DATEX II v2.0 RC2

SCHEMA GENERATION TOOL GUIDE

Document version: 1.1

15 January 2010

European Commission

Directorate General for Transport and Energy

Prepared by :			
	Date	**Comment**	**Version**
DATEX Technical Group	01/07 2009		1.0
DATEX Technical Group	15/01 2010		1.1

Reviewed by :			
	Date	**Comment**	**Version**
DATEX Technical Group	15/01 2010		1.1

Approved by :			
	Date	**Comment**	**Version**
DATEX Technical Group	15/01 2010		1.1

TABLE OF CONTENTS

INTRODUCTION



1. Introduction

1.1. Objective

This deliverable documents the work on converting the DATEX II UML PIM into an XML Schema. The first chapter “UML To XSD Conversion Process” describes the used tools and the entire conversion process. Necessary mapping rules for such a conversion are written in the second chapter. The last chapter describes in detail the derived XML Schema.

1.2. Document structure

This document is structured as follows:

- Section **Fel! Hittar inte referenskälla.** “**Fel! Hittar inte referenskälla.**” gives an overview on the objectives of this document, its structure and how it fits into the whole set of DATEX II reference documents.
- Section 2 describes the UML to XSD conversion process

1.3. DATEX II reference documents

Reference in this document	DATEX II document	Document version	Date
DATEX II Methodology	DATEX II Methodology	1.2	15/01/2010

UML TO XSD CONVERSION PROCESS



2. UML To XSD Conversion Process

To derive an XML Schema from an UML model a conversion process is needed. Some tools must be used to facilitate a more or less automated way of converting UML into XML Schema. The first subchapter lists the needed tools and explains which software program is suitable for which part of the work.

A tailor-made transformation has to be used to create an XML Schema which is easy to generate and easy to use. The second subchapter explains the work flow of that automated process in detail.

Despite the automated process there may still be a couple of issues left which need to be resolved manually. These issues and the work required are described in the next part of this chapter.

This transformation runs in the context of a windows based application. The usage of the programme is also described in a separate subchapter.

2.1. Used Tools

2.1.1. Enterprise Architect

Enterprise Architect of Sparx Systems has been used to create the platform independent DATEX II UML model. EA has a typical Windows look and feel and is easy to use. A free trial version and a full version for purchase are downloadable at http://www.sparxsystems.com.au. EA provides the possibility to use UML version 2.0 to create models. Its integrated XSD export capabilities are very useful for some quick results. Particular attention should be drawn to the export of UML models in XMI 1.1 which is the basis for DATEX II conversion.

2.1.2. XMLSpy

Altova's XMLSpy is a convenient software tool to work with the derived XML Schema. A downloadable version is available at http://www.altova.com. It supports both XML Schema and XML. Its integrated XSL processor enables easy transformation. Also the validation feature has been used to check the correctness of the XML Schema and the derived XML files.
The above tool is only an example other tools free or commercial exists the can validate, view and process XML files and XML Schemas.

2.1.3. Tailor-made transformation

To convert the XMI file derived from the UML model into the XSD it is necessary to use a tailor made conversion tool. The tool is built on Microsoft's ".NET 2.0" framework.
The rules for the transformation are described in the DATEX II Methodology document.

2.2. Automated Conversion Process

The following figure shows the work flow for an automated conversion process with the help of tools described in the previous section.

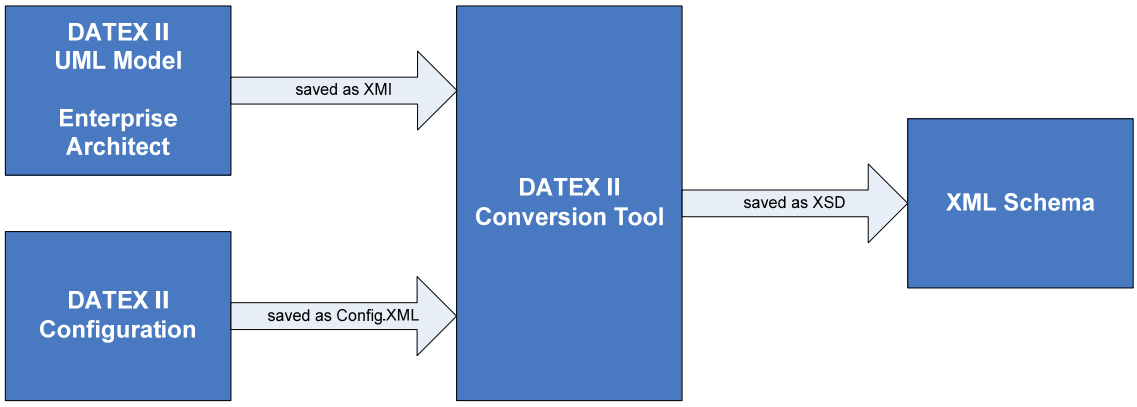


Figure 1 - conversion work flow

Having produced the XMI file from the UML model, a configuration file is required to control the conversion process. Upon selecting the XMI file and choosing the destination folder with the DATEX II conversion tool, the transformation process from XMI file to XSD schema file takes place.

These generated XSD files can be validated using a variety of XML tools including a web form offered by the World Wide Web Consortium W3C (http://www.w3.org/2001/03/webdata/xsv) or Altova's XML Spy.

2.3. Manual Work

This chapter describes the creation of the XMI file within Enterprise Architect and the way to configure the tailor-made conversion tool.

2.3.1. Export an XMI file

At first you have to select the root package “D2LogicalModel” within the project view.

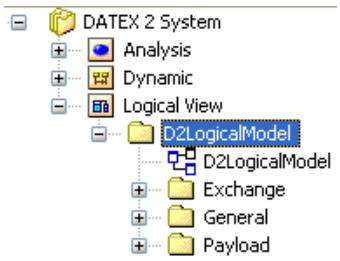


Figure 2 - select the package "D2LogicalModel"

Then you have to click the right mouse button and select the menu item “Import/Export” → “Export package to XMI file…”.

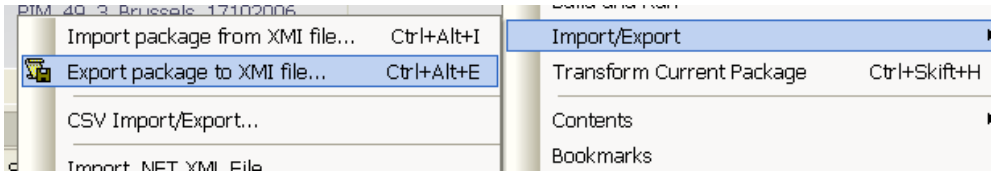


Figure 3 - the menu item "Export package to XMI file…"

In the following dialog please select the path and file name for the resulting XMI file and make sure that only the option “Export Diagrams” is selected, XMI version is 1.1 and start the export by pressing the “Export” button.

