

THE QUICK START-UP GUIDE TO ESMP PROFILING

André Maizener

Jean-Luc Sanson

Wout van Voornveld

Contents

Introduction.....	3
CIM basics.....	4
CIM profiling methodology	4
UML Repository using the CIM.....	7
My First Profile (MFP).....	10
Objectives.....	10
How to get started	11
Preparation	11
The use of CreateProfilePackage construct	16
The use of the IsBasedOn construct	17
The use of the Edit Hierarchical connectors construct	21
Adding (multiple) connector qualifiers	22
The use of AttributeOrder construct	26
The use of Property grouping construct	27
Creating an XSD export.	29
Creating an JSON export.	31
Modifying your profile	33
Adding and Removing attributes from the classes in the messages	33
Making optional attributes mandatory	34
Changing namespace, codelist directory, file export locations, etc.	35
Appendix 1: Overview of user interface functions for different contexts (CIMConteXtor)	36
Appendix 2: Overview of user interface functions for different contexts (CIMSyntaxGen).....	37

Introduction

This startup guide is meant as a lightweight, step-by-step introduction into working with the Common Information Model (CIM) electricity standard, especially the European variant, the European Style Market Profile (ESMP). For those active in the Energy Transition, the exchange of data is crucial and imposes challenges as the recent developments like the regulated sharing of metering and consumption data, customer switching and demand response are not reflected in the current standards yet. Research projects need to build upon existing standards as much as possible, but face the shortcomings of the current energy profiles. This document gives a general overview of the structure of the ESMP and leads the reader to a step-by-step instruction to create a new profile that can be extended to the specific needs. The extensibility is part of a different document that will be published later on this year, entitled 'The pragmatics guide to the Common Information Model extensibility'. The quick start-up guide is a way to get familiar with the basics of the CIM framework and the use of the CIM specific tools, CimContreXtor and CimSyntaxGen, that are freely available as Sparx Enterprise Architect Add-ins.

CIM basics

The Common Information Model (CIM in short) started in the United States in 1999: it is a set of open standards for representing power system components originally developed by the Electric Power Research Institute (EPRI) in North America and now a series of standards under the IEC. Its aim is to allow consistent [management](#) of Power System components, independently of their manufacturer or provider. The CIM is developed, standardized and maintained by the UCAUIG community (Utility Communication Architecture- International Users Group).

CIM is regarded to have a strong semantic model. Opinions differ if CIM can be regarded as an ontology, when you define this as “An ontology is a description of data structure of classes, properties, and relationships in a domain of knowledge.” Leaving out the ontology part, CIM can be defined as follows “an [open standard](#) that defines how managed elements in an Energy [IT environment](#) are represented as a common set of [objects](#) and relationships between them.” The CIM is expressed using UML (Unified Modeling Language).

CIM consists of several parts, targeted at different energy segments and defined by IEC as a series of standards:

- Grid level – IEC 61970
- Enterprise level – IEC 61968
- Market level: IEC 62325

CIM is used to allow the exchange of data between Power systems applications, independent of their internal software architecture or operating platform. The CIM provides a methodology to define data format exchanges (called “Profiles”) whose semantics is conforming to the CIM. Those profiles are defined by several IEC standards, targeted also for different energy applications. One of the data exchanges is targeted for the European Electricity Market (ESMP)

For the sake of this document, we will concentrate on the ESMP.

CIM profiling methodology

The CIM profiling methodology consists of several hierarchical, interconnected layers. The idea behind this is to enable different implementation messages based on common building blocks (The CIM) with clearly defined meaning. In the ESMP, we distinguish three layers (from top to bottom):

- The CIM information model, (building blocks expressed in UML)
- The profile derivation level, (information used for an exchange -the context- described in UML)
- The implementation derivation model. (Information exchange defined in a given syntax: XSD, Json Schema...)

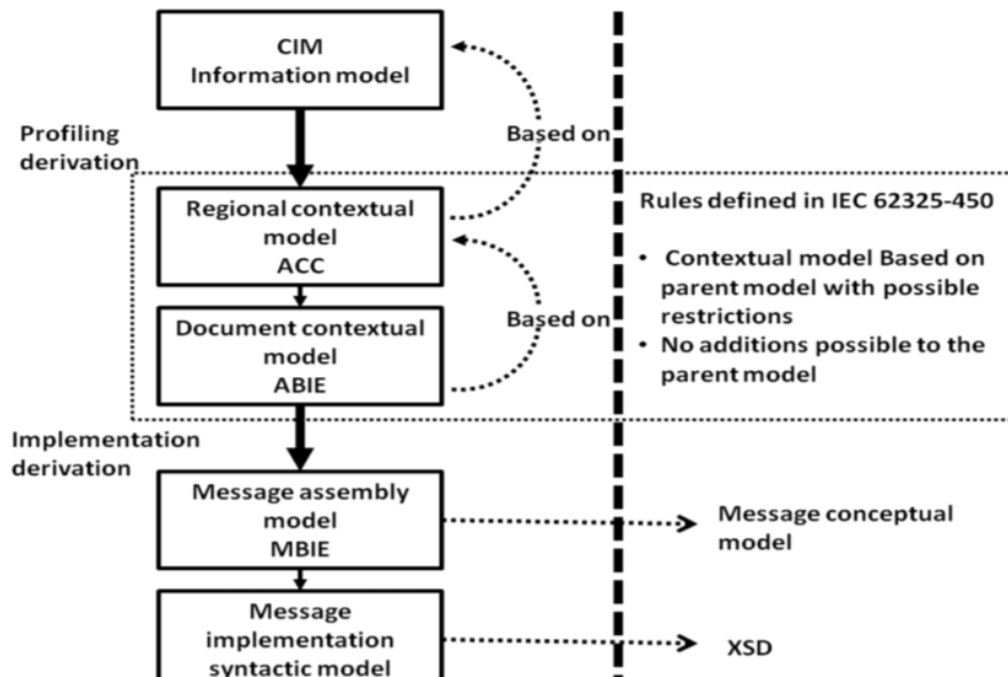
You could compare the three layers with Lego building blocks at the top level, Lego theme offerings (like Pirates, Knights or Disney) at the middle layer and the off the shelf packages you find in the toyshops at the bottom layer.

The profile derivation model consists of two interrelated components:

- Regional Contextual model which is a general “Profile” with all the buildings blocks necessary for the European Electricity Market exchanges and is described using UML. This model is derived from the CIM Information Model according to profiling rules defined by the IEC62325-450 standard and is standardized by IEC as IEC62325-351. UML Entities in this layer are exposed with a ‘**ACC**’ UML stereotype, which stands for Aggregated Core Components.
- Document Contextual model which is also a UML profile derived from the ESMP Regional Model (IEC62325-351) according to IEC62325-450 profiling rules. UML Entities in this layer are exposed with an ‘**ABIE**’ UML stereotype, which stands for Aggregated Business Information Entity.

The implementation derivation model consists of two interrelated components:

- A Message Assembly model in UML: model which is generated from the Document Contextual Model (ABIE objects) using the CimConteXtor tool. UML Entities in this layer are stereotyped as '**MBIE**', which stands for Message Business Information Entity.
- A Message Implementation Syntactic model which is generated from the Message Assembly model (MBIE) using the CimSyntaxGen tool. These syntactic models (XML or Json Schema) are generated according to IEC62361-100 and 104 standards.

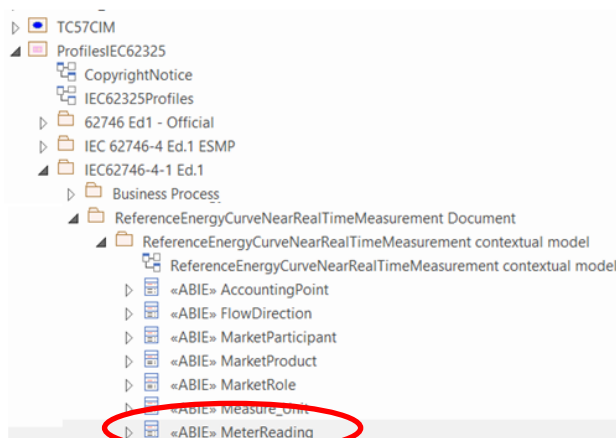


UML Repository using the CIM

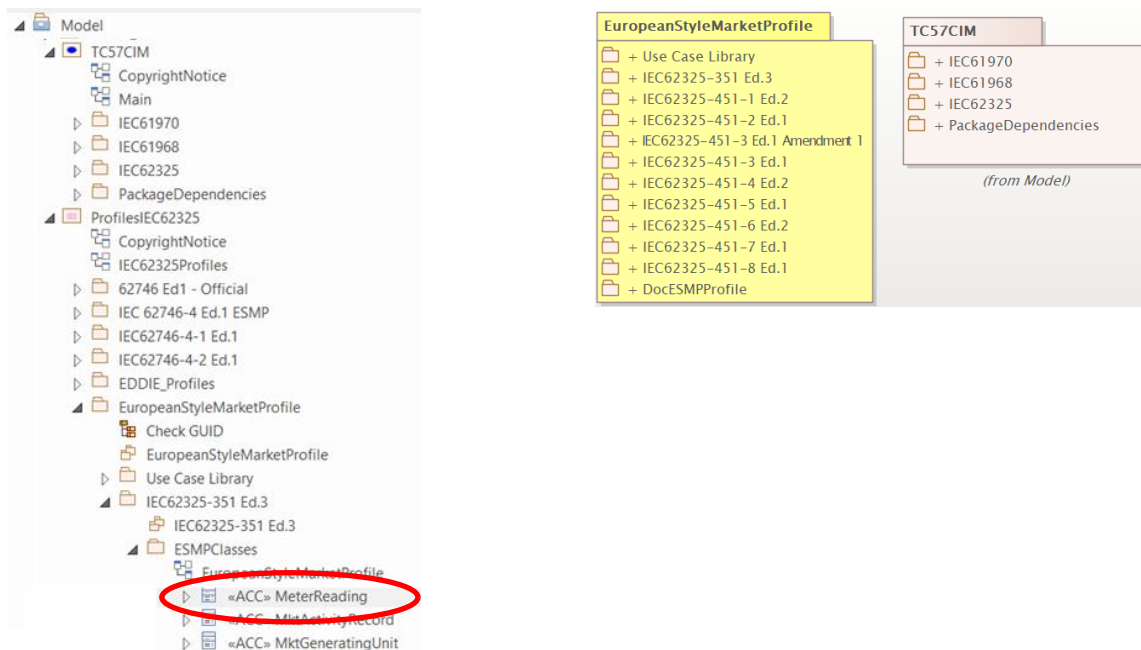
How can you recognize these layers in a CIM repository? Opening the Sparx Enterprise Architect file that supports the IEC 62746-4 standard, you see the following structure:

