How to deal with association classes in the GML encoding

1. GML encoding of association classes according to ISO 19136-2

ISO 19136-2 defines an encoding rule for association classes. The encoding rule comprises several steps for deriving a GML encoding from association classes which will be explained in the following based on the UML model provided in Figure 1.

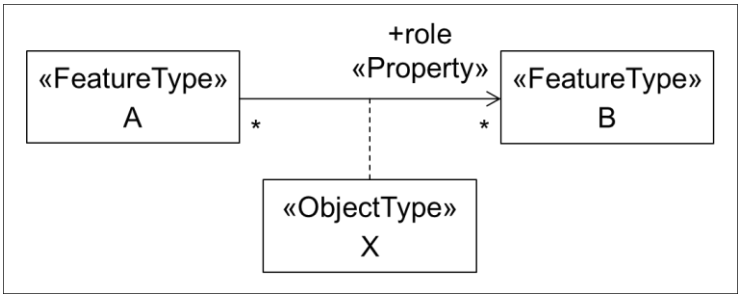

Figure 1: Association class

The UML model in Figure 1 defines association class X with an association between source class A and target class B. The following steps are defined by the encoding rule, resulting in the UML model shown in Figure 2:

- Association class X is converted into intermediate class X. The new intermediate class has the same name, stereotype, tagged values, constraints, attributes, and relationships as the original association class.
- The association between source class A and target class B is replaced by two associations, association 1 between the classes A and X, and association 2 between the classes X and B.
- The association ends at class X of association 1 and at class B of association 2 receive the role name, navigability, stereotype, and tagged values of the association end at the original target class B. In addition, the association end at class X receives the multiplicity of the association end at the original target class B. The association end at class B receives multiplicity 1.
- The association ends at class A of association 1 and at class X of association 2 receive the role name, navigability, stereotype, and tagged values of the association end at the original source class A. In addition, the association end at class X receives the multiplicity of the association end at the original source class A. The association end at class A receives multiplicity 1.

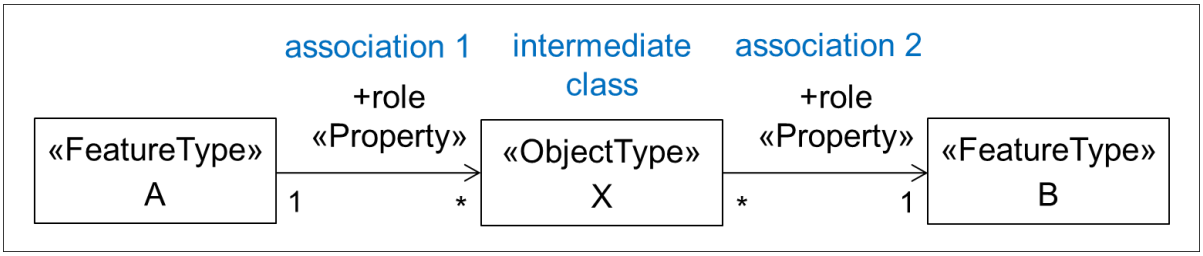

Figure 2: UML model resulting from applying the encoding rule.

The encoding rule can be applied to association classes that are defined with unidirectional, bidirectional or unspecified navigability. Correspondingly, the UML model that results from the encoding rule will also have the associations defined unidirectional, bidirectional or unspecified. The examples here focus on unidirectional associations, since all associations in the CityGML 3.0 Conceptual Model are defined with unidirectional navigability.

2. Encoding of the tagged value “inlineOrByReference” within association classes

The tagged value “inlineOrByReference” from ISO 19136-1 is commonly used for associations to define how a feature (referenced feature) that is referenced by another feature (referencing feature) is to be represented in GML instance documents. Three different values are defined for this tagged value:

- inline: the referenced feature is embedded inside the referencing feature
- byReference: the referenced feature is provided elsewhere in the same or an external GML instance document and is referenced from the referencing feature using XLink
- inlineOrByReference: both representations, i.e. inline and byReference, are possible and, in addition, a mixture of both representations.

When making use of this tagged value in association classes, the encoding rule described above will add this tagged value to the corresponding association ends of association 1 and 2 after having created the intermediate class. Figure 3 and 4 illustrate this. Figure 3 assumes that for the association end at class B the tagged value “inlineOrByReference” is set to the value “inline”. After applying the encoding rule, both the association ends at class X and at class B, will exhibit the value “inline” as is shown in Figure 4. Similarly, when the association in Figure 3 will have the value “byReference” or “inlineOrByReference”, both associations in Figure 4 will exhibit the value “byReference” or “inlineOrByReference”, respectively.

