Copy No._____

Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Describing Generic Ocean Environmental Data Objects Using the Geography Markup Language

Anthony W. Isenor

Defence R&D Canada – Atlantic

Technical Memorandum
DRDC Atlantic TM 2004-087
July 2004

Canada

This page intentionally left blank.

Copy No:

Describing Generic Ocean Environmental Data Objects Using the Geography Markup Language

Anthony W. Isenor

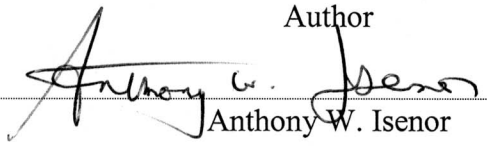
Defence R&D Canada – Atlantic

Technical Memorandum

DRDC Atlantic TM 2004-087

July 2004

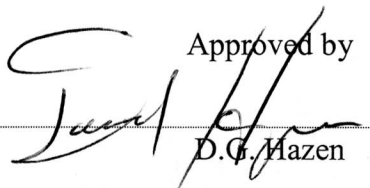
Author



Anthony W. Isenor

Defence Scientist

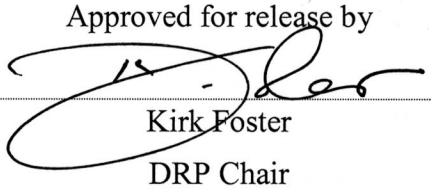
Approved by



D.G. Hazen

Head, Technology Demonstration Section

Approved for release by



Kirk Foster

DRP Chair

Abstract

This investigation examines the Geography Markup Language (GML) structure and its application to the transfer of one-dimensional ocean profile data (e.g., eXpendable Bathythermograph (XBT) data). GML was designed for the storage and transfer of geography-related data and is noted to be a NATO emerging standard for geographic data interchange. The investigation compares the GML-based structure for an XBT data profile to the same dataset stored in the Keeley Brick data structures. From the perspective of ocean data management, the results indicate that the GML nomenclature is difficult to understand. However, compared to the Keeley Brick approach GML provides benefits such as a standardized structure and efficiencies in file sizes.

Résumé

Cette étude examine la structure du langage GML (Geography Markup Language) et ses applications au transfert des données unidimensionnelles sur les profils océaniques (par exemple les données XBT (eXpendable Bathythermograph)). GML a été conçu pour le stockage et le transfert de données géographiques, et c'est un standard émergent de l'OTAN pour l'échange de ce type de données. L'étude compare la structure, basée sur GML, d'un profil de données XBT, et le même ensemble de données conservées dans des structures de données à base de briques de Keeley. Du point de vue de la gestion des données océaniques, les résultats indiquent que la nomenclature de GML est difficile à comprendre. Cependant, comparé à l'approche des briques de Keeley, GML présente des avantages, notamment une structure standardisée et une meilleure efficacité en ce qui concerne la taille des fichiers.

This page intentionally left blank.

Executive summary

Background

The ocean data community is examining data transfer structures for use within the community of international data centres. One potential transfer structure is Geography Markup Language (GML). The Open Geographic Information System (GIS) Consortium (OGC) has developed GML for the transfer, storage and manipulation of geographic data.

This investigation considers GML as compared to a previous data transfer investigation that developed the Keeley Brick structures for ocean data. The report provides a comparison between a one-dimensional dataset stored in the XML structures developed from the Bricks, to the same dataset stored in GML structures.

Principal Results

GML is shown to be a viable option for the transfer of ocean environmental data. Results show GML is a highly structured framework with intricate and often complicated internal structure. However, from the perspective of ocean datasets, the geography nomenclature in GML makes it difficult to relate to ocean dataset properties.

Significance of Results

Understanding the intricacies of data transfer structures is an important step in discerning the strengths and weaknesses of these structures. Such data transfers will be important for network centric warfare where the successful transfer of geographic-related data is important for all types of military operations. For the navy, geographic data includes those data pertaining to the ocean environment. By understanding and investigating these data structures, we hope to influence and encourage the ocean data community to standardize the data delivery structures. This standardization should lead to efficiencies in the generation of products beneficial to the navy.

GML is also important from the perspective of the North Atlantic Treaty Organisation (NATO). The NATO Data Administration Group is monitoring GML, considering it an emerging standard that may be applicable in future geography-related data transfers.

Future Plans

Further investigations will be conducted on data interchange services. Particular interest will be given to those standards listed as mandatory and emerging by the NATO Data Administration Group. Knowledge of such standards will be helpful in determining available solutions for the information transfer required during the Networked Underwater Warfare (NUW) Technology Demonstration Project, currently underway at DRDC Atlantic.

Isenor, Anthony W. 2004. Describing Generic Ocean Environmental Data Objects Using the Geography Markup Language, DRDC Atlantic TM 2004-087, Defence R&D Canada – Atlantic.

Sommaire

Situation générale

La communauté des données océaniques est en train d'examiner les structures de transfert de données pouvant être utilisées pour les échanges entre les centres de données internationaux. GML (Geography Markup Language) constitue une structure de transfert potentielle. L'OGC (Open Geographic Information System (GIS) Consortium) a développé GML pour le transfert, le stockage et la manipulation des données géographiques.

Cette étude compare GML et les structures à base de briques de Keeley pour le transfert des données océaniques. Le rapport compare un ensemble de données unidimensionnelles conservées dans des structures XML construites à partir des briques de Keeley, et le même ensemble de données conservées dans des structures GML.

Principaux résultats

On démontre que GML constitue une solution viable pour le transfert des données environnementales sur les océans. Les résultats montrent que GML est un cadre fortement structuré, dont la structure interne est subtile et souvent compliquée. Cependant, du point de vue des ensembles de données océaniques, la nomenclature géographique de GML rend difficile d'établir les relations avec les propriétés des ensembles de ce type.

Importance des résultats

Comprendre les subtilités des structures de transfert de données constitue un pas important pour en déterminer les points forts et les points faibles. Les transferts de données de cette nature sont importants pour la guerre réseaucentrique, où le transfert réussi de données géographiques est important pour tous les types d'opérations militaires. Pour la marine, les données géographiques comprennent les données qui concernent l'environnement océanique. En comprenant et en étudiant ces structures de données, nous comptons influer sur la communauté des données océaniques et l'inciter à standardiser les structures de distribution des données. Cette standardisation devrait rendre plus efficace la génération de produits avantageux pour la marine.

GML est également important du point de vue de l'Organisation du traité de l'Atlantique Nord (OTAN). Le groupe d'administration des données de l'OTAN surveille GML, et considère qu'il s'agit d'un standard émergent pouvant être utilisé dans l'avenir pour le transfert de données géographiques.

Plans pour l'avenir

D'autres études seront effectuées sur les services d'échange de données. On s'intéressera tout particulièrement aux standards dont le groupe d'administration des données de l'OTAN considère qu'ils sont obligatoires et émergents. La connaissance de ces standards aidera à déterminer les solutions de transfert de l'information requises dans le cadre du projet de démonstration de la technologie de la guerre sous-marine en réseau, actuellement en cours à RDDC Atlantique.

Isenor, Anthony W. 2004. Describing Generic Ocean Environmental Data Objects Using the Geography Markup Language, DRDC Atlantic TM 2004-087, Defence R&D Canada – Atlantic.

Table of contents

List of figures

List of tables

This page intentionally left blank.

1. Introduction

Geography is common to everyone. Even on a very local level people travel past rivers, hills and valleys, thereby developing an appreciation for the local geography, distances and geographical features.

We are also very familiar with abstract representations of geography. One only need consider the local newspaper and the weather report. The report typically shows a local map with overlaid atmospheric conditions. The data, which may be represented by temperatures or sun and cloud symbols, are referenced to locations on the map. The casual observer can quickly determine the local forecast in part, due to their experience with maps.

In many fields of investigation, the data and information contain a component of geography. In the above weather map example, the information is presented in relation to geography. The reader is responsible for orienting themselves geographically and then obtaining the weather information they require. In the military, the required information may be weather, or possibly assets such as ships, planes and vehicles. At any instant in time, these objects may exist at a specific location in space and thus have a geographic component to their description.

The geographic component of military or environmental data is critical to the proper description of these data. Without the geographic coordinates, the data may be useless to the client. However, the transport of this information to the client must also be clear and unambiguous. If the client has the information, but cannot understand it, the data are essentially useless.

The unambiguous transport of such data and information has become more important as the exchange of data increases. Using existing technologies, data is easily exchanged with colleagues around the world. In a military operational scenario, one could envisage systems exchanging data on a local level to generate a more complete picture of the operational space. Such sharing may help maximize the use of the assets in the operational space.

However, the successful transfer of data is not trivial. The transfer must meet two requirements to be considered successful: (1) it must deliver the data that is required by the receiving client, and (2) it must deliver the data in an understandable form. If either condition is not met, the transfer is a failure.

The transfer of data, and in particular the form of the transferred data, has traditionally caused problems for the receiving party. In the 1970's, data were often transferred on magnetic tape, typically in a binary format particular to the provider of the dataset. Although the specific media on which the data exist has changed through the years, centre-specific binary representation of the data files is still commonplace.

Of course not all data files are binary. Smaller data files may exist as comma separated columns of data in ASCII form. This is perfectly acceptable and readable by many modern-day applications. However, the problem with such data files is the inability to explicitly represent data structure.

The introduction of the eXtensible Markup Language (XML) [1] has changed this. XML allows the explicit representation of structured and varied data types in any character set (e.g., ASCII). This enhances the readability of the received dataset, and thus an improved level of understanding (addresses requirement (2) above). As well, the communities of interest may define the XML structures used to contain the data. This provides the community with the ability to define the important content of the dataset (addresses requirement (1) above).

In the community of XML developed languages, the Geography Markup Language (GML) [2] is one option for transferring data that contains a geographic component. Of course other specialized markup languages have also been defined. In the field of oceanographic data, the definition of generic data objects (sometimes referred to as Keeley Bricks) [3, 4] has provided the ocean data community with content requirements from which to build a markup language. However, the Bricks are not a standard, nor is the XML-based language that was built from the Bricks.

The development of the Keeley Bricks originated as a Canadian contribution to an international study group. The International Council for the Exploration of the Sea (ICES) and the Intergovernmental Oceanographic Commission (IOC) have formed the ICES/IOC Study Group on the Development of Marine Data Exchange Using XML (or the Study Group on XML, SGXML) [5]. The SGXML was initially tasked to develop generic data objects for use in transferring ocean data between international data centres.

Inherent in the Brick development process was the distinction between Brick development and application of the Bricks to XML. This is an important difference, as the Brick development aspect is defining the content and the relationships between the content. The application to the XML environment is a second independent step in the process.

By separating the steps, the Brick definition step becomes applicable to many application environments. Such environments span beyond XML, to include a multitude of other formats. Also, within the XML developed languages, there are many application options.

In the SGXML context, the generic objects or Keeley Bricks, have been used to construct a structure suitable for one-dimensional data stream [3]. A one-dimensional stream is any data that has one independent variable. For example, when the independent variable is depth (or pressure) the data stream constitutes a vertical profile such as an eXpendable BathyThermograph (XBT) profile. When the independent variable is time, the one-dimensional profile is a time series. An example would be current measurements from a fixed mooring location.

The initial development used the Brick definitions and an XML developed language specific to the bricks. This allowed the one-to-one relationships to be created between the Brick definitions and the XML element/attribute definitions. This development path is the simplest to understand and apply. This is because an understanding of the Bricks easily leads to an understanding of the XML elements and attributes, because the naming is identical.