1. TC57CIM - this is the top level repository. TC57 stands for Technical Committee 57, the IEC working group for CIM management and standard profiles definition.
 - a. IEC 61970 - the standard that defines the Grid components.
 - b. IEC 61968 – the standard that defines Energy Management Systems components.
 - c. IEC 62325 - The standard that defines Electricity Market components.
2. ProfileIEC62325 - the profile specifications for applications using the European Style Electricity Market profiling methodology (IEC62325-450)
 - a. European Style Market Profile - the ESMP specific profiles standardized by IEC
 - i. IEC62325-351 Ed.3 - the latest version of the ESMP data model. Home of the ACC components.
 - ii. IEC62325-451-1 Ed.2 - the ESMP latest acknowledgement business process.
 1. Acknowledgement contextual model. Home of the Acknowledgement document ABIE Components
 2. Acknowledgement assembly model. Home of MBIE Components
 - iii. IEC62325-451-xx other ESMP business processes profiles
 1. Contextual model...
 - b. ENTSO-E packages – Specific Entsoe profiles using the ESMP profiling methodology.
 1. Contextual model. Home of the ABIE Components
 2. Assembly model. Home of the MBIE Components

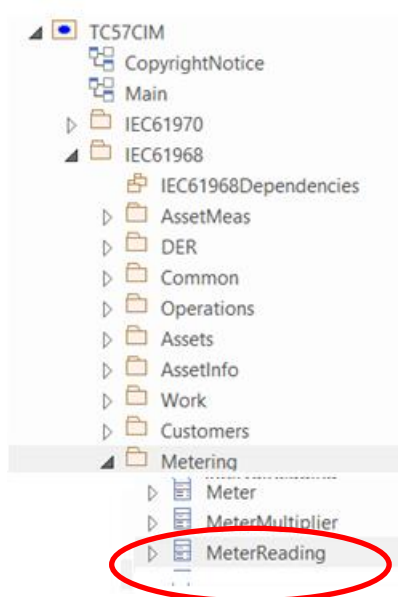
Here follows an example of the hierarchical dependencies of the MeterReading object from the IEC 62746-4 profile. It starts at the Aggregated Business Information Entity (ABIE)



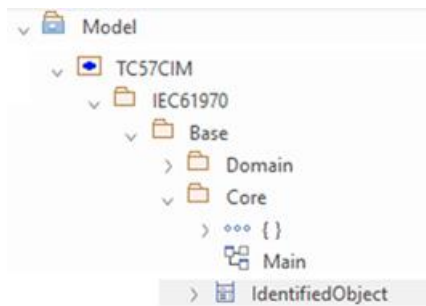
The MeterReading object from the IEC 62746-4 is based on the Aggregated Core Component (ACC) in the EuropeanStyleMarketProfile (ESMP).



In return, this ESMP object is based on the MeterReading object from the IEC 61968 Enterprise profile



This object is an implementation of the generic IdentifiedObject in the IEC61970 Grid profile:



<i>IdentifiedObject</i>	
Metering::MeterReading	
+	valuesInterval: DateTimeInterval [0..1]
+	isCoincidentTrigger: Boolean [0..1]

My First Profile (MFP)

Objectives

In this exercise you will create your first profile: a contextual Model derived from the IEC 62325-351 Ed. 3 ESMP core library.

Prerequisite

1. UML basics
2. Enterprise Architect basics

After this exercise you will be able to:

Install CimContextor and CimSyntaxGen Enterprise Architect add-ins

Configure add-in options

Have the basis of the profiling methodology

Understand what is an Information Model like CIM and how you can use it

Understand what is a regional model like ESMP

Start your own UML contextual model profile in a copy of the ENTSO-E ESMP repository file.

Start using CimContextor for contextual models

Use the CreateProfilePackage menu

Select classes from the base library (ESMP), and select appropriate class attributes and relationships

Use Qualifiers

Start using CimContextor for message assembly models

Use AttributeOrder menu

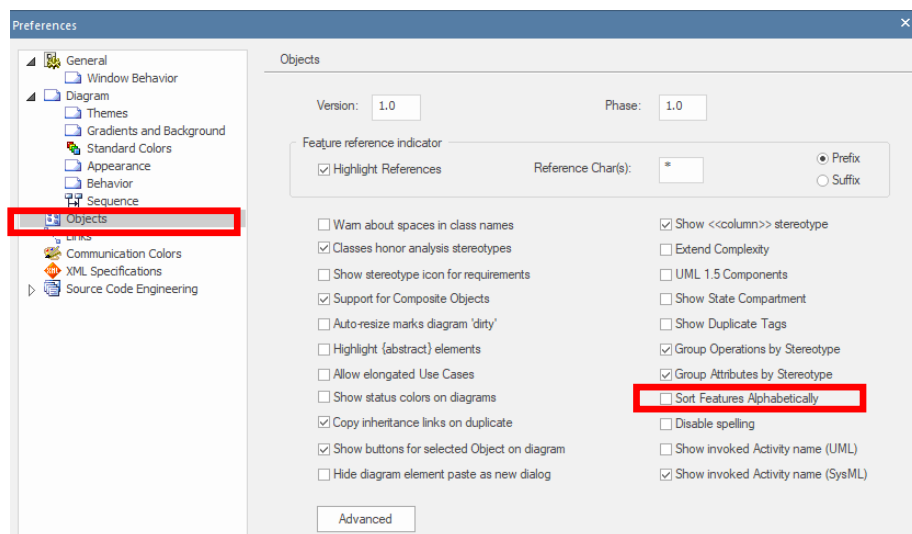
Use Property grouping for all 0-to1 and 1-to-1 relationships.

Start using CimSyntaxgen for XSD or Json Schema exports

How to get started

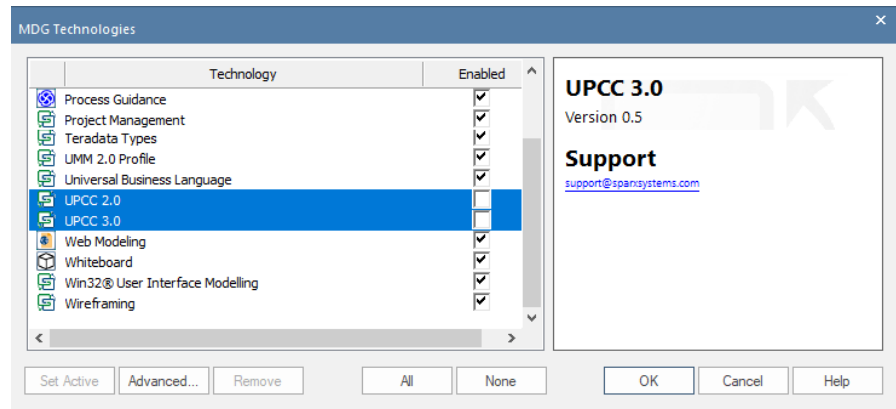
Preparation

- You should have basic knowledge of Unified Modelling Language (UML) otherwise it will be hard to follow the concepts explained here.
- Make sure you have installed Sparx Enterprise Architect (from here on referred to as EA). The preferred version is a 32 bits edition of either EA 14, 15 or 16. Although the Add-ins to be installed work with 64 bits versions, the importing of the ENTSO-E repository works better with the 32 bits.
- Some Enterprise Architect settings are important to check before starting. The screenshot below are taken from version 15. More recent versions may have them hidden under different menu options, shortcut key often work under all versions:
 - a. Some functions in the CIM add-ins expect folder (in EA called packages) to be in a certain order. To prevent EA to order the packages alphabetically, check the following setting under Start/Preferences/Preferences [shortcut keys Ctrl+F9], then select ' Objects' on the left and make sure the ' Sort Features Alphabetically' is unchecked.



- b. Uncheck two add-ins that might disturb the functionality of the CIM add-ins. Go to Specialize/Manage Technology opening the MDG Technologies dialog box. Scroll down in the list and make sure the UPCC2.0 and UPCC3.0 are both

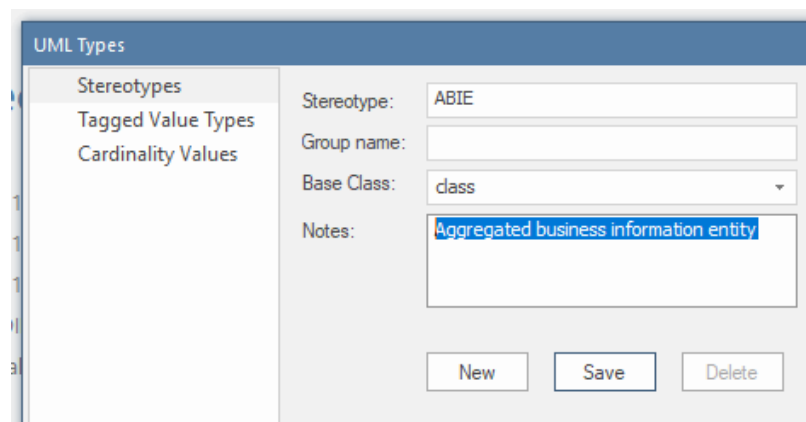
unchecked.



