Review of Technological Paths of Application Schema Transformation from UML to GML

Agnieszka CHOJKA and Joanna KUCZYŃSKA, Poland

Key words: UML, GML, application schema, data modeling

SUMMARY

The capability to communicate and transfer data, called interoperability, is needed to ensure successful spatial data exchange between two or more different information systems. One of the most important aspects of achieving interoperability is specifying system independent data structure. The formal description of this data structure and content is provided by an application schema, which is also suitable for transport and storage. According to the ISO 19100 series of geographic information standards, UML (Unified Modeling Language) and GML (Geography Markup Language) are formal languages recommended to describe an application schema. UML is an object-oriented language used among others to spatial data modeling, in turn GML is the XML grammar to express geographical features including features, coordinate reference systems, geometry, topology, time, units of measure and generalized values. GML provides the ability to integrate all forms of geographic information. The transformation from UML application schema according to ISO 19109 to the corresponding GML application schema is based on a set of encoding rules specified in ISO 19136.

The paper presents a variety of possibilities concerning mapping from UML to GML, from manual to automatic ones. Using the simple case study, these solutions will be reviewed, compared and rated. This compilation could be helpful in choosing the optimal way of application schemas transformation.

STRESZCZENIE

Schemat aplikacyjny, zawierający formalny opis struktury danych, stanowi podstawę pomyślnej wymiany danych pomiędzy różnymi systemami informacyjnymi. Językami formalnymi, rekomendowanymi przez normy ISO serii 19100 dotyczące informacji geograficznej, umożliwiającymi taki opis są UML (ang. Unified Modeling Language) oraz GML (ang. Geography Markup Language). UML jest zorientowanym obiektowo językiem służącym głównie do modelowania informacji geograficznej, natomiast GML jest opartym na XML-u językiem opisu danych przestrzennych oraz formatem wymiany tych danych. Zasady mapowania schematów UML na GML zawiera norma ISO 19136.

W pracy przedstawiono różnorodne sposoby przekształcania schematów aplikacyjnych UML w schematy aplikacyjne GML. Wykorzystując przykład dokonano szeregu transformacji, od metody manualnej, poprzez wspomaganą, do automatycznej. Wyniki opisano, porównano i oceniono.

Review of Technological Paths of Application Schema Transformation from UML to GML

Agnieszka CHOJKA and Joanna KUCZYŃSKA, Poland

1. INTRODUCTION

The capability to communicate and transfer data, called interoperability, is needed to ensure successful spatial data exchange between two or more different information systems. One of the most important aspects of achieving interoperability is specifying system independent data structure. The formal description of this data structure and content is provided by an application schema, which is also suitable for transport and storage. An application schema contains the descriptions of both geographic data and other related data. According to the ISO 19100 series of geographic information standards, UML (Unified Modeling Language) and GML (Geography Markup Language) are formal languages recommended to describe an application schema.
UML is an object-oriented language, defined by Object Management Group OMG (not-for-profit computer industry specifications consortium, http://www.uml.org/), used among others to spatial data modeling. GML is the XML grammar to express geographical features including features, coordinate reference systems, geometry, topology, time, units of measure and generalized values. GML provides the ability to integrate all forms of geographic information. It is defined by Open Geospatial Consortium, the international industry consortium of 416 companies, government agencies and universities participating in a consensus process to develop publicly available interface standards (http://www.opengeospatial.org/).
On the basis of OGC standards, a lot of ISO standards was prepared. UML application schema pose the platform – independent semantic description of data structure, in turn GML application schema is an implementation specification for a various technologies, for example it can be a transfer data schema (Pachelski W., Parzyński Z., Zwirowicz A., 2007; see Figure 1). In addition to being a mark-up language that describes objects in the world around us, GML also can be used to transport descriptions over the Internet (Lake R., 2004).

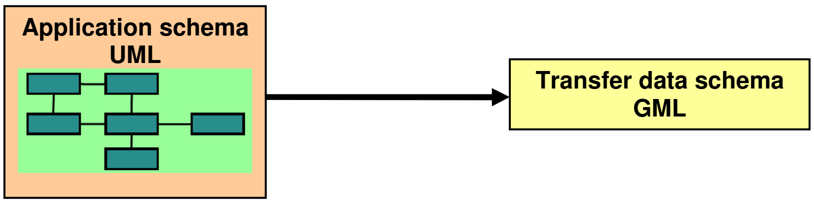


Figure 1. Implementation of UML and GML application schema

2. REVIEW OF UML TO GML MAPPING METHODS AND TOOLS

2.1 Manual

The transformation from UML application schema, according to ISO 19109, to the corresponding GML application schema is based on a set of encoding rules specified in ISO 19136. These rules are given in Annex E (ISO 19136:2007) and based on the general idea that the class definitions in the UML application schema are mapped to type and element declarations in GML Schema according to the following relation:

– Package → One XML Schema document per package (default mapping),
– <<Application Schema>> → XML Schema document,
– <<DataType>> → Global element, whose content model is a globally scoped XML Schema complexType, property type,
– <<Enumeration>> → Restriction of xsd:string with enumeration values,
– <<CodeList>> → Union of an enumeration and a pattern (default mapping, an alternative mapping is a reference to a dictionary),
– <<Union>> → Choice group whose members are GML objects or features, or objects corresponding to DataTypes,
– <<FeatureType>> → Global element, whose content model is a globally scoped XML Schema type derived by direct/indirect extension of gml:AbstractFeatureType, property type,
– No stereotype or <<Type>> → Global element, whose content model is a globally scoped XML Schema type derived by direct/indirect extension of gml:AbstractGMLType, property type,
– Operations → Not encoded,
– Attribute → Local xsd:element, the type is either a property type (if the type is a complex type) or a simple type,
– Association role → Local xsd:element, the type is always a property type (only named and navigable roles),
– General OCL constraints → Not encoded.