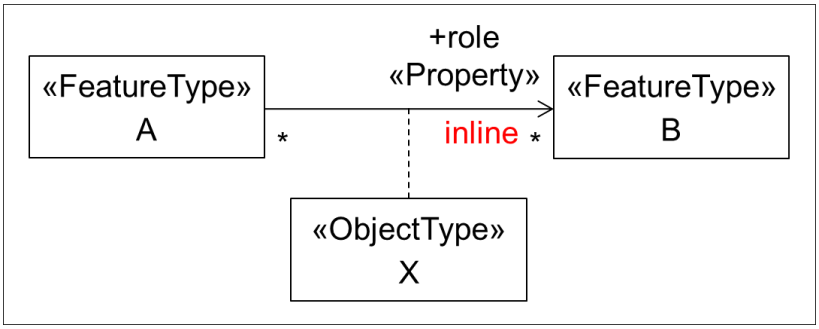

Figure 3: Association class with the tagged value “inlineOrByReference” set to “inline”

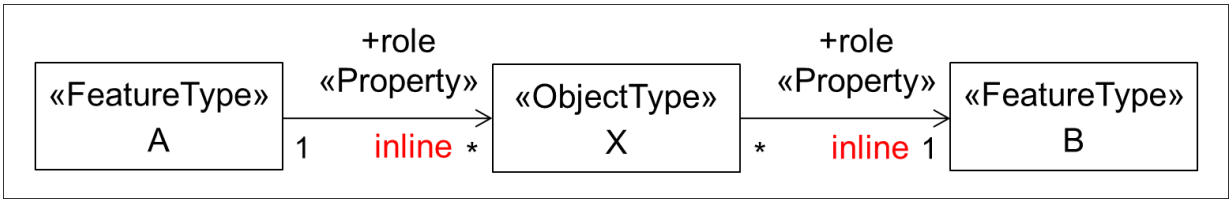

Figure 4: “inlineOrByReference” settings after applying the encoding rule

Using this tagged value means that three different encodings can be obtained depending on which of the three values is set. These different encodings are illustrated in the following by

three different GML instance documents. The source class A, the target class B, and the intermediate class X are represented by corresponding XML elements <A>, <B>, and <X>. The associations between the classes A and X and between X and B are both represented by the property element <role>. For illustration purposes, the GML instance documents do not contain root elements and namespaces.

The first GML instance document (Figure 5) is obtained when setting the tagged value to “inline”. Here, element X needs to be provided inline element A and element B inline element X.

```
<A gml:id="f1">
  <role>
    <X gml:id="f3">
      <role>
        <B gml:id="f2">
        </B>
      </role>
    </X>
  </role>
</A>
```

Figure 5: GML instance document for the value “inline”

The second GML instance document (Figure 6) results from setting the tagged value to “byReference”. Here, element A references element X and element X references element B using XLink.

```
<A gml:id="f1">
  <role xlink:href="#f3"/>
</A>
<X gml:id="f3">
  <role xlink:href="#f2"/>
</X>
<B gml:id="f2">
</B>
```

Figure 6: GML instance document for the value “byReference”

The third GML instance document (Figure 7) is obtained when the tagged value is set to “inlineOrByReference”. Here, the inline and byReference representations are combined, i.e. element X is provided inline element A and element B is referenced by element X using XLink. Alternatively, it is also possible that element X is referenced by element A and element B is provided inline element X. In addition, also the GML instances as shown above for “inline” and “byReference” can be represented with the value “inlineOrByReference”.

```
<A gml:id="f1">
  <role>
    <X gml:id="f3">
      <role xlink:href="f2"/>
    </X>
  </role>
</A>
<B gml:id="f2">
</B>
```

Figure 7: GML instance document for the value “inlineOrByReference”

3. Solutions for restricting the combination of inline and byReference representations in the GML encoding

As described in the previous section, four different instance representations are possible in the case of the “inlineOrByReference” value. This behaviour is not desired in the GML encoding of the CityGML 3.0 Conceptual Model, as it allows for too many possibilities of how to reference features and, thus, needs to be restricted.

In the CityGML 3.0 Conceptual Model, this setting affects two association classes, *CityObjectRelation* in the Core module and *Role* in the CityObjectGroup module[1]. For both, the only desired way of representing them in GML instance documents is the structure shown in Figure 7.

This structure can be specified in the UML model as shown in Figure 8. After converting the association class into an intermediate class, the tagged value of association 1 receives the value “inline” and the tagged value of association 2 receives the value “byReference”.

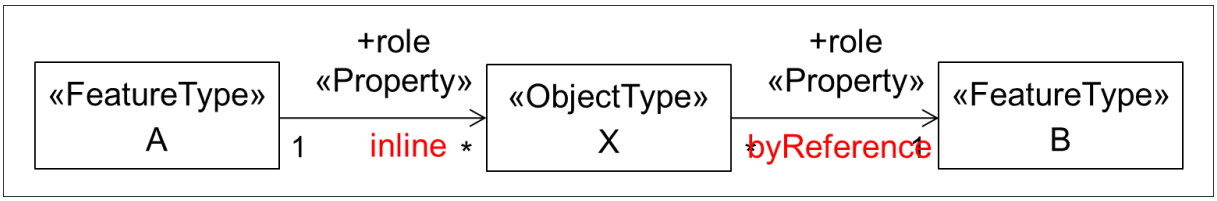

Figure 8: Desired settings for the value “inlineOrByReference” after applying the encoding rule

The following two solutions for obtaining the restricted structure in GML instance documents are introduced:

1) A new value “inlineThenByReference” is defined. This value informs the encoding rule for association classes that during the encoding all association classes to which the new value is applied are to be converted as shown in Figure 8.

