GML Encoding of NADM C1

By Eric Boisvert[1*], Bruce R. Johnson[2*], Simon Cox[3], and Boyan Brodaric[4]

[1]Geological Survey of Canada
490 de La Couronne
Québec, G1K 9A9 Canada
e-mail: eboisver@nrcan.gc.ca

[2]U.S. Geological Survey, 954 National Center, Reston, VA 20192, U.S.A.
[3]CSIRO, ARRC, PO Box 1130, Bentley, WA 6102, Australia
[4]Geological Survey of Canada, 615 Booth Street, Ottawa, K1A 0E9, Canada

*Members of NADM DITT (Digital Interchange Technical Team)

ABSTRACT

The North American Geologic Map Data Model Steering Committee's (NADM) Digital Interchange Technical Team (see http://nadm-geo.org) is tasked to create an interchange format compliant with the North American Geologic Map Data Model conceptual model, known as "NADM C1". XML was unanimously selected as the technology of choice, and after initial attempts, it was realised that leveraging existing work on GML (Geographic Markup Language; OpenGis, 2004) would improve the interchange format. GML is a library that provides essential GIS features that can be reused in any geospatial application, such as NADM. GML provides reusable objects and design patterns. The NADM conceptual model has been analysed to create a GML application; this paper describes that process and provides examples of NADM GML encoding for interchange of geoscience data.

INTRODUCTION

In the latest report on this technical team (Digital Interchange Technical Team, 2003), we discussed the challenges of encoding the NADM conceptual model in an XML document. The process involved converting classes from the UML diagram into meaningful XML tags. We pointed out that the principal difficulty involved creation of a consistent logical schema (in UML), such that patterns in the schema could be mapped in a regular manner onto XML document structures, easing the transition from modeling to encoding. The solution to this problem came to us in the fall of 2003 at an international meeting in Edinburgh (Laxton & Brodaric, 2003), where it was decided that GML could be used to constrain and direct the XML encoding process, and thus that GML will be used as the encoding standard for sharing geological datasets among the participants (see http://ncgmp.usgs.gov/intdb/dmic/dmic-rep1.html). GML is itself an XML encoding of ISO standards designed to represent geographical features. Of importance to this discussion is the notion that GML provides a much needed design pattern which focuses encoding choices to a manageable subset of the very large set of choices possible in XML.

WHAT IS GML?

GML is the abbreviation for *Geographic Markup Language*. It is an XML encoding of an ISO *feature model* standard that describes classes required to represent common geospatial features (see Open GIS Consortium, 2004, for complete normative references). In the standard, there are descriptions for spatial objects (points, lines, polygons …), projections, dictionaries, topology, time, etc. The state of a geographic feature is described by properties, where each property is formed by a name, a type, and a value. GML is a framework, or a library, of reusable building blocks for any group that needs to create a "*GML application*". GML by itself is not intended to be used directly, because it is domain neutral; it only describes geography, and there are only a small number of concrete feature types. To turn GML into a useful application, it must be expanded to describe the content in a specific domain, such as biology, forestry or geology. XMML is one example of such an application which extends GML to address specific issues of mining and mineral exploration (https://www.seegrid.csiro.au/twiki/bin/view/Xmml/WebHome).

WHY GML?

Several benefits can be obtained by adapting GML rather than starting with a new XML encoding. First, GML is an international standard developed by the Open GIS Consortium (OGC) that is currently being revised for pub-

lication as ISO 19136, being derived from a related set of international standards (the ISO 19100 series). It provides a formal development framework and comes with a large group of practitioners to help guide and support development. It also ensures that encodings developed in other domains can be reused by sharing common GML constructs. Some examples of reuse will be discussed later.

GML is also at the root of other standard technologies and protocols fostered by OGC. One important technology is WFS (Web Feature Service), which provides a mechanism for interaction with a geospatial database using GML. Software developed according to this standard can be reused for any GML encoding. By connecting to the GML community, we benefit from software vendors who support WFS in their server and client products. One possible scenario is to integrate information from various sources for decision making. Figure 1 shows a typical architecture of servers (on the left) translating their content into GML and serving the content through WFS to a GML-enabled client. Because the schemas are GML-based, the application can handle the common parts (the geography) without any prior knowledge of the domain.