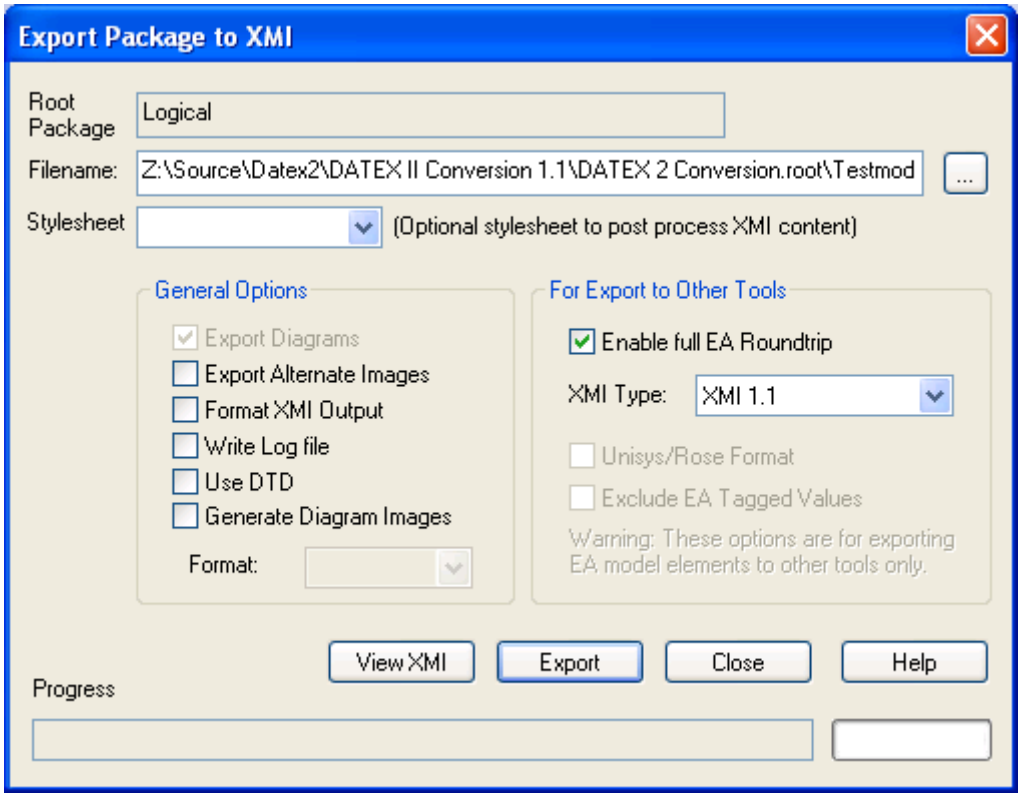


Figure 4 - Enterprise Architect XMI export dialog

The diagrams are needed to determine if dead links are contained in the model. The check itself and the exigency are described further in the “The model contains unused links or inheritances” chapter.

Now the XMI file should be created at the specified location and can be used by the tailor-made conversion tool.

2.3.2. Conversion Tool configuration

The conversion tool can be configured using an XML file found in the same directory as the tool. This configuration file lists the names of the packages which are used to generate namespaces.

2.3.2.1 Structure of the Configuration File

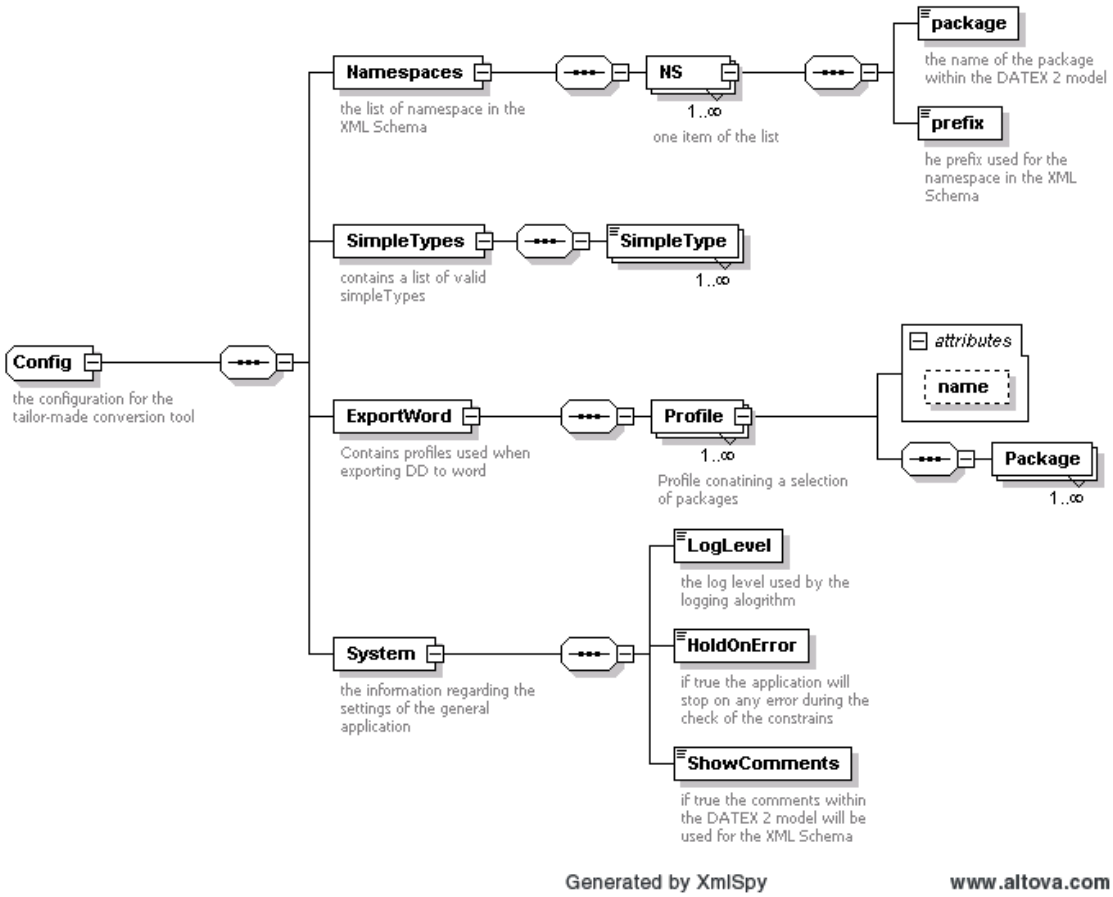


Figure 5 - XML Schema of the configuration file

Figure 5 shows the structure of the set-up. Any changes made need to follow this approach. Currently no more than one namespace can be used.

2.3.2.2 Configuration of the logging algorithm

The logging algorithm provided by this conversion tool has four logging levels which can be used in the configuration file.

Level	Name	Description
0	System	the logging algorithm is switched off and no log file will be produced
1	Error	only error messages will be shown in the log file
2	Warning	warnings and error messages will be shown in the log file
3	Debug	all information will be shown in the log file

2.3.2.3 System Flags

There are two system flags embedded in the configuration file to steer the general behaviour of the conversion tool.

The first flag is “HoldOnError”. If this flag is set to “true” the check of constrains at the beginning of the conversion process will stop on any error or violation found, otherwise only a warning will be shown and the conversion process will continue.

The second flag is “ShowComments”. If this flag is set to “true” the definitions (tagged value) of the classes and elements will be converted into the XML Schema, otherwise the definitions will be left out.

2.4. Conversion Tool

As described earlier, a tailor-made windows based program is used to carry out the tailor-made conversion between a DATEX II model and XML Schema.

