

Aeronautical Information Exchange Model (AIXM)

AIXM 5 Profile of GML

Copyright: 2006 - EUROCONTROL and Federal Aviation Administration

All rights reserved.

This document cannot be copied, in whole or in part, by anyone external to EUROCONTROL, United States Federal Aviation Administration (FAA) or United States National Geospatial Agency (NGA) without the expressed permission of EUROCONTROL or FAA. Any copy must retain the above copyright notice.

For all inquiries, please contact:

Brett BRUNK - brett.brunk@faa.gov

Eduard POROSNICU - eduard.porosnicu@eurocontrol.int

Edition No.	Issue Date	Author	Reason for Change
0.1	2006/06/22	Brett Brunk	First Edition

Contents

1	SCOPE	1
1.1	Introduction	1
1.2	References	1
2	AIXM PROFILE OF GML	1
2.1	GML Subset	1
3	XML ENCODING USING GML	2
3.1	Implications of GML's Object-Property model	2
3.2	AIXM Relationships	3
3.2.1	Naming convention	3
3.2.1.1	Feature associations	3
3.2.1.2	Feature complex properties	3
3.3	Feature identification in GML	3
3.3.1	Identification in GML 3.1.x	3
3.3.2	Expected GML 3.2 evolution	3
3.3.3	AIXM 5 implementation	4
3.4	Feature Relationships in GML	4
3.4.1	Resolving Xlinks	5
3.5	Null and optional feature property	5

1 Scope

1.1 Introduction

This document presents the AIXM profile of GML. The document contains two sections:

- Description of the AIXM profile of GML
- GML encoding conventions and style guidelines

1.2 References

1. Geographic Information – Spatial Schema. ISO 19107. First Edition, 2003-05-01
2. AIXM Profile of 19107. AIXM Technical Team. June 2006.
3. Geography Markup Language (GML). ISO/TC 211/WG 4/PT 19136 OGC GML RWG. Committee Draft. 2004-02-07.

2 AIXM Profile of GML

The AIXM profile of GML is based on the methodology described in Section 22 and Annex G of draft ISO 19136.

2.1 GML Subset

AIXM uses the following GML features and objects:

- gml:TargetPropertyType
- gml:ObservationType
- gml:AssociationType
- gml:_Feature
- gml:Geodesic
- gml:Arc
- gml:ArcByCenterPoint
- gml:CircleByCenterPoint
- gml:Ring
- gml:TimePeriod
- gml:TimeInstant
- gml:DynamicFeatureType
- gml:AbstractFeatureCollectionType
- gml:SurfacePropertyType
- gml:MeasureType
- gml:CodeType
- gml:Point
- gml:MultiPoint
- gml:LineString
- gml:LineStringSegment
- gml:Polygon
- gml:LinearRing
- gml:CompositeCurve
- gml:Curve
- gml:OrientableCurve
- gml:CompositeSurface
- gml:CompositeSurfacePropertyType

3 XML Encoding using GML

To the extent possible XML Encoding of AIXM is based on GML modeling styles. Some of the important concepts are discussed in the subsections below.

3.1 Implications of GML's Object-Property model

GML follows a structured object-property pattern: Features have one or more properties that encode the attributes or relationships of that feature. The content of a property is a value or another feature or complex object.

The GML object-property model changes the structure of complex AIXM properties such as the TimeTable. The structure of the TimeTable in AIXM 4.0 is:

```
<Vtt>
  <codeWorkHr>TIMSH</codeWorkHr>
  <Timsh>
    <codeTimeRef>UTCW</codeTimeRef>
    ...
  </Timsh>
  <Timsh>
    <codeTimeRef>UTCW</codeTimeRef>
    ...
  </Timsh>
  ...
</Vtt>
```

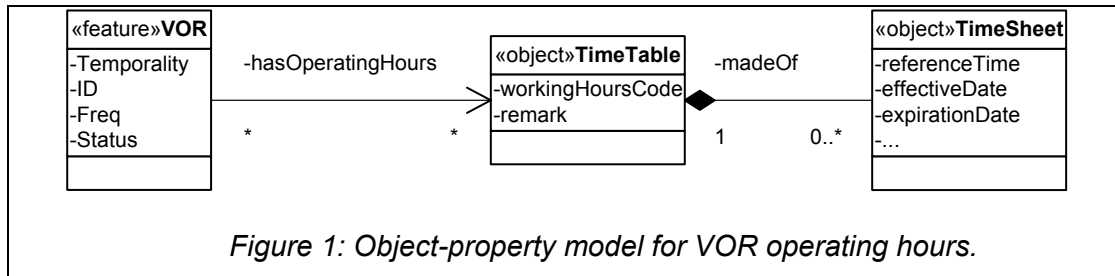
Where <Vtt> is the VOR operating hours Time Table

Following the GML object-property model the TimeTable changes as shown:

```
<hasOperatingHours>
  <TimeTable>
    <codeWorkHr>TIMSH</codeWorkHr>
    <madeOf>
      <TimeSheet>
        <codeTimeRef>UTCW</codeTimeRef>
        ...
      </TimeSheet>
      <TimeSheet>
        <codeTimeRef>UTCW</codeTimeRef>
        ...
      </TimeSheet>
    </madeOf>
  </TimeTable>
</hasOperatingHours>
```

Where <operatingHours> is the operating hours for the VOR.

