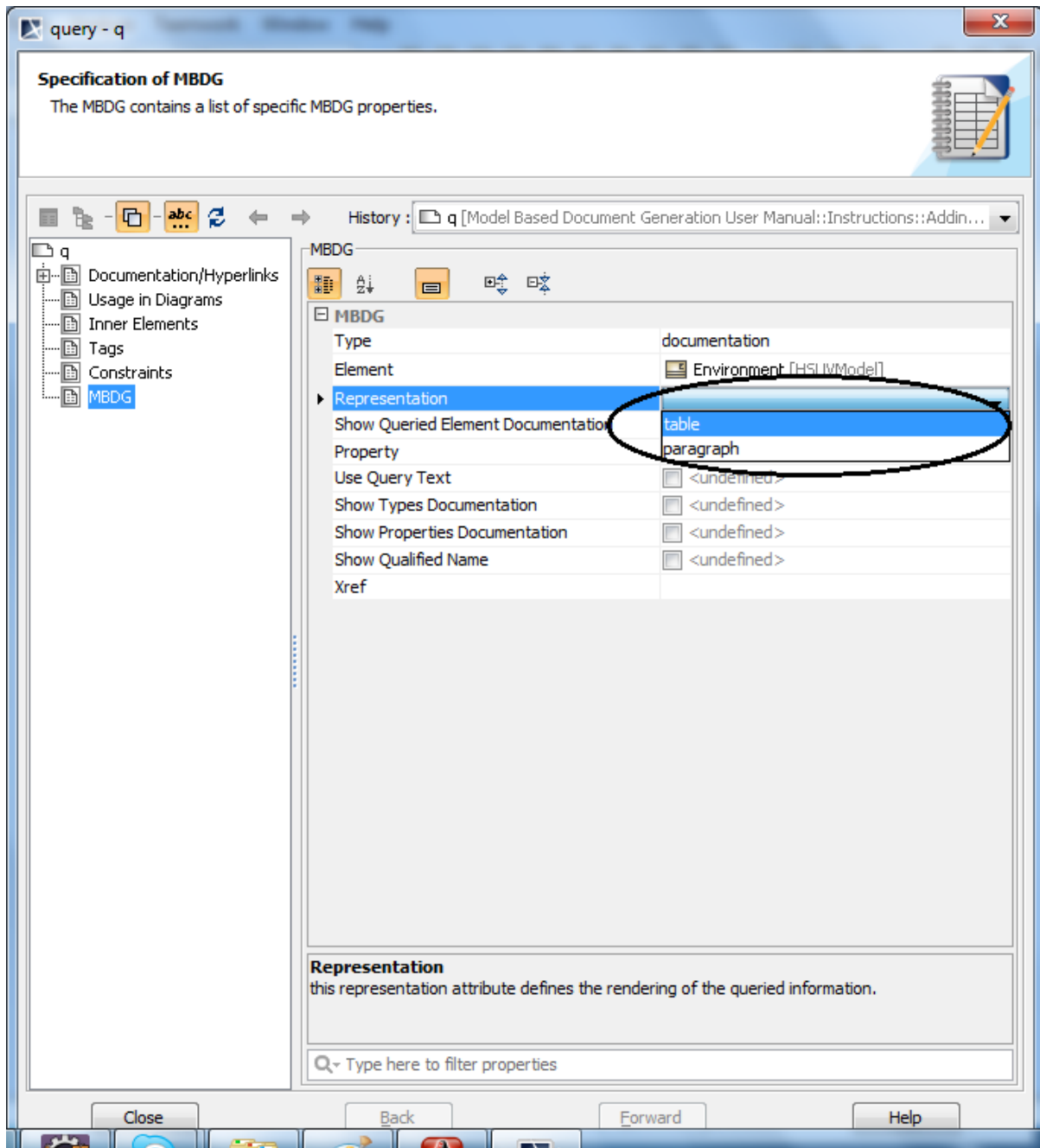


**Figure 4.24. queryDialog**



Specification of the query like specifying the output format to be a table or paragraph can be done by editing the Representation tag value like the image below.