2.4.1. Title bar

Title bar System requirements

This conversion tool requires the Microsoft .Net-Framework 2.0 as a system requirement. The .Net-Framework can be downloaded without charge from the Microsoft Download Centre - .Net-Framework 2.0

The conversion tool consists of the following files which have to be within the application directory.

Filename	Description
Config.xml	the configuration file
Config.xsd	the XML Schema of the configuration file
D2Conversion.chm	the online help file
D2Conversion.exe	the conversion tool itself
Logging.dll	the library with logging algorithm
RuleSet.dll	the library containing the conversion rules
MultiLingualString.xsd	definition of the MultiLingualString type
DATEXIIDD_template.dotx	a word template used when generating data dictionary
Reference.xsd	definition of Reference data type
VersionedReference.xsd	definition of VersionedReference data type

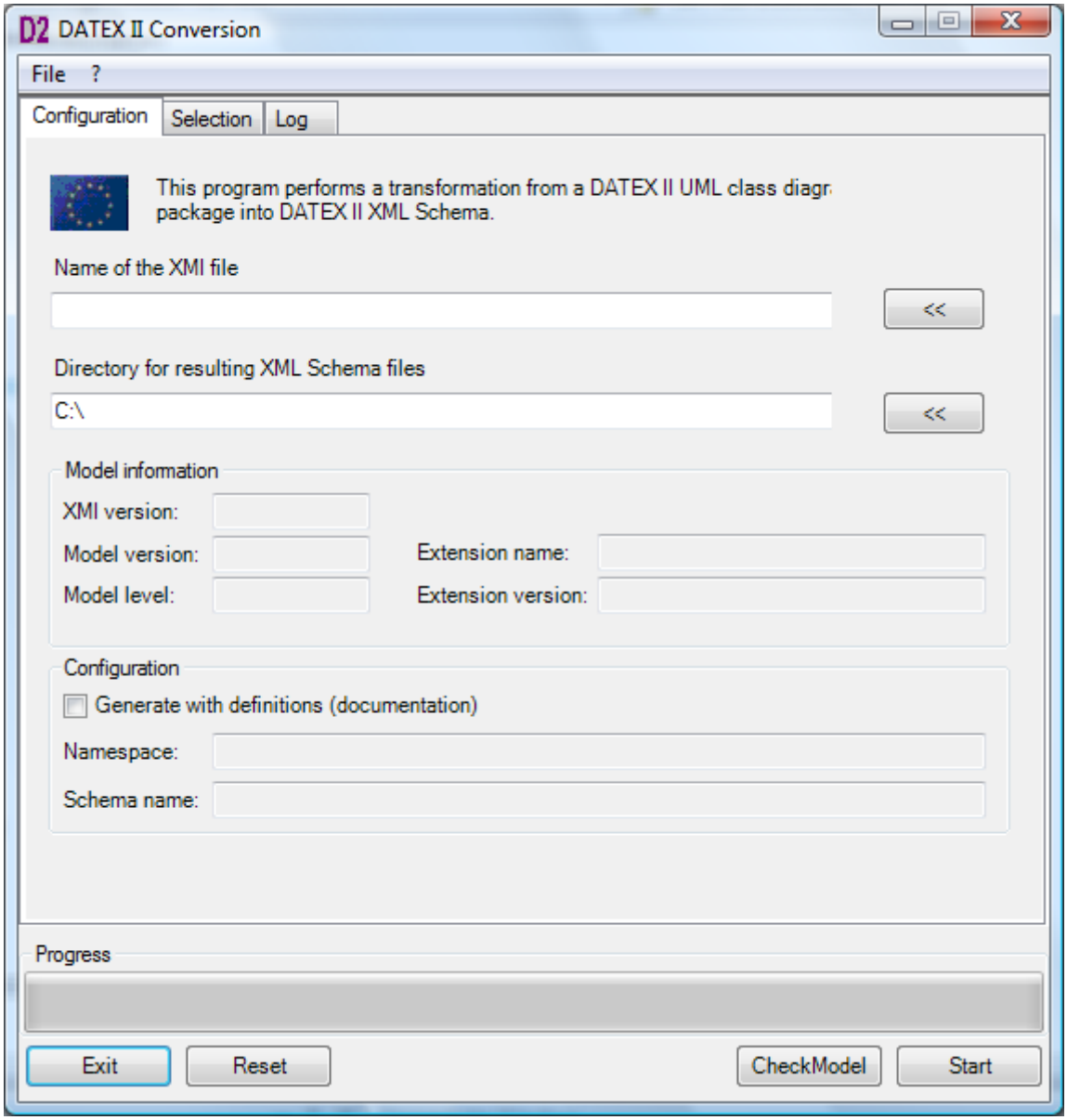


Figure 6 - graphical user Interface

2.4.2. Title bar

The title bar is the horizontal bar at the top of a window indicating the name of the window. It also contains the program symbol, the buttons **Minimize** and **Close**.

Figure 7 - menu bar

2.4.3. Menu bar

The functions offered by the button bar and the button of the entry field can be accessed via the menu bar. Online help and version display are also possible from here.

Figure 8 - menu bar

2.4.3.1 Menu "File"

The menu "File" offers the possibility to select the source file and the target directory, to start the conversion and to exit the application.

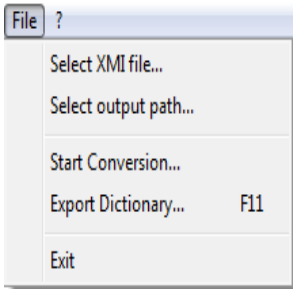


Figure 9 - menu "File"

2.4.3.2 Menu "?"

The menu "?" offers the possibility to get the about dialog and the online help.

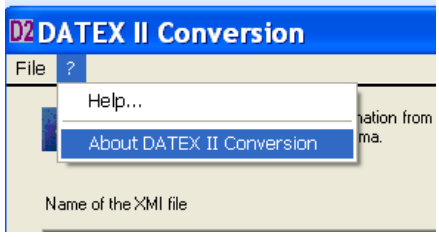


Figure 10 - menu "?"

2.4.4. Entry field

You can either enter the XMI source file and output directory path in the entry fields or use the buttons on the right to navigate to the file and directory.

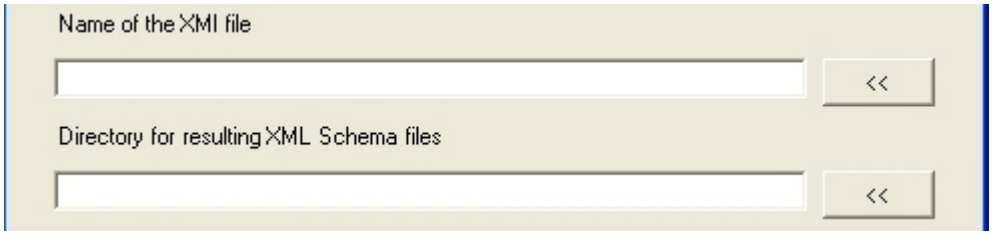


Figure 11 - entry fields

By using the buttons a number of checks are performed after the selection.

2.4.5. Model information

These fields will be set when the XMI files is opened. Model version, Extension name and Extension version are read from tagged values. Extension Level is set according to what extensions are found in the model.