In the object-property approach, the <operatingHours> property contains a complete TimeTable object. The TimeTable object contains a property called timeSheets that contains an array of TimeSheet objects. A UML diagram for the timeTable property is provided in Figure 1. Notice that the properties involved in the object-property relationship are encoded as UML associations between the parent object and the object referenced in the property.



3.2 AIXM Relationships

3.2.1 Naming convention

In AIXM, both attributes and relationships from the AICM UML model are encoded as feature properties.

3.2.1.1 Feature associations

The names of the feature properties that represent associations with other features will be constructed using the in three parts:

- The first part indicates the role of the relationship in lowerCamelCase.
- The second part is an underscore, “_”, separator
- The third part of the name of the feature type that is pointed to by the relationship.

For example, the starting point of a Route Segment would be written as “startingAt_SignificantPoint”.

In the AICM UML model the third part of the relationship name is omitted because it can be derived from the target of the relationship.

3.2.1.2 Feature complex properties

The names of the complex feature properties that are represented as associations with <<object>> in UML will be constructed using role of the relationship in lowerCamelCase.

For example, the timetable complex property of a VOR would be written as “hasOperatingHours”.

3.3 Feature identification in GML

3.3.1 Identification in GML 3.1.x

In GML 3.1.x all GML objects have an optional gml:ID. The gml:ID must be unique within an instance document. Ideally the gml:IDs are globally unique so that features are unique across distributed systems.

3.3.2 Expected GML 3.2 evolution

The GML 3.2 specification is expected to include the following:

- gml:ID will be mandatory and of type XSD type ID
- gml:ID will be the local feature identifier. That is, the gml:ID will be unique within the XML instance document.
- An optional property called gml:identifier will be used to provide globally unique identification on features. The gml:identifier will be a combination of context (e.g., namespace) along with an identification string.

3.3.3 AIXM 5 implementation

AIXM 5 will incorporate the recommended `gml:identifier` property on all features. The property will be used to globally identify aeronautical features. We recommend that the `gml:identifier` include the namespace of the data provider as well as an artificial key that is unique to the data provider.