## Environment



**Figure 4.25. queryDialog2**

Representation tag to set the output appearance of the element, in table or paragraph format.

The type of information to be displayed in the table or paragraph can be specified through editing the Type tag values. User can choose the element's documentation, properties, constraints, ports, receptions, operations etc. to be displayed in the table.

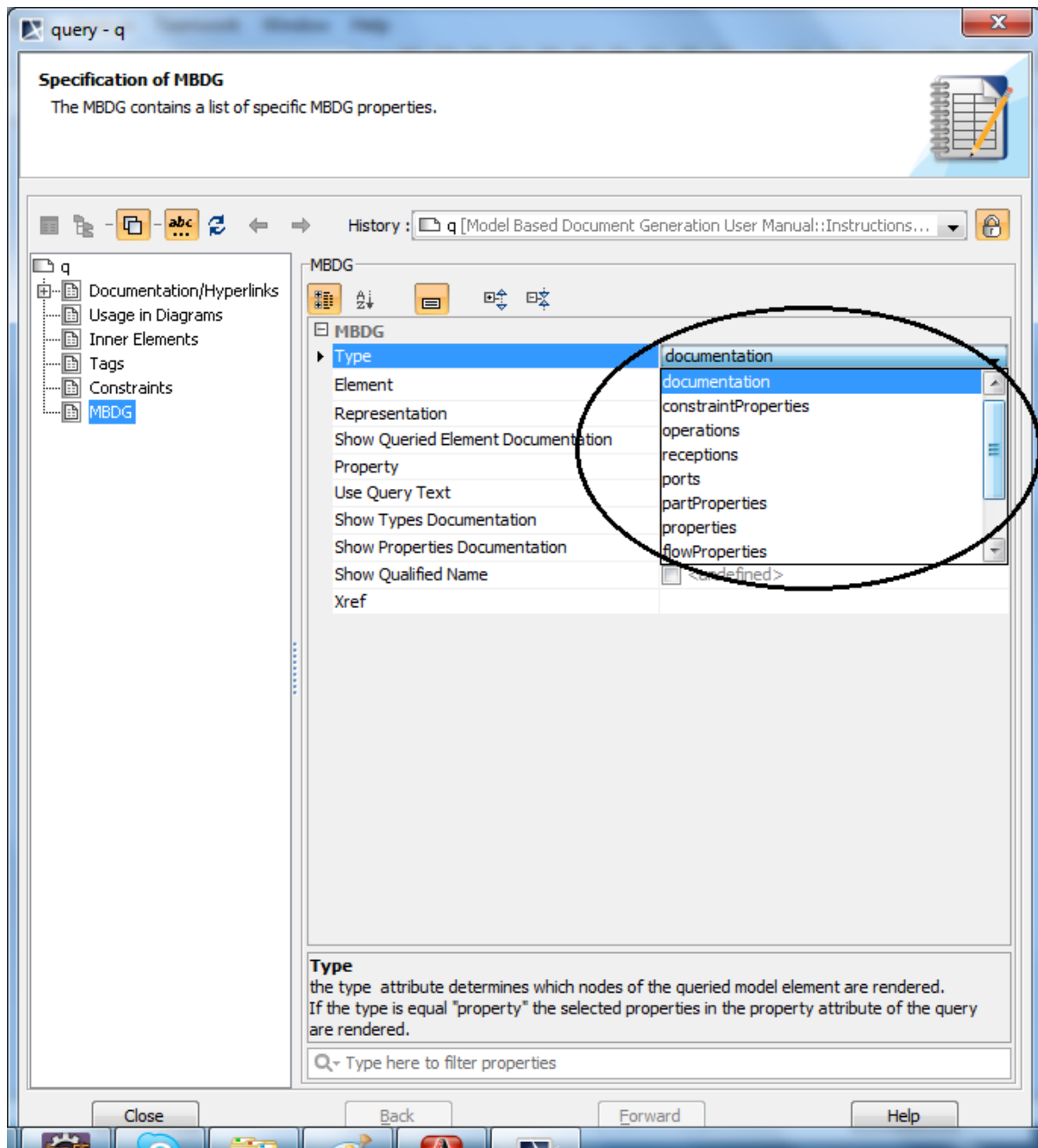


Figure 4.26. queryDialog3

This image shows the Type tag value that can be specified by the user to query different type of information.