However, the Bricks do not need to be applied in their own XML environment. The Brick content and the structure within the Bricks can be applied in other XML environments. For example, GML may be used to define structures with the content as defined by the Bricks. Thus, an investigation began to relate the Keeley Brick XML representation to GML.

1.1 Outline of Report

The following report will apply the Brick definitions to the GML environment. However, the application will not be a mapping between Bricks and GML objects. Such a mapping would be extremely difficult. This is because the Bricks themselves have only internal structure. Combining Bricks in a manner appropriate to the data type develops the external structure. In the GML case, there is internal and external structure already defined in the specification. This makes a direct comparison to the Bricks, which contain only internal structure, extremely difficult.

This report will describe the results of the comparison between the Keeley Brick XML implementation and GML. This comparison does not include all Bricks in the initial definition. Rather, an example data file structured in a one-dimensional profile XML document is converted to GML.

The following report first provides background information on XML. The reader will be assumed to have an intermediate knowledge of XML and XML schema language. References will be provided for those requiring additional resources.

The Open Geographic Information System (GIS) Consortium (OGC) and GML itself will then be described. GML is one specification from a large geography-interested consortium. GML is also of interest to the military, as indicted by the North Atlantic Treaty Organisation (NATO) standards that identify GML as an emerging standard for data interchange service. The role of GML in NATO will be described.

Finally, GML will be used to represent a one-dimensional profile of ocean environmental data. The data will be described based on the initial Keeley Brick XML application and then from the perspective of GML.

Within the text, a specific syntax will be followed for describing the XML elements, attributes and objects. Elements will be denoted with the angle brackets (e.g., <name>). Attributes will be identified using italic text (e.g., *name*). The content

of the attribute or element will be denoted using Arial font and double quotes (e.g., “name”).

There are also three naming conventions in use in this document. The conventions follow the original documentation for the specification. The Keeley Bricks will be described using all lowercase characters. In GML, objects will be implemented using XML elements with upper camel case names. *Upper camel case* has capitalization of all word first letters, with no spaces (e.g., AbstractFeatureCollection). GML properties are XML elements denoted by lower camel case naming. *Lower camel case* is a capitalization pattern where the first letter of the first word is not capital, but all remaining word first letters are capital, with no spaces between words (e.g., codeSpace).

2. Background

This section will provide background information on several topics that are central to the report. The section will briefly introduce XML and GML. Also, the organisation that created GML will be described. This will be followed by an overview of GML and its role in NATO.

2.1 XML Introduction

It will be assumed that the reader is familiar with XML essentials. Much of the required XML knowledge may be obtained from books [6], online sources [7, 8] or reports [3, 9]. For a terse overview of XML as it relates to the present subject, see reference [9].

For the following report, the reader should also be aware of the XML schema language. The XML schema language [10, 11] is a World Wide Wed Consortium (W3C) specification for describing the objects and structure of XML documents. The schema language is itself an XML-based language.

Part of the schema definition is particularly important for the following report. The schema language is used to define XML data objects. Once an object is defined, the schema specification allows another object to be defined based on the first object. The process may utilize schema *extensions* [10, 11]. An *extension* is the process of defining one object by adding components to a previously defined object. In this way, the previously defined object is extended to create the new object.

In a similar way, an object may be defined using schema restrictions. A *restriction* [10, 11] defines a new object from a previously defined object by restricting the permitted content of the previously defined object. Thus, the new object occupies a smaller data domain as compared to the previously defined object.

Another important component is substitution. In XML schema language, a *substitution* [10, 11] uses one object in place of another object. Consider an example where object A is defined in the schema, and object B is considered a substitution for object A. In this case, object B may appear anywhere in the instance document where object A is permitted. Here, B is the only member of the substitution group.

Many objects in GML are also defined as abstract XML objects. *Abstract* objects [10, 11] may be considered a high-level object representation that cannot exist as a concrete object in an instance document. Other objects must be defined based on substitutions from the abstract objects.

Finally, one should be aware of XML nodes. The XML specification for addressing parts of an XML instance document is called XML Path Language (XPath, see [9] for

a terse overview or the XPath specification [12]). In a conceptual sense, XPath considers an XML document as a tree structure. Nodes may be considered the branching points in the tree. Various types of nodes exist, such as element nodes, attribute nodes and text nodes.

Also in the XML descriptions contained in this document, are occurrence numbers for attributes and elements. Occurrence will be depicted using square brackets with the occurrence numbers denoted as shown in Table 1.

Table 1. In some figures in this report, the occurrence of XML attributes and elements will be described using square brackets. This table relates the bracket content to the number of allowed occurrences of the item.

COMMAND	FUNCTION
[1]	One and only one item is allowed.
[0, 1]	Zero or one item is allowed.
[1, *]	One to many items are allowed.
[0, *]	Zero to many items are allowed.

2.2 Open GIS Consortium

The Open Geographic Information System (GIS) Consortium (OGC) Inc. [13] (formed in 1994) is an international group of 257 companies, government bodies and universities. OGC produces publicly available specifications for data and information exchange oriented toward geospatial data. At the time of writing, the OGC have developed 14 specifications covering such topics as data cataloguing, coordinate transformations, and simple feature manipulations using tools such as Common Object Request Broker Architecture (CORBA) and Structured Query Language (SQL). The Geography Markup Language is one specification developed under the OGC.

In terms of Canadian participation, there are a total of 17 Canadian groups involved in OGC. The Canadian federal government is involved through Natural Resources Canada. Other Canadian involvement includes University of Northern British Columbia, the City of Toronto, and Laval University (Center for Research in Geomatics). Involved Canadian companies include PCI Geomatics Inc., CARIS and Galdos Systems Inc.

2.3 Geography Markup Language

GML is an XML-based markup language. According to the specification [2], GML *"is an encoding for the modelling, transport and storage of geographic information including both spatial and non-spatial properties of geographic features"*[2]. The data and information is represented as properties of the defined object, where properties can be either spatial (e.g., location of an object) or non-spatial (e.g., name of an object).

The GML specification [2] defines the GML objects using the XML schema language [10]. A set of 27 XML schemas comprises the GML specification (Figure 1). At present, GML exists as version 3.0.1. This version was released in January 2003, and represents a considerable expansion as compared to the previous version 2.1.2. For example, the schema representation for version 3.0.1 is eight times the size of version 2.1.2.

GML also uses three additional schemas covering Synchronized Multimedia Integration Language (SMIL) [14] and XML Linking Language (XLink) [15] (Figure 1). SMIL in an XML-based language for interactive, multimedia presentations. XLink provides the framework for creating Uniform Resource Identifier (URI) link structures within XML documents.

The 27 GML schemas have been developed taking advantage of the object inheritance characteristics found in the XML schema specification. This has resulted in a somewhat difficult to follow implementation. GML schemas utilize XML abstractions, substitutions, extensions and restrictions that can result in a complicated trail of object creations.

2.4 NATO and GML

The North Atlantic Treaty Organisation (NATO) continually monitors and evaluates existing technologies for use across NATO partners. NATO technical groups are tasked to consider technologies that support a variety of service areas including data interchange, or the exchange of data to support interoperability. A component of data interchange deals specifically with geography data. GML is recognized as an emerging standard in the geography data category.

NATO recognizes the potential of GML by including it as an emerging standard. However, this status indicates that GML does not yet meet the full requirements of NATO. The details of the NATO organizational structure, and GMLs place in this structure, are given in Annex 1.

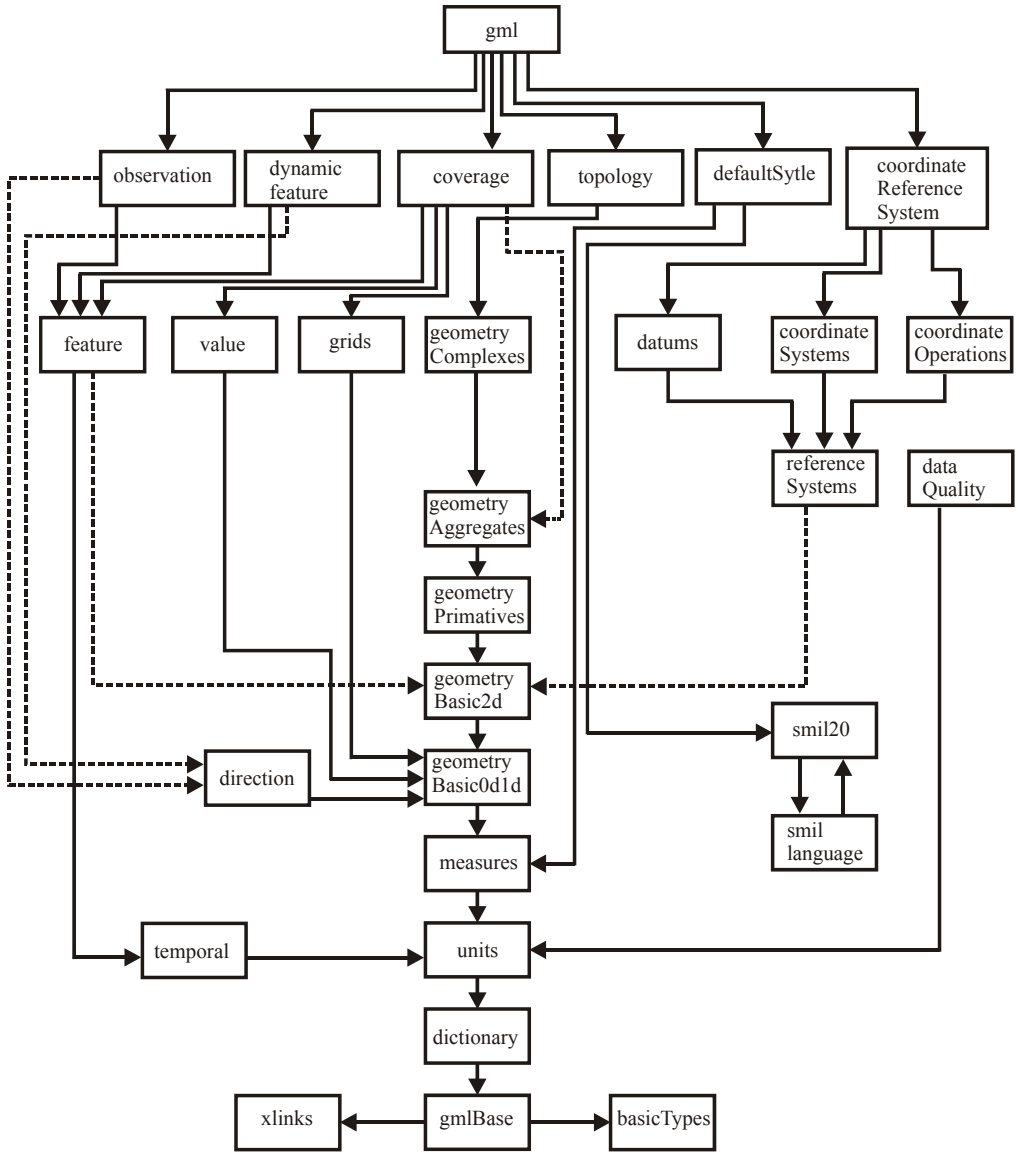


Figure 1. The 27 GML schemas. Three additional schemas are also shown (smil20, smil language, and xlinks). The solid and dashed lines (line difference for clarity only) indicate how schemas are included in other schemas. For example, the gml schema includes six schemas: observation, dynamicfeature, coverage, etc. Reproduced from [2].