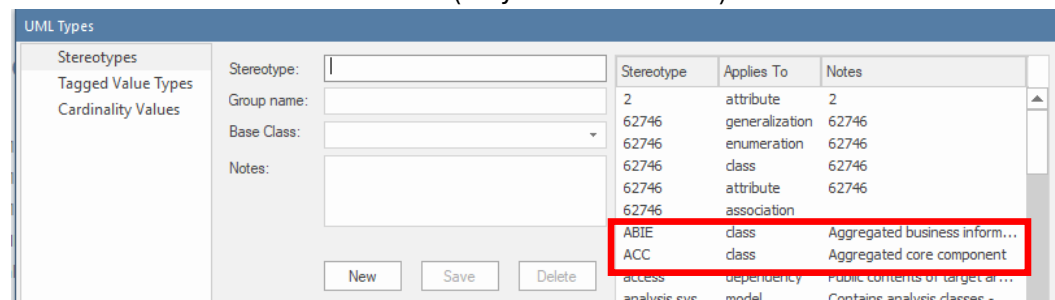
- c. The ESMP needs four UML stereotypes, which you need to define; three are of type class and one is of type dependency.

StereoType	Base Class	Notes
ABIE	Class	Aggregated business information entity
ACC	Class	Aggregated Core Component
MBIE	Class	Aggregated message information entity
IsBasedOn	Dependency	IsBasedOn

Go to menu Configure/UML Types opening the UML Type dialog box. Check to see if the four entries are there. If not Click New, enter the values from the table and click Save.



You should see all four entries now (only two shown here):

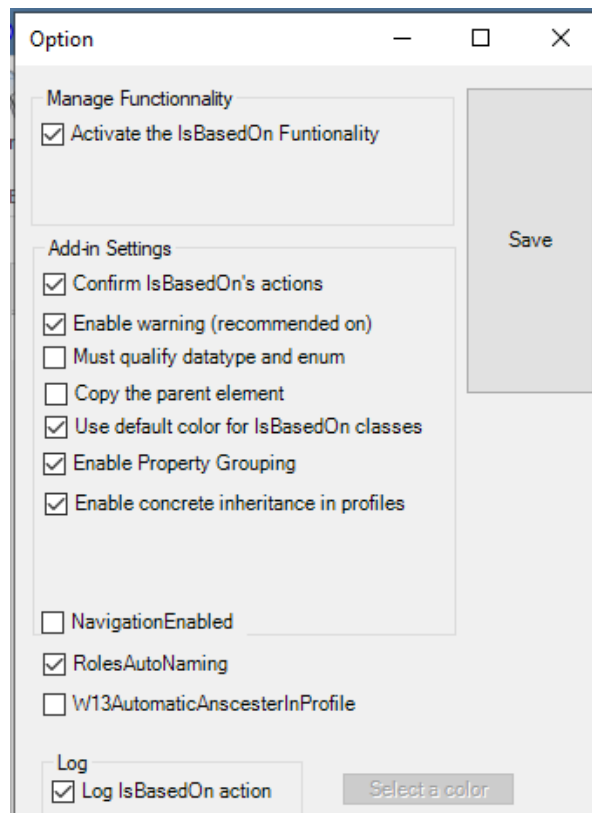


- Install both CIMConteXtor and CIMSyntaxGen from the Zamiren Website. You need to sign up (which is free) in order to download the packages:

<https://www.zamiren.fr/index.php/en/downloads>

Read the installation instruction carefully.

- Set the following Options in the Specialise/CIMConteXtor/Options menu:



- Copy the ENTSO-E ESMP repository from their website:

https://www.entsoe.eu/Documents/EDI/Library/cim_based/20230522_ESMPv4.zip

Unzip the file

20230517_ESMPv4_iec61970cim17v34_iec61968cim13v12_iec62325cim04v09.eap

and rename it to

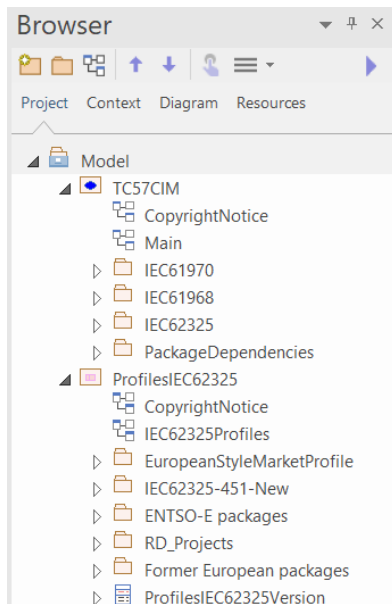
20230517_ESMPv4_iec61970cim17v34_iec61968cim13v12_iec62325cim04v09_MFP.eap

thus keeping the source content in the title:

iec61970cim version 17v34

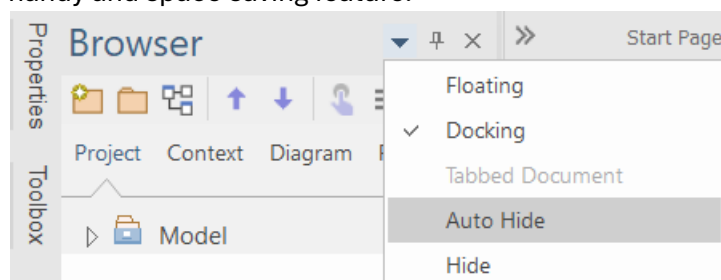
iec61968cim version 13v12

iec62325cim version 04v09

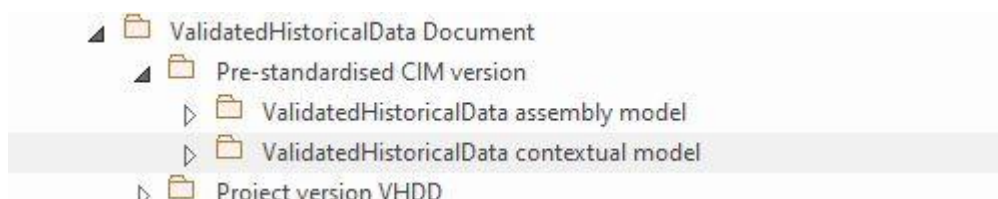


- General note on working with Sparx Enterprise Architect add-ins in combination with the Auto Hide feature of the Project Browser.

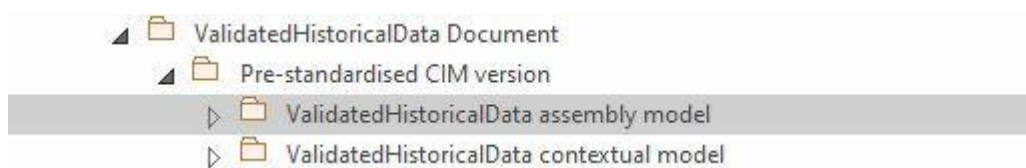
The Project Browser window has a setting that automatically hides when it loses the focus. A handy and space saving feature.



However, add-ins frequently act on the currently selected package in the Project Browser. Here is the twist: you need explicitly to select the package because if you change the focus from the Project Browser in Auto Hide mode and return later on, it suggests that the latest selection is still active. It looks like this:



But when selected, it looks like this:

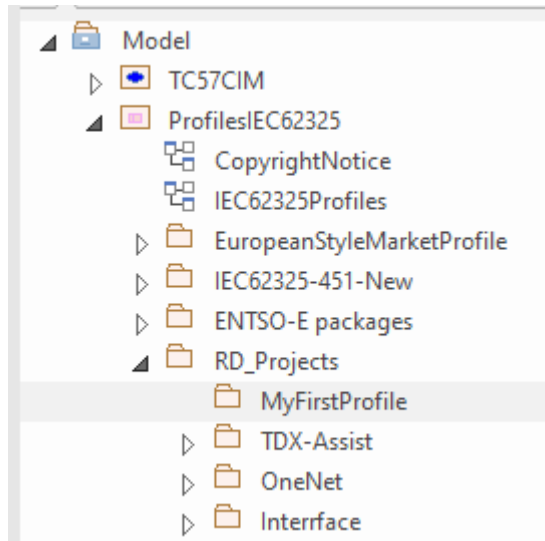


So, there is a triple state: unselected, previously selected and currently selected. Do make sure your package is properly selected, otherwise your add-ins will not work as expected.

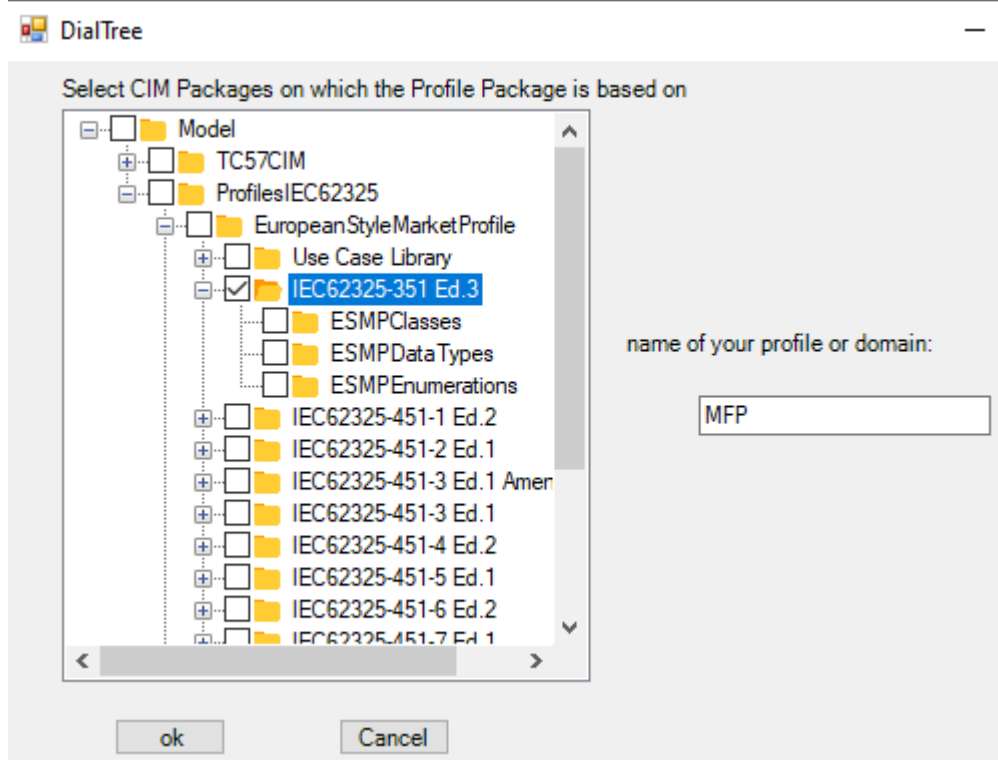
The use of CreateProfilePackage construct