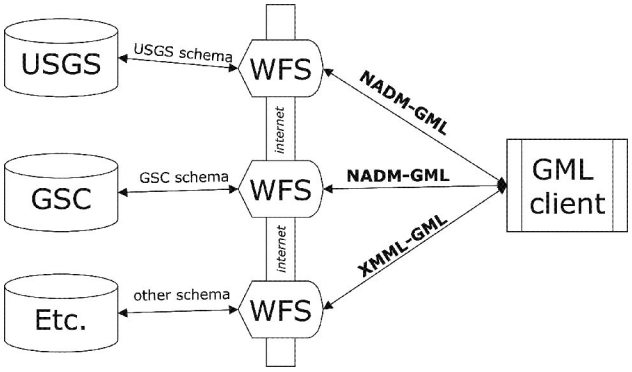

Figure 1. Example of a distributed system sharing geological information using GML and WFS (Web Feature Service).

GML also provides a 'Design pattern' that is one of the most significant additions to our design process. Although the literature does not 'sell' GML as such, the rules and constraints set by GML designers are a great help to any team working on XML schema design. GML in effect provides guidelines for consistent schema design. Without such rules and constraints XML (and UML) is almost too flexible, allowing many alternative ways to encode the same content, and requiring potentially different tools to be developed in order to read and manipulate the content. This implicit and understated aspect of GML proved to be of significant importance in our design process.

XML VS GML

It is important to realise that GML is XML. GML components are defined using XML schemas. An XML schema is a W3C standard that defines the structure of an XML document (see http://www.w3.org/XML/Schema). We extend GML by using an XML schema that allows us to re-use tools from our first attempt to encode the NADM conceptual model in pure XML. GML provides a conceptual model based on the ISO feature model. A GML application must reuse core GML features defined in the conceptual model. Figure 2 shows a UML representation of a very simple example of a *Street* feature that we might want to model. The street inherits from the abstract GML feature and reuses standard GML spatial geometries and property structures for the new street features. By inheriting the Street from a GML Feature, we turned our street into a formal GML feature (the diagram reads 'A Street is a kind of GML feature').

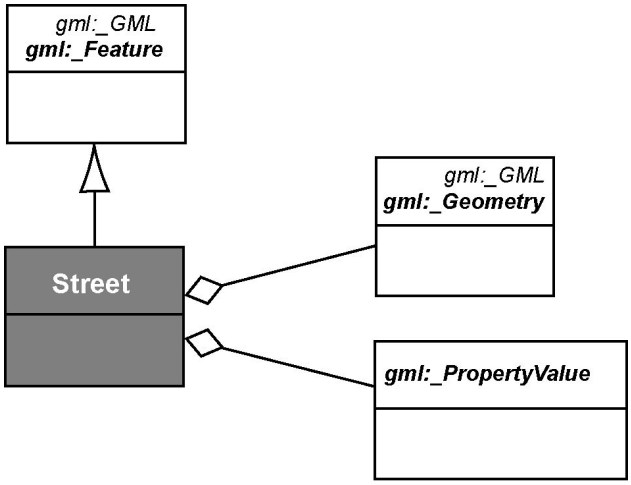

Figure 2. UML schema representing how GML components are reused in a specific application. In UML, a line with a triangle can be read as 'is a kind of' and the lines with diamonds are read as 'has'. This diagram is read as follows: 'A **street** is a kind of GML feature that has geometry and property'.

However, GML also introduces design patterns, conformant with the General Feature Model defined in ISO 19109:

- Classes are associated to other classes (or to simple data types such as strings or integers) using GML properties (i.e., the Class-property model). GML properties must have meaningful names and defined types. This rule forces relations to be qualified, documenting why or how classes are related.
- All class names must start with a capital letter (`'Street'`, and not `'street'`) and all property names must start with a lower case letter. More precisely, GML uses the 'CamelCase' structure where names can be formed from several words using capitalisation as a separator, for

example **'MainStreet'** as a class name and **'coveringMaterial'** as a property name.

- All classes must have IDs. In a complex document, the same feature can be referenced several times in the same document, or from outside the document. To ensure that every piece of information can be successfully reused, it must be uniquely identified. GML provides this functionality with the **gml:id** attribute.
- XML element attributes should be avoided. GML uses attributes in two very specific cases: 1) to assign ID and references to other features in the document or in another document; and 2) for limited metadata, in particular indicating the reference system for simple values (e.g. spatial coordinates, quantities, codes). This means that all first order information should be modelled as XML elements.