3. Environmental Profile Data in GML

As noted previously in Figure 1, the GML specification consists of six high-level schemas from which all other schemas are included in the structure. The six schemas define the main functions of the GML. One of these six schemas, the observation schema, will be used for this investigation.

The observation schema is based on the concept of an observation. In GML terminology, an observation is considered to be a feature with an associated time element, which in this case is the time the observation was made.

The coverage schema was also examined. A coverage representation would be a useful GML construct for data types involving a grid. Using coverage, one may define a grid and incrementing scheme for the grid. The data values are then associated with the grid by placing data in all or some of the grid cells.

Unfortunately, the coverage XML schema would not validate using any of the XML validators readily available (five validators in all; note that validation is the process of checking the schema objects against the schema specification). Although the implementation of the schema specification in the validators is being questioned [16], the fact remains that the inability to validate the coverage schema creates uncertainty when using the coverage schema. For this reason, the coverage schema was not considered further.

3.1 The GML Observation Schema

In the observation schema, there are two basic types of observations available to the user: directed and non-directed observations. Directed observations account for observations that are based on a specific direction. For example, a photograph may be taken while standing at a particular location and looking north. In this case, the photograph represents the observation and the direction is north. Non-directed observations, or simply observations, account for observations that do not have a directional component (e.g., a temperature reading at a specific location).

For the purpose of this investigation, we consider scalar observations of temperature in a one-dimensional profile. For scalar observations, the schema defines the ObservationType (Figure 2) as an extension of the GML AbstractFeatureType (Figure 3). The ObservationType is therefore built from a base set of elements in AbstractFeatureType (Figure 3) plus the elements defined in the extension sequence (Figure 2). The extension adds XML elements for time stamping the observation, for describing how the observation was obtained, the intended target of the observation and the result of the observation.

From the extension elements, only the intended target requires clarification. This is the subject of the observation. Consider photography, where the subject of the photograph may be included with various other objects in the photo. The target element would identify the intended subject for the photo.

```
<complexType name="ObservationType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:timeStamp"/>
        <element ref="gml:using" minOccurs="0"/>
        <element ref="gml:target" minOccurs="0"/>
        <element ref="gml:resultOf"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Figure 2. The XML schema syntax for the GML ObservationType. ObservationType is based on the AbstractfeatureType with the extension elements for timeStamp, using, target and resultOf. Note that the GML namespace is indicated by gml:.

The AbstractFeatureType shown in Figure 3 is indicated using the subelements in the structure. The main element is indicated with "..." to indicate that the defined element may take any name as defined by the author. AbstractFeatureType is an abstract object and therefore cannot exist as an object in the defined structure. Note that the attributes of AbstractFeatureType are indicated with the attribute occurrence.

```
<... gml:id="[0,1]" fid="string [0,1]">
  <gml:metaDataProperty> ... </gml:metaDataProperty> [0,*]
  <gml:description> ... </gml:description> [0,1]
  <gml:name> ... </gml:name> [0,*]
  <gml:boundedBy> ... </gml:boundedBy> [0,1]
  <gml:location> ... </gml:location> [0,1]
</...>
```

Figure 3. The GML AbstractFeatureType as described in the feature schema. Schema annotations and comments have been removed for clarity. The occurrence of each element and attribute are indicated according to Table 1.

3.2 Internal verses External Structure

Before considering the details of the GML implementation, the user needs to be aware of an issue involving internal and external structure. Here, we use the terms internal and external to describe structure related to the data objects, either Bricks or GML

objects. We consider internal structure to be the structure internal to the object or Brick. External structure is the structure of the entire XML document, and thus represents a structure related to a specific combination of objects.

The difference between the internal and external structure is better described with an example. Consider a single Brick, such as provenance. The provenance Brick has internal structure in that it contains content describing the agency, country, data grouping, etc. In the original XML application [3], the provenance brick could be placed at any level within the XML structure. The inclusion of the Brick depends only on the need for this particular information. The data type being used in the application in part determined the need. For the profile example, this resulted in the provenance Brick being placed in two locations, within the <data_collection> and <data_set> elements.

Now consider the GML implementation. Here, the provenance information will ultimately be placed in the <gml:description> and <gml:name> elements within the <additional_metadata> element. The <additional_metadata> element is a substitution for gml:_Feature. This means that any structure location that uses the gml:_Feature can substitute in the <additional_metadata> element.

Now consider that <gml:target> may be substituted with <gml:subject> In turn, <gml:subject> contains a gml:_Feature. Thus, it is possible for <gml:subject> to contain the <additional_metadata> element, which in turn could contain the provenance information. However, <gml:subject> refers to the subject of the observation in something like a photograph. Recall that the provenance Brick refers to the information regarding the source of the data.

In this case, the concept of provenance does not naturally fit in the subject of a photograph. The Bricks, as created, were simply packages of information that were not given any external structure (i.e., the external structure was only defined by combining the Bricks for a particular data type, not by the Bricks themselves). In GML, the objects have been defined as well as the external structure. The complication exists when the Bricks, with no external structure, are placed in the GML, with external structure.

3.3 The One-Dimensional Profile

The following will detail the conversion of the one-dimensional temperature profile obtained from an XBT from the initial XML implementation of the Keeley Bricks to GML. The report addresses the specific XML instance document rather than the generic Bricks. This is because of the difficulty in relating Bricks, with only internal structure, to externally structured GML. The instance document expressed in the XML Brick implementation is shown in Annex 2.

3.3.1 Data Collection

The data collection represents the document element in the initial XML implementation of the Bricks. In the GML implementation, the data collection will be represented as an extension to gml:AbstractFeatureCollectionType. The extension is shown in Figure 4.

The AbstractFeatureCollectionType provides a description and name for the data collection. However, the collection also needs a date definition and supporting datasets. As such, the AbstractFeatureCollectionType is extended to provide the date (see Figure 4, ldate) and dataset properties (see Figure 4, data_set) for the collection.

```
<complexType name="data_collection_brick">
  <complexContent>
    <extension base="gml:AbstractFeatureCollectionType">
      <sequence>
        <element name="ldate" type="gml:TimeInstantType"
                    minOccurs="0" maxOccurs="unbounded"/>
        <element name="data_set" type="kb:data_set_brick"
                    minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Figure 4. The schema representation of the data_collection Brick as built from a GML object. The representation extends the gml:AbstractFeatureCollectionType to include elements for date/time and data_set. The data_collection_brick is the object type used to define the <data_collection> element.

The Brick structures are identified in Figure 4 by the namespace kb: representing Keeley Bricks. The kb: namespace was not used in the initial implementation. Here, the kb: namespace is used to manage actual names from the initial implementation. However, the schema assigns an inaccessible URI for the Keeley Bricks. The complete schema is included in Annex 3.

The initial XML resulting from the <data_collection> document element is shown in Figure 5 (complete document in Annex 4). A prominent property of Figure 5 is the XML namespace list in the <data_collection> document element. The namespace list is important as it provides reference to the many namespaces that may be used within the GML document. Note that in this implementation, the GML namespace requires the gml: prefix. Thus, in the following XML document snippets, the GML elements and attributes will be indicated by the gml: prefix.

Note also that the default namespace of the XML document is the namespace assigned to the kb: reference in the schema. Thus, the Brick elements in the document do not need to be qualified (i.e., do not need the kb: prefix).

```
<?xml version="1.0" encoding="UTF-8"?>
<data_collection xmlns="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/Prog_Int/ICES" xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/Prog_Int/ICES
file:///c:/Anthony/International/Standards/GML/gml3.0/bricks_in_GML_3
.xsd">
  <gml:description>A test application to convert XML Keeley Brick
definitions to GML.  Note that I initially wanted to use the
coverage.xsd schema, but changed to observation.xsd   because of
validation problems.  The validators currently available cannot agree
on whether or not coverage.xsd is valid.  Given my limited knowledge
of the schema specification, and   the complexities in the gml set of
schemas, I cannot judge the validators for correctness.


  Anthony W. Isenor    -   February 2004</gml:description>
  <gml:name>Data set</gml:name>
```

Figure 5. The XML content at the beginning of the GML instance document for the 1-demsional profile data.

3.3.2 Bounding Box

The GML specification includes bounding box definitions. The hierarchical structure of ocean data can utilize this bounding box concept. As an example of this, consider the bounding box for a data collection. We will consider this example in three different ways: using the Brick definitions, using the XML specification of the Bricks, and using the GML specification of the Bricks.

In the Brick definitions, a location is defined using the location set information. A location set is a group of x,y,z,t variables that specify the spatial-temporal position of the data. Each location set specifies one spatial-temporal position. The location set information is shown in Figure 6.

As shown in Figure 6, the location set also contains unstructured comments. The z component is represented as depth_pressure, and x,y as longitude and latitude, respectively. The ldate contains date/time, and quality provides an indication of the quality of the position.

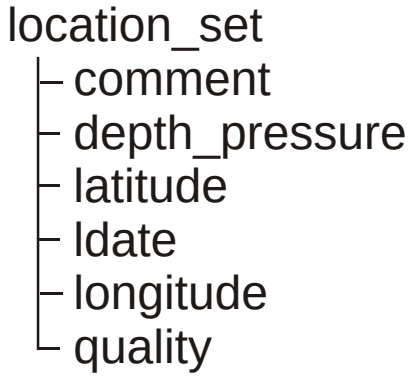


Figure 6. The Brick content for a location set. The vertical branching line illustrates the brick structure encapsulated within location_set. The rational behind the specification is given in [3].

The XML brick representation of a location set is as shown in Figure 7. This representation defines four spatial locations (in the figure no temporal position is described). In the XML structure defined for one-dimensional data, the cruise level of the dataset was defined to contain [0,n] occurrences of the location_set Brick (Figure 6). This allows the specification of multiple points, which may implicitly form a bounding box. In this example, we are considering a bounding box at the data collection level. However, similar bounding boxes could be constructed within any <data_set>.

However, in the Keeley Brick (kb:) namespace this description is implicit. The characteristics of the latitude/longitude values are set by the property attribute in the individual subelements of the <location_set>. This is also shown in Figure 7 by the fact that each sub-element contains an attribute property. However, the initial brick specification did not define the property content of "corner" is indicated here. The property content of "corner" would be required to extend the initial Brick definitions to include a bounding box capability. However, "corner" does not explicitly link the <location_set> elements. In this case, the multiple corners imply the creation of a bounding shape.

By specifying four corners using four <location_set> elements, we implicitly indicate a bounding area. This set of four <location_set> elements would be included within a data collection, or possibly within a dataset.

Now consider a bounding box defined in the GML environment. Here, the bounding box could be implemented as shown in Figure 8. An envelope is used as a container for a specific location. The <gml:description> and <gml:name> elements may contain details of the envelope. The <gml:coordinates> then contain the four defining points in the bounding box.

```
<location_set>
  <latitude property="corner">42.5</latitude>
  <longitude property="corner">-56.4</longitude>
</location_set>
<location_set>
  <latitude property="corner">46.0</latitude>
  <longitude property="corner">-56.4</longitude>
</location_set>
<location_set>
  <latitude property="corner">46.0</latitude>
  <longitude property="corner">-55.3</longitude>
</location_set>
<location_set>
  <latitude property="corner">45.0</latitude>
  <longitude property="corner">-55.3</longitude>
</location_set>
```

Figure 7. The XML implementation of XML location_set Brick. Here, four locations are defined indicating the corners of a bounding box. Note that attribute content of "corner"' is not part of the initial Brick development [3].