2.2 Tools with XML Schema support

There are many tools, leveraging XML technology, which can support a creation of GML application schema. It is possible to use many XML editors, such as oXygen (http://www.oxygenxml.com), EditiX (http://www.editix.com) or XMLmind (http://www.xmlmind.com). One of the most popular is Altova XMLSpy (http://www.altova.com). This is the advanced XML editor for modeling, editing, transforming, and debugging XML-related technologies.
To develop the GML application schema with the use of XMLSpy, the first step is to create a new document with no schema content and enter the target namespaces with prefixes: XML Schema namespace (http://w3.org/2001/XMLSchema), GML namespace (http://www.opengis.net/gml) and, optionally, a target namespace for the XML document instance. The next step is to import the gml core schemas (gml.xsd). Then, it is possible to create schema through creating features, adding feature properties, extending a simple type,

creating feature relationship and association etc. During work, we can choose between graphical XML editing and text-based XML editing views. To validate XML documents XMLSpy's built-in validator can be used. It provides two evaluations of the XML document, both a well-formedness check and a validation check.

2.3 GML dedicated tools

2.3.1 ShapeChange

ShapeChange is a software that can generate a valid GML application schema (GML 3.2.1), when an UML model follows a well-defined set of guidelines that were described in a "UGAS Guidelines and Encoding Rules" document (Portele C., 2008a). This tool accepts UML models as input in XMI 1.0 format and it also recognizes and generates diagnostic error messages for incorrectly constructed input. ShapeChange is a Java application, that can be run from the command line. It accepts a number of parameters that influence the encoding rules from the UML metamodel to the GML application schema documents.
ShapeChange is an encoding service in the sense of ISO 19118 and implements the "generateXMLSchema" operation.
The current version of this tool is 1.0. It is based on the following documents of ISO/TC 211 and the Open Geospatial Consortium:
– GML 3.2.1 (ISO 19136:2007),
– ISO/TS 19139:2007,
– ISO 19118 rev 1 (draft),
– ISO/TS 19103:2005 and ISO 19103 (draft),
– ISO 19109:2005 (Portele C., 2008b).

2.3.2 Hollow World and FullMoon

Hollow World is an environment to enable specialists in a domain that utilizes geospatial information (GI) to develop an information model for their application domain, which conforms to international standards for interoperable GI. A model developed in that framework can be easily transformed into a GML-conformant XML Schema, which specifies the document format for transfer of domain data as a standard XML document, compatible with OGC-WFS.
HollowWorld provides an UML template which includes the components described in the ISO 19100 series of geographic information standards.
The ISO 19100 components are primarily from the ISO/TC 211 "Harmonized Model". These are augmented with UML representations of components provided in ISO 19136 (GML) but do not implement elements of the Harmonized Model.
Moreover, for users of Sparx Systems Enterprise Architect tool, an "UML Profile" containing the standard stereotypes and tagged values from ISO 19136 is provided.
The XML Schema implementation of the model can be generated automatically, providing enough detail in the model to:
– fully specify the information model required for the domain,
– accommodate the properties of XML Schema, and the GML-based patterns for the use of

XML to encode UML-designed models.
The UML model must follow the profile described in Annex E. of ISO 19136, in particular concerning:

– association navigability and role names,
– use of only the standard stereotypes,
– assignment of all the necessary tagged values.

The FullMoon UML processing environment can verify conformance to the ISO 19136 UML profile as a precursor to generating the implementation.
The FullMoon XML Processing Framework allows to easily transform the application schema defined in conceptual terms of a particular domain into its physical representation – a set of W3C XML Schemas. It was originally designed for processing large UML models using XML mapping rules defined in ISO 19118, 19136 and 19139 standards. FullMoon processes the XML Metadata Interchange (XMI) representation of a model, generating XML schemas and some other views, with the mapping rules maintained as separate XQuery scripts (Solid Earth and Environment GRID, 2011).

3. GENERATION OF GML APPLICATION SCHEMA

3.1 Case study in UML

To make a comparison of tools capabilities in GML application schema generating, the following UML application schema was used (Figure 2). This model presents classes concerning Record of Places, Streets and Addresses according to Polish administration documents. It consist of 6 classes: 2 with <<FeatureType>> stereotype, 2 with <<DataType>> stereotype and 2 with <<Enumeration>> stereotype. Association and aggregation roles occur between classes.