This snippet of GML code is an example of these encoding rules applied to the model depicted in Figure 3 (adapted from Galdos System Inc., 2003):

```
<app:RoadSegment gml:id="RS1"[1]>
   <app:name>Handbury Road North</
      app:name>
   <gml:centerLineOf[2]>
      <gml:LineString gml:id="L1">
         <gml:posList srsName="#
              localCRS2a">
         1,2 2,3 3,4 4,0
         </gml:posList>
      </gml:LineString>
   </gml:centerlineOf>
   <app:numberOfLanes>2</app:
      numberOfLanes>
   <app:surfaceType>Asphalt</app:
      surfaceType>
   </app:RoadSegment>
```

[Note the typical

```
class (UpperCase "RoadSegment")
   property (lowercase "center
      LineOf")
      class (UpperCase "Line
           String")
```

structure].

[1]Required ID.

[2]Since RoadSegment inherits from Feature, it can have 0 or more geometries. One type of GML geometry is **centerLineOf**.

"app:" and **"gml:"** are namespace prefixes. The namespace mechanism provided by XML prevents 'name clashing' when several schemas are brought together in the same document. In this case, the designer chose to create a **<app:name>** tag to hold the name, but a

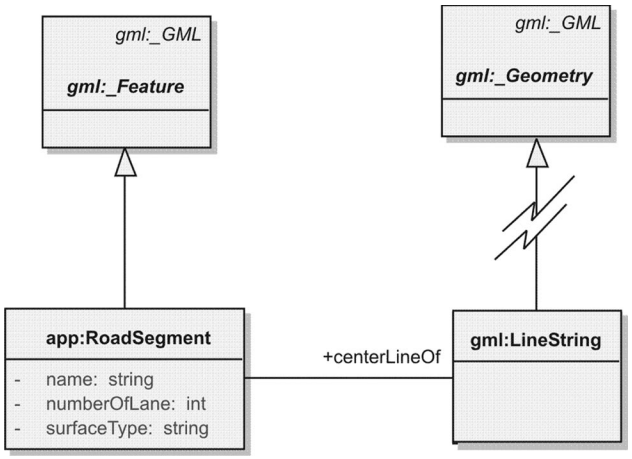

Figure 3. UML representation of a GML **RoadSegment**. RoadSegment inherits from the abstract **gml:_Feature** , so a **RoadSegment** is a GML Feature. The **RoadSegment** declares a series of simple properties (**numberOfLane**, etc..) and a more complex property named '**centerLineOf**' that relates to a complex geometric feature from gml called **LineString** (which is a special kind of **Curve**) that derives (through a series of intermediate geometry types) from an abstract **_Geometry** class.

<gml:name> also exists in GML. To differentiate between those two elements, a namespace mechanism is used.

Another way to encode the same piece of information (Galdos System Inc., 2003) is:

```
<app:RoadSegment gml:id="RS1">
   <app:name>Handbury Road North</
      app:name>
   <gml:centerLineOf href="#L1" />
      This points to a LineString
   <app:numberOfLanes>2</app:
      numberOfLanes>
   <app:surfaceType>Asphalt</app:
      surfaceType>
   <app:width uom="m">7.0</app:
      width>
</app:RoadSegment>…
…
…
<gml:LineString gml:id=»L1»>  This is
   the LineString, located further in document
   <gml:posList srsName="#localCR
      S2a">1,2 2,3 3,4 4,0</gml:
      posList>
</gml:LineString>
```

where "localCRS2a" is the value of an ID attribute on the definition of the coordinate-reference system.

ENCODING OF NADM C1

The NADM Data Model Design Team (DMDT) recently released version 1 of its conceptual model (C1), which is available at http://nadm-geo.org. The model is described in a document containing UML diagrams, descriptions of classes, and accompanying text. The conceptual model is 'implementation neutral', in that it does not describe how it can be implemented in any specific technology, be it XML or in a relational database. Conversion of the conceptual model to an XML implementation first required adaptation of the conceptual model into a logical model expressed in a "GML-friendly" profile of UML, in preparation for formal encoding. This required: a) converting NADM C1 into a feature model, and b) replacing some specific UML structures with structures that are directly compatible with GML/XML. The main objective was to retain the original meaning of NADM C1 and to only reinterpret it for conversion to the GML application.

The Class-property model used in GML encoding is represented in a UML logical model (Figure 4) where:

- GML Classes (including Feature types) correspond to UML Classes defined in NADM C1, in a derivation hierarchy; all NADM C1 classes derive ultimately from abstract **_Feature** and abstract ***_GML***. Note that the underline *(_)* preceding the name is a syntaxic convention to represent abstract classes.
- GML properties are modelled as UML attributes and associations (i.e., the lines between the classes), where the GML property name is the same as the UML attribute name or the role name associated with the target class from the original NADM C1 model.
- Thus, all associations must carry role names at the end corresponding to the child element. See association between **_EarthMaterial** and **Fabric** on Figure 6.
- Association classes from the original NADM C1 are replaced in GML by intermediate classes, with additional role names as required and fixed cardinalities (as demonstrated later on Figure 6).