The <gml:coordinates> specification uses a comma separator for the coordinates, and a space separator for the pairs (both are defaults, and can be altered to suit needs). This has the advantage of compacting the data to a single XML element. However, the disadvantage is that the XML parsers cannot parse the content to unique nodes. Additional parsing would be required by the application to obtain the discrete positions of the bounding box.

As well, there is a validation consequence to this type of specification. The <gml:coordinates> element is based on string content and thus validators will not check for numbers or proper structure within the <gml:coordinates> element content. This means that any string content will pass validation. The XML Brick implementation restricted the latitude (e.g., -90 to +90) and longitude (e.g., -180 to +180) content.

```
<gml:boundedBy>
  <gml:Envelope>
    <gml:description>Bounding box for collection.</gml:description>
    <gml:coordinates>45.0,-56.4 46.0,-56.4 46.0,-50.3
45.0,-50.3</gml:coordinates>
  </gml:Envelope>
</gml:boundedBy>
```

Figure 8. The GML implementation of a bounding box at the level for the entire dataset. The coordinates of the bounding box are represented as a series of y,x values in the <gml:coordinates> element.

3.3.3 Provenance

In the brick definitions, the provenance of a dataset was described using the provenance Brick. This Brick contained metadata on the origin of the dataset. In the GML implementation we construct a metadata element named <additional_metadata> for this information.

The <additional_metadata> element is based on the GML AbstractFeatureType (Figure 3). For the implementation (Figure 9), we utilize <gml:description> and <gml:name> elements. We utilize the <gml:name> element to store a key word (or words) as defined in the hypothetical code list identified by the *codeSpace* attribute. The content associated with the key word is then stored in the <gml:description> element.

```
<gml:featureMembers>
  <additional_metadata>
    <gml:description>BIO</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">Agency</gml:name>
  </additional_metadata>
  <additional_metadata>
    <gml:description>1899</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">Institute
Code</gml:name>
  </additional_metadata>
  <additional_metadata>
    <gml:description>2003-02-18Z</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">Date
Created</gml:name>
  </additional_metadata>
  <additional_metadata>
    <gml:description>Shark Fishery</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">Project</gml:name>
  </additional_metadata>
  <additional_metadata>
    <gml:description>97025</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">Identifier</gml:name>
  </additional_metadata>
</gml:featureMembers>
```

Figure 9. The information pertaining to the owner of the data is stored in the <additional_metadata> element. This element contains a single <gml:description> and <gml:name>.

Admittedly, this may exceed the proper bounds for the intention of the <gml:description> and <gml:name> elements. However, we are using <gml:name> to store the declared name of a metadata property. In this sense, the <gml:description> of that name then represents the content associated with the name. In other words, once we define a metadata property such as an "Agency", we consider the content of the defined "Agency" to be "BIO".

3.3.4 Dates

Next we consider the specification of dates. Here, we use the ldate element from the initial Brick construction and define the ldate to be a GML timeInstantType. The timeInstantType (Figure 10) provides specification for a timePosition. The timePosition format allows varying degrees of precision for the content. Here we use the same format for the content as in the initial Brick implementation.

```
<... gml:id="[0,1]" frame="anyURI [0,1]">
  <gml:metaDataProperty> ... </gml:metaDataProperty> [0,*]
  <gml:description> ... </gml:description> [0,1]
  <gml:name> ... </gml:name> [0,*]
  <gml:timePosition> ... </gml:timePosition> [1]
</...>
```

Figure 10. An example XML representation of <gml:timeInstantType>. Since the type may be used to define any named element, the start and end tags are represented as " ...". The element content attributes and elements are also shown.

The implementation allows the description of the time position using the <gml:description> and <gml:name> elements as shown in Figure 11. Again, the *codeSpace* attribute is used to define the named content of the <gml:name> element. The <gml:description> is used to describe the name, while the <gml:timePosition> actually contains the date value.

```
<ldate>
  <gml:description>Start date of the collection.</gml:description>
  <gml:name codeSpace="www.meds.ca/namelist">Start</gml:name>
  <gml:timePosition>2003-02-18Z</gml:timePosition>
</ldate>
<ldate>
  <gml:description>End date of the collection.</gml:description>
  <gml:name codeSpace="www.meds.ca/namelist">End</gml:name>
  <gml:timePosition>2003-03-18Z</gml:timePosition>
</ldate>
```

Figure 11. A date value is implemented using the ldate element. ldate was initially defined as content in the location_set Brick. Here, the implementation is via the <gml:timeInstantType>.

3.4 Datasets

The dataset implementation in GML is similar to the initial Brick implementation in that a dataset is a self-recursive element (i.e., a <data_set> element may contain zero or more <data_set> elements). This is similar to the GML feature collection object.

A GML feature collection is any collection of geographic features. A collection may represent a university campus (with buildings and streets), or a wildlife preserve (with streams and forest). Individual items within the collections are represented as individual features. A feature collection may also be a feature in a larger collection. This is similar to the concept of a dataset, where one dataset may contain many other datasets.

The <data_set> element will be implemented as an extension of the GML AbstractFeatureCollectionType as shown in Figure 12 (AbstractFeatureCollectionType element is shown in Figure 13). The resulting dataset Brick contains elements for naming, description, bounds, date/time and data points. The content of the <data_set> element as described by Figure 12 can now be compared with the initial dataset description [3] as shown in Figure 14. The initial dataset definition is well documented [3, 4] and is comprised of all the information required to document the history, quality and related characteristics of the dataset. The individual elements of the dataset are also shown in Table 2 in an incomplete mapping to GML elements.

```
<complexType name="data_set_brick">
  <complexContent>
    <extension base="gml:AbstractFeatureCollectionType">
      <sequence>
        <element name="ldate" type="gml:TimeInstantType"
                              minOccurs="0" maxOccurs="unbounded"/>
        <element name="data_point" type="kb:data_point_brick"
                              minOccurs="0" maxOccurs="unbounded"/>
        <element name="data_set" type="kb:data_set_brick"
                              minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Figure 12. The XML schema definition for data_set_brick type is defined as an extension of gml:AbstractFeatureCollectionType. The extension is used to include the date/time, data points, and other datasets.

```
<...gml:id="[0,1]"   fid="string [0,1]">
    <gml:metaDataProperty> ... </gml:metaDataProperty> [0,*]
    <gml:description> ... </gml:description> [0,1]
    <gml:name> ... </gml:name> [0,*]
    <gml:boundedBy> ... </gml:boundedBy> [1]
    <gml:location> ... </gml:location> [0,1]
    <gml:featureMember> ... </gml:featureMember> [0,*]
    <gml:featureMembers> ... </gml:featureMembers> [0,1]
</...>
```

Figure 13. The gml:AbstractFeatureCollectionType as described by the content attributes and elements. The gml:featureMember and gml:featureMembers elements are the extension elements as compared to the base gml:BoundedFeatureType.

```
<data_set>
    <availability> ... </availability> [0,n]
    <comment> ... </comment> [0,n]
    <data_point> ... </data_point> [0,n]
    <data_set_id> ... </data_set_id> [0,n]
    <provenance> ... </provenance> [0,n]
    <quality> ... </quality> [0,n]
    <quality_testing> ... </quality_testing> [0,n]
    <variable_set> ... </variable_set> [0,n]
    <location_set> ... </location_set> [0,n]
    <history_set> ... <history_set>  [0,n]
    <data_set> ... </data_set> [0,n]
</data_set>
```

Figure 14. The initial definition of a dataset from [3].

The GML element mapping (Table 2) illustrates two points from the investigation. First, those bricks that were not present in the input XML instance have not been considered in the mapping. The properties of the dataset described by the availability, quality, and quality testing Bricks would need to be implemented via specialized extensions to the data_set_brick (shown in Figure 12) in a similar fashion as ldate and data_point.

Second, the <gml:description> element is used for comment and history_set. The <gml:description> may contain free-format strings and thus can be used in place of multiple comments. As well, the entire history of a dataset may be included in a single <gml:description> (as shown in Annex 4).

However, caution needs to be exercised when using free-format comments. In the past, some organisations have utilized comment fields for storing metadata that did not have an appropriate or natural location in the organisations in-house data structure. Free-format comments should not be used to store data that systems need to access. The parsing problem is simply too difficult. The need to store this information in free-

form comments indicates that the data structure being used is not capable of properly storing the important data.

Table 2. An incomplete mapping between key components of the <data_set> element [3] to the GML elements.

BRICK ELEMENT	GML ELEMENT(S)
availability	See Note 1
comment	gml:description
data_point	data_point
data_set_id	gml:name
provenance	gml:featureMembers
quality	See Note 1
quality testing	See Note 1
variable set	See Note 2
location set	gml:boundedBy
history_set	gml:description

1: The indicated Brick element was not present in the XML instance document and was therefore not considered in this investigation.

2: The variable definitions in the GML implementation are referenced by links to code sets, rather then defined within the GML instance document. In the initial Brick implementation, variables could be referenced or defined within the instance document. This functionality was not extensively explored for the GML implementation.

The dataset mapping (Table 2) only shows the sub-elements of the <data_set> element, not individual attributes of the sub-elements. One attribute of particular importance is the *level* attribute (not shown) in the <data_set_id> sub-element. This attribute describes the granularity of the dataset, thereby identifying datasets as cruise, station, profile, and record. The dataset content of the GML instance document contains this information in the <gml:name> element. The <gml:name> element is also used to store specifics regarding the granularity. For example, in Figure 15 <gml:name> is used to store the cruise number. Similar information may exist at the station and profile levels of the dataset. We will note later, that the record level does not exist in the GML implementation.

```
<gml:description>This is the cruise level of the Keeley Brick
structure.</gml:description>
<gml:name codeSpace="www.meds.ca/namelist">cruise</gml:name>
<gml:boundedBy>
  <gml:Envelope>
    <gml:description>This is a bounding box for the
cruise.</gml:description>
    <gml:name>97025</gml:name>
    <gml:coordinates>45.0,-56.4 46.0,-56.4 46.0,-55.3 45.0,-
55.3</gml:coordinates>
  </gml:Envelope>
</gml:boundedBy>
```

Figure 15. The initial content in the <data_set> element in the GML implementation.

3.5 Data Points

The data point brick is implemented as a gml:ObservationType with a single extension element. The extension is a construction to contain the dependent parameter data. The data_point definition is shown in Figure 16.

The dependent parameter is a new construction specifically for the GML implementation. This element will be used to store a comma-separated list of all dependent parameter values in the profile. The comma-separated list is used for consistency with the overall philosophy of GML. However, the dependent parameter element is defined as unbounded, to also allow the specification of numerous such elements.

```
<complexType name="data_point_brick">
  <complexContent>
    <extension base="gml:ObservationType">
      <sequence>
        <element name="dependent_param"
                 type="kb:dependent_param_construction" minOccurs="1"
                 maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Figure 16. The data_point brick is defined using an extension to ObservationType. The data point is extended using the dependent parameter element. The dependent parameter was not part of the initial Brick implementation [3].

The data_point Brick is based on the GML Observationtype (Figure 2), which is an extension of an AbstractFeatureType (Figure 3). AbstractFeatureType is in turn an

extension of AbstractGMLType. This particular implementation illustrates the nesting of types that is common in the GML structure.