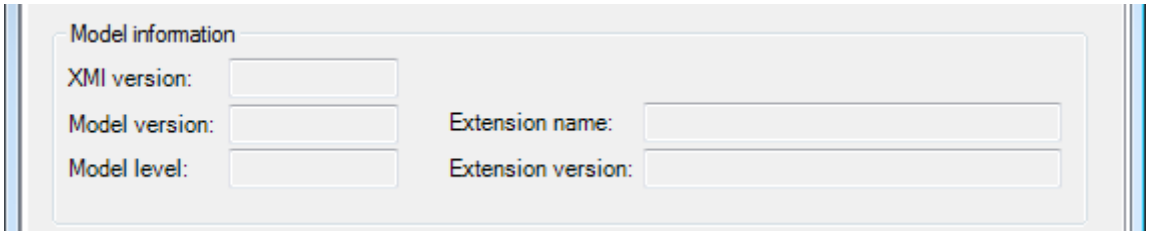


Figure 12 – model information

2.4.6. Configuration

In this section you can select whether you would like to generate a schema with documentation. If a Level A schema is generated then Namespace and Schema name are set automatically by the tool. If it's a Level C schema then these two fields have to be set manually.

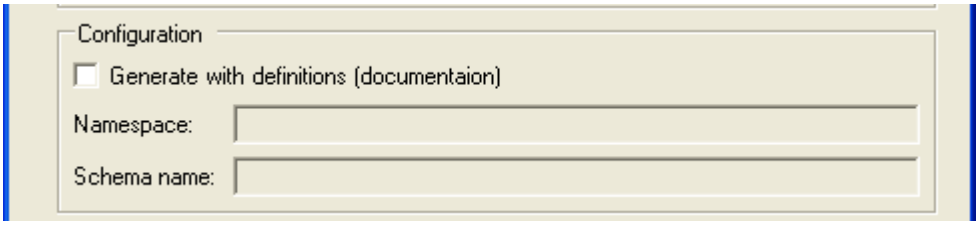


Figure 13 - configuration

2.4.7. Button bar

The button bar provides access to the main function of this conversion tool.

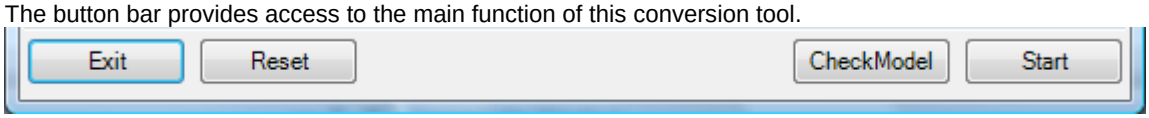


Figure 14 - button bar

The button "Exit" closes the dialog and finishes the program.
The button "Reset" resets the dialog and clears the entry fields for a new conversion.
The button "Start" launches the conversion of the given DATEX II model.

2.4.8. Progress bar

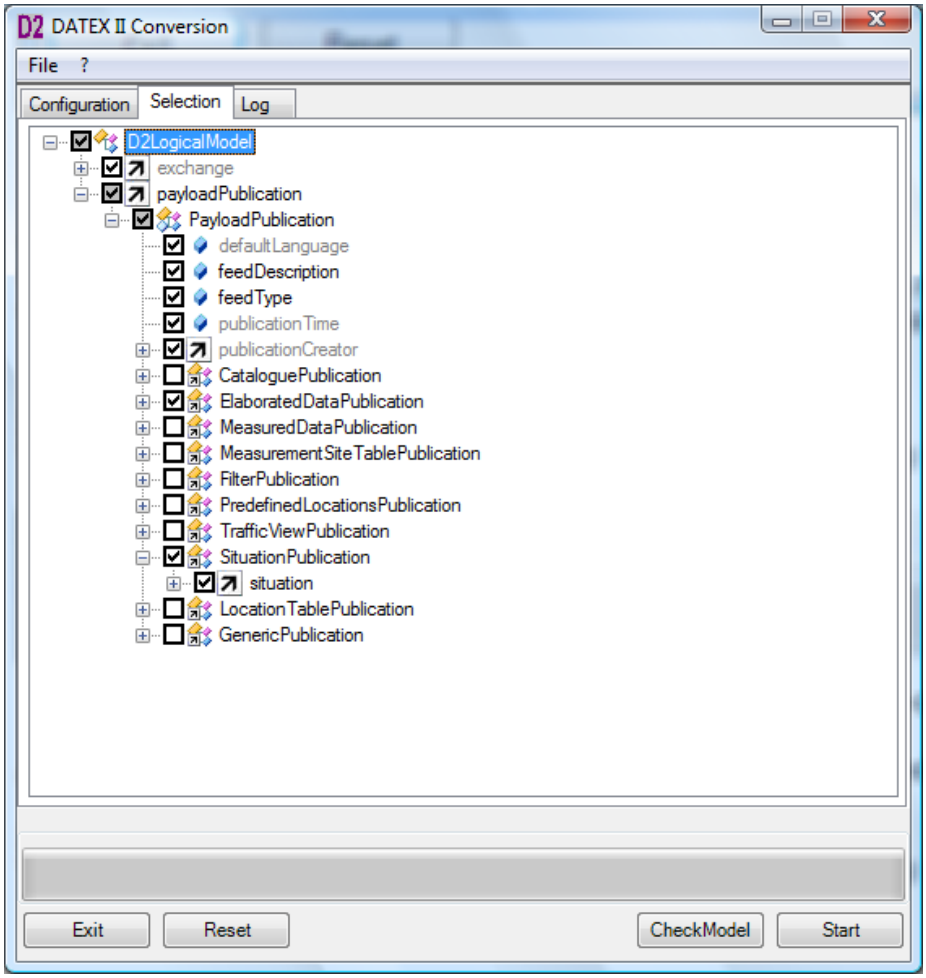
The progress bar shows the progress of the constraint checking and the conversion process.

Figure 15 - progress bar showing the constrain-checking progress

After pressing the "Start" button a constraints check will be performed before the real conversion starts.