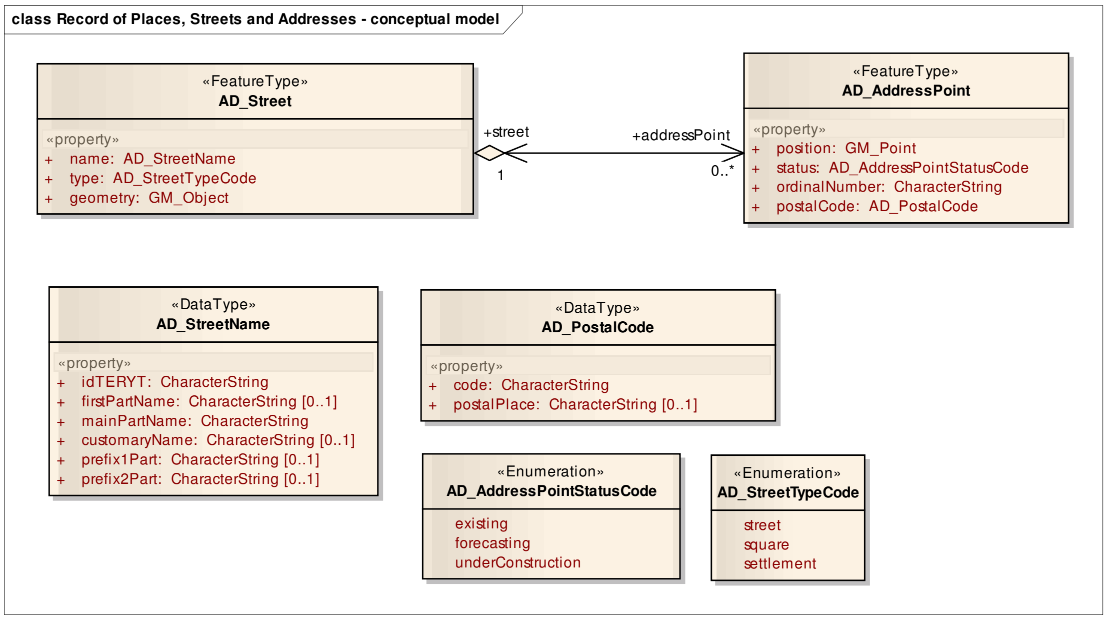
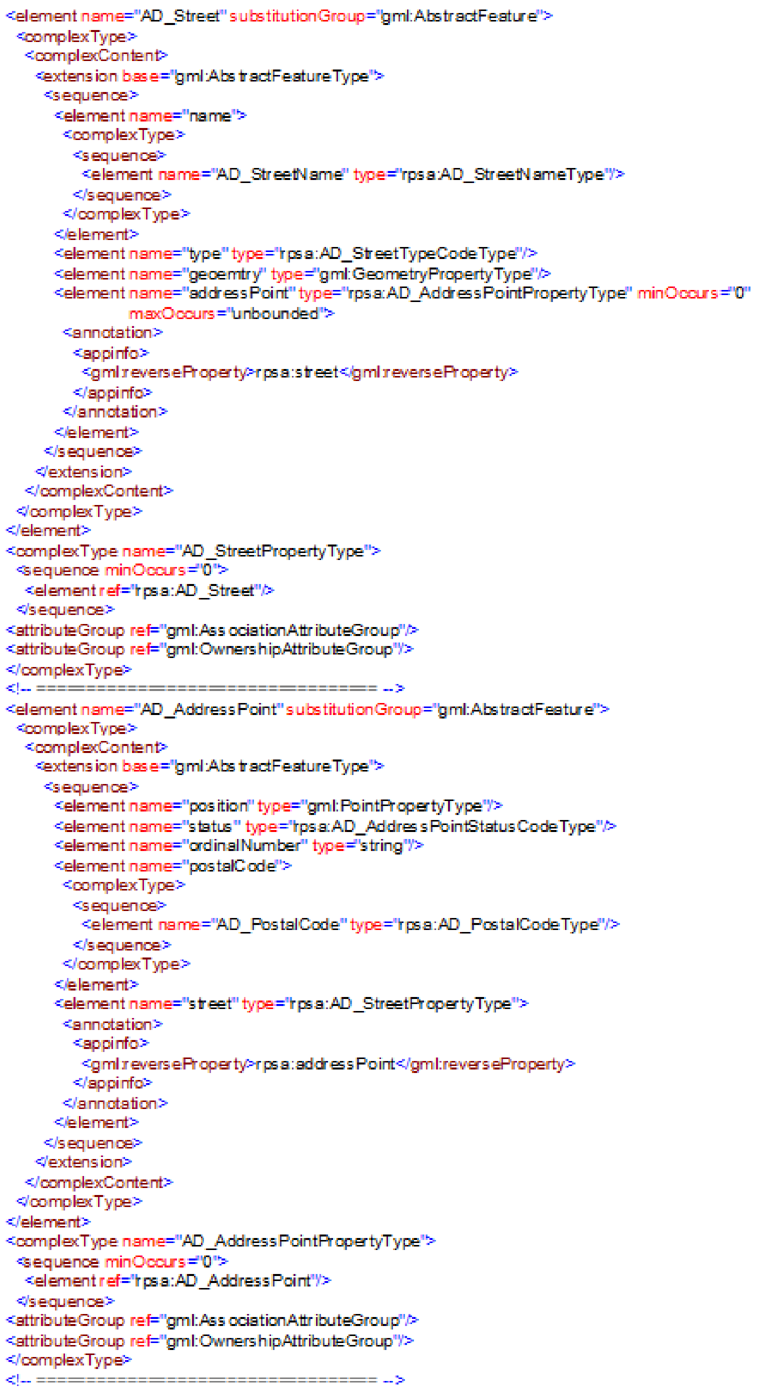