Several difficulties in GML encoding were encountered, for example:

- We discovered possible improvements to NADM C1, such as modifying the relations between the **ParticleSize** and **ParticleShape** classes and the **ParticleGeometry** class. We then struggled to choose between the improved model and strict adherence to the published (NADM C1) model.
- Some UML constructs could not be ported to GML; for example, multiple inheritance (see **Fossil** class in NADM C1), although some techniques can be used in GML/XML to simulate multiple inheritance.

The resulting GML code or implementation, although depicted as a UML model is not a representation of the original conceptual model, but rather is a model of the GML application in which model elements are derived from the GML classes and some relation types are interpreted as XML structures, e.g., aggregation implies nesting tags into another tag in the XML document. This is a very important distinction; the GML-friendly UML diagram is not an amendment or change to the official NADM C1 conceptual model. It is a tool that is used to bridge between the conceptual model and the GML application.

Shown on Figure 5 are structural changes to the hierarchy of NADM C1. In GML, all classes must inherit from core GML classes (**_Object**, **_GML**, **_Feature**).

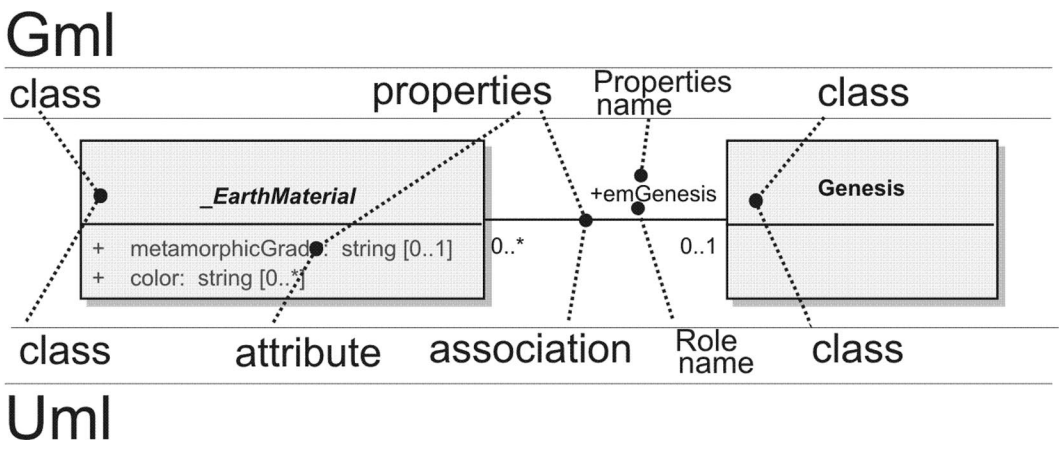

Figure 4. Representing GML using UML. The original UML diagram (top) is converted in a GML-friendly representation (bottom). Changes are described in the text.