Now everything is in place, let us start with the profile:

- Create a new package under RD_project, give it the full name MyFirstProfile



- From the CIMConteXtor Menu select “Create Profile Package’ and select the IEC62325-351 ED.3 from the menu and give the MFP name to your profile as below:

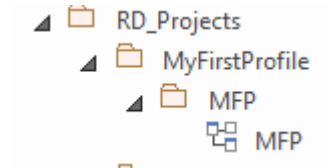


name of your profile or domain:

MFP

The use of the IsBasedOn construct

Now we are going to select classes from the ESMP library to use in our profile. Open the class diagram created in the previous step:



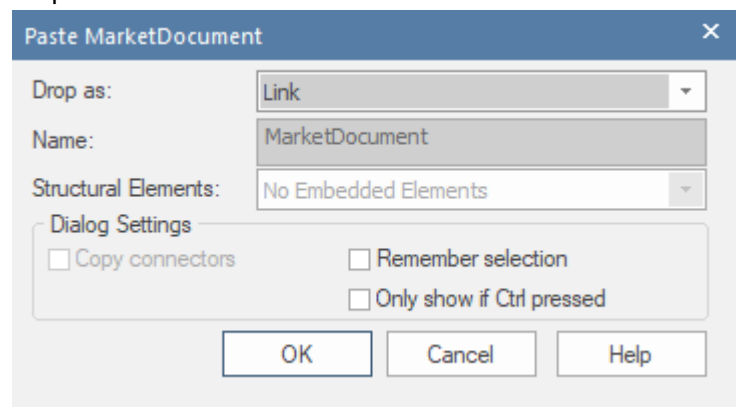
Rename the MFP package to read 'MFP contextual model' as this is exactly what we will be working on.

Open the ESMPClasses in the Project browser from here:

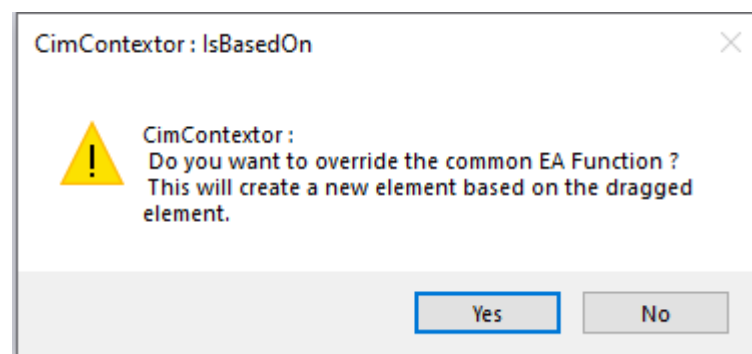
[Model.ProfilesIEC62325.EuropeanStyleMarketProfile.IEC62325-351 Ed.3.ESMPClasses] and drag and drop the following <<ACC>> stereotyped classes to your class diagram:

MarketDocument
MarketParticipant
MarketRole
Time_Period
TimeSeries
MarketEvaluationPoint

When dropping a class, first you will be asked to confirm it through the standard EA dialog, respond 'OK'



Subsequently you will be asked to confirm whether you want to use the CIMContextor specific function. Respond 'Yes'

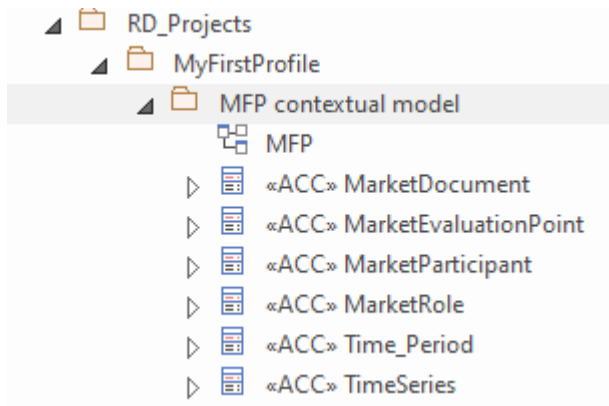


At the end you will be presented the 'IsBasedOn functionality; dialog which for now we will excepting all defaults by clicking on 'Execute IsBasedOn'. The dialog may be hidden , but can be shown using ALT+TAB key sequence.

Possible inheritance	Attribute name	LowerB...	UpperB...
<input checked="" type="checkbox"/>	mRID	0	1
<input checked="" type="checkbox"/>	description	0	1
<input checked="" type="checkbox"/>	revisionNumber	0	1
<input checked="" type="checkbox"/>	type	0	1
<input checked="" type="checkbox"/>	createdDateTime	0	1
<input checked="" type="checkbox"/>	docStatus	0	1
<input checked="" type="checkbox"/>	title	0	1
<input checked="" type="checkbox"/>	status	0	1

Do this for all six classes.

The result should look like this:



Editing stereotypes

All classes still have the stereotype of an aggregated core component (ACC). We need to change them to the appropriate stereotype for the contextual model, which is aggregated business information object (ABIE).

To do so, select each class, click on Specialize/CimConteXtor/Edit an IsBasedOn deselect 'Copy parent's stereotype' and click on 'Edit class's stereotype' to select 'ABIE' from the list and deselect 'ACC'. Click 'Save' to close the dialog and click 'Execute IsBasedOn'.

Do so for all five other classes.

The restriction of attributes

You might not need all attribute from the ESMP. For the sake of this exercise we will restrict

ourselves to the following:

MarketDocument

mRID

createdDateTime

MarketParticipant

mRID

MarketRole

Type

Time_Period

timeInterval

TimeSeries

version

MarketEvaluationPoint

mRID

In order to select the attribute, select the class, , click on Specialize/CimConteXtor/Edit an IsBasedOn and deselect the unneeded attributes from the list. click 'Execute IsBasedOn'.

Attribute				
	Possible inheritance	Attribute name	LowerB...	UpperB...
<input checked="" type="checkbox"/>		mRID	0	1
<input type="checkbox"/>		description	0	1
<input type="checkbox"/>		revisionNumber	0	1
<input type="checkbox"/>		type	0	1
<input checked="" type="checkbox"/>		createdDateTime	0	1
<input type="checkbox"/>		docStatus	0	1
<input type="checkbox"/>		title	0	1
<input type="checkbox"/>		status	0	1

The use of entity qualifiers

The MarketDocument object features is in all of the messages created from the ESMP. To distinguish one message from the other, it is good practice to use a qualifier (prefix) to the MarketDocument. Again this can be done using Specialize/CimConteXtor/Edit an IsBasedOn. With the MarketDocument class selected, we first create a new Qualifier called “ MyFirstProfile”

IsBasedOn functionality : Updating Marke

Qualifier
Enter a qualifier or create one :

Select

No qualifier

Delete

Create

MyFirstProfile

☒ Allowed to class and role

Create

Subsequently we can select it from 'Select'

IsBasedOn functionality : Updating Marke

Qualifier
Enter a qualifier or create one :

Select

MyFirstProfile

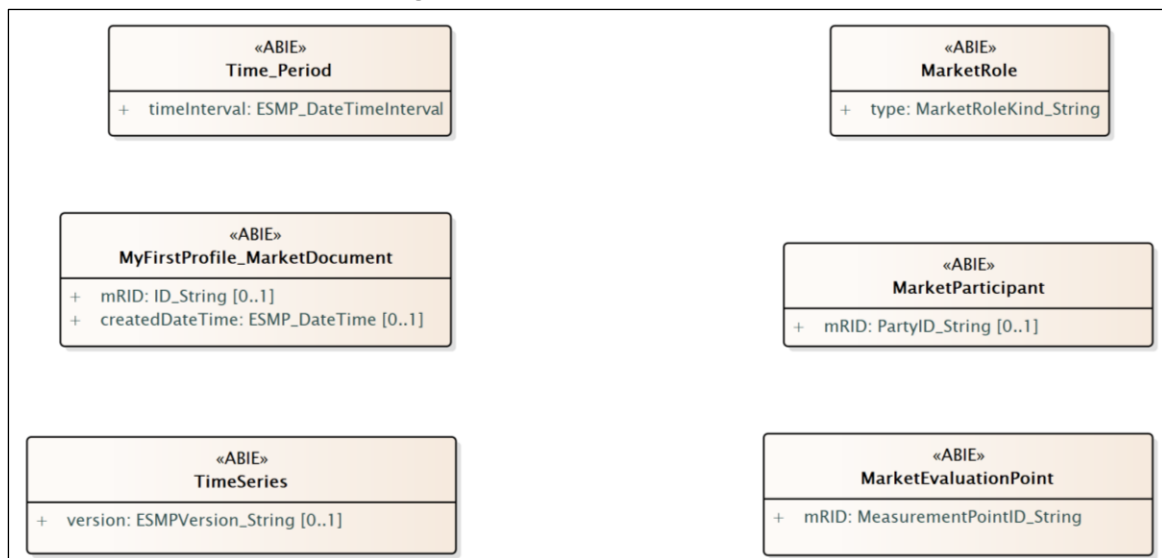
Delete

Create

☒ Allowed to class and role

Create

Your class should look something like this:



The use of the Edit Hierarchical connectors construct

Now we have created six classes without (visible) connectors. As the classes derive from the ESMP library, they do have connectors, which we will need to select here.

Start at the top level of your class structure, usually that is the MarketDocument class. Select 'MyFirstProfile_MarketDocument' class and go to Specialize/CimConteXtor/Edit Hierarchical connectors. First we will connect the Time-Period class.