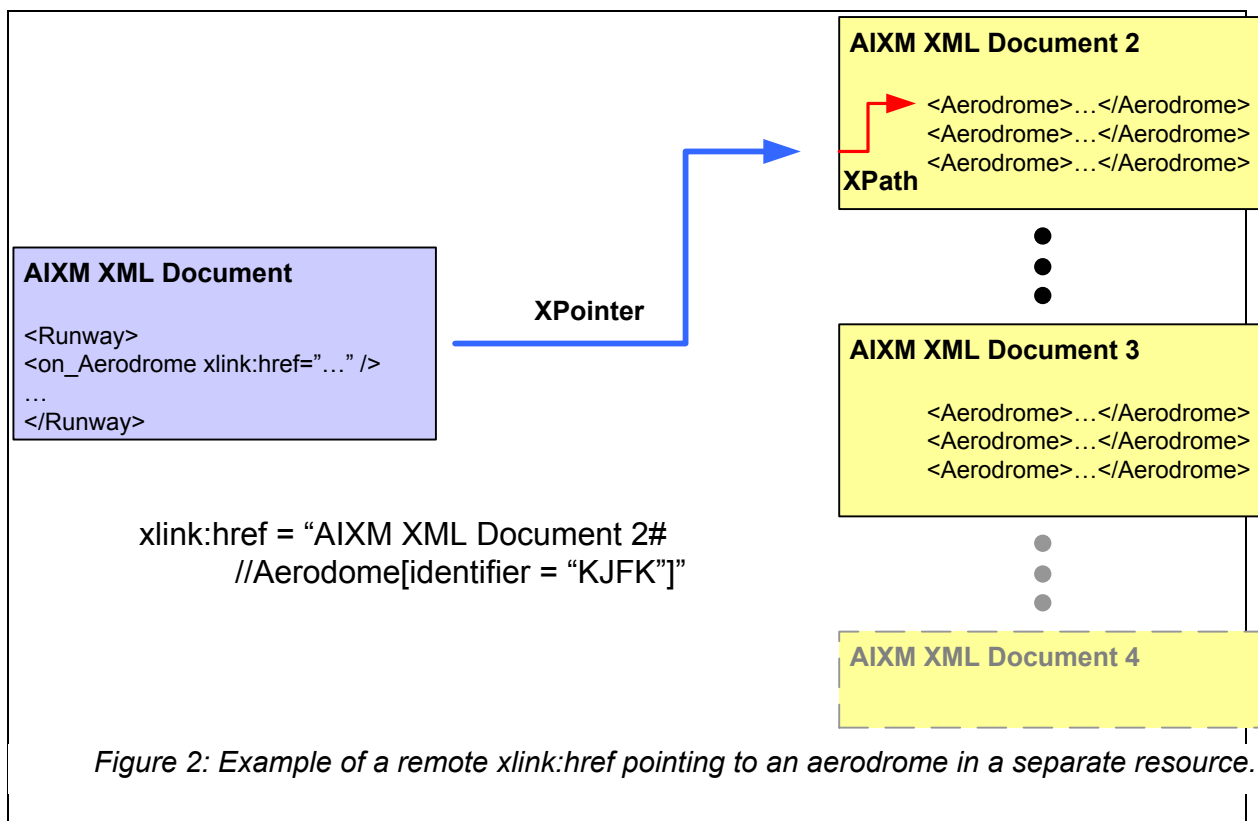
3.4 Feature Relationships in GML

In GML associations are created using properties. Association properties can point to the target object by either including the target object inline or through an `xlink:href`. The `xlink:href` specification supports the use of XPath and XPointer to identify XML elements in local or remote resources. For more on XLink, XPointer and XPath see <http://www.brics.dk/~amoeller/XML/linking/>,

At its most basic XLink has a format “XPointer#XPath” where

- XPointer points to a remote resource
- XPath locates a specific XML element within the context specified by the XPointer.

This concept is illustrated in Figure 2.



Xlink works directly and simply for artificial identifiers coded using the Feature identifier property. In addition, Xlink can accommodate natural identifying property in a complex but standard syntax.

For example a possible `xlink` to an Aerodrome located in the FAA's NASR database might look like:

```
xlink:href="http://www.faa.gov/nasr/database.xml#
//aixm:AerodromeHelicopter[
aixm:icaoCode="EDDF" and
aixm:position/gml:Point/gml:pos='50.0333 8.5704']"
```

A more complex example is the natural identification for a Runway. In this case we logically assume that the associations are inline when the xlink is created:

```
xlink:href='http://www.faa.gov/nasr/database.xml#
//aixm:Runway[
  aixm:designator="09/27" and
  aixm:on_AerodromeHeliport/AerodromeHeliport [
    aixm:icaoCode="EDDF" and
    aixm:position/gml:Point/gml:pos="50.0333 8.5704"
  ]
],
```

The xlink mechanism provides a standardized syntax for representing relationships between features using queries. As a result the XML document can accommodate the use of artificial identifiers and/or flexible use of natural properties to identify features.

3.4.1 Resolving Xlinks

With xlink encoding, the association property points to the location of the associated feature. Xlinks can be either local or remote.

A local xlink resolves to a location within the same AIXM XML document. Local xlinks are the easiest for AIXM document consumers to resolve since all the features and associated features are included within the same document. Of course, local xlinks can greatly increase the size of the transmitted document. Depending on the application requirements and the data requirements this can be a problem. For example, for a data file containing a version of an entire aeronautical database; local xlinks would be most efficient since all of the features must be included in the AIXM document anyway.

Remote xlinks can be used to reference AIXM features that are not included in the AIXM document. In order for an AIXM Document consumer to be able to resolve a remote xlink the consumer must either 1) already know about the features being remotely referenced or 2) the AIXM document supplier should provide a mechanism for the consumer to request information about the remote features.

An AIXM document may include both local and remote xlinks depending on the criticality of the data. Also there may be business cases where transmitting documents must always include locale copies of associated features. One can imagine safety critical applications where all referenced features must be transmitted so that the state of the system is unambiguously known.

Note that the mechanism used to resolve remote xlinks is beyond the scope of this standard. It is assumed that service level agreements between AIXM data providers and AIXM data consumers will document the data transmission approach used as well as protocols for requesting information on remotely referenced features.

3.5 Null and optional feature property

All AIXM 5 feature properties are optional and may be null. Optionality is required to support timeSlices where one or more feature properties may be changed. Null properties indicate that the feature property has not been provided.

AIXM 3.2 implements null values by adopting the XML schema `xsi:nil` attribute for all nullable feature properties. Using the `xsi:nil` flag means that null values can be explicitly transmitted.

For example, `<valElevation nilReason="unknown"></valElevation>` indicates that the `<valElevation>` property is null.

GML 3.2 also provides a `nilReason` attribute that can be used to refine how the null values are interpreted. The GML 3.2 `nilReason` attribute has the following enumerated values:

- `inapplicable`
- `missing`
- `template`
- `unknown`
- `withheld`