The data point consists of the metaDataProperty, a description and a name as identified in Figure 3. The only element utilized from the AbstractFeatureType (Figure 3) base is <gml:boundedBy>. This element is used to store the minimum and maximum values for the independent variables. In this example, this corresponds to the minimum and maximum depths of the profile. The <gml:name> element identifies the code list being used to define the parameters in the bounding range. The range values are specified in the <gml:coordinates> element in a comma separated list in the same order as the <gml:name> specification (Figure 17).

This implementation of data_point primarily uses the extension elements as defined in Figure 3. The timeStamp element is used to position the profile in time. The format of this element may take many forms, all of which are ISO 8601 compliant [17]. Note that the format used in Figure 17 is slightly different than the format defined for the initial brick implementation. This is because the initial implementation stressed the requirement for the "Z", which indicates time in GMT. The date/time format shown in Figure 17 uses a "T" indicator, which is not permitted in the initial brick implementation. The initial brick implementation may be considered an additional restriction on the ISO 8601 specification.

Following from the <gml:timeStamp> element is the <gml:using> element. The <gml:using> element is intended to describe details of the sensor, instrument or procedure used to collect the data. Here, we utilize the *href* attribute to define a code list for the data type. We have also used the *title* attribute to specify the parameter code being utilized from this code list.

Enclosed in the <gml:using> element is the <instrument> element. The <instrument> is defined as an AbstractFeatureType (Figure 3) and thus substitutes the _Feature in the <gml:using> element (Figure 18). The substitution thus allows zero or one <instrument> elements within the <gml:using> element.

The single occurrence of _Feature in <gml:using> (Figure 18) restricts the implementation to one instrument description. The result is that the <data_point> element can contain data from only one instrument. This creates a problem if a dataset contains data from multiple instruments. For example, the dataset in this exercise could also contain temperature values from a Conductivity-Temperature-Depth instrument (CTD), interpolated to the same depths as the XBT. Such a dataset may be constructed to provide verification for the XBT data. Such a dataset would require multiple instances of the <data_point> element, each with a defined instrument. The link between the CTD temperature and the XBT depth variables could be made implicitly through the hierarchical structure of the XML document.

```
<data_point>
  <!--Some points to be made:
- The timeStamp and resultOf are mandatory inside data_point.
- The using element defines the independent parameters.  The resultOf
then contains the independent parameter values.
- This single using element could be used to define more than one
indepenedent parameter (e.g. xlink:title="LATD,LOND").  Then the
point_value element could be a comma separated list.
             (e.g. <point_value>32.4,-45.6</point_value>) -->
  <gml:boundedBy>
    <gml:Envelope>
      <gml:description>min/max depth of profile</gml:description>
      <gml:name codeSpace="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/">min_depth</gml:name>
      <gml:name codeSpace="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/">max_depth</gml:name>
      <gml:coordinates>3.2400,589.8800</gml:coordinates>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:timeStamp>
    <gml:TimeInstant>
      <gml:timePosition>1999-02-28T09:02:00</gml:timePosition>
    </gml:TimeInstant>
  </gml:timeStamp>
  <gml:using xlink:href="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/" xlink:title="DEPH">
    <instrument>
      <gml:description>SOC BT/SV Processor</gml:description>
      <gml:name>XBT</gml:name>
    </instrument>
  </gml:using>
  <gml:resultOf>
    <gml:Array>
      <gml:members>
        <point_value>3.24</point_value>
        <point_value>3.88</point_value>
      </gml:members>
    </gml:Array>
  </gml:resultOf>
  <!--There can be any number of dependent parameters.  -->
  <dependent_param xlink:href="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/" pt_code="TEMP">
    <gml:Array>
      <gml:members>
        <point_value>4.653</point_value>
        <point_value>4.482</point_value>
      </gml:members>
    </gml:Array>
  </dependent_param>
</data_point>
```

Figure 17. The data profile is contained in the <data_point> element. This element contains bounding data for the independent parameter, time of the profile, instrument information and data values.

The <gml:resultOf> element is used to store the data members of the profile. Since the present implementation has a single independent variable, the independent variable is stored in the <gml:resultOf> element as a series of point values in the <point_value> element. In this implementation, <point_value> is simply a substitution of the gml:_Object.

```
<gml:using xlink:type="simple [0..1]" xlink:href="[0..1]"
xlink:role="[0..1]" xlink:arcrole="[0..1]" xlink:title="[0..1]"
xlink:show="[0..1]" xlink:actuate="[0..1]" gml:remoteSchema="[0..1]">
   <gml:_Feature> ... </gml:_Feature> [0..1]
</gml:using>
```

Figure 18. The <gml:using> element definition. A single gml:_Feature type is permitted in <gml:using>.

Implicit in the implementation is the decision to use the <gml:resultOf> element to store only the independent variable values. If multiple independent variables were in the data stream, the multiple values could be stored in the <point_value> element (in the <gml:resultOf>) using separators (e.g., Figure 19). In this case, the *title* attribute of the <gml:using> element would need to include the multiple codes. This could be implemented in the comma-separated format in the same order as the data values. The present GML structure does not allow multiple <gml:Array> or <gml:members> elements.

```
<gml:resultOf>
  <gml:Array>
    <gml:members>
      <point_value>3.24,44.520</point_value>
      <point_value>3.88,44.522</point_value>
    </gml:members>
  </gml:Array>
</gml:resultOf>
```

Figure 19. Multiple independent variables may be stored in point_value element as shown.

For this implementation, the dependent variables are stored in the <dependent_param> element (this element was defined in Figure 16). It is worth noting that the <gml:resultOf> element could be used to store both the independent and dependent variable values. Again, a comma-separated list would be utilized with multiple codes stored in the *title* attribute. This structure would eliminate the need for the <dependent_param> element.

The <dependent_param> element is included here to illustrate a method of separating variable values within the structure. For variable separation, each <dependent_param> element would contain a single parameter code in the *pt_code* attribute, and a series of <point_value> elements containing the data. Alternately, one <point_value> element could be used with the data in a comma-separated form.

However, there is no need to include the <dependent_param> element if one wants to utilize comma-separated values. A complete record (i.e., independent and dependent variables) of variable content could be stored in the <point_value> element in the <gml:resultOf> element.

The <dependent_param> type references the GML AssociationType. This type is based on the gml:_Object. This specification allows the use of the <gml:Array>, <gml:members>, and <point_value> structure. This structure will be used to store the temperature values for our example implementation (Figure 17). Again multiple dependent variables can be described using a similar structure as shown in Figure 19.

The *pt_code* attribute in the <dependent_param> element is similar to the *pt_code* attribute in the initial XML Brick implementation [3]. However, in the GML version we actually indicate the parameter dictionary using the *xlink:href* attribute. This provides the ability to use more than one parameter dictionary in a single GML instance document. The initial Brick implementation assumed one dictionary per instance document.

4. Issues Implementing Bricks in GML

There are many issues surrounding the use of GML for transfer of oceanographic data. Many of the specific issues were noted in the main text of this report. What follows in this section, are the more general issues related to the comparison between the initial XML Brick and GML implementations.

4.1.1 Parsing versus File Size

The implementation of both the dependent and independent data values causes a parsing problem that was not present in the initial brick implementation. In the initial implementation, an individual data value was stored in an individual XML element. This allows standard XML parsers to parse the instance document to nodes that contain the individual data values.

In the GML implementation, the node content will not contain individual values. Rather, the nodes will contain string content that represents a comma-separated record of values. This content would need further parsing by application software to obtain the individual data values. Similar additional parsing would be needed in the *title* attribute content to determine the parameter codes for the record structure. The <gml:coordinates> content within a bounding box also shows this problem.

The issue of compacted records verses individual values relates directly to XML document file size. XML tags use considerable space in any XML instance document. By combining the data values into a comma-separated format, the file size will be reduced substantially as compared to the initial Brick implementation. However, if the individual values are required, each application requiring those values will need additional code to parse the instance content.

4.1.2 Nomenclature

An initial problem during the construction of the GML data file was the nomenclature. For someone familiar with the terms and naming conventions associated with the ocean data community, the conversion to terms utilized in the geography community is a non-trivial exercise. The GML documentation is also very generalized adding to the difficulty. Finally, the size of the GML documentation is somewhat intimidating, being about 530 pages in total.

4.1.3 GML Schema Validation

The inability of standard XML software to validate the GML schemas is a cause for concern. The coverage.xsd schema would not validate using numerous XML validators. In communication with GML authorities [16], it was recognized that the existing validator implementations are often inconsistent.

Although the validators may be the cause of the validation problems, the inability to validate does cause suspicion regarding the schemas and thereby makes one question implementing the structures used in these schemas. OGC would be wise to work with software vendors and W3C to clarify any inconsistencies in the XML schema specification and software implementation.

4.1.4 Content Control versus Specification Acceptance

This is a common issue when considering standards for implementation. Should one accept the standard and thereby limit themselves to the data representation described by the standard, or should one develop and use a solution that is tailored to the particular problem being considered.

In the GML case, the standard provides much of the functionality required for ocean datasets. In those cases when the functionality is not in the standard, extensions can enhance the structure to include those elements that are required by the uniqueness of the ocean datasets.

The Brick implementation provides the counter example. In this case, the Brick content is well defined. The external structure constructed from the Bricks is dependent on the dataset. The structure could also be expanded to include other content particular to the exchanging parties.

Ultimately, the question relates to the use of the packaged data. If the transferred data is intended for a Geographical Information System (GIS), then a strong argument exists for the use of GML. However, if the transfer is intended to provide concise and clear data to users familiar with ocean datasets, then the Keeley Brick approach may be more applicable. The international community will need to critically examine both aspects of this problem.

5. Concluding Remarks

When clients or systems request data, there are two conditions that must be met for the successful delivery of the data. First, the delivered data must contain the data required by the requesting client. Second, the supplier must deliver the data in a form that is understandable by the requesting client. If the client understands the data structure, then it is possible for the client to manipulate the structure to make the data usable in other systems or applications to meet particular needs.

The latter point of understanding the structure agrees well with the sixth design goal in the XML specification [1], which states;

"XML documents should be human-legible and reasonably clear."

XML provides the global community with the ability to define the data structures and deliver the datasets that address these conditions. As a result, communities of interest have developed structures based on the XML specification. In this investigation, the ocean data community has investigated the use of a geography-based XML language, GML, and compared this to a previous investigation that implemented an XML language based on the Keeley Bricks.

One of the primary difficulties related to the use of GML in an ocean data context is nomenclature. One of the strengths of XML is that it provides communities of interest with the ability to define objects with community specific names and structures that are well known within the community. If the community utilizes an XML structure with naming conventions unfamiliar to the community, then some of the benefits of the language are lost.

One advantage of GML is that it is a widely recognized specification. The North Atlantic Treaty Organisation recognizes GML as an emerging standard, and as such is monitoring its evolution. NATO considers GML potentially useful for the exchange of geographic data.

A general comparison of GML and the XML Brick implementation is provided in Table 3. The comparison is based solely on the findings presented in this report, rather than an all-inclusive comparison. The table lists several attributes or functionality that may be used in the comparison between the two implementations. The table illustrates the complexity associated with deciding between implementations. In reality, the functional requirements of the oceanographic data management problem need to be defined and each implementation compared to the functional requirements.

The efforts toward an international oceanographic data exchange structure continue to evolve. However, the community must recognize that the investigations of alternate structure representations could be a never-ending process. International groups must start to make the more difficult decisions to actually build a consensus and commitment around a specific representation. The commitment must drive the

adoption process. Only through use, will the full potential of any resulting standard be realized.

Table 3. A summary comparison between the GML implementation and the XML Brick implementation for oceanographic data. Present capabilities are used to define implementation characteristics.