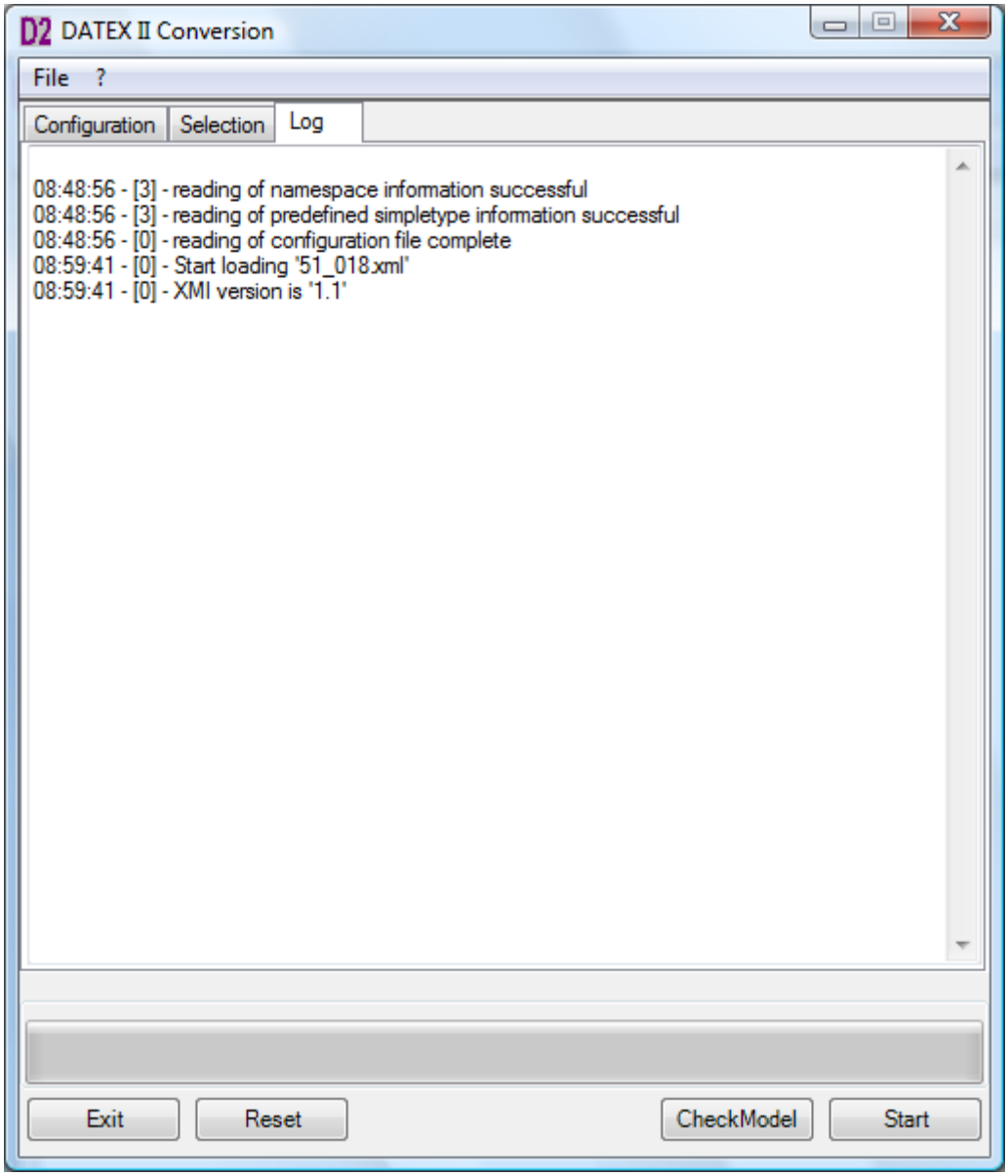
2.4.9. Selection tab

On the selection tab it's possible to select/deselect parts of the UML model. This will create a Sub-Schema.
By right clicking on the tree it's possible to save or load a selection.



2.4.10. Log tab

On the Log tab the same information that is written to the log is shown.



2.4.11. Application configuration

The configuration of this application is located in the XML file "Config.xml" in the program directory.

If the configuration file can not be read at start up, the application will not be able to make the conversion.

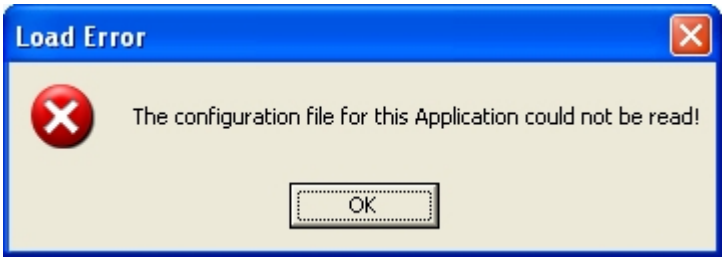


Figure 16 - error message "Load Error"

If this error occurs, please ensure that there is a configuration file located in the application directory and that it is called “Config.xml”.

2.4.12. Conversion process

The following paragraphs describe the steps of the conversion of a DATEX II UML model into DATEX II XML Schema.

2.4.12.1 select source file

To select the source file, either enter the name with its full path for the extracted XMI file in the entry field for "Name of the XMI file", or use the button on the right to navigate to the XMI file.

Using the latter, the following dialogue box will appear:

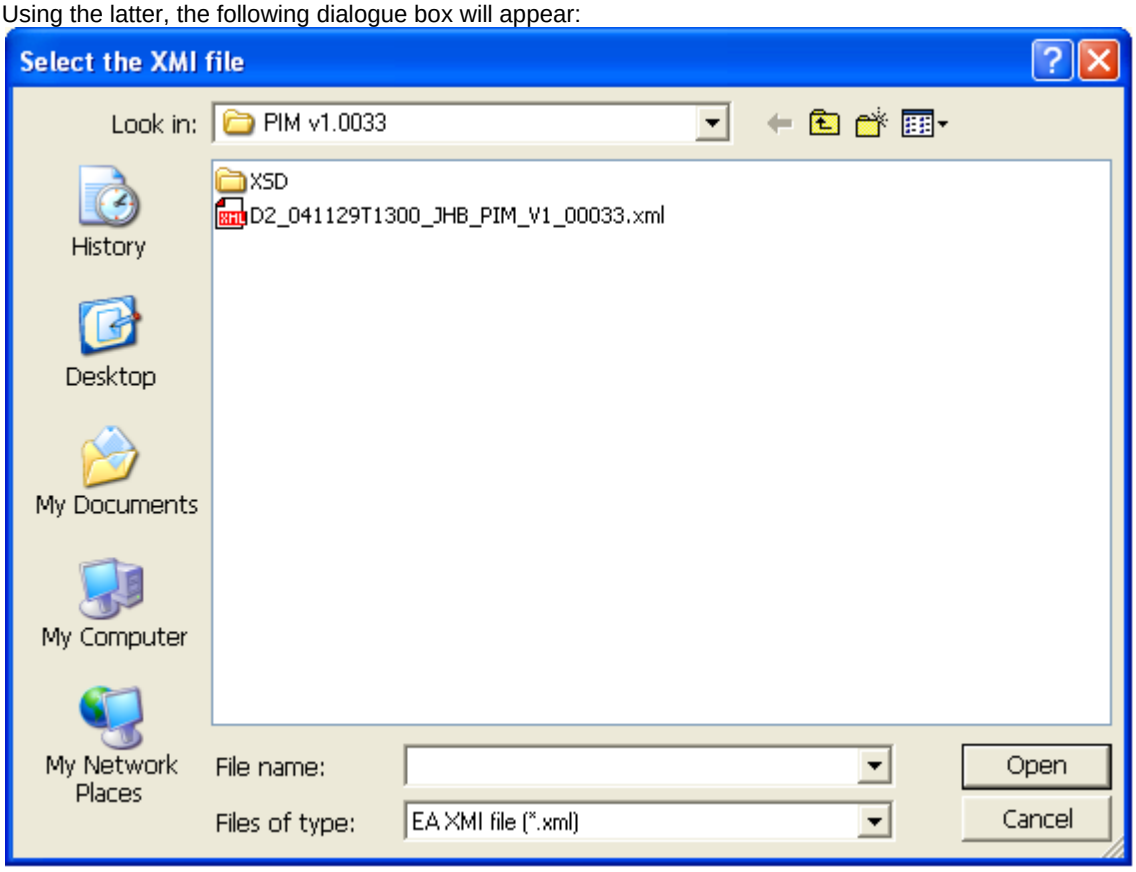


Figure 17 - select the XMI file

After confirming the entry a check of the source XMI file will be performed. If the test is successful the path will be shown in the entry field, otherwise the following message box will appear.