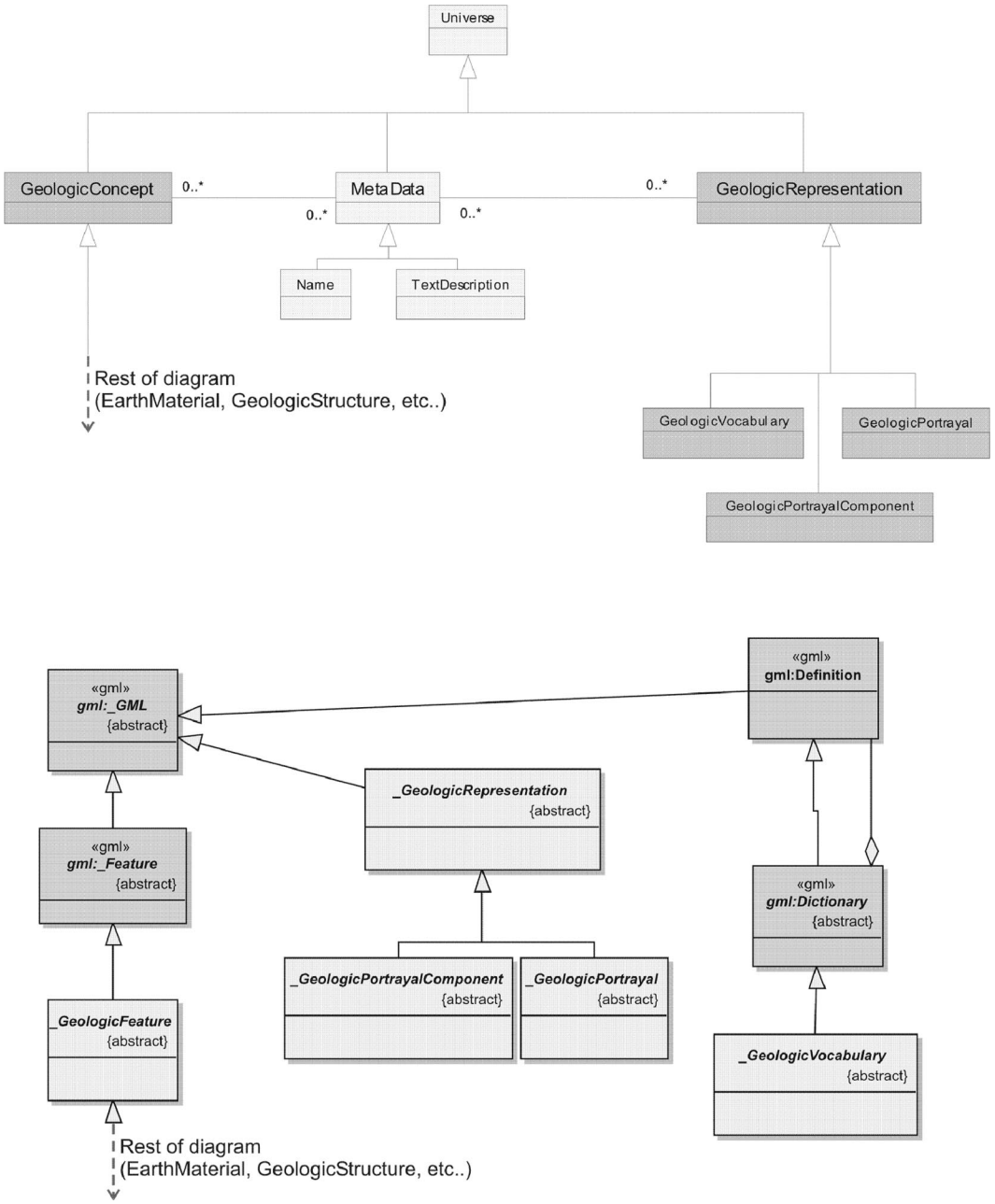

Figure 5. Adaptation of the top level of the model, essentially redirecting top level classes from NADM C1 to GML. Top part is original NADM C1 while bottom part is the GML interpretation. The original root of NADM C1 is a concept called **`Universe`**, from which derives **`GeologicConcept`**, **`MetaData`** and **`GeologicRepresentation`**. GML offers an alternative set of root concepts from which GML application must derive. For instance, **`GeologicConcept`** has been renamed **`_GeologicFeature`** to match GML syntax and now derives from GML's **`_Feature`**, which provides **`_MetaData`** functionalities. **`_GeologicRepresentation`** now derives from **`_GML`**, which is a high level abstract object. **`_GeologicVocabulary`** has been moved under GML **`_Dictionary`**, since this GML class offers the functionalities **`GeologicVocabulary`** intended to offer.

Therefore, top level classes of NADM C1 have been remodelled as descendents of GML features. Although multiple inheritance is allowed in UML, it is almost always suggested to avoid it in any kind of implementation, hence we did not keep the original top classes from NADM C1, but renamed the original **GeologicConcept** to **_GeologicFeature**. GML already provides the naming and documenting mechanism (**_MetaData**) so they don't need to be duplicated in NADM C1 encoding. The **GeologicVocabulary** from NADM C1 takes advantage of the GML Dictionary that provides the same core functionality.