ATTRIBUTE	GML IMPLEMENTATION	XML BRICK IMPLEMENTATION
Compact record structure for data content	Yes	No
Parser access to individual data values	No	Yes
Recognized Standard	No[1]	No
Capable of defining a spatial bounding box	Yes	No
Validation of latitude - longitude values	No	Yes
Nomenclature (Tag names represent common oceanographic terms)	No	Yes
Date/Time specification based on ISO 8601 standard	Yes	Yes
Multiple data dictionary support in a single XML document	Yes	No[2]
Schema(s) validate	No	Yes
Complex schema structure	Yes	No
Allows parameter definition within the XML document	No[3]	Yes

1: GML has been submitted to ISO for consideration. It is presently at the committee comment stage under Technical Committee (TC) 211. It is identified as ISO 19136.

2: Two dictionaries cannot be defined in a single instance document. However, variable definitions can be included to describe parameters not in the defined dictionary.

3: The GML specification allows dictionary definitions but does not appear to support the requirements of an oceanographic parameter dictionary.

6. References

1. WC3. 2004. XML specification at http://www.w3.org/TR/2004/REC-xml-20040204/, (March 2004).

2. Cox, Simon, Paul Daisey, Ron Lake, Clemens Portele and Arliss Whiteside. 2003. OpenGIS Geography Markup Language (GML) Implementation Specification, January 29, 2003, Document number OGC 02-023r4, Version 3.00, p 529.

3. Isenor, Anthony W., J. Robert Keeley and Joe Linguanti. 2003. Developing an eXtensible Markup Language (XML) Application for DFO Marine Data Exchange via the Web. (DRDC Atlantic ECR 2003-025). Defence R&D Canada – Atlantic.

4. Isenor, Anthony W. 2003. Canadian Investigations of the Keeley Bricks With Application to Profile Data Transfer Using XML, DRDC Atlantic SL 2003-029, Defence R&D Canada – Atlantic. Published in Meeting Report for the ICES-IOC Study Group on the Development of Marine Data Exchange Systems Using XML, Gothenburg, Sweden, May 26-27, 2003, 115-131.

5. SGXML, 2002. Report of the ICES-IOC Study Group on the Development of Marine Data Exchange Systems Using XML, Helsinki, Finland, 15–16 April 2002, ICES CM 2002/C:12, p 59.

6. Ray, Erik T. 2001. Learning XML, 2nd Edition. O'Reilly and Associates Inc., January 2001, p 416.

7. Walsh, Norman. A Technical Introduction to XML, http://www.xml.com/lpt/a/98/10/guide0.html, (March 2004).

8. The Annotated XML Specification, see http://www.xml.com/axml/testaxml.htm, (March 2004).

9. Isenor, Anthony W. 2003. XML Based Manipulation of Codes Exchanged Between Data Systems. (DRDC Atlantic TM 2003-132). Defence R&D Canada – Atlantic.

10. WC3, XML Schema specification primer at http://www.w3.org/TR/xmlschema-0/, (March 2004).

11. van der Vlist, Eric. 2002. XML Schema, O'Reilly and Associates Inc., p 379.

12. WC3. 1999. XPath specification at http://www.w3.org/TR/xpath, (March 2004).

13. OGC, http://www.opengis.org/, (March 2004).

14. WC3. 2001. SMIL specification at http://www.w3.org/TR/smil20/, (March 2004).

15. WC3. 2001. XLink specification at http://www.w3.org/TR/2001/REC-xlink-20010627/, (March 2004).

16. Cox, Simon. CSIRO Exploration & Mining. Lead of the OGC GML Revision Working Group. February 2004. Personal Communication.

17. ISO 8601 Standard, see http://www.iso.ch/iso/en/prods-services/popstds/datesandtime.html, (March 2004).

18. Allied Data Publication 34 (ADatP-34), NATO Technical Architecture, Volume 1 Management, December 15, 2003, p 45.

19. Allied Data Publication 34 (ADatP-34), NATO Technical Architecture, Volume 2 Architectural Descriptions and Models, December 15, 2003, p 53.

20. Allied Data Publication 34 (ADatP-34), NATO Technical Architecture, Volume 3 Base Standards and Profiles, December 15, 2003, p 348.

21. Allied Data Publication 34 (ADatP-34), NATO Technical Architecture, Volume 4 NC3 Common Standards Profile, December 15, 2003, p 67.

22. ISO. 2000. Geographic information – Metadata, ISO 19115-3.

23. Isenor, Anthony W. 2003. The Importance of Metadata in System Development and IKM. (DRDC Atlantic TM-2003-011). Defence R&D Canada – Atlantic, February 2003.

Annex 1: GML as an Emerging Standard in NATO

NATO attempts to utilize community-of-practice developments by actively evaluating existing products that cover a wide relevance. GML is one product included in the NATO geography data category as an emerging standard that is being monitored as it evolves. However, GML is only one product being monitored in the geography data category. To fully appreciate GMLs place in the NATO data categories, we must first appreciate the NATO structure that supports interoperability requirements.

As an organisation, NATO is very involved in specifying interoperability requirements for the participating partners. The NATO Consultation, Command and Control Board (NC3B) is tasked with establishing policy for achieving interoperability goals. Directives and guidance based on the policy is managed by a subgroup under NC3B, specifically the Information Systems Sub-Committee (ISSC). Under the ISSC is the NATO Data Administration Group (NDAG). NDAG is involved in establishing the specifications, working definitions, and enabling technologies for data interoperability. For a complete management description, see [18].

Part of the NDAG effort has been in establishing the NATO Consultation, Command and Control Technical Architecture (NC3TA). The NC3TA represents the technical rules for development procedures, concepts and data standards. These rules are to be applied to NATO developments, which focus on interoperability. In essence, the rules help ensure the development meets the requirements for a NATO View Model.

The NC3TA represents an enabler technology, and falls under the responsibility of the ISSC. The NC3TA was developed by the NATO Open Systems Working Group, a sub-group under the ISSC. The NC3TA is structured into a five-volume set of documentation.

Part of the NC3TA is the NATO Technical Reference Model (NTRM). The NTRM provides a structure for common standards used in NATO [19]. The NTRM is structured into five areas of applicability, with the intention to separate data, applications and platforms. These five areas are related to entities (three areas of applicability) and the interfaces (two areas of applicability) between these entities:

- Application Software Entity
- Application Platform Entity
- External Environment Entity
- Application Program Interface
- External Environment Interface

In the Application Platform Entity, there are numerous service areas identified in the NTRM. These services areas are:

- user interface services,
- data management services,
- data interchange services,
- graphical services,
- communication services, protocols and profiles,
- operating system services,
- internationalisation,
- system management services,
- security services,
- distributed computing services, and
- software engineering services.

Of particular interest in this study are the base standards for data interchange services. These services support the interchange of data between applications on heterogeneous computers or systems. GML is included in this service area because it supports the transport, or interchange, of data between applications.

The focus in this service category is toward those standards that permit easy manipulation by Internet browsers. Here, manipulation is defined as the compression, decompression, processing and display of data [20]. The categories of data interchange services [21] covered by the publication include:

- graphics,
- video/audio,
- text/data,
- tactical information links,
- tactical messages,
- technical business,

- geographical data,
- documents,
- file compression and archive,
- military symbology,
- encoding, and
- hardware.

The geographic data category includes a number of standards or specifications that detail structures for data exchange. Included in the geographic data exchange standards [20] are:

- Digital Feature Analysis Data (DFAD),
- Digital Geographic Information Exchange Standard (DIGEST v2.0 and v2.1),
- Digital Nautical Chart (DNC),
- Digital Terrain Elevation Data (DTED),
- Environmental Data Coding Specification (EDCS),
- Geospatial Symbols for Digital Displays (GeoSym),
- Geographical Tagged Image Format (GeoTIFF),
- GML,
- ISO 19115,
- Source for Environmental Representation and Interchange (SEDRIS),
- Spatial Reference Model (SRM),
- STANAG 2211,
- STANAG 7170,
- Vector Product Format (VPF),
- Standard for Warship Electronic Chart Display and Information System (WECDIS), and

- World Geodetic System 84 (WGS 84).

Several points are worth noting regarding the named list. First, note that the International Organisation for Standardisation (ISO) specification 19115 is in this list [22, 23]. This specification is from ISO Technical Committee 211 (TC 211) and defines the information set required for describing digital geospatial data. The standard has been investigated from the perspective of comparison to other structures for geospatial metadata [23].

Also of note are the STANAG (Standardization Agreement) specifications. As an agreement, the STANAGs may or may not actually contain detailed specifications.

In the category for data interchange services, all NATO standards are listed as either emerging or mandatory. The *mandatory status* indicates that the standard has received unanimous consensus of participating nations in tests involving 10 criteria [21] that includes the standardisation requirement, legacy issues, procurement issues, etc. The *emerging status* indicates that it is expected to become a mandatory standard, but at the moment does not meet all the required criteria. The transformation from emerging to mandatory typically involves the evolution of the standard via request-for-change proposals.

It is worth noting the present mandatory standards in the geographical category. At the moment, DFAD, DIGEST v2.0, DTED, STANAG 2211 and the World Geodetic System 84 are the mandatory standards. GML is included in the geographical data category and is listed as an emerging standard that specifically supports interoperability.

Annex 2: Keeley Brick Instance of XBT Data

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<data_collection xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xsi:noNamespaceSchemaLocation="c:\Anthony\Projects\xml\odf_conve
rsion\bricks_v2.xsd">
  <data_dictionary>
    <dictionary_name>GF3+</dictionary_name>
  </data_dictionary>
  <data_set>
    <data_set_id level="cruise">97025</data_set_id>
    <provenance>
      <agency>Bedford Institute</agency>
      <date_created>2003-02-18Z</date_created>
      <institute_code>1899</institute_code>
      <originator_identifier>97025</originator_identifier>
      <project>Shark Fishery</project>
    </provenance>
    <data_set>
      <data_set_id level="station">240</data_set_id>
      <data_set>
        <data_set_id level="profile"/>
        <provenance>
          <agency>Bedford Institute</agency>
          <data_grouping>XBT</data_grouping>
          <date_created>2003-02-18Z</date_created>
        </provenance>
        <variable_set>
          <instrument type="XBT">
            <description>SOC BT/SV Processor</description>
            <manufacturer>XBT</manufacturer>
          </instrument>
          <sampling pt_code="DEPH">
            <interval>-99.00000</interval>
          </sampling>
          <units pt_code="DEPH" stored_units="metres"/>
          <variable kind="I" pt_code="DEPH" typing="R">
            <decimal_places>2.000000</decimal_places>
            <maximum_value>589.88</maximum_value>
            <minimum_value>3.24</minimum_value>
            <null_value>-99.00</null_value>
            <variable_name>Sensor Depth below Sea
Surface</variable_name>
          </variable>
        </variable_set>
        <variable_set>
          <instrument type="XBT">
            <description>SOC BT/SV Processor</description>
            <manufacturer>XBT</manufacturer>
          </instrument>
```