class Record of Places, Streets and Addresses - conceptual model
«FeatureType»
AD_Street
«property»
+ name: AD_StreetName
+ type: AD_StreetTypeCode
+ geometry: GM_Object
+street
1
+addressPoint
0..*
«FeatureType»
AD_AddressPoint
«property»
+ position: GM_Point
+ status: AD_AddressPointStatusCode
+ ordinalNumber: CharacterString
+ postalCode: AD_PostalCode
«DataType»
AD_StreetName
«property»
+ idTERYT: CharacterString
+ firstPartName: CharacterString [0..1]
+ mainPartName: CharacterString
+ customaryName: CharacterString [0..1]
+ prefix1Part: CharacterString [0..1]
+ prefix2Part: CharacterString [0..1]
«DataType»
AD_PostalCode
«property»
+ code: CharacterString
+ postalPlace: CharacterString [0..1]
«Enumeration»
AD_AddressPointStatusCode
existing
forecasting
underConstruction
«Enumeration»
AD_StreetTypeCode
street
square
settlement

Figure 2. UML application schema

3.2 Manual method

Underneath graphic (Figure 3) presents the part of GML application schema (for the above UML application schema) that was created manually using the UML-to-GML application schema encoding rules (Annex E, ISO 19136:2007) and the simple text editor.

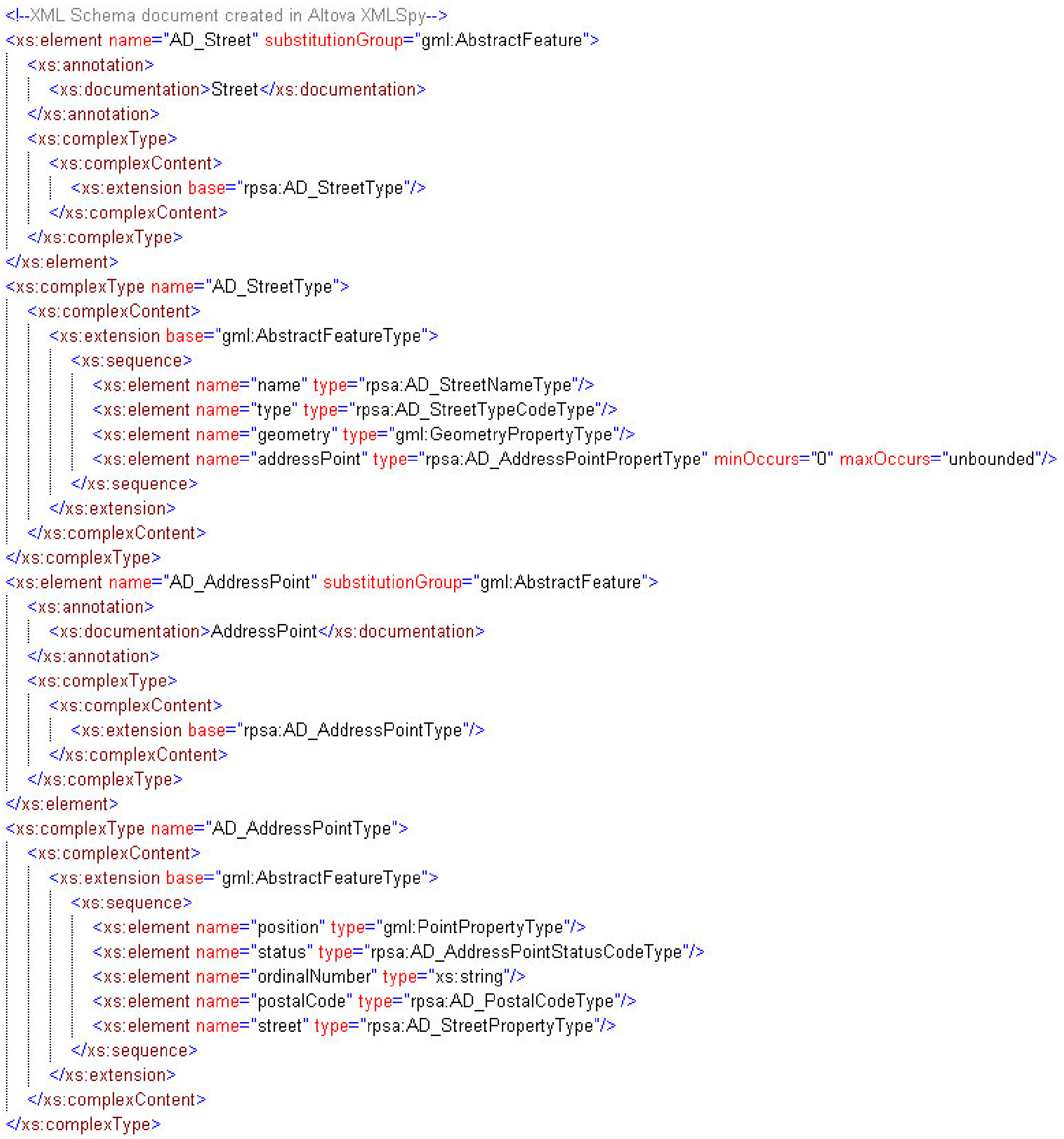


```
<element name="AD_Street" substitutionGroup="gml:AbstractFeature">
 <complexType>
  <complexContent>
   <extension base="gml:AbstractFeatureType">
    <sequence>
     <element name="name">
      <complexType>
       <sequence>
        <element name="AD_StreetName" type="rpsa:AD_StreetNameType"/>
       </sequence>
      </complexType>
     </element>
     <element name="type" type="rpsa:AD_StreetTypeCodeType"/>
     <element name="geoemtry" type="gml:GeometryPropertyType"/>
     <element name="addressPoint" type="rpsa:AD_AddressPointPropertyType" minOccurs="0"
             maxOccurs="unbounded">
      <annotation>
       <appinfo>
        <gml:reverseProperty>rpsa:street</gml:reverseProperty>
       </appinfo>
      </annotation>
     </element>
    </sequence>
   </extension>
  </complexContent>
 </complexType>
</element>
<complexType name="AD_StreetPropertyType">
 <sequence minOccurs="0">
  <element ref="rpsa:AD_Street"/>
 </sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
<attributeGroup ref="gml:OwnershipAttributeGroup"/>
</complexType>
<!-- ================================ -->
<element name="AD_AddressPoint" substitutionGroup="gml:AbstractFeature">
 <complexType>
  <complexContent>
   <extension base="gml:AbstractFeatureType">
    <sequence>
     <element name="position" type="gml:PointPropertyType"/>
     <element name="status" type="rpsa:AD_AddressPointStatusCodeType"/>
     <element name="ordinalNumber" type="string"/>
     <element name="postalCode">
      <complexType>
       <sequence>
        <element name="AD_PostalCode" type="rpsa:AD_PostalCodeType"/>
       </sequence>
      </complexType>
     </element>
     <element name="street" type="rpsa:AD_StreetPropertyType">
      <annotation>
       <appinfo>
        <gml:reverseProperty>rpsa:addressPoint</gml:reverseProperty>
       </appinfo>
      </annotation>
     </element>
    </sequence>
   </extension>
  </complexContent>
 </complexType>
</element>
<complexType name="AD_AddressPointPropertyType">
 <sequence minOccurs="0">
  <element ref="rpsa:AD_AddressPoint"/>
 </sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
<attributeGroup ref="gml:OwnershipAttributeGroup"/>
</complexType>
<!-- ================================ -->
```