If an invalid source file was selected the following dialog box appears and the start button will be disabled.

Figure 18 - no namespace found in source file

2.4.12.2 Select target directory

To select a target directory you can either enter the path in the entry field by clicking on the right button or you can select the menu item "select output path".

Using the second method the following dialogue box will appear.

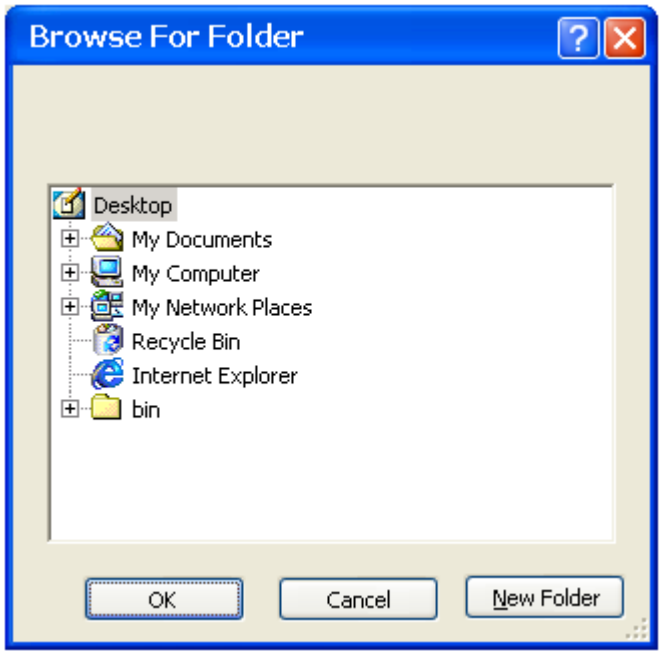


Figure 19 - select target folder

After confirming the selection, the program checks whether the pre-defined namespace file exists in the selected folder. If it finds relevant files, the following message will pop up.

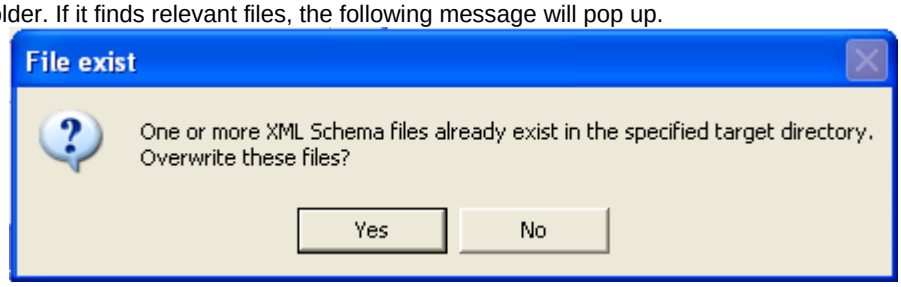


Figure 20 - files in target directory already exists

Clicking on "Yes" the existing files will be overwritten. Clicking on "No" the path will not be selected.

2.4.12.3 Starting the conversion

By pressing the Start button the conversion will be started. While the conversion is in progress only the Exit button and the help is available. The conversion process first consists of checking the constrains in the model and afterwards the conversion itself.

2.4.12.4 Failures during constrains checking and conversion

A failure of any kind during the checking and the conversion process will stop the conversion program, no XML Schema files will be created and the following dialogue box appears. In addition a log file entry with a further description of the failure will be generated.

Figure 21 - dialog "Failure Conversion"

The following errors can occur during the constraints checking and conversion process.

2.4.12.5 No diagram information within the XMI file

If the XMI does not contain diagram information the following dialog appears.

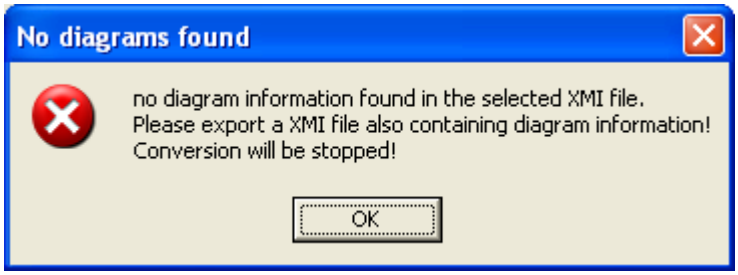


Figure 22 - dialog "No diagrams found"

Without the diagram information the conversion can not be performed.

2.4.12.6 Violation of an constrains found

If the model contains an aggregation or composition which is of an invalid direction the following dialog appears.