Figure 6 shows some typical examples of changes that were made:

1. **EMConstituent** has been converted from an association class into a bridge class—a class that acts as a connector between two classes (**EMConstituent** was linked to the association between **EarthMaterial** and **CompoundMaterial**). Therefore, **EMConstituent** will behave as a container in GML that will wrap the **_EarthMaterial** class (see GML example, Figure 7).
2. **EMCRelation** has been remodelled as a GML property, because it essentially links two classes, **EMConstituent** and **EMCRole**. We also redesigned the way relations were described in the model. **EMCRelation** is a kind of **_GeologicRelation** (see NADM C1 documentation) and shows the parent class to emphasise that **EMConstituent** can have any kind of relation.
3. **Proportion** and **EMCRole** have been wrapped into properties of **EMConstituent**. This representation has the same meaning as the change made to **EMCRelation** (#2), but this representation is preferred when we feel that the property will

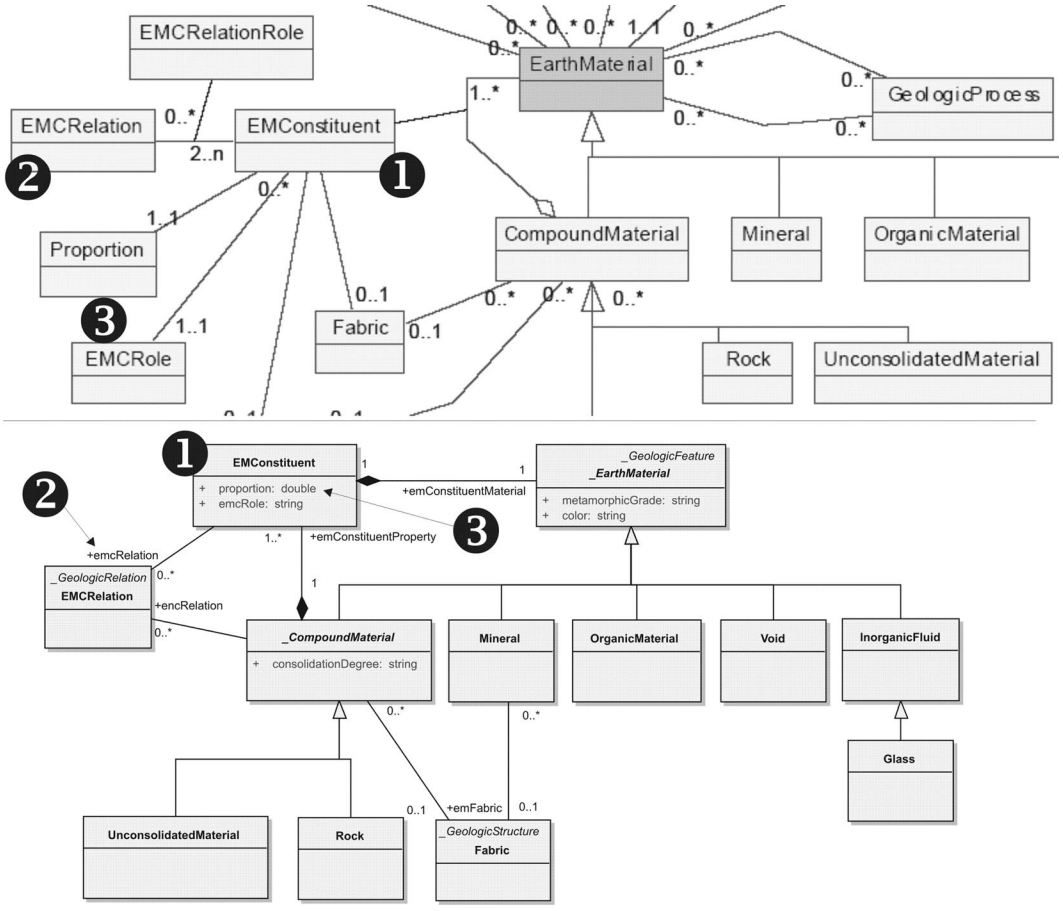

Figure 6. Reinterpretation of the conceptual model to conform to GML (and XML) constraints. The top part of the figure is from the original NADM C1 document while the bottom part shows adaptations to create a GML-friendly model. Principally, NADM association classes are characterized as GML bridge classes (see label #1), some NADM classes are characterized as GML complex properties that are depicted as UML associations (see label #2), and some NADM classes are characterized in GML as simple properties that are depicted as UML attributes (see label #3).

contain simple values (such as text or number), as opposed to complex values. We expect these properties in GML to be modelled as simple XML entities, ie, tags that cannot contain any other tags.

In some instances, the adaptation was not trivial. For example, **GeologicAge** might be replaced by the more detailed encoding designed by the XMML group. The issue at this point is: do we keep the simple structure that NADM C1 proposes (follow the guiding principle of being consistent with NADM C1) or do we simply reuse XMML, which essentially is the same thing but uses a different vocabulary and is more detailed (so, we would not reinvent the wheel).