Figure 3. GML application schema created manually

3.3 XMLSpy support

In Figure 4 is shown the fragment of GML application schema that was created with the support of XMLSpy tool, according to the UML-to-GML application schema encoding rules (Annex E, ISO 19136:2007).



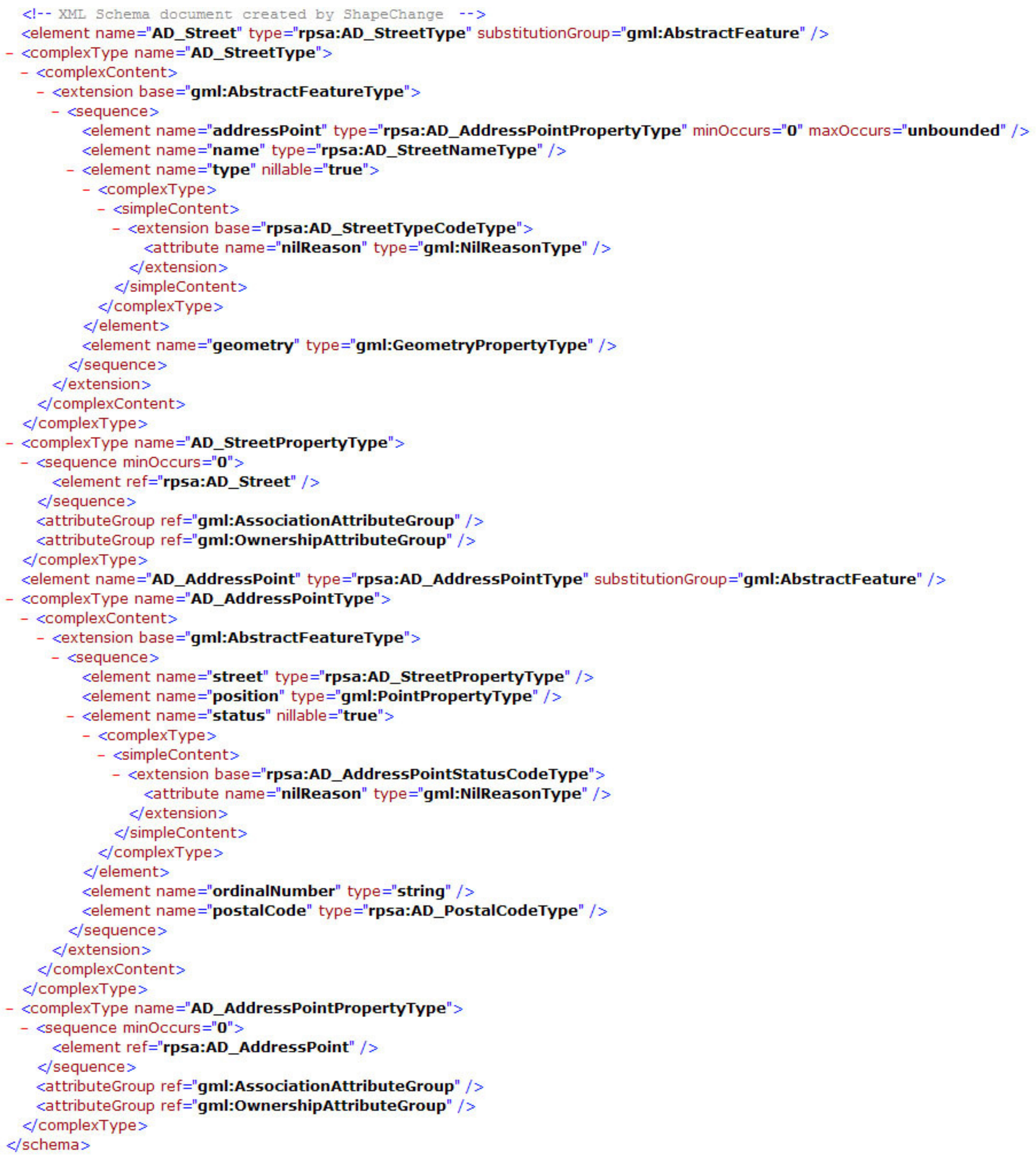
```xml
<!--XML Schema document created in Altova XMLSpy-->
<xs:element name="AD_Street" substitutionGroup="gml:AbstractFeature">
  <xs:annotation>
    <xs:documentation>Street</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="rpsa:AD_StreetType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:complexType name="AD_StreetType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractFeatureType">
      <xs:sequence>
        <xs:element name="name" type="rpsa:AD_StreetNameType"/>
        <xs:element name="type" type="rpsa:AD_StreetTypeCodeType"/>
        <xs:element name="geometry" type="gml:GeometryPropertyType"/>
        <xs:element name="addressPoint" type="rpsa:AD_AddressPointPropertType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:element name="AD_AddressPoint" substitutionGroup="gml:AbstractFeature">