Additional information regarding the documentation of elements, types or properties can be chosen to display as a new column in the table. This can be done by editing the tag values shown in the following image.

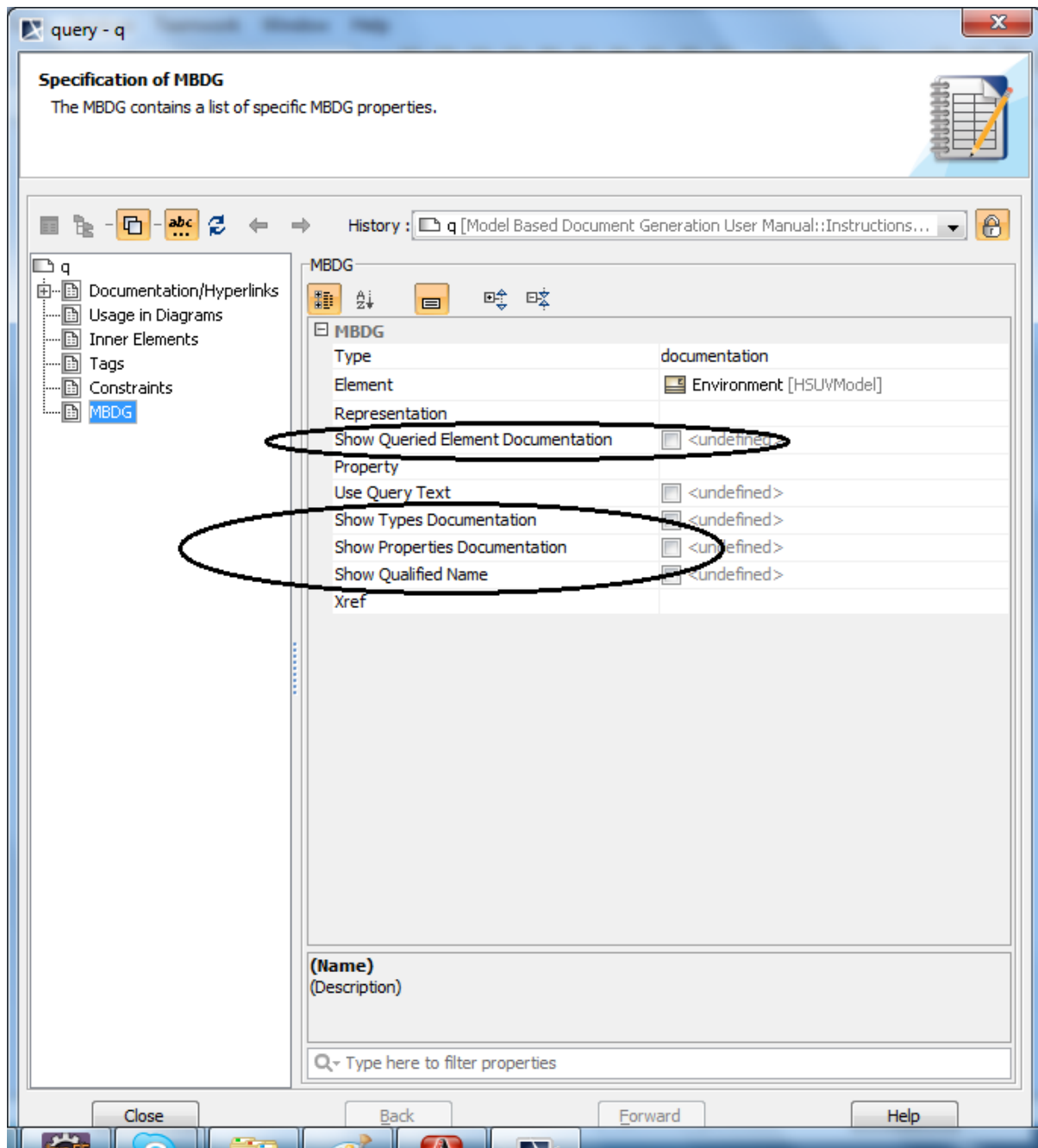


Figure 4.27. queryDialog4

## 4.12. Using Generic Tables

MagicDraw offers the possibility to create generic tables which contain any kind of information of model element, e.g. documentation, its members, its operations, etc.

The generic tables can be referenced in a document and are converted into native Docbook tables.

A tableDiagram element needs to be created in the model which references the generic table. The tableDiagram can have a caption text (by default the name of the generic table is taken).

Table 4.20, "Diagram StereotypesAndBlocks" shows such a table where the selected element types are Block and Stereotype and the columns show name, documentation, and members.

CAVEAT: if the width of the generic table columns is not adjusted manually, the size is not properly set in MagicDraw (as of 17.0 SP3) and the generation will fail. Therefore the column width has to be adjusted the first time the table is created.

null

#	Name	Documentation	Member
1	CapacityEquation	The documentation of the CapacityEquation	unnamed2=pcap=Sum(Vi)-V1 : Vol-V2 : Vol-V3 : Vol-vc : SysML::Blocks::Real
2	CapacityContext	The documentation of the CapacityContext	-ad : HSUVModel::HSUV Structure::AutomotiveDomain [1]-cap : HSUVModel::HSUV Analysis::CapacityEquation
3	figureImage	<p>To scale a graphic to fit the available width in printed output, use width="100%" and scalefit="1" attributes. For indented text as in a list, the available width is from the current indent to the right margin.</p> <p>To keep a graphic for printed output at its natural size unless it is too large to fit the available width, in which case shrink it to fit, use scalefit="1", width="100%", and contentdepth="100%" attributes.</p>	-imageContainer : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Class- useText : DocBookProfile::usageTextKind- base_Comment : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Comment- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- captionText : SysML::Blocks::String [0..1]-scalefit : Boolean = true- width : Integer = 100-contentdepth : Integer = 100-base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element