EXAMPLE OF A NADM GML DOCUMENT

This example sums up the discussion about GML encoding. The portion of the document (the full document is located at https://www.seegrid.csiro.au/twiki/pub/CGIModel/EarthMaterial/earthMaterial.xml) should be compared to our model depicted on Figure 7 (numbers correspond on document, comments and figure).

```
<?xml version="1.0" encoding="UTF-8"?>
<NADM xmlns="http://geology.usgs.gov/dm/NADM/v1.0" xmlns:xsi="http:
//www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://
geology.usgs.gov/dm/NADM/v1.0 NADM-0_1.xsd" xmlns:xlink="http://www.w3.org/
1999/xlink">
  <!-- Current max id number = 27 -->
  <featureMember>
    <Rock id="nadm-1">  1
      <!-- Rock connected to a genetic event, w/constituent minerals -->
      <name>Joe's Granite</name>
      <description>Intrusive rock</description>
      <color>light gray</color>
      <consolidationDegree>consolidated</consolidationDegree>
      <emGenesis xlink:href="earthMaterial.xml#nadm-19"/>
      <emFabric>  2
        <!-- fabric of Rock -->
        <Fabric id="nadm-26">
          <name>Fabric description</name>
          <pervasiveness/>
        </Fabric>
      </emFabric>
      <emConstituentProperty>  3
        <EMConstituent id="nadm-10">
          <!-- mineral w/ it's own fabric -->
          <proportion>35</proportion>
          <emcRole>Mineral</emcRole>
          <emConstituentMaterial>
            <Mineral>
              <name>Quartz</name>
              <description>Silica</description>
              <color>white</color>
              <emFabric>
                <Fabric>
                  <name>Aligned C-axes</name>
                  <pervasiveness>pervasive</pervasiveness>
                </Fabric>
              </emFabric>
            </Mineral>
          </emConstituentMaterial>
        </EMConstituent>
      </emConstituentProperty>
```

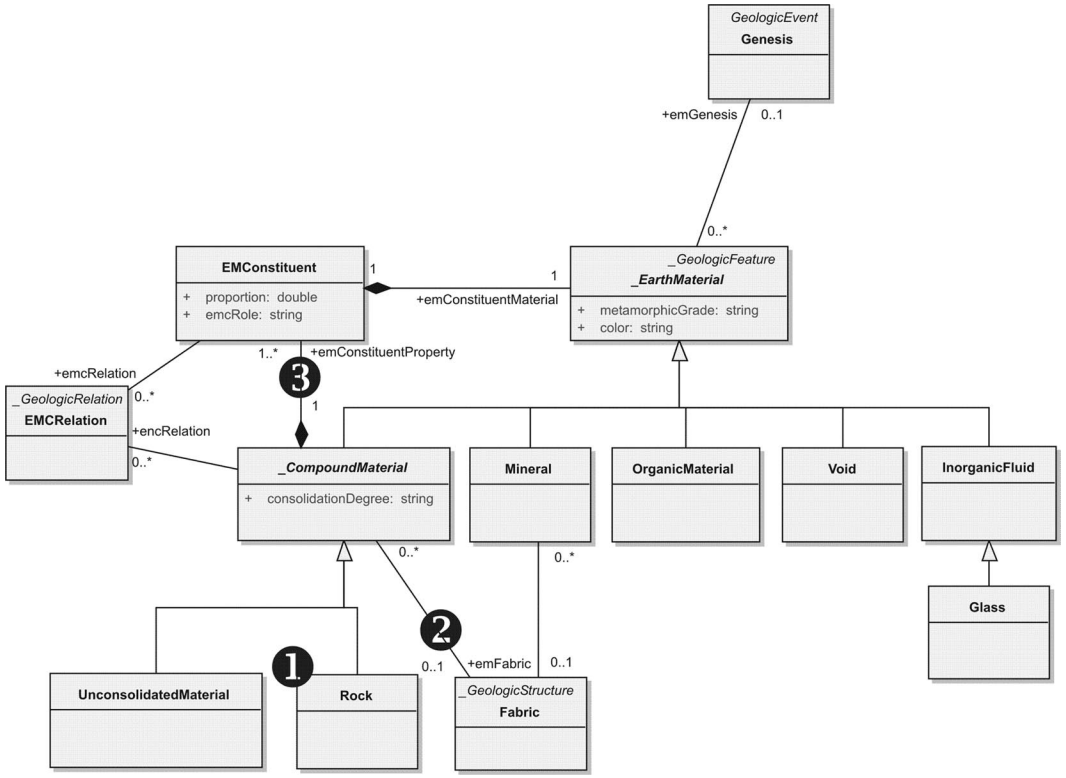

Figure 7. Example of a GML-compliant XML document with accompanying UML representation. The numbers on the figure refer to the example document and the notes in the text.