  <xs:annotation>
    <xs:documentation>AddressPoint</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="rpsa:AD_AddressPointType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:complexType name="AD_AddressPointType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractFeatureType">
      <xs:sequence>
        <xs:element name="position" type="gml:PointPropertyType"/>
        <xs:element name="status" type="rpsa:AD_AddressPointStatusCodeType"/>
        <xs:element name="ordinalNumber" type="xs:string"/>
        <xs:element name="postalCode" type="rpsa:AD_PostalCodeType"/>
        <xs:element name="street" type="rpsa:AD_StreetPropertyType"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Figure 4. GML application schema created in Altova XMLSpy

3.4 GML dedicated tools

3.4.1 ShapeChange

Figure 5 presents the part of GML application schema generated automatically by the ShapeChange tool.

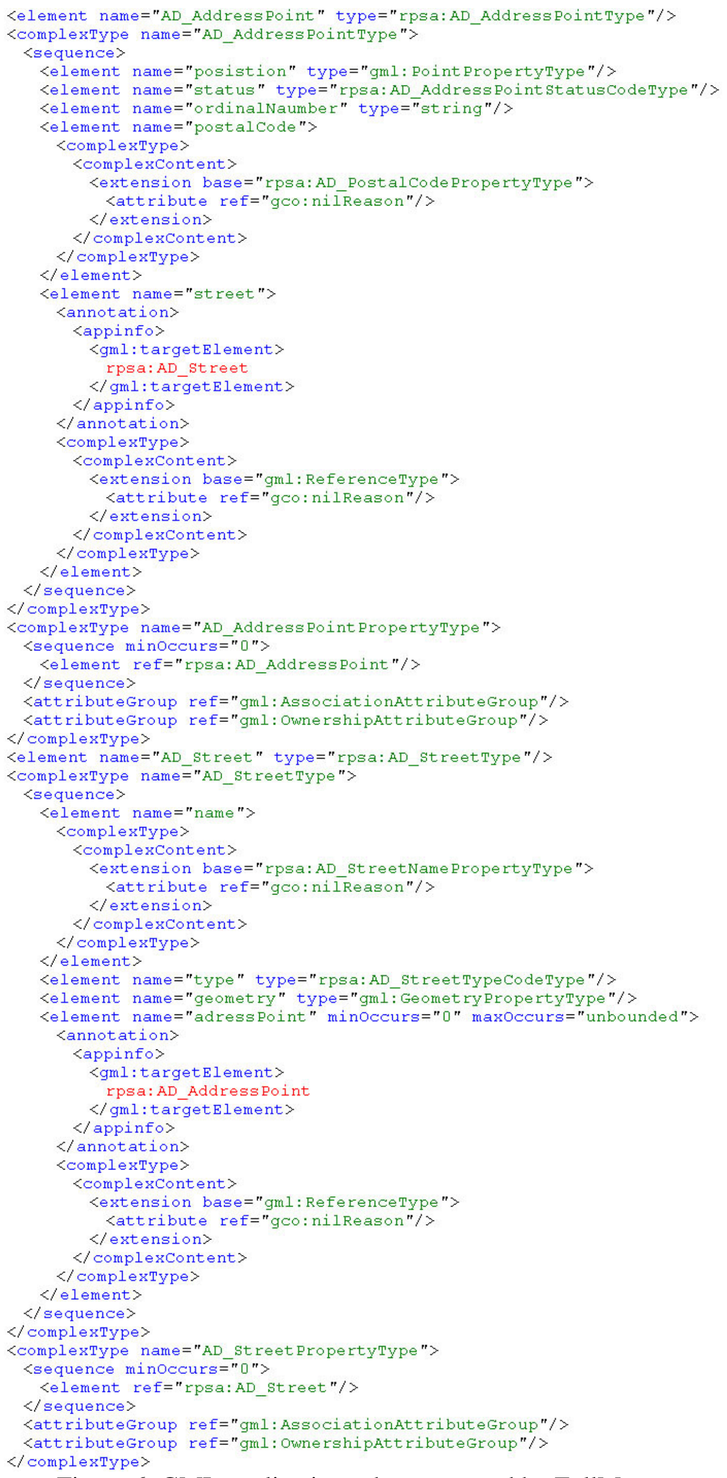