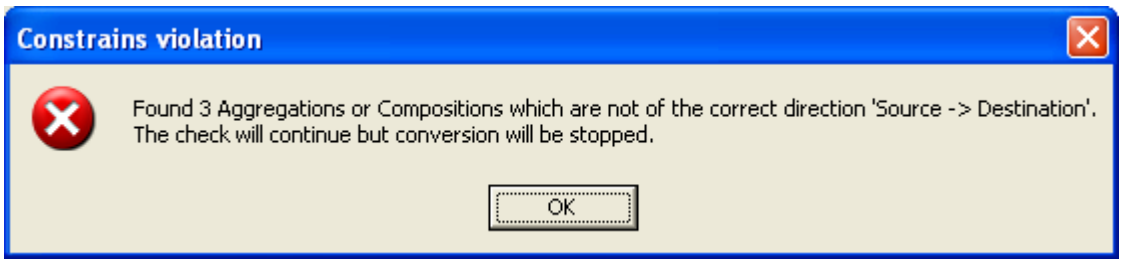


Figure 23 - dialog "Constrains violation" with holdOnError true

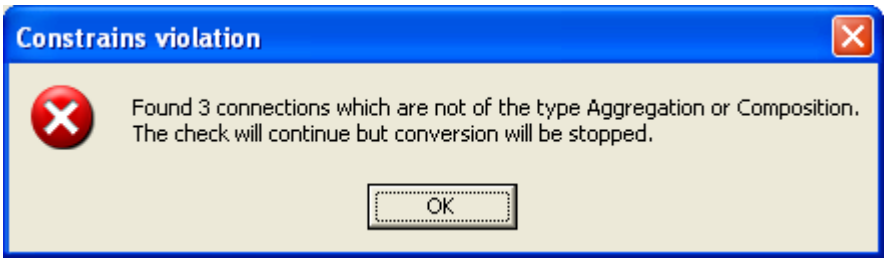


Figure 24 - dialog "Constrains violation" with holdOnError true

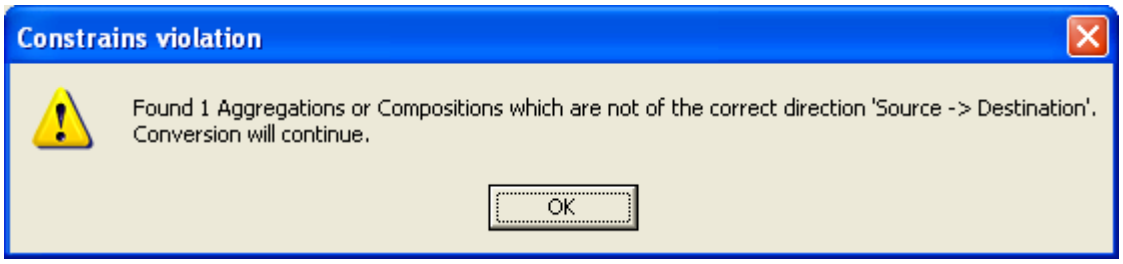


Figure 25 - dialog "Constrains violation" with holdOnError false

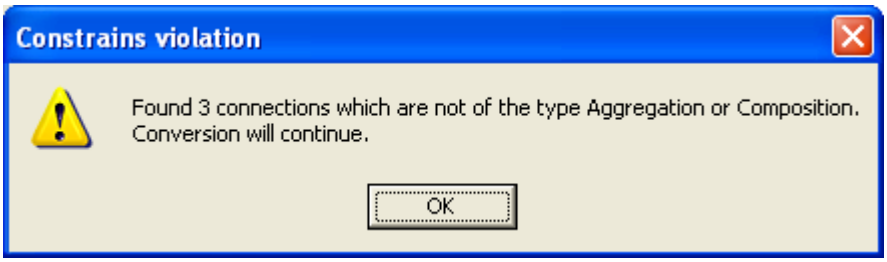


Figure 26 - dialog "Constrains violation" with holdOnError false

For every violation one log file entry is generated.

```
15:18:30 - [1] - found a connection of type Aggregation which has a wrong direction relating class 'Vehicle'
15:18:30 - [1] - found a connection of type Aggregation which has a wrong direction relating class 'Vehicle'
```

Figure 27 - log file entry for each violation

2.4.12.7 Cyclic loops found

If the model contains forbidden cyclic loops with an association with a class as start and end of the link the following dialog appears.

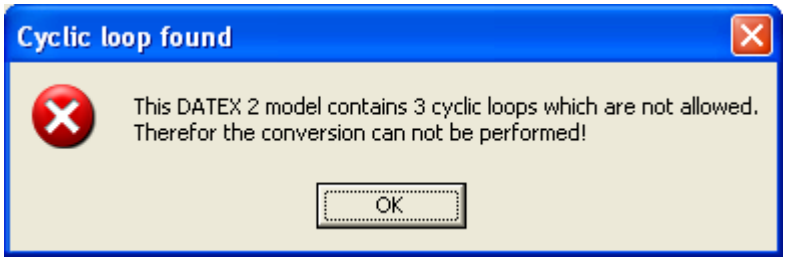


Figure 28 - dialog "cyclic loop failure" with holdOnError true

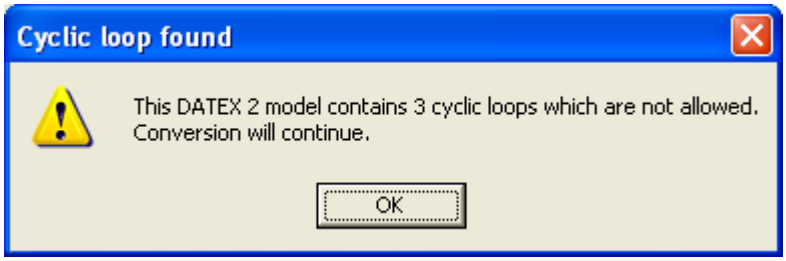


Figure 29 - dialog "cyclic loop failure" with holdOnError false

Also a log file entry for each found cyclic loop is generated like the following.

```
09:00:54 - [1] - An cyclic loop found relating class Class1
```

Figure 30 - log file entry for a cyclic loop

2.4.12.8 Multiple inheritance found

If a multiple inheritance is used in the DATEX II model the following dialog appears and the conversion will be stopped.

Figure 31 - multiple inheritances found with holdOnError true

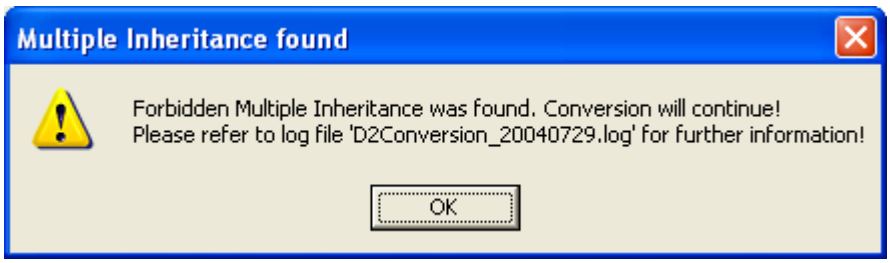


Figure 32 - multiple inheritances found with holdOnError false

The log file will have an entry for every multiple inheritance found in the model.

```
10:52:38 - [1] - A forbidden multiple inheritance was found relating class TrafficElement.
10:52:38 - [1] - A forbidden multiple inheritance was found relating class ExhaustPollution.
10:52:38 - [1] - A forbidden multiple inheritance was found relating class OperatorAction.
```

Figure 33 - log file entry for three multiple inheritance

2.4.12.9 The model contains unused links or inheritances

During model development it may happen that an aggregation, composition or inheritance is not deleted correctly and therefore this connection is still part of the model.

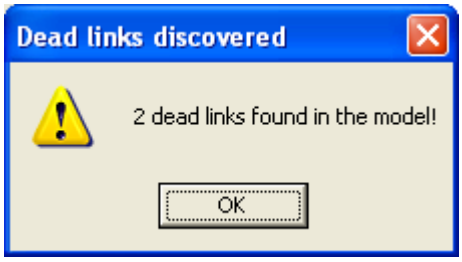


Figure 34 - dialog "Dead link discovered" for both values of holdOnError

The log file will have an entry for every dead link found in the model.

```
08:41:21 - [2] - a dead link between 'Record' and 'Situation' found!
08:41:22 - [2] - a dead link between 'Record' and 'TrafficView' found!
```

Figure 35 - log file entry for two dead links

2.4.12.10 Violation of the naming convention
If the model contains a violation of the naming convention the following dialog appears.

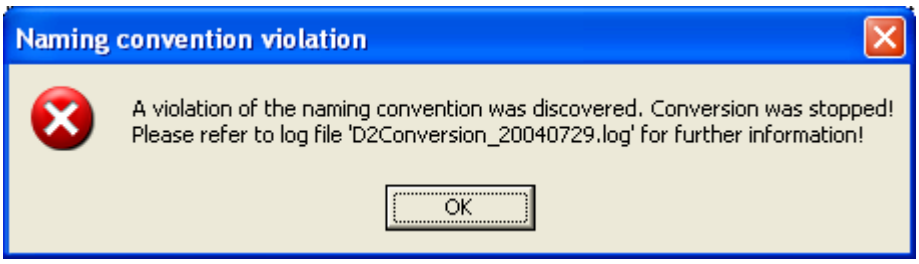


Figure 36 - dialog "Naming convention violation" with holdOnError true

Figure 37 - dialog "Naming convention violation" with holdOnError false

Also a log file entry for each violation is generated.

```
09:19:20 - [1] - The name of the class 'class1' is a violation of the naming convention!
```

Figure 38 - log file entry for a violation

2.4.12.11 Not every package contains an diagram
If the model contains a package which has no related class diagram then the following dialog appears.