Activate the checkbox in front of the class and select the row.

Click on 'Modify selected association'

Select the link to edit

Selected class: MFP contextual model::MyFirstProfile_MarketDocument

N°	Associated class	Selected class role	Associated class
<input type="checkbox"/> 0	MFP contextual model::MarketParticipant		MarketParticipant
<input type="checkbox"/> 1	MFP contextual model::MyFirstProfile_MarketDocument		MarketDocument
<input checked="" type="checkbox"/> 2	MFP contextual model::Time_Period		Period
<input type="checkbox"/> 3	MFP contextual model::TimeSeries	Original_MarketDocument	TimeSeries
<input type="checkbox"/> 4	MFP contextual model::TimeSeries		TimeSeries

Detail of the connector n° : 2

N°	Selected class role	Associated class end role	Oriented

Options

☒ Duplicate parent's constraint

Modify selected association

Modify edited association

Save

Cancel

The next dialog appears where you can customise the connector.

Editing Association MarketDocument - Time_Period

MarketDocument

class role

Parent's role : No qualifier

☒ Containment

☐ By Ref

Cardinality

Selected class cardinality : 1

Time_Period

Other class role

Parent's role : Period

No qualifier

Period

☐ Containment

☐ By Ref

Other end cardinality : 1

Options

☒ Copy parent's TaggedValues

☒ Copy parent's notes

Stereotype

☒ Copy Parent's stereotype

Constraints

☒ copy parent's constraints

No Constraint

ManageConstraint

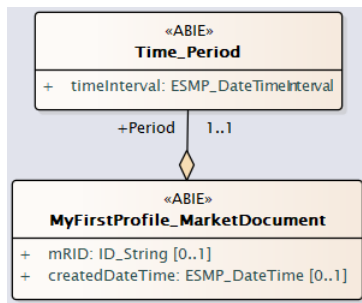
Manage qualifier

Save

Cancel

As we want to have just one Time_Period in our message, we change the other end cardinality to read 1: 1. Click on Save in this dialog box and on Save in the previous dialog box.

Your connection is now established:



Do this for the following connections as well:

MyFirstProfile_MarketDocument connected to MarketParticipant – cardinality 1:1

MarketParticipant connected to MarketRole – cardinality 1:1

MyFirstProfile_MarketDocument connected to TimeSeries – cardinality 1:*

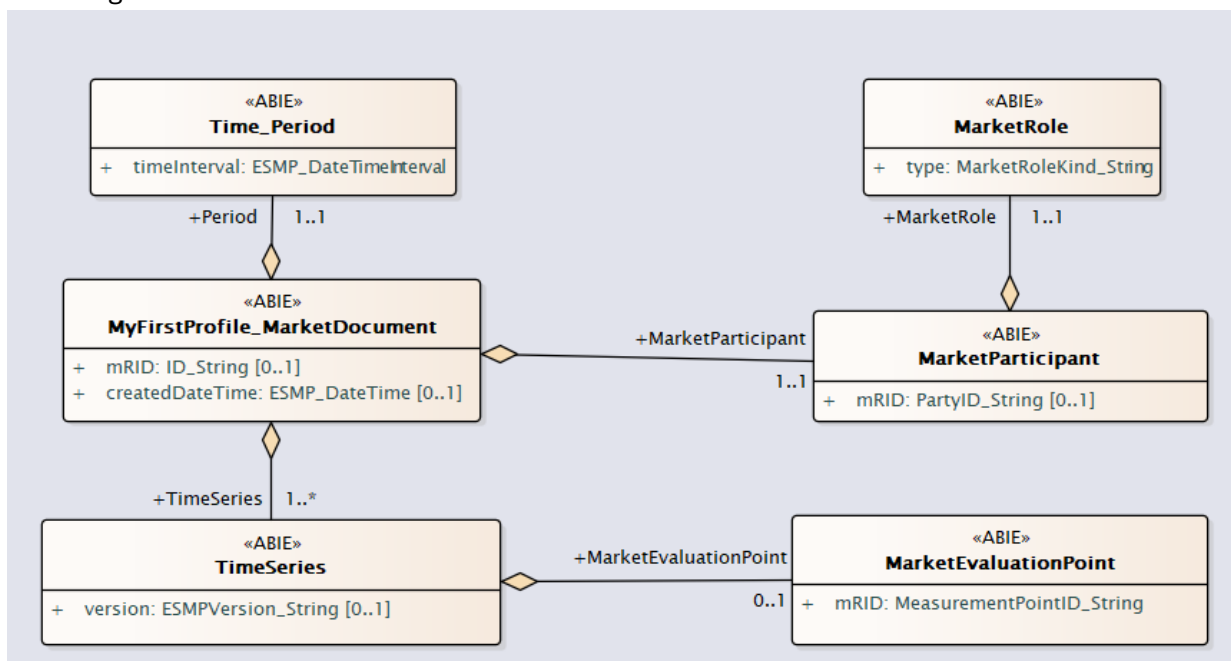
TimeSeries connected to MarketEvaluationPoint – cardinality 0:1

(hint: take the TimeSeries where the Associated Class and the associated Class End has the value TimeSeries)

	Associated class	Selected class role	Associated class end
0	MFP contextual model::MarketParticipant		MarketParticipant
1	MFP contextual model::MyFirstProfile_MarketDocument		MarketDocument
2	MFP contextual model::Time_Period		Period
3	MFP contextual model::TimeSeries	Original_MarketDocument	
4	MFP contextual model::TimeSeries		TimeSeries

If you accidentally connected the wrong class, you can delete it directly from the diagram. There is no need to use the CimConteXtor tool for this.

Your diagram should now look like this:



Adding (multiple) connector qualifiers

Entities can be given qualifiers, like we saw in MyFirstProfile_MarketDocument, but connections can have qualifiers as well. Especially, if we need to have multiple connectors, we have to

distinguish the one from the other. When would we want to use this? Well, a MarketDocument can be send from a certain MarketParticipant to another MarketParticipant. In this case we can use two qualifiers 'sender' and 'receiver'.

Here is how:

First, we add the qualifier to the already established MarketParticipant. Select 'MyFirstProfile_MarketDocument' class and go to Specialize/CimConteXtor/Edit Hierarchical connectors. Select the MarketParticipant in the top left corner of the dialog and select in the details section the MarketParticipant as shown below. Click on 'Modify edited association'

N°	Selected class role	Associated class end role	Oriented
<input checked="" type="checkbox"/> 0		MarketParticipant	true

Options

☒ Duplicate parent's constraint

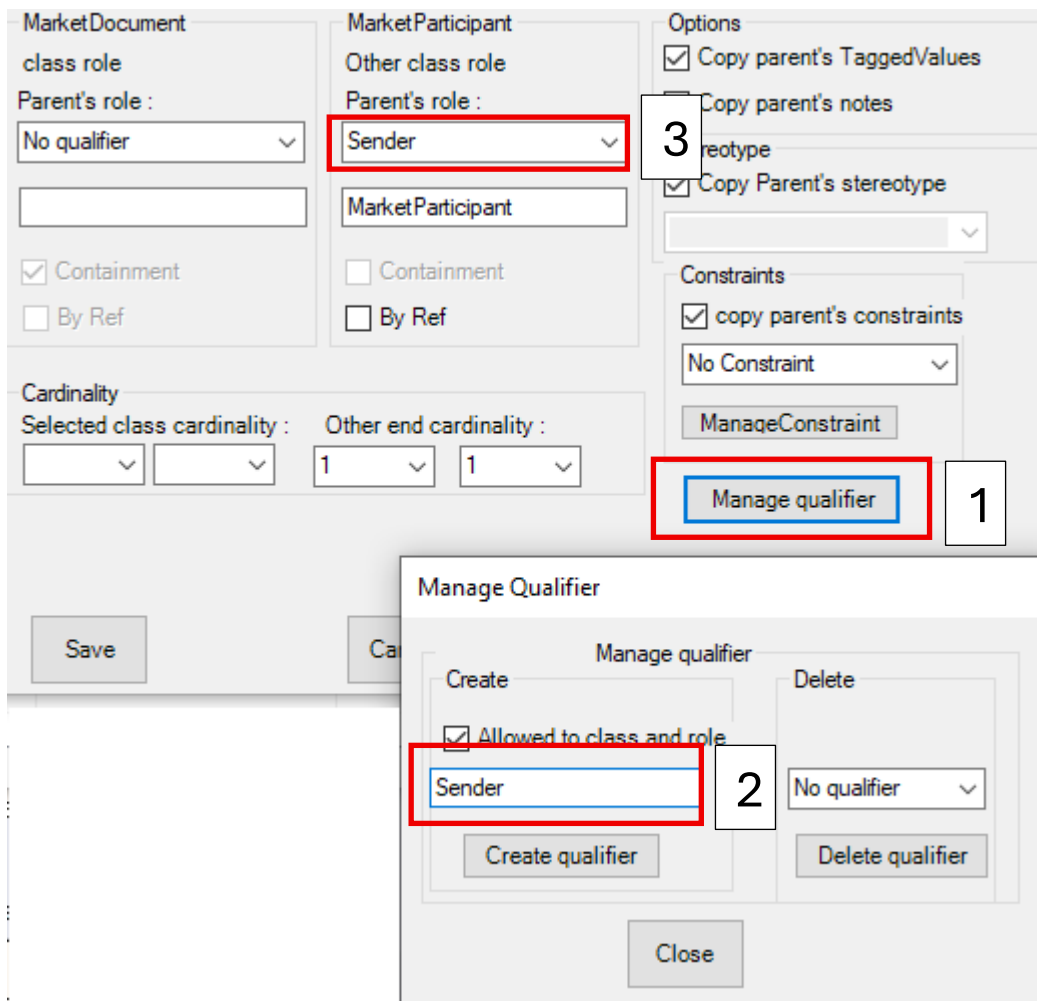
Modify selected association

Modify edited association

Save

Cancel

If you have not created a qualifier with the name 'Sender', now is the time to do so. Click on 'Manage Qualifier' and enter 'Sender' as the new qualifier. Save and select the newly entered value in the main dialog. Click on Save to close the dialog