```
   <!-- XML Schema document created by ShapeChange  -->
   <element name="AD_Street" type="rpsa:AD_StreetType" substitutionGroup="gml:AbstractFeature" />
 - <complexType name="AD_StreetType">
   - <complexContent>
     - <extension base="gml:AbstractFeatureType">
       - <sequence>
           <element name="addressPoint" type="rpsa:AD_AddressPointPropertyType" minOccurs="0" maxOccurs="unbounded" />
           <element name="name" type="rpsa:AD_StreetNameType" />
         - <element name="type" nillable="true">
           - <complexType>
             - <simpleContent>
               - <extension base="rpsa:AD_StreetTypeCodeType">
                   <attribute name="nilReason" type="gml:NilReasonType" />
                 </extension>
               </simpleContent>
             </complexType>
           </element>
           <element name="geometry" type="gml:GeometryPropertyType" />
         </sequence>
       </extension>
     </complexContent>
   </complexType>
 - <complexType name="AD_StreetPropertyType">
   - <sequence minOccurs="0">
       <element ref="rpsa:AD_Street" />
     </sequence>
     <attributeGroup ref="gml:AssociationAttributeGroup" />
     <attributeGroup ref="gml:OwnershipAttributeGroup" />
   </complexType>
   <element name="AD_AddressPoint" type="rpsa:AD_AddressPointType" substitutionGroup="gml:AbstractFeature" />
 - <complexType name="AD_AddressPointType">
   - <complexContent>
     - <extension base="gml:AbstractFeatureType">
       - <sequence>
           <element name="street" type="rpsa:AD_StreetPropertyType" />
           <element name="position" type="gml:PointPropertyType" />
         - <element name="status" nillable="true">
           - <complexType>
             - <simpleContent>
               - <extension base="rpsa:AD_AddressPointStatusCodeType">
                   <attribute name="nilReason" type="gml:NilReasonType" />
                 </extension>
               </simpleContent>
             </complexType>
           </element>
           <element name="ordinalNumber" type="string" />
           <element name="postalCode" type="rpsa:AD_PostalCodeType" />
         </sequence>
       </extension>
     </complexContent>
   </complexType>
 - <complexType name="AD_AddressPointPropertyType">
   - <sequence minOccurs="0">
       <element ref="rpsa:AD_AddressPoint" />
     </sequence>
     <attributeGroup ref="gml:AssociationAttributeGroup" />
     <attributeGroup ref="gml:OwnershipAttributeGroup" />
   </complexType>
 </schema>
```

Figure 5. GML application schema created by ShapeChange

3.4.2 FullMoon

In Figure 6 is shown the fragment of GML application schema received automatically from the FullMoon application.