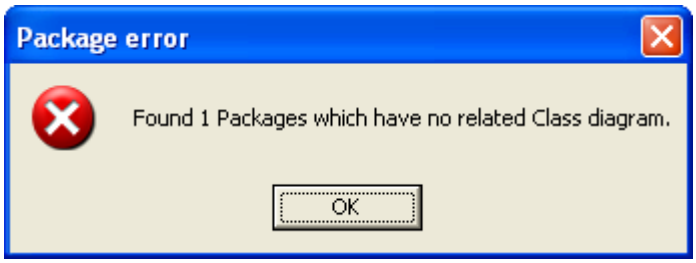


Figure 39 - dialog "Package error" for both values of holdOnError

The log file has an entry for every package without a related class diagram.

```
15:11:27 - [1] - no diagram for the package 'Exchange' found!
```

Figure 40 - log file entry for a violation

2.4.12.12 Error while converting the packages
If an error occurs during the conversion of the packages the following dialog appears.

Figure 41 - error while converting packages

The error may be due to a missing element in the XMI file for example.

2.4.12.13 Error while converting the classes
If an error occurs during the conversion of the classes the following dialog appears.

Figure 42 - error while converting classes

The error may be due to a missing element in the XMI file for example.

2.4.12.14 Missing data type of an attribute
If an attribute has no data type the following dialog appears. The data type of this attribute will be set as "xs:string" and the conversion will **not** be stopped.

Figure 43 - no data type for an attribute found

2.4.12.15 Cyclic references found
Before finishing the conversion process it is checked if any cyclic references exist between the namespaces. These cyclic references cause problems while validating the XML Schema with several tools.

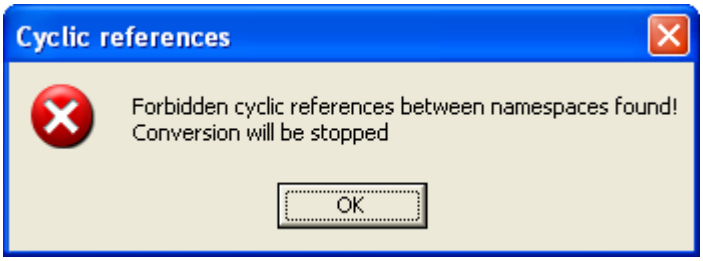


Figure 44 - dialog "Cyclic references" with holdOnError true

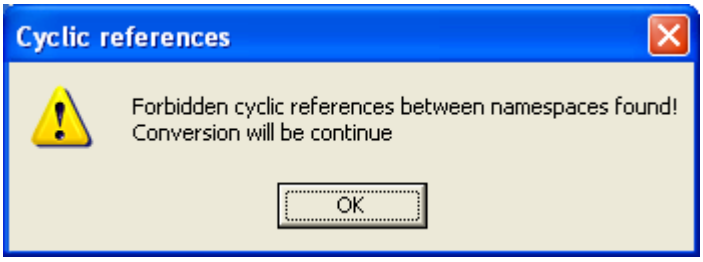


Figure 45 - dialog "Cyclic references" with holdOnError false

The log file contains two entries for every cyclic loop.

```
11:28:41 - [1] - Forbidden cyclic references between namespace 'Payload' and 'General' found!
11:28:41 - [1] - Forbidden cyclic references between namespace 'General' and 'Payload' found!
```

2.4.12.16 Extension check

Checks will be done for the tagged value *extension.* If the tag contains any values other than “levelb” or “levelc” then an error will be raised.

2.4.12.17 General conversion errors

If a general error occurs the description of the error will be shown in the following dialog.
For example if the existing XML Schema files in the specified location are read-only. Therefore the new XML Schema files cannot be written to the specified location.

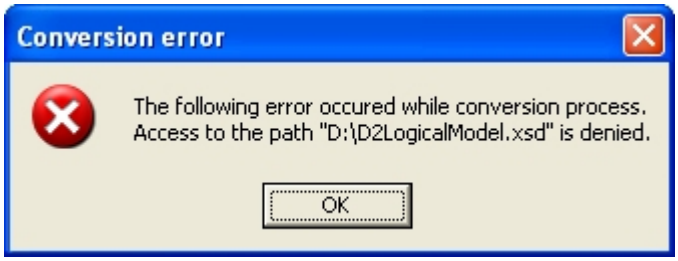


Figure 46 - a general conversion error has been occurred

2.4.12.18 Successful conversion

If the conversion finishes successfully the following dialog appears indicating that the XML Schema files have been created at the specified location.

Figure 47 - conversion successful

2.4.13. logging algorithm

The logging algorithm provides additional information about the conversion process. For example the result of the constraints checking is listed within the log files. The log files are created on a daily basis. The entries of the log file have different levels as described in a previous chapter.

```
09:23:58 - [0] - log started with log level = Debug = 3
09:23:58 - [3] - reading of namespace information successful
09:23:58 - [3] - reading of the data type conversion table successful.
09:23:58 - [0] - reading of configuration file complete
09:24:11 - [0] - start checking the constrains
09:24:11 - [3] - the model does not contain forbidden cycles.
09:24:16 - [3] - the model is according to the naming convention
09:24:19 - [3] - no forbidden multiple inheritance found
09:24:19 - [0] - checking the constrains successful finished
09:24:19 - [0] - start conversion of 'PIM v29.xml'
09:24:19 - [3] - creation of XML Schema file for namespace D2LogicalModel
09:24:19 - [3] - creation of XML Schema file for namespace General
09:24:19 - [3] - creation of XML Schema file for namespace Publication
09:24:21 - [2] - The class 'PoliceOperation' is an empty element.
09:24:31 - [3] - XML Schema output path D:\
09:24:31 - [3] - saving of XML Schema file D2LogicalModel.xsd for namespace D2LogicalModel
09:24:31 - [3] - saving of XML Schema file General.xsd for namespace General
09:24:31 - [3] - saving of XML Schema file Publication.xsd for namespace Publication
09:24:31 - [3] - number of packages = 83
09:24:31 - [3] - number of classes = 270
09:24:31 - [3] - number of attributes = 333
09:24:31 - [3] - number of enumerations = 772
09:24:31 - [3] - number of associations = 163
09:24:31 - [3] - number of generalization = 104
09:24:32 - [0] - the conversion finished successfully
09:24:33 - [0] - log finished
```

Figure 48 - sample log file of a successful conversion

The figure above shows a sample log file of a successful conversion with the logging level 3 defined in the configuration file. The first column shows the time of the event followed by the logging level number. The last column is the underlying text of this log file entry.

2.5. Constraints that are checked by the conversion tool

The following table shows the constraints which are checked by the conversion tool.

Constraint
An aggregation or composition is not navigable.
Only aggregations and compositions are allowed.
Cyclic references are not allowed.
A multiplicity other than 1 or not set at the source of an aggregation or composition is not allowed.
Multiple Inheritance is not allowed
The naming convention has to be fulfilled.
No cyclic references between the namespaces are allowed.
Naming convention
Extension tagged values
Attribute scope check
If two or more associations in a class points to the same class then a role is required
Target class tagged value should point to a existing class

ANNEX



3. Annex

3.1. Table Of Figures