Your first qualifier connection is now made:

Detail of the connector n° : 0			
N°	Selected class role	Associated class end role	Oriented
<input checked="" type="checkbox"/> 0		Sender_MarketParticipant	true

Now we create the second connector with the 'Receiver' qualifier. Select the MarketParticipant in the top left corner of the dialog and select 'Modify selected association' in the right lower corner of the dialog. This brings up the details dialogue where we first are going to create the 'Receiver' qualifier, select it subsequently from the list. Do not forget to change the cardinality to read 1:1 as shown below:

Editing Association MarketDocument - MarketParticipant

MarketDocument class role Parent's role : No qualifier <input checked="" type="checkbox"/> Containment <input type="checkbox"/> By Ref Cardinality Selected class cardinality : <input type="text"/>	MarketParticipant Other class role Parent's role : MarketParticipant Receiver MarketParticipant <input type="checkbox"/> Containment <input type="checkbox"/> By Ref Other end cardinality : <input type="text"/>	Options <input checked="" type="checkbox"/> Copy parent's TaggedValues <input checked="" type="checkbox"/> Copy parent's notes Stereotype <input checked="" type="checkbox"/> Copy Parent's stereotype Constraints <input checked="" type="checkbox"/> copy parent's constraints No Constraint ManageConstraint Manage qualifier
---	---	--

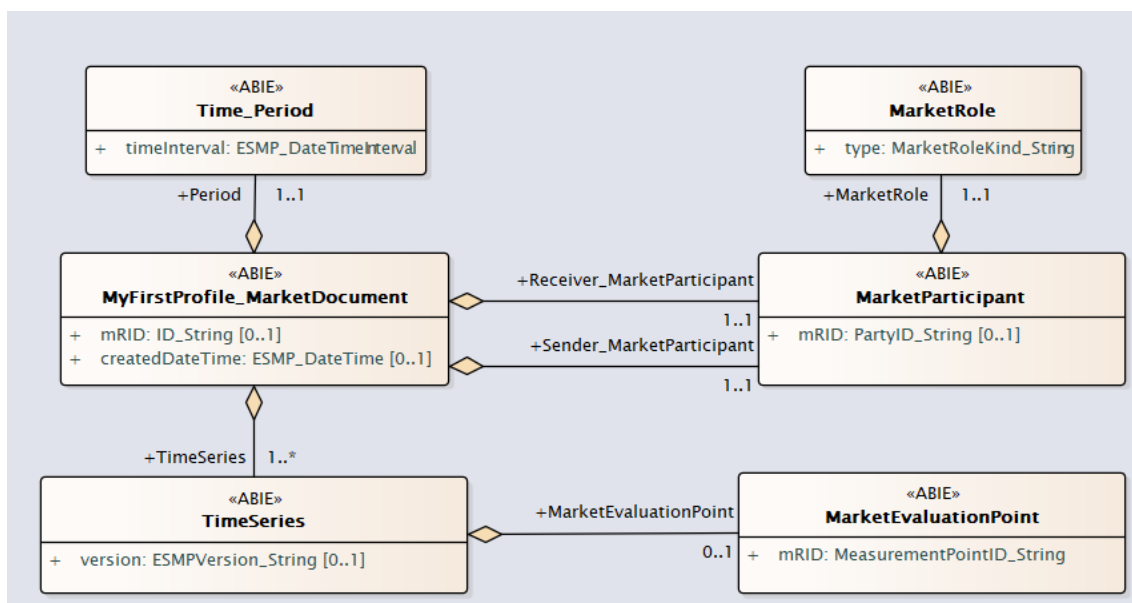
Save Cancel

This will give you two connector between MarketDocument and MarketParticipant:

Detail of the connector n° : 0

N°	Selected class role	Associated class end role	Oriented
<input checked="" type="checkbox"/> 0		Sender_MarketParticipant	true
<input checked="" type="checkbox"/> 1		Receiver_MarketParticipant	true

Save your changes, should see this class diagram (the two connectors may be positioned on top of each other, so it looks like there is only one connector created):



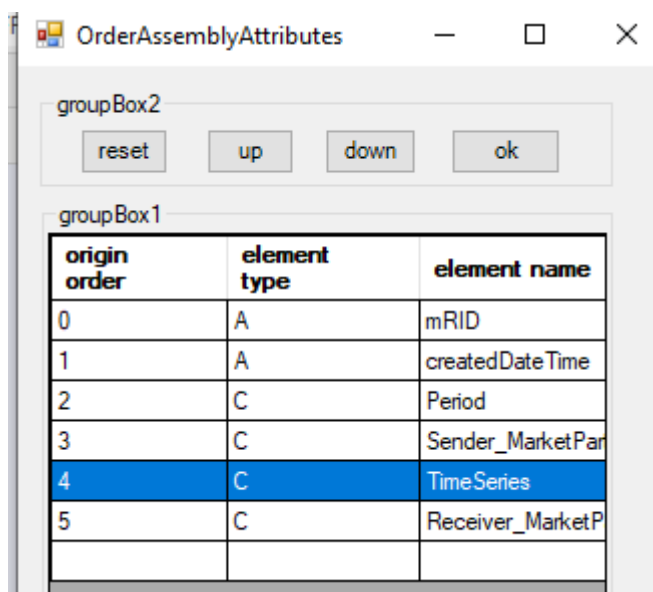
The use of AttributeOrder construct

Attribute ordering is used to determine the order of attributes in the eventual message. It also generates a tagged value that is used in the rest of the process. For this reason you do have to walk through all classes of your diagram, even though – in our case – we have mostly just one attribute.

Starting from the bottom classes to the root classes, do the following:

Select a class, choose from the menu Specialize/CimConteXtor/AttributeOrder. Use the “up” or “down” button to order each attribute of the class (element type “A”) or the association end role name for the associated classes (element type “C”). Click Ok.

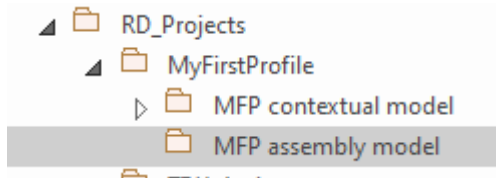
Note: 1..* (one to many) relations have to be at the end of the connectors. In the example below this applies to TimeSeries, which we will need to put at the end of the list:



The use of Property grouping construct

The “PropertyGrouping” option is to be used to generate from a contextual model the associated assembly model. All the classes associated with a multiplicity of 0..1 or 1..1 to a class are inserted within this later class.

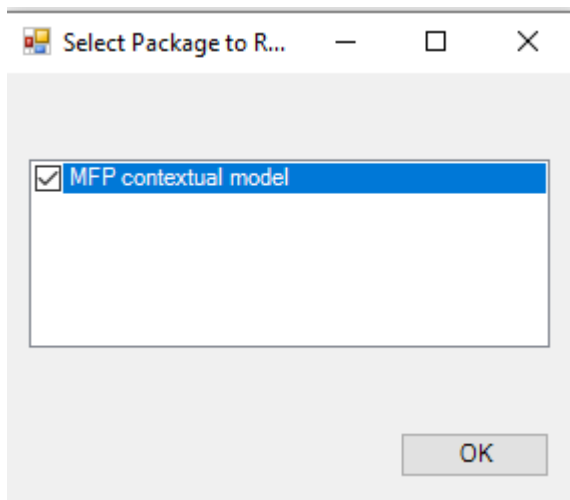
First, make a assembly package to hold the generated classes, call it ‘ MFP assembly model’



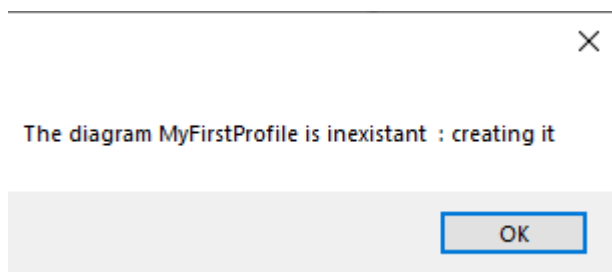
Make sure this package is selected, then choose Specialize/CimConteXtor/PropertyGrouping.

Note: if the assembly package is not empty, all its content will be deleted.

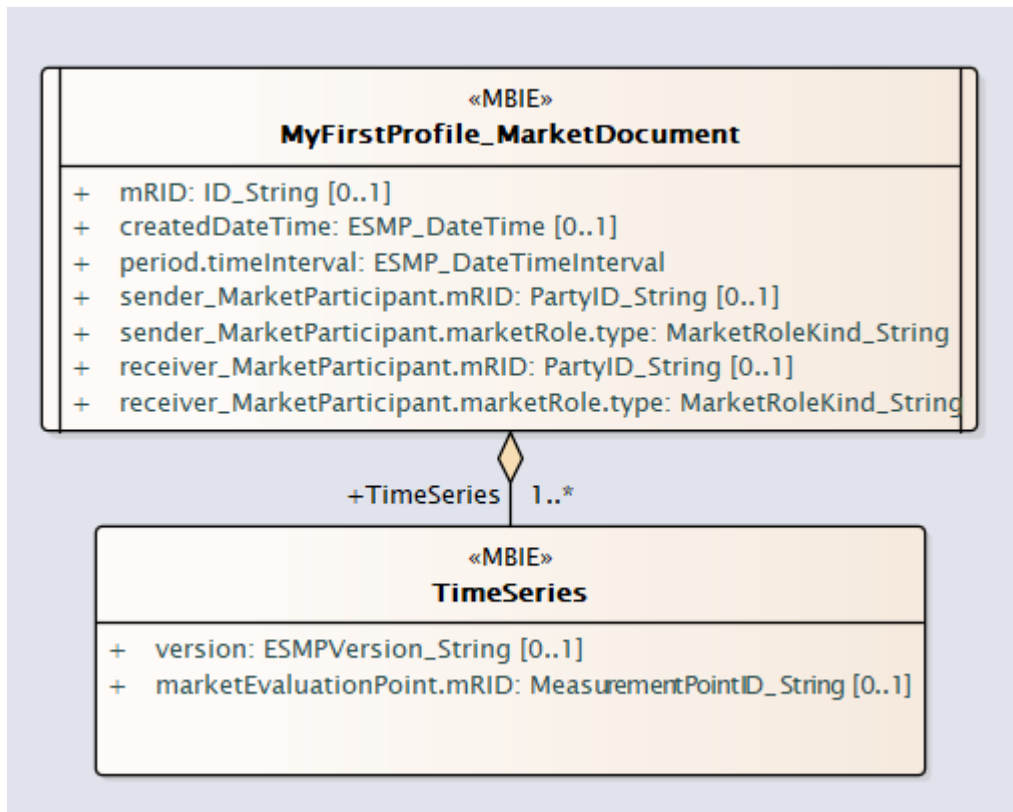
Confirm the contextual package used as input in the dialog



As we have not created a diagram to hold the assembly model, CinConteXtor helps to create one for you.