```
<element name="AD_AddressPoint" type="rpsa:AD_AddressPointType"/>
<complexType name="AD_AddressPointType">
  <sequence>
    <element name="posistion" type="gml:PointPropertyType"/>
    <element name="status" type="rpsa:AD_AddressPointStatusCodeType"/>
    <element name="ordinalNaumber" type="string"/>
    <element name="postalCode">
      <complexType>
        <complexContent>
          <extension base="rpsa:AD_PostalCodePropertyType">
            <attribute ref="gco:nilReason"/>
          </extension>
        </complexContent>
      </complexType>
    </element>
    <element name="street">
      <annotation>
        <appinfo>
          <gml:targetElement>
            rpsa:AD_Street
          </gml:targetElement>
        </appinfo>
      </annotation>
      <complexType>
        <complexContent>
          <extension base="gml:ReferenceType">
            <attribute ref="gco:nilReason"/>
          </extension>
        </complexContent>
      </complexType>
    </element>
  </sequence>
</complexType>
<complexType name="AD_AddressPointPropertyType">
  <sequence minOccurs="0">
    <element ref="rpsa:AD_AddressPoint"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
  <attributeGroup ref="gml:OwnershipAttributeGroup"/>
</complexType>
<element name="AD_Street" type="rpsa:AD_StreetType"/>
<complexType name="AD_StreetType">
  <sequence>
    <element name="name">
      <complexType>
        <complexContent>
          <extension base="rpsa:AD_StreetNamePropertyType">
            <attribute ref="gco:nilReason"/>
          </extension>
        </complexContent>
      </complexType>
    </element>
    <element name="type" type="rpsa:AD_StreetTypeCodeType"/>
    <element name="geometry" type="gml:GeometryPropertyType"/>
    <element name="adressPoint" minOccurs="0" maxOccurs="unbounded">
      <annotation>
        <appinfo>
          <gml:targetElement>
            rpsa:AD_AddressPoint
          </gml:targetElement>
        </appinfo>
      </annotation>
      <complexType>
        <complexContent>
          <extension base="gml:ReferenceType">
            <attribute ref="gco:nilReason"/>
          </extension>
        </complexContent>
      </complexType>
    </element>
  </sequence>
</complexType>
<complexType name="AD_StreetPropertyType">
  <sequence minOccurs="0">
    <element ref="rpsa:AD_Street"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
  <attributeGroup ref="gml:OwnershipAttributeGroup"/>
</complexType>
```

Figure 6. GML application schema created by FullMoon

4. DISCUSSION AND CONCLUSION

Applied UML to GML application schema transformation methods have been rated regarding the following parameters (Table 1):

– ease to use – installation and configuration process, operation,
– quickness – time of preparing the processing environment and receiving result,
– error probability – probability of occurring errors in outcome.

In authors' opinion, the most user friendly method is GML application schema generation supported by XMLSpy software. The user can choose between a graphical and text-based view of XML Schema. Moreover this tool has built-in validator that enables controlling correctness of application schema creation. In turn the worst method appeared FullMoon for the sake of very complex way of preparation the processing environment. By the same reason ShapeChange also got a low note.
Working out the complete GML application schema was the most time consuming (very complicated installation path) in FullMoon environment. The fastest way occurred usage of XMLSpy.
The probability of errors appearance in outcome is the highest using manual method. This way of creation application schema does not enable the correctness of the result controlling. Making use of automatic tools solves this problem.
In authors' transformation methods ranking, the best score achieved XMLSpy. It should be emphasized that this is very subjective assessment and comparison.

	Manual	**XMLSpy**	**ShapeChange**	**FullMoon**
Ease to use	3	4	2	1
Quickness	3	4	2	1
Error probability	1	3	4	4
Overall	**7**	**11**	**8**	**6**

Table 1. Comparison of methods and tools: rates from 4 (the best) to 1 (the worst)

Particular of presented UML to GML application schema transformation methods allow to receive valid GML application schemas for case study UML application schema (Figure 2). However, GML application schemas differs in the convention of notation for particular elements of schema, e.g. reference to data type, description of relationship and used namespaces, that can cause some problems during interpreting schemas by different software tools.

REFERENCES

ISO 19136:2007, Geography Markup Language (GML).
Lake R., Burggraf D., Trninić M., Rae L., 2004, Geography Mark-up Language (GML), John Wiley&Sons Ltd.
Pachelski W., Parzyński Z., Zwirowicz A., 2007, Aspekty implementacyjne modeli pojęciowych informacji geograficznej, Vol. 17b, ss. 591 – 602, Archiwum Fotogrametrii, Kartografii i Teledetekcji.
Portele C., 2008a, Mapping UML to GML Application Schemas. Guidelines and Encoding Rules. Interactive Instruments GmbH, http://www.interactive-instruments.de/ugas/UGAS-

Guidelines-and-Encoding-Rules.pdf.
Portele C., 2008b, Mapping UML to GML Application Schemas. ShapeChange – Architecture and Description. Interactive Instruments GmbH, http://www.interactive-instruments.de/ugas/ShapeChange.pdf.
Solid Earth and Environment GRID, SEE GRID community website, 2011, https://www.seegrid.csiro.au/wiki/bin/view/AppSchemas/HollowWorld, https://www.seegrid.csiro.au/wiki/bin/view/Siss/FullMoon.

BIOGRAPHICAL NOTES

Agnieszka Chojka, since 2003 MSc – computer science engineer, Technical University of Lodz (Poland), Faculty of Physics, Computer Science and Mathematics, major in Computer Science with specialization in Computer Graphics. Since 2004 research worker and academic teacher at the University of Warmia and Mazury in Olsztyn (Poland), Faculty of Geodesy and Land Management, Chair of Land Surveying and Geomatics. In 2008 defended PhD thesis on the Faculty of Geodesy and Land Management, at the University of Warmia and Mazury in Olsztyn.

Joanna Kuczyńska in 2007 completed studies in the field of Geodesy and Cartography, majoring in Geodesy and The Spatial Information System on Faculty of Geodesy and Land Management of University of Warmia and Mazury in Olsztyn (Poland). Since 2008 research worker and academic teacher in Chair of Land Surveying and Geomatics at UWM in Olsztyn.

CONTACTS

PhD, Eng. Agnieszka Chojka
University of Warmia and Mazury in Olsztyn
Oczapowskiego 2
Olsztyn
POLAND
Tel. +48 89 523 48 78
Fax +48 89 523 48 78
Email: agnieszka.chojka@uwm.edu.pl

MSc, Eng. Joanna Kuczyńska
University of Warmia and Mazury in Olsztyn
Oczapowskiego 2
Olsztyn
POLAND
Tel. +48 89 523 48 78
Fax +48 89 523 48 78
Email: joanna.kuczynska@uwm.edu.pl