```
            <units pt_code="TEMP" stored_units="degrees C"/>
            <variable kind="D" pt_code="TEMP" typing="R">
              <decimal_places>3.000000</decimal_places>
              <maximum_value>10.483</maximum_value>
              <minimum_value>4.15</minimum_value>
              <null_value>-99.00</null_value>
              <variable_name>Sea Temperature</variable_name>
            </variable>
          </variable_set>
          <location_set>
            <ldate property="creation">
              <pdate>2003-02-18Z</pdate>
              <ptime>19:33:44.14Z</ptime>
            </ldate>
          </location_set>
          <location_set>
            <ldate property="original">
              <pdate>2000-08-23Z</pdate>
              <ptime>14:27:45.3Z</ptime>
            </ldate>
          </location_set>
          <location_set>
            <latitude property="start">42.75000</latitude>
            <ldate property="start">
              <pdate>1999-02-28Z</pdate>
              <ptime>09:02:00Z</ptime>
            </ldate>
            <longitude property="start">-62.91670</longitude>
          </location_set>
          <location_set>
            <latitude property="end">42.75000</latitude>
            <longitude property="end">-62.91670</longitude>
          </location_set>
          <history_set>
            <comment>Conversion</comment>
            <history>
              <application_date>2000-08-23Z</application_date>
            </history>
          </history_set>
          <history_set>
            <comment>GF3 Name Checking and Code
Formatting</comment>
            <comment>Name check performed</comment>
            <comment>Edit Cruise: Country_Institute_Code=1899
Organization=Clearwater Foods Cruise_Name=Shark Fishery
Cruise_Number=96999</comment>
            <comment>Edit Instrument: Inst_Type=XBT
Description=SOC BT/SV Processor</comment>
            <comment>Edit Event: Data_Type=XBT Event_Number=99
Initial_Latitude=42.9167 Initial_Longitude=-62.3333
End_Latitude=42.9167 End_Longitude=-62.3333 Start_Date_Time=10-
MAR-1996 06:31:00.00 End_Date_Time=10-MAR-1996
06:31:00.00</comment>
            <comment>The code of Parameter `Sensor Depth below Sea
```

```
Surface´ will be changed to: `DEPH_01´.</comment>
          <comment>Add_Parameter: A.Data.DEPH_01 =
evalin(</comment>
          <comment>The code of Parameter `Sea Temperature´ will
be changed to: `TEMP_01´.</comment>
          <comment>Add_Parameter: A.Data.TEMP_01 =
evalin(</comment>
          <history>
            <application_date>2000-08-23Z</application_date>
          </history>
        </history_set>
        <history_set>
          <comment>GF3 Name Checking and Code
Formatting</comment>
          <comment>Name check performed</comment>
          <comment>Edit Cruise: Country_Institute_Code=1899
Organization=Clearwater Foods Cruise_Name=Shark Fishery
Cruise_Number=97025</comment>
          <comment>Edit Instrument: Inst_Type=XBT
Description=SOC BT/SV Processor</comment>
          <comment>Edit Event: Data_Type=XBT Event_Number=240
Initial_Latitude=42.75 Initial_Longitude=-62.9167
End_Latitude=42.75 End_Longitude=-62.9167 Start_Date_Time=28-
FEB-1999 09:02:00.00 End_Date_Time=28-FEB-1999
09:02:00.00</comment>
          <history>
            <application_date>2000-08-24Z</application_date>
          </history>
        </history_set>
        <history_set>
          <comment>GF3 Name Checking and Code
Formatting</comment>
          <comment>Name check performed</comment>
          <history>
            <application_date>2003-02-18Z</application_date>
          </history>
        </history_set>
        <data_set>
          <data_point pt_code="min_depth">3.24000</data_point>
          <data_point pt_code="max_depth">589.88000</data_point>
          <data_set_id level="related"/>
          <variable_set>
            <instrument type="XBT"/>
            <units pt_code="max_depth" stored_units="metres"/>
            <variable kind="I" pt_code="max_depth" typing="R">
              <decimal_places>5.000000</decimal_places>
              <variable_name>maximum depth</variable_name>
            </variable>
          </variable_set>
          <variable_set>
            <instrument type="XBT"/>
            <units pt_code="min_depth" stored_units="metres"/>
            <variable kind="I" pt_code="min_depth" typing="R">
              <decimal_places>5.000000</decimal_places>
```

```
            <variable_name>minimum depth</variable_name>
          </variable>
        </variable_set>
      </data_set>
      <data_set>
        <data_point pt_code="DEPH">3.24</data_point>
        <data_point pt_code="TEMP">4.653</data_point>
        <data_set_id level="record"/>
      </data_set>
      <data_set>
        <data_point pt_code="DEPH">3.88</data_point>
        <data_point pt_code="TEMP">4.482</data_point>
        <data_set_id level="record"/>
      </data_set>
    </data_set>
  </data_set>
</data_set>
</data_collection>
```

Annex 3: Schema Implementation

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds" xmlns:kb="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" version="3.0">
  <annotation>
    <appinfo>XML Keeley Bricks Expressed in GML.  Brick Schema
Version 2.0. 2004-Feb-23</appinfo>
    <documentation xml:lang="en">
    </documentation>
  </annotation>
  <import namespace="http://www.opengis.net/gml"
schemaLocation="c:\Anthony\International\Standards\GML\gml3.0\ba
se\observation.xsd"/>

  <!--
==========================================================
       global element declarations
==========================================================
=== -->
  <element name="data_collection"
type="kb:data_collection_brick"
substitutionGroup="gml:_FeatureCollection"/>

  <!--
==========================================================
       type definitions for Keeley Brick Definitions
==========================================================
=== -->
  <complexType name="data_collection_brick">
    <complexContent>
      <extension base="gml:AbstractFeatureCollectionType">
        <sequence>
          <element name="ldate" type="gml:TimeInstantType"
minOccurs="0" maxOccurs="unbounded"/>
          <element name="data_set" type="kb:data_set_brick"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="data_set_brick">
    <complexContent>
      <extension base="gml:AbstractFeatureCollectionType">
        <sequence>
          <element name="ldate" type="gml:TimeInstantType"
minOccurs="0" maxOccurs="unbounded"/>
```

```
          <element name="data_point" type="kb:data_point_brick"
minOccurs="0" maxOccurs="unbounded"/>
          <element name="data_set" type="kb:data_set_brick"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="data_point_brick">
    <complexContent>
      <extension base="gml:ObservationType">
        <sequence>
          <element name="dependent_param"
type="kb:dependent_param_construction" minOccurs="1"
maxOccurs="unbounded"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>

  <element name="point_value" substitutionGroup="gml:_Object"/>
  <element name="metadata_value" type="gml:GenericMetaDataType"
substitutionGroup="gml:_MetaData"/>
  <element name="instrument" type="kb:instrument_brick"
substitutionGroup="gml:_Feature"/>
  <element name="additional_metadata" type="kb:instrument_brick"
substitutionGroup="gml:_Feature"/>

  <complexType name="dependent_param_construction">
    <complexContent>
      <extension base="gml:AssociationType">
        <attribute name="pt_code"/>
      </extension>
    </complexContent>
  </complexType>

  <complexType name="instrument_brick">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
      </extension>
    </complexContent>
  </complexType>
</schema>
```

Annex 4: GML Instance of XBT Data

```
<?xml version="1.0" encoding="UTF-8"?>
<data_collection xmlns="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/Prog_Int/ICES"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/Prog_Int/ICES
file:///c:/Anthony/International/Standards/GML/gml3.0/bricks_in_
GML_3.xsd">
  <gml:description>A test application to convert XML Keeley
Brick definitions to GML.  Note that I initially wanted to use
the coverage.xsd schema, but changed to observation.xsd
because of validation problems.  The validators currently
available cannot agree on whether or not coverage.xsd is valid.
Given my limited knowledge of the schema specification, and
the complexities in the gml set of schemas, I cannot judge the
validators for correctness.


  Anthony W. Isenor    -   February 2004</gml:description>
  <gml:name>Data set</gml:name>
  <gml:boundedBy>
    <gml:Envelope>
      <gml:description>Bounding box for
collection.</gml:description>
      <gml:coordinates>45.0,-56.4 46.0,-56.4 46.0,-50.3
45.0,-50.3</gml:coordinates>
    </gml:Envelope>
  </gml:boundedBy>
  <!--  Here the individual metadata items are defined using a
namelist at some established centre.
  The gml:name element uses a name from the namelist.  The
gml:description element is used for the content.  For example,
the "Agency" is "BIO".-->
  <gml:featureMembers>
    <additional_metadata>
      <gml:description>BIO</gml:description>
      <gml:name
codeSpace="www.meds.ca/namelist">Agency</gml:name>
    </additional_metadata>
    <additional_metadata>
      <gml:description>1899</gml:description>
      <gml:name codeSpace="www.meds.ca/namelist">Institute
Code</gml:name>
    </additional_metadata>
    <additional_metadata>
      <gml:description>2003-02-18Z</gml:description>
      <gml:name codeSpace="www.meds.ca/namelist">Date
```

```
Created</gml:name>
    </additional_metadata>
    <additional_metadata>
      <gml:description>Shark Fishery</gml:description>
      <gml:name
codeSpace="www.meds.ca/namelist">Project</gml:name>
    </additional_metadata>
    <additional_metadata>
      <gml:description>97025</gml:description>
      <gml:name
codeSpace="www.meds.ca/namelist">Identifier</gml:name>
    </additional_metadata>
  </gml:featureMembers>
  <!--  Multiple ldates may be use to specify important dates.
Here, the gml:name element is again used to indicate a namelist
where "Start" would be defined.  The timePosition element would
contain the date corresponding to the content in the gml:name
element.-->
  <ldate>
    <gml:description>Start date of the
collection.</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">Start</gml:name>
    <gml:timePosition>2003-02-18Z</gml:timePosition>
  </ldate>
  <ldate>
    <gml:description>End date of the
collection.</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">End</gml:name>
    <gml:timePosition>2003-03-18Z</gml:timePosition>
  </ldate>
  <!--  Here, dataset is used to describe the top level cruise
information.  A bounding box must be defined for the cruise, as
boundedBy element is mandatory.  boundedBy is the only mandatory
element.-->
  <data_set>
    <gml:description>This is the cruise level of the Keeley
Brick structure.</gml:description>
    <gml:name codeSpace="www.meds.ca/namelist">cruise</gml:name>
    <gml:boundedBy>
      <gml:Envelope>
        <gml:description>This is a bounding box for the
cruise.</gml:description>
        <gml:name>97025</gml:name>
        <gml:coordinates>45.0,-56.4 46.0,-56.4 46.0,-55.3 45.0,-
55.3</gml:coordinates>
      </gml:Envelope>
    </gml:boundedBy>
    <data_set>
      <!--The next level describes the station.-->
      <gml:description>This is the station level of the Keeley
Brick structure.</gml:description>
      <gml:name
codeSpace="www.meds.ca/namelist">station</gml:name>
      <gml:boundedBy>
```

```
        <gml:Envelope>
          <gml:description>This is the station position, but
described as some general name: e.g. Station Bravo, or Charlie.
In this case, the position could be the agreed position of the
station.</gml:description>
          <gml:name>Station 35</gml:name>
          <gml:coordinates>42.7,-63.0</gml:coordinates>
        </gml:Envelope>
      </gml:boundedBy>
      <data_set>
        <!--Next, the processing history is included in the
gml:description element.-->
        <gml:description>GF3 Name Checking and Code Formatting
          Name check performed
          Edit Cruise: Country_Institute_Code=1899
Organization=Clearwater Foods Cruise_Name=Shark Fishery
Cruise_Number=96999
          Edit Instrument: Inst_Type=XBT Description=SOC BT/SV
Processor
          Edit Event: Data_Type=XBT Event_Number=99
Initial_Latitude=42.9167 Initial_Longitude=-62.3333
End_Latitude=42.9167 End_Longitude=-62.3333 Start_Date_Time=10-
MAR-1996 06:31:00.00 End_Date_Time=10-MAR-1996 06:31:00.00
          The code of Parameter `Sensor Depth below Sea Surface´
will be changed to: `DEPH_01´.
          Add_Parameter: A.Data.DEPH_01 = evalin(
          The code of Parameter `Sea Temperature´ will be
changed to: `TEMP_01´.
          Add_Parameter: A.Data.TEMP_01 = evalin(

          GF3 Name Checking and Code Formatting
          Name check performed
          Edit Cruise: Country_Institute_Code=1899
Organization=Clearwater Foods Cruise_Name=Shark Fishery
Cruise_Number=99999
          Edit Instrument: Inst_Type=XBT Description=SOC BT/SV
Processor
          Edit Event: Data_Type=XBT Event_Number=240
Initial_Latitude=42.75 Initial_Longitude=-62.9167
End_Latitude=42.75 End_Longitude=-62.9167 Start_Date_Time=28-
FEB-1999 09:02:00.00 End_Date_Time=28-FEB-1999 09:02:00.00

          GF3 Name Checking and Code Formatting
          Name check performed</gml:description>
        <gml:name>profile</gml:name>
        <gml:boundedBy>
          <gml:Envelope>
            <gml:description>This is the station position and
the station number for the start of actual data collection.  The
position could be the actual position, as opposed to the agreed
position of the repeating station.</gml:description>
            <gml:name>240</gml:name>
            <gml:coordinates>42.75000,
-62.91670</gml:coordinates>
```

```
          </gml:Envelope>
        </gml:boundedBy>
        <gml:featureMember>
          <additional_metadata>
            <gml:boundedBy>
              <gml:Envelope>
                <gml:description>This is the station position
and the station number for the END of actual data collection.
The position could be the actual position, as opposed to the
agreed position of the repeating station.</gml:description>
                <gml:name>240</gml:name>
                <gml:coordinates>42.75000,
-62.91670</gml:coordinates>
              </gml:Envelope>
            </gml:boundedBy>
          </additional_metadata>
        </gml:featureMember>
        <data_point>
          <!--Some points to be made:
        - The timeStamp and resultOf are mandatory inside
data_point.
        - The using element defines the independent parameters.
The resultOf then contains the independent parameter values.
        - This single using element could be used to define more
than one indepenedent parameter (e.g. xlink:title="LATD,LOND").
Then the point_value element could be a comma separated list.
             (e.g. <point_value>32.4,-45.6</point_value>)
        -->

          <gml:boundedBy>
            <gml:Envelope>
              <gml:description>min/max depth of
profile</gml:description>
              <gml:name codeSpace="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/">min_depth</gml:name>
              <gml:name codeSpace="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/">max_depth</gml:name>
              <gml:coordinates>3.2400,589.8800</gml:coordinates>
            </gml:Envelope>
          </gml:boundedBy>

          <gml:timeStamp>
            <gml:TimeInstant>
              <gml:timePosition>1999-02-
28T09:02:00</gml:timePosition>
            </gml:TimeInstant>
          </gml:timeStamp>
          <gml:using xlink:href="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/" xlink:title="DEPH">
            <instrument>
              <gml:description>SOC BT/SV
Processor</gml:description>
              <gml:name>XBT</gml:name>
            </instrument>
```

```
          </gml:using>
          <gml:resultOf>
            <gml:Array>
              <gml:members>
                <point_value>3.24</point_value>
                <point_value>3.88</point_value>
              </gml:members>
            </gml:Array>
          </gml:resultOf>
          <!--There can be any number of dependent parameters.
-->
          <dependent_param xlink:href="http://www.meds-sdmm.dfo-
mpo.gc.ca/meds/About_MEDS/standards/" pt_code="TEMP">
            <gml:Array>
              <gml:members>
                <point_value>4.653</point_value>
                <point_value>4.482</point_value>
              </gml:members>
            </gml:Array>
          </dependent_param>
        </data_point>
      </data_set>
    </data_set>
  </data_set>
</data_collection>
```