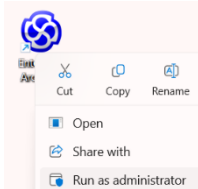


The result is this:



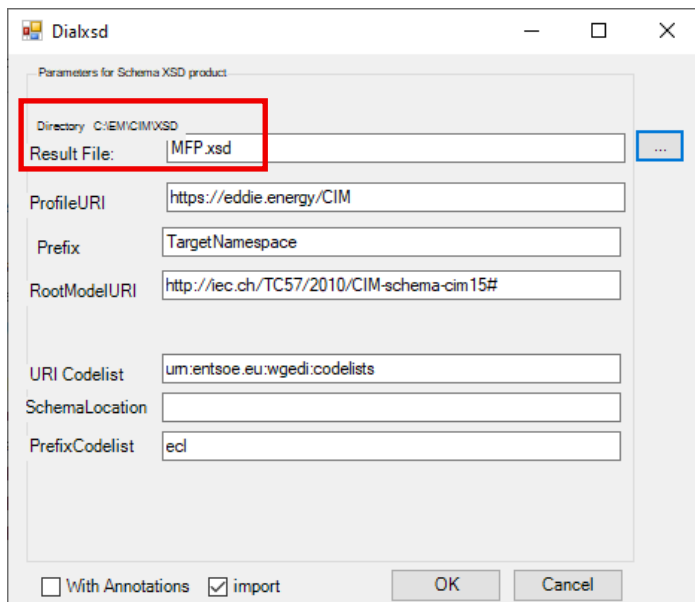
Creating an XSD export.

Before using CimSyntaxGen make sure you have started Enterprise Architect with administrative rights otherwise the file creation might fail.

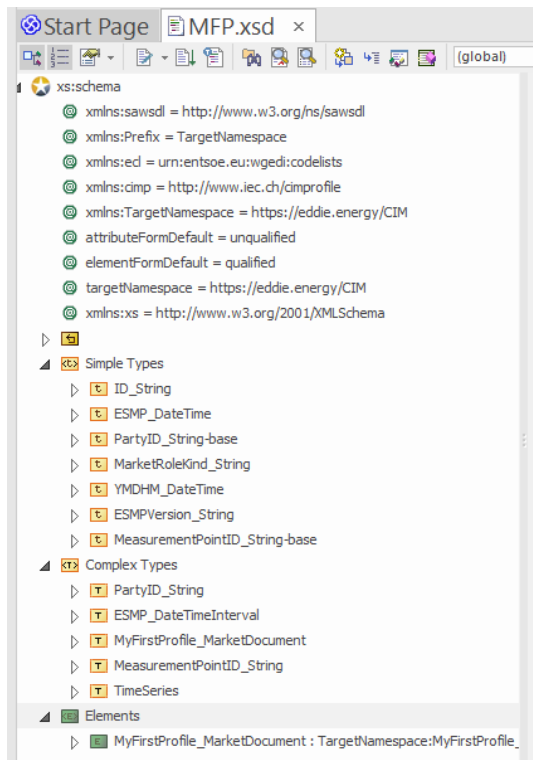


Make sure the assembly package is selected, then choose Specialize/CimSyntaxGen/XSD/XSD WG16.

Make sure the XSD gen dialog resembles the following screen:



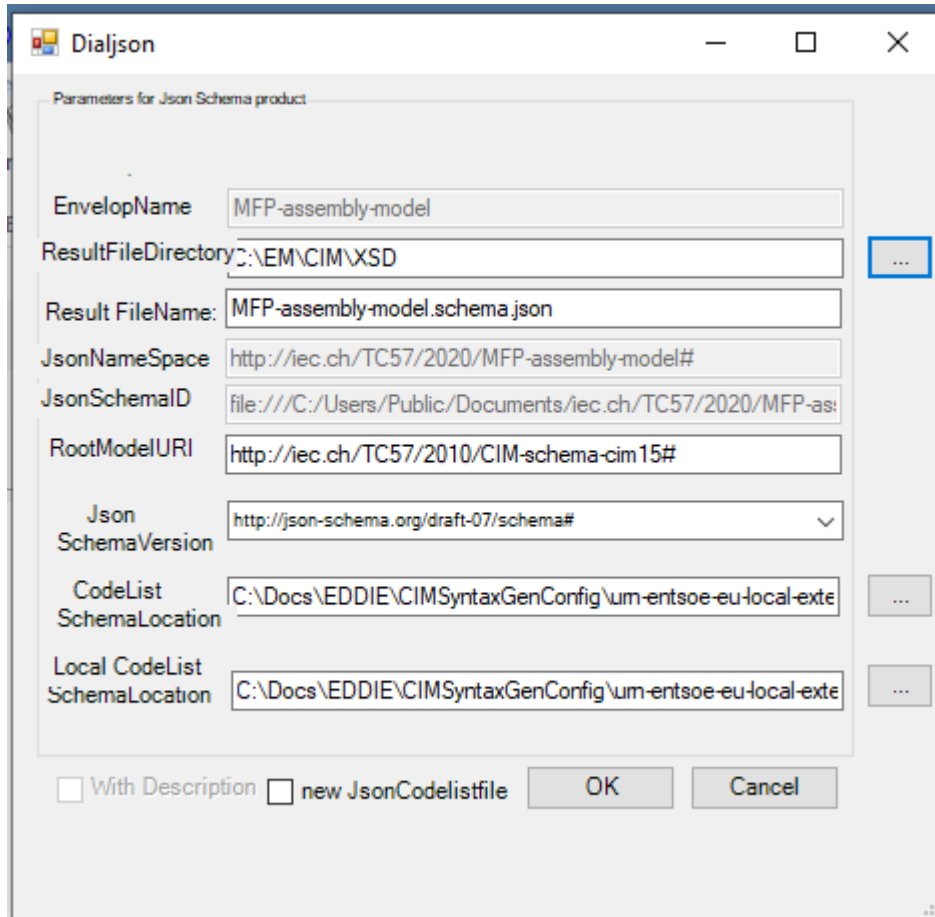
Choose your export directory and filename according to your environment. Click 'OK' . Your XSD is created:



Creating an JSON export.

Make sure the assembly package is selected, then choose Specialize/CimSyntaxGen/JSON/JSON WG16.

Make sure the Dialjson dialog resembles the following screen:



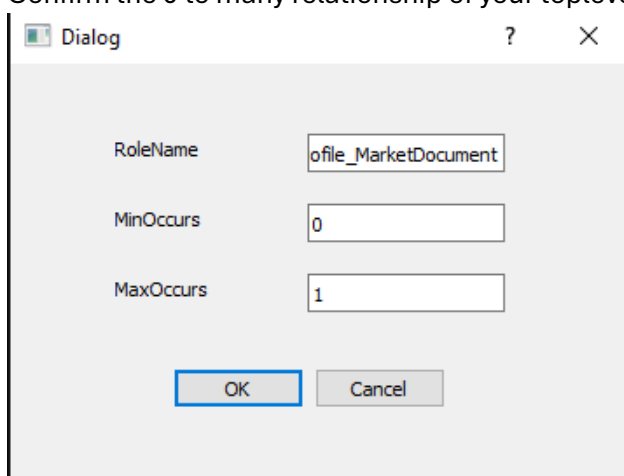
The screenshot shows the 'Dialjson' dialog box with the following fields and values:

- EnvelopName: MFP-assembly-model
- ResultFileDirectory: C:\EM\CIM\XSD
- Result FileName: MFP-assembly-model.schema.json
- JsonNameSpace: http://iec.ch/TC57/2020/MFP-assembly-model#
- JsonSchemaID: file:///C:/Users/Public/Documents/iec.ch/TC57/2020/MFP-as
- RootModelURI: http://iec.ch/TC57/2010/CIM-schema-cim15#
- Json SchemaVersion: http://json-schema.org/draft-07/schema#
- CodeList SchemaLocation: C:\Docs\EDDIE\CIMSyntaxGenConfig\um-entsoe-eu-local-exte
- Local CodeList SchemaLocation: C:\Docs\EDDIE\CIMSyntaxGenConfig\um-entsoe-eu-local-exte

At the bottom, there are two checkboxes: 'With Description' and 'new JsonCodelistfile', both of which are unchecked. To the right of these checkboxes are 'OK' and 'Cancel' buttons.

Choose your export directory and filename according to your environment. Click 'OK'.

