VOM Workarounds

# Introduction

These are things you have to do in VOM.

* Some need to be done after you have done a VOM import from an OWL file, because the importer misses something
* Some need to be done after you have edited or created certain things in VOM, because the Cameo tool puts there where VOM would rather they are not put. For these ones, if you’ve imported directly from an OWL text file you won’t see these issues.

# 1. Things to do After an Import from OWL (text) to VOM

These are:

* Add XML Base
* Name the IRI Literal

## Add XML Base

Reason: The VOM importer ignores the XMLBase declaration in the OWL text file if it’s there. So you have to add it.

### Procedure:

(use the imports diagram)

* Open up the UML Folder properties dialog for the ontology you are in
* Go to the “Default Namespace”
* Copy that
* Paste it into “XMLBase”

## Name the IRI Literal

Reason: This does not affect FIBO spec production but having this done makes it easier for other people to work with this ontology in VOM. What it does is make the name of the IRI Literal the same as the URI itself.

### Procedure:

(use the imports diagram)

* Open up the UML Folder properties dialog for the ontology you are in (or continue from above)
* Go to Relations
* Find the relationship to the IRI Literal
* Right click -> Select in Containment Tree
* Drag into the diagram
* Show relations from it (so others can see it’s the IRI literal for the right ontology)
* Open the properties dialog for this Literal
* Find the “Specification” field, which contains the literal value of the IRI itself
* Copy that
* Paste it into the “Name” field at the top of the dialog.

# 2. Things you have to Do after Editing in VOM

There are a few things that end up in the wrong places. These are:

* Classes, Restrictions and Properties
* Logical unions
* Annotation-related relationships
* IRI Features
* Disjoints and Inverses

## Classes, Restrictions and Properties

Places these end up include the Diagrams package(s) where you were when you did the edit, and the Proxies package for anything that’s a property of Thing or anything else that uses the local Proxy for something.

### Classes

**1. Diagrams UML Package.**

Drag them to the top of the structure for this ontology (the UML Package with the name of this ontology)

**2. Proxies: this doesn’t happen for Classes**

### Restrictions

These are also UML classes, so they behave the same (end up in Diagrams).

* Drag them to the folder called “Auxiliary Elements/Restrictions
* If they complain about extra elements, say Yes:
  + The associated relationships, onProperty and \*ValuesFrom should also go in the Restrictions folder (I presume!)

***TIP:*** *If you move the UML class for the restriction before adding the relationships, these should end up in the right place already. This probably happens with other things that end up in Diagrams as well.*

### Properties (“Relations” in VOM package naming)

These are tougher to shift. And they end up in more places. Object Properties and Datatype Properties behave the same.

They all need to end up in /Relations under the main (root) for this ontology (the one with the name of the ontology)

*General Rule: Drag relations to a level above where you think they are going to go.*

**1. Properties that ended up in Diagrams**

You need to shift these to the root.

* If there are any of these, they will be grouped under a /Relations sub-thing
* Click on that thing and drag to the Root
  + If you try and drag it to the /Relations sub-thing it won’t work

**2. Properties that ended up in Proxies**

Any property whose domain is Thing will have ended up in the Proxies folder

* If there are any of these, they will be grouped under a /Relations sub-thing
* Click on that thing and drag to the Root

## Logical Unions

These are similar to Restrictions in that they are blank nodes. There is not a dedicated folder for them so I assume they go under “Blank Nodes”.

Unions involve a union UML class and 2 or more <unionOf>> relations to classes

**Union Class:** modeled as UML classes, so they end up in the diagrams folder.

* Move the Union class to “Blank Nodes”
* IF it asks about moving the relationships, say Yes, and will end up in the Blank Nodes folder
  + I don’t know if this is where they are intended to go.

## Annotation-related relationships

There are 2 of these, for any annotation for any class or property:

1. predicate
2. fact

This is in addition to a Literal (see below)

**1. Predicate**

* Move these from Relations, to Auxiliary Elements/Facts-Individual
  + They should end up in the Relations sub-thingy of this.

**2. fact**

* Move these from Relations, to Auxiliary Elements/Annotations
  + They should end up in the Relations sub-thingy of this.

***TIP:*** *Move the predicates before moving the facts. If you move the facts first, the predicates will go wherever they went.*

### Annotation Literals

Where annotations have been done within VOM, the literals may also have ended up misplaced. As with Classes and Properties, these will have ended up in the Diagrams folder

* Move them to the folder Auxiliary Elements/Literals

## IRI Features

When you add an IRI to a class or property, there are two features of this:

* The IRI Literal
* The dependency relationship <<identifier>> between the class or property, and that literal.

Move these as follows:

* The IRI literal will have ended up in the same folder as the diagram you were working on. Move the literal to the folder Auxiliary Elements/URIs (NOT “Literals”)
* The relationship called “identifies” will have ended up in a new Relations thingy in the Diagrams folder. Move the ISI identifier relationship(s) to the Auxiliary Elements URIs folder as above (drag and let go of the relationship(s) at the level of “URIs” itself; it will create or add to the Relations sub-thingy under that).

## Disjoints and Inverses

### Disjoints

These should also not be in Root/Relations

* Move these to Auxiliary elements/Facts-Model

### Inverses

These behave much like Disjoints, in that they end up in the main Relations place.

* Move these to Auxiliary elements / Facts-Model
  + Have confirmed this is where they should go, by inspection of Relations.rdf VOM

# Oddities

An oddity:

If you create annotations in the OWL text file and import them, the “predicate” for the annotation points to the association element of the annotation property. Yet in editing these in VOM we have always made the Predicate point to the class element for the Annotation Property.

Hopefully this makes no difference to anything?