The main points noted in this example are:

1. A **Rock** class inherits from **_CompoundMaterial**, which itself inherits from **_EarthMaterial.** In other words, a **Rock** is a **_CompoundMaterial,** which is an **_EarthMaterial**. So, a **Rock** inherits the **color** and **consolidationDegree** properties (and also **metamorphicGrade**, but this property is optional). **color** is a simple property, shown as a UML attribute instead of an association. A rock also has a **Genesis**; in this example, we chose to point to a description in another document (earthMaterial.xml). **name** and **description** are inherited from far above the hierarchy shown in Figure 7, they are properties of the top-most class of the model.
2. A **_CompoundMaterial** can have a **Fabric**; note on the UML diagram that **emFabric** is the name of the association. Note the *CamelCase* structure; you can distinguish properties from classes quite easily by looking at how the name is capitalised.
3. A **Rock** is made of constituents (because it's a **_CompoundMaterial**); **proportion** and **EMCRole** are inherited from the **EMConstituent** class. The constituent material in this example is a Mineral, but other kind of **_EarthMaterial** can be substituted (**Rock** or **Glass** for instance).

INTERNATIONAL ACTIVITIES

This activity is a contribution to a larger international project hosted by the IUGS Commission for the Management and Application of Geoscience Information (CGI) (see http://www.bgs.ac.uk/cgi_web/tech_collaboration/tech_collab.html). As stated there, "*The overall objective of the Working Group is to develop international standards for the structure of geological information (i.e. data model standards) to enable interoperability among several national geological survey agencies.*" (see http://www.bgs.ac.uk/cgi_web/tech_collaboration/data_model/data_model.html). The working group progress is documented at https://www.seegrid.csiro.au/twiki/bin/view/CGIModel/WebHome.

CONCLUSION

Moving from generic XML encoding to GML has been beneficial for several reasons: a) it allowed encoding of the model to follow broadly accepted standards (ISO, OGC); b) it provided a much needed design pattern that resolved many consistency issues; and c) it opened NADM to other standards that are based on GML, such as WFS, and to other tool developers working with GML. As a by-product, encoding of the model was also an excellent review process and, for better or worse, generated a series

of revision requests and comments to the NADM Data Model Design Team.

ACKNOWLEDGMENTS

Peter Davenport, Andrée M. Bolduc and Kathleen Lauzière (Geological Survey of Canada) have kindly reviewed this manuscript. Dave Soller's (U.S. Geological Survey) comments greatly improved this paper. Eric Boisvert has been financially supported by *the Consolidation of Canadian Geocience Knowledge* and *Groundwater* programs of the Earth Sciences Sector of Natural Resources Canada. Simon Cox's contributions were supported by the Minerals and Energy Research Institute of Western Australia and the sponsors of project M340, and by the Open GIS Consortium. Boyan Brodaric was supported by *Sustainable Development through Knowledge Integration* and *Consolidation of Canadian Geocience Knowledge* programs of Earth Science Sector of Natural Resources Canada.

REFERENCES

Digital Interchange Technical Team, 2003, XML Encoding of the North American Data Model, *in* Soller, D.R., ed., Digital Mapping Techniques '03—Workshop Proceedings: U.S. Geological Survey Open-File Report 03–471, p. 215-221, accessed at http://pubs.usgs.gov/of/2003/of03-471/boisvert/index.html.

Galdos System Inc., 2003, Developing and Managing GML Application Schemas, http://www.geoconnections.org/developersCorner/devCorner_devNetwork/components/GML_bpv1.3_E.pdf.

Laxton J., and Brodaric, B., 2003, Data Model Collaboration Working Group Report, Edinburgh, Nov. 2003, accessed at http://ncgmp.usgs.gov/intdb/dmic/dmic.html.

North American Geologic-map Data Model Steering Committee, 2003, NADM Conceptual Model 1.0. A Conceptual Model for Geologic Map Information: preliminary website release under the auspices of the Geological Survey of Canada, the U.S. Geological Survey, and the Association of American State Geologists, accessed at http://nadm-geo.org, now published and available at http://pubs.usgs.gov/of/2004/1334/.

OpenGIS Consortium, 2004, OpenGIS Geography Markup Language (GML) Implementation specification version 3.1, Cox et. al., ed.: OGC Document 03-105r1, accessed at http://portal.opengis.org/files/?artifact_id=4700.