Confirm the 0 to many relationship of your toplevel class:



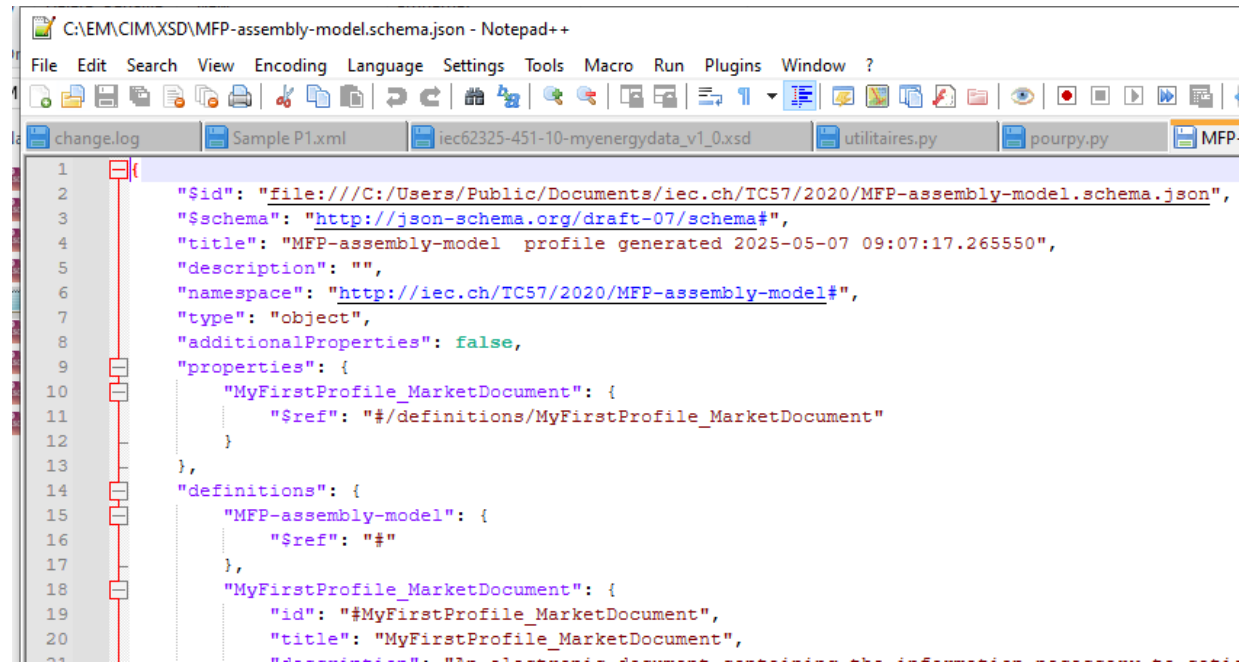
The screenshot shows a 'Dialog' box with the following fields and values:

- RoleName: ofile_MarketDocument
- MinOccurs: 0
- MaxOccurs: 1

At the bottom, there are 'OK' and 'Cancel' buttons.

Click 'Ok'.

Your JSON is created:



The screenshot shows a Notepad++ window with the title bar 'C:\EM\CIM\XSD\MFP-assembly-model.schema.json - Notepad++'. The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations and editing. The tab bar shows several open files: change.log, Sample P1.xml, iec62325-451-10-myenergydata_v1_0.xsd, utilitaires.py, pourpy.py, and MFP. The main text area displays a JSON schema document with the following content:

```
1 {
2   "$id": "file:///C:/Users/Public/Documents/iec.ch/TC57/2020/MFP-assembly-model.schema.json",
3   "$schema": "http://json-schema.org/draft-07/schema#",
4   "title": "MFP-assembly-model profile generated 2025-05-07 09:07:17.265550",
5   "description": "",
6   "namespace": "http://iec.ch/TC57/2020/MFP-assembly-model#",
7   "type": "object",
8   "additionalProperties": false,
9   "properties": {
10     "MyFirstProfile_MarketDocument": {
11       "$ref": "#/definitions/MyFirstProfile_MarketDocument"
12     }
13   },
14   "definitions": {
15     "MFP-assembly-model": {
16       "$ref": "#"
17     },
18     "MyFirstProfile_MarketDocument": {
19       "id": "#MyFirstProfile_MarketDocument",
20       "title": "MyFirstProfile_MarketDocument",
21       "description": "The first profile document created by the MFP-assembly-model profile"
```


Modifying your profile

Introduction

Once you have created your initial profile, you will need at some stage to make changes. You can revisit the tool constructs you used to create it, to make changes afterwards.

- Use the EA delete feature to delete class
- Use the drag and drop CimContextor functionality to add class
- Use the 'Edit an Is Based on' construct to add or remove classes attributes.
- Use the 'Edit Hierarchical connectors' to edit, remove or add associations.

A good practice is to do save the diagram you have modify

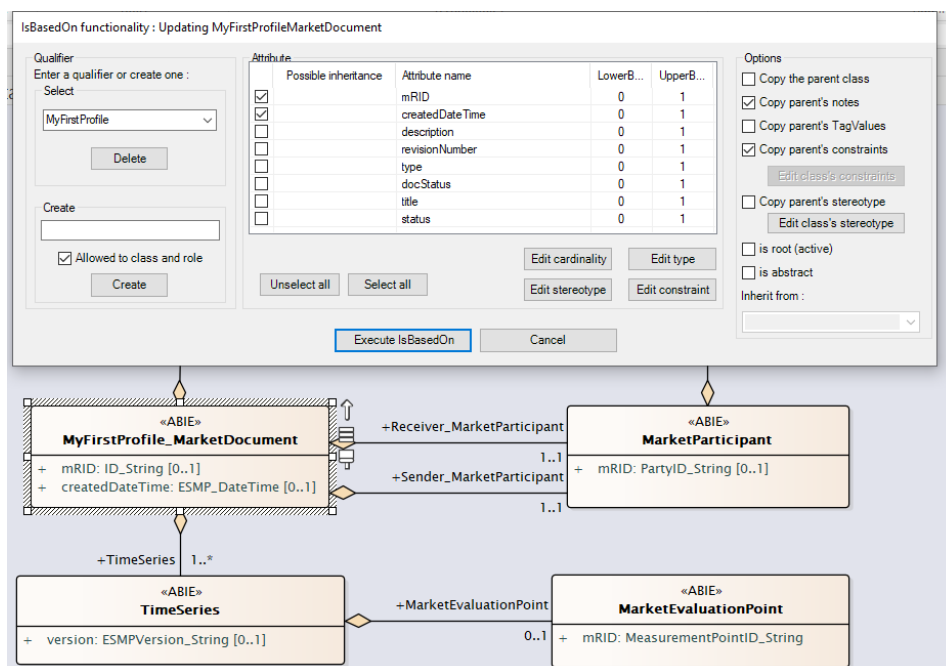
Remember that after making changes this way, you will always have to go to the process of Attribute Ordering and Property Grouping before generating your messages

Adding and Removing attributes from the classes in the messages

Open the Contextual Model diagram and select the class of which you want to add an attribute. In this example we will add the revisionNumber attribute of the MarketDocument class to our message.

Select the MyFirstProfile.MarketDocument class, choose from the menu Specialize/CimConteXtor/Edit an IsBasedOn to popup the 'Updating MyFirstProfile MarketDocument' dialog box.

Select the attribute you want to add (or remove for that part) and click on 'Execute IsBasedOn'.

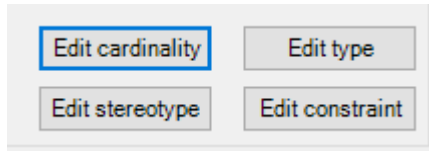


Continu with Attribute Ordering and Property Grouping before generating your messages.

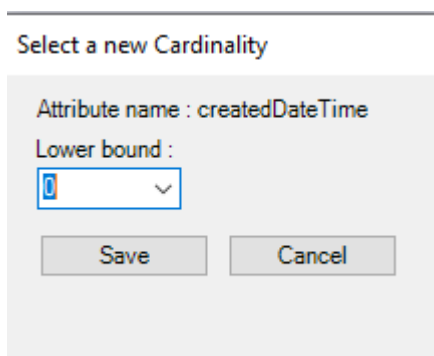
Making optional attributes mandatory

Follow the steps in last paragraph until you reach the popup 'Updating MyFirstProfile MarketDocument' dialog box.

Click on 'Edit Cardinality' and make the appropriate changes, changing Lower bound from 0 to 1.



Theoretically, you can change a mandatory to an optional attribute, but is not recommended.



Click on 'Save' in this dialog and click on 'Execute IsBasedOn' in the previous screen.

Continu with Attribute Ordering and Property Grouping before generating your messages.

Changing namespace, codelist directory, file export locations, etc.

[to be added]

Appendix 1: Overview of user interface functions for different contexts (CIMConteXtor)

	IEC/TC57/WG16 ESMP	IEC/TC57/WG13 (CGMES)	IEC/TC57/WG14
CIMConteXtor			
CreateProfilePackage	X	X	X
Edit an IsBasedOn	X	X	X
Edit Connectors for WG13 (CGMES)		X	
Edit Graph Connectors (All inheritance)			X
Edit Hierarchical Connectors	X		X
PropertyGrouping	X		
Attribute Order	X		
Utilities			
IntegrityCheck	X		X
ESMPIntegrityCheck	X		
CGMESIntegrityCheck		X	
ReplaceSimpleFloatByFloat		x	
LocalizeDataTypes		x	
CreateGlobalProfile		x	
copyAllNotes		x	
recoverProfileDataTypes		x	
MakeMembersMandatory		x	
Options			
NavigationEnabled		X	X
RolesAutoNaming	X		X
W13AutomaticAncestorInProfile		X	
About	X	X	X

Appendix 2: Overview of user interface functions for different contexts (CIMSyntaxGen)

	IEC/TC57/WG16 ESMP	IEC/TC57/WG13 (CGMES)	IEC/TC57/WG14
CIMSyntaxGen			
RDF			
EC 61970-501-2006		X	
IEC 61970-501-2006 augmented (2019)		X	
IEC 61970-501-2006 augmented (2020)		X	
IEC 61970-501-Ed2		X	
XSD			
XSD WG19			X
XSD WG19 : by Ref		X	
XSD WG16	X		
XsdToProf	X		X
Html documentation			
Html documentation		X	X
ESMPHtmlDocumentation	X		
Contextual and Assembly	X		
Creation 351	X		
FullPackage	X		
ENTSO-E Documentation	X		
HtmlENTSO-E Documentation		X	
ManagedCodeLists			
ImportCodeLists	X		
ExportCodeLists	X		
Json			
Json WG19			X
Json WG16	X		

LastExportWg16ToAvro	X		
Options	X	X	X
CodeComponent	X		
About	X	X	X