List of symbols/abbreviations/acronyms/initialisms

ADatP	Allied Data Publication
ASCII	American Standard Code for Information Interchange
BIO	Bedford Institute of Oceanography
CORBA	Common Object Request Broker Architecture
CSIRO	Commonwealth Scientific & Industrial Research Organisation (Australia)
CTD	Conductivity-Temperature-Depth
DFAD	Digital Feature Analysis Data
DFO	Fisheries and Oceans
DIGEST	Digital Geographic Information Exchange Standard
DNC	Digital Nautical Chart
DND	Department of National Defence
DRDC	Defence R&D Canada
DTED	Digital Terrain Elevation Data
EDCS	Environmental Data Coding Specification
GeoSym	Geospatial Symbols for Digital Displays
GeoTIFF	Geographical Tagged Image Format
GIS	Geographic Information System
GML	Geography Markup Language
gml:	GML namespace
ICES	International Council for the Exploration of the Sea
IOC	Intergovernmental Oceanographic Commission

ISO	International Organisation for Standardisation
ISSC	Information Systems Sub-Committee
kb:	Keeley Brick namespace
NATO	North Atlantic Treaty Organisation
NC3B	NATO Consultation, Command and Control Board
NC3TA	NATO Consultation, Command and Control Technical Architecture
NDAG	NATO Data Administration Group
NTRM	NATO Technical Reference Model
NUW	Networked Underwater Warfare
OGC	Open GIS Consortium
SEDRIS	Source for Environmental Representation and Interchange
SGXML	ICES-IOC Study Group on the Development of Marine Data Exchange Systems Using XML (or the Study Group on XML)
SMIL	Synchronized Multimedia Integration Language
SRM	Spatial Reference Model
STANAG	Standardization Agreement
SQL	Structured Query Language
TC	Technical Committee
URI	Uniform Resource Identifier
VPF	Vector Product Format
W3C	World Wide Web Consortium
WECDIS	Standard for Warship Electronic Chart Display and Information System
WGS 84	World Geodetic System 84
XBT	eXpendable Bathythermograph

XLink	XML Linking Language
XML	eXtensible Markup Language
XPath	XML Path Language

Distribution list

Document No.: DRDC Atlantic TM 2004-087

LIST PART 1:

5 DRDC Atlantic LIBRARY FILE COPIES
1 M. E. LEFRANÇOIS
1 W. A. ROGER
1 G. R. MELLEMA
1 D.G. HAZEN
2 A. W. ISENOR (1 CD COPY, 1 HARD COPY)

11 TOTAL LIST PART 1

LIST PART 2: DISTRIBUTED BY DRDKIM 2-2-5

1 NDHQ/DRDKIM 2-2-5

1 Pekka Alenius
Finnish Institute of Marine Research
P.O.Box 33 (Asiakkaankatu 3)
FIN-00931 Helsinki
Finland

1 Keita Furukawa
Head, Marine Environment Division,
National Institute for Land and Infrastructure Management
3-1-1, Nagase, Yokosuka,
239-0826 Japan

1 Jean Gagnon
Marine Environmental Data Services Branch
Fisheries and Oceans
W12082 - 200 Kent
Ottawa, ON K1A-0E6

1 Robert D. Gelfeld
National Oceanographic Data Center
1315 East West Highway, Room 4230
Silver Spring, MD 20910-3282

1 Doug Gregory
Ocean Sciences Division
Bedford Institute of Oceanography
PO Box 1006
Dartmouth, N.S.
B2Y 4A2

1 Marta Gutierrez
CCLRC Rutherford Appleton Laboratory
Chilton, Didcot
Oxon, UK
OX11 0QX

1 Pieter Haaring
Ministry of Transport, Public Works and Water Management
National Institute for Coastal and Marine Management (RIKZ)
P.O. box 20907, 2500 EX The Hague,
The Netherlands

1 J. Robert Keeley
Marine Environmental Data Service
Department of Fisheries and Oceans
W12082 - 200 Kent Street
Ottawa, Ontario Canada
K1A 0E6

1 Kirstin Kleese-Van Dam
CCLRC Daresbury Laboratory
Daresbury, Warrington
Cheshire, UK
WA4 4AD

1 Janus Larsen
International Council for the Exploration of the Sea (ICES)
2-4 Palægade
DK-1261
Copenhagen K
Denmark

1 Sue Latham
CCLRC Rutherford Appleton Laboratory
Chilton, Didcot
Oxon, UK
OX11 0QX

1 Bryan Lawrence
CCLRC Rutherford Appleton Laboratory
Chilton, Didcot
Oxon, UK
OX11 0QX

1 Roy Lowry
British Oceanographic Data Center,
Bidston Observatory,
Bidston Hill, Prenton,
Merseyside CH43 7RA
UK

1 Kevin O'Neill
CCLRC Rutherford Appleton Laboratory
Chilton, Didcot
Oxon, UK
OX11 0QX

1 Lesley Rickards
Chair, IODE
British Oceanographic Data Center,
Bidston Observatory,
Bidston Hill, Prenton,
Merseyside CH43 7RA
UK

1 Tsuneki Sakakibara
University of Tokyo
Center for Spatial Information Science
Cw-502, 4-6-1, Komaba, Meguro-ku
Tokyo 153-8505
Japan

1 Tobias Spears
Informatics
Bedford Institute of Oceanography
PO Box 1006
Dartmouth, N.S.
B2Y 4A2

1 Andrew Woolf
CCLRC Rutherford Appleton Laboratory
Chilton, Didcot
Oxon, UK
OX11 0QX

19 TOTAL LIST PART 2

30 TOTAL COPIES REQUIRED

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence R&D Canada – Atlantic
PO Box 1012
Dartmouth, Nova Scotia
B2Y 3Z7

2. SECURITY CLASSIFICATION (overall security classification of the document including special warning terms if applicable).

UNCLASSIFIED

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title).

Describing Generic Ocean Environmental Data Objects Using the Geography Markup Language

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)

Anthony W. Isenor

5. DATE OF PUBLICATION (month and year of publication of document)	6a. NO. OF PAGES (total containing information Include Annexes, Appendices, etc).	6b. NO. OF REFS (total cited in document)
July 2004	53	23

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered).

Technical Memorandum

8. SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include address).

Defence R&D Canada – Atlantic
PO Box 1012
Dartmouth, NS, Canada B2Y 3Z7

9a. PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant).	9b. CONTRACT NO. (if appropriate, the applicable number under which the document was written).
11cg01	

10a ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity. This number must be unique to this document.)	10b OTHER DOCUMENT NOs. (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC Atlantic TM 2004-087	

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(x) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected).

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

This investigation examines the Geography Markup Language (GML) structure and its application to the transfer of one-dimensional ocean profile data (e.g., eXpendable Bathythermograph (XBT) data). GML was designed for the storage and transfer of geography-related data and is noted to be a NATO emerging standard for geographic data interchange. The investigation compares the GML-based structure for an XBT data profile to the same dataset stored in the Keeley Brick data structures. From the perspective of ocean data management, the results indicate that the GML nomenclature is difficult to understand. However, compared to the Keeley Brick approach GML provides benefits such as a standardized structure and efficiencies in file sizes.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title).

XML
extensible markup language
Keeley Bricks
GML
Geography Markup Language
ocean environmental data
Networked Underwater Warfare

This page intentionally left blank.

Defence R&D Canada

Canada's leader in defence and national security R&D

R & D pour la défense Canada

Chef de file au Canada en R & D pour la défense et la sécurité nationale

www.drdc-rddc.gc.ca