2) A CityGML 3.0 Implementation Model is manually created from the CityGML 3.0 Conceptual Model in which the association classes are represented as shown in Figure 8. With this solution, the encoding rule for association classes does not need to be applied, the encoding is simply performed on the Implementation Model.

[1] The CityGML 3.0 Conceptual Model also defines the association class *TextureAssociation* in the Appearance module. This association class, however, is not affected here, because it makes use of the value “inline” for which the encoding is correct.

Both solutions guarantee for representing references between features according to Figure 7 in GML instance documents. The first solution, however, requires that the semantics of the new value is implemented in ShapeChange or any other conversion tool used for the encoding. The second solution can be applied directly without any changes to the conversion tools. For this reason, the second solution is used for the GML encoding of the CityGML 3.0 Conceptual Model as a simple and fast workaround.

In the following, the solutions will be exemplified based on the association class CityObjectRelation which is shown in Figure 9. This association class can be used to specify relationships between different features, e.g. it can be expressed that the WallSurface of one building shares the Polygon geometry with the WallSurface of a second Building.

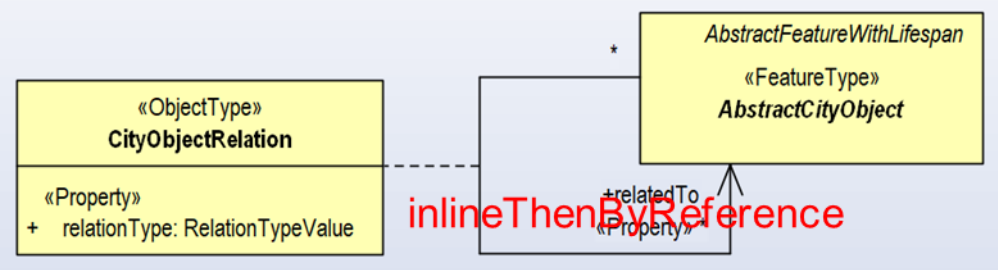

Figure 9: Association class CityObjectRelation

By setting the tagged value of the association to “inlineThenByReference”, the encoding rule will convert the UML model into the representation illustrated in Figure 10. The same representation is obtained by manually converting the association class in the Implementation Model.

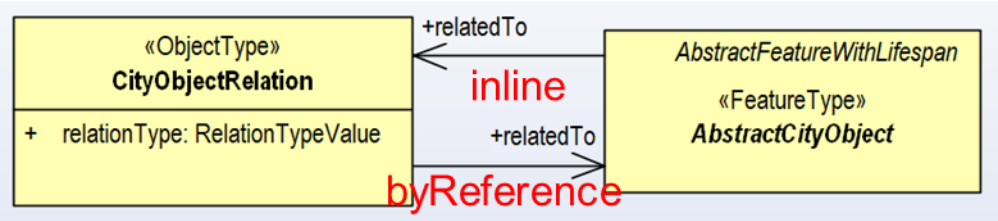

Figure 10: Association class CityObjectRelation represented as intermediate class

As shown in Figure 11, a GML instance document would then specify in the following way that the WallSurfaces of two buildings share the same geometry. The source and target classes are represented by two XML elements <WallSurface> and the intermediate class by the element <CityObjectRelation>. The element WallSurface of building 1 provides the element CityObjectRelation inline, whereas the element CityObjectRelation references the WallSurface element of building 2 using XLink. In the same way, building 2 provides the element CityObjectRelation inline its WallSurface, and the CityObjectRelation references the WallSurface of building 1 using Xlink.

```
<bldg:Building gml:id="bldg_1">
  <boundary>
    <con:WallSurface gml:id="bldg_1_ws_2">
      <relatedTo>
        <CityObjectRelation>
          <relationType>shared</relationType>
          <relatedTo xlink:href="#bldg_2_ws_4"/>
        </CityObjectRelation>
      </relatedTo>
      <lod2MultiSurface> ... </lod2MultiSurface>
    </con:WallSurface>
  </boundary>
</bldg:Building>
<bldg:Building gml:id="bldg_2">
  <boundary>
    <con:WallSurface gml:id="bldg_2_ws_4">
      <relatedTo>
        <CityObjectRelation>
          <relationType>equal</relationType>
          <relatedTo xlink:href="#bldg_1_ws_2"/>
        </CityObjectRelation>
      </relatedTo>
      <lod2MultiSurface> ... </lod2MultiSurface>
    </con:WallSurface>
  </boundary>
</bldg:Building>
```

Figure 11: GML instance document for the value “inlineThenByReference”