4	figureDiagram	<p>To scale a graphic to fit the available width in printed output, use width="100%" and scalefit="1" attributes. For indented text as in a list, the available width is from the current indent to the right margin.</p> <p>To keep a graphic for printed output at its natural size unless it is too large to fit the available width, in which case shrink it to fit, use scalefit="1", width="100%", and contentdepth="100%" attributes.</p>	-diagram : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Diagram- useText : DocBookProfile::usageTextKind- base_Comment : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Comment- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- captionText : SysML::Blocks::String [0..1]-scalefit : Boolean = true- width : Integer = 100-contentdepth : Integer = 100-base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element-

#	Name	Documentation	Member
			base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element
5	tableDiagram		-diagramTable : UML Standard Profile::MagicDraw Profile::DiagramTable [0..1]-base_Class : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Class- captionText : String [0..1]-base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element- base_Element : UML Standard Profile::UML2 Metamodel::Classes::Kernel::Element

**Table 4.20. Diagram StereotypesAndBlocks**

### 4.13. Adding Tables

TBD.

### 4.14. Adding a Table Paragraph

A table paragraph can be created through invocation of "SE2 : create Table Paragraph" command under Section or Chapter elements etc. The aim of creating a table paragraph is to allow user to reference the table through a query mechanism or in normal paragraphs (see Section 4.10, "Adding internal references" ) as well as adding a caption to the table, which would not be possible through normal html tables. Invoking the "SE2 : create Table Paragraph" command, user will be asked to fill the caption of the table and then the plugin will generate an empty stub html table, in which the user can edit it normally. During the conversion process (generating a pdf) this html table is converted into a Docbook conform table. An example table is created below (Table 4.21, " Table Paragraph 1 ")

Header1	Header2
c1	a1
c2	a2
c3	a3
c4	a4
c5	internal ref to Section 4.10, "Adding internal references"

**Table 4.21. Table Paragraph 1**

### 4.15. Changing the ordering of elements

The ordering of each docbook elements like chapter, section, paragraphs or diagrams can be changed by drag and drop action. The elements in the Edit Panel tree can be drag and drop to be placed at a

new location under the same parent or different parents. Note that the movement should be conformed to DocBook profile, for e.g. a paragraph cannot be moved under a bibliography element.

## 4.16. Generating the final Document

In order to generate your DocBook document you must select an element of type book in your model (there might be more than one in there) from the content tree, and use the action "MBSE->SE: Generate DocBook".

The final artifact of the generation process is a standard PDF file, produced by means of XSL transformation from a generated XML file. These two steps are available as separate actions from the user interface with the Actions "Generate PDF", "Generate XML" respectively. A third step "Generate XML to PDF" is available to carry out the XSLT step only.

The specific XSLT to be used can be selected from MagicDraw from the menu "Options", then "Environment" and then selecting the MBSE group, which looks like in Figure 4.28, "xsltSelector" . A file chooser permits the selection of the appropriate XSLT file.

The currently shipped XSL transformations are briefly outlined in Section 5.2, "Transformation XSL" .

If the model which you're using is coming from the Teamwork Server, a file chooser dialog will appear. Once the file has been chosen the generation will start (no progress bar is available) and the plug-in will notify you of the successful completion with an acknowledgment pop-up.

The file chooser remembers the selected directory across multiple generations.

Notice that the chosen directory will be populated with all the images which are required to produce the XML and PDF files. The presence of many diagrams and external images may contribute significantly to the generation time.

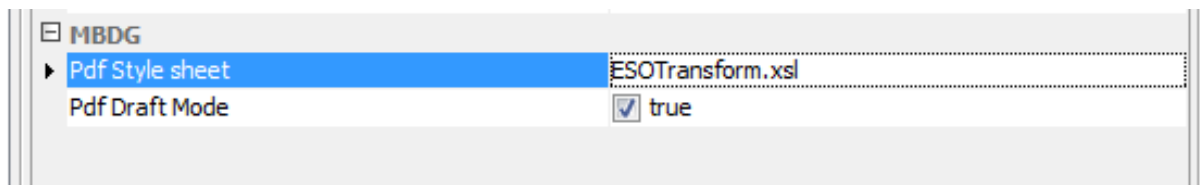


Figure 4.28. xsltSelector

## Chapter 5. Installation

### 5.1. MagicDraw Plugin

The plugin can be downloaded from SourceForge at: <http://sourceforge.net/projects/mbse4md/>.

Please consult the README file for complete installation instructions.

### 5.2. Transformation XSL

In MagicDraw's Options->Environment->MBSE you select the appropriate transformation file:

- ESOTransform.xsl
- SE2Transform.xsl: refines ESOTransform by adding different logo and different rendering of the book information
- ELTTransform.xsl: refines ESOTransform by adding different logo and different rendering of the book information, specific to ESO EELT project.

Furthermore, you select if a "Draft" Watermark is added.

---

## **Chapter 6. Maintenance**

### **6.1. Extension Procedure**

After initial release, changes to this plug-in will have to be coordinated by the SE2 Challenge team members.

### **6.2. Fault identification procedure**

Not applicable.

### **6.3. Diagnostic tools and procedures**



## Chapter 7. Known Problems

- DO NOT copy/paste from word into HTML (hidden formatting instructions are copied as well). cut & paste into ASCII FIRST and then turn on HTML.
- Very rarely the edit panel displays the document twice.

## **Chapter 8. Future Enhancements**

- Add a shortcut key for the generation
  - Direct editing of paragraphs in preview panel
  - Drag'n'drop of diagrams directly into the preview panel
  - More sophisticated queries
  - The file chooser for book generation should only display XML files.
-