GML implementations

Luc Donéra – ERDAS APOLLO Product Manager
Philippe Duchesne – ERDAS APOLLO Architect
GML in RedSpider and APOLLO products

GML is first implemented as output data format for the WFS service

- Implementation of GML 3.2.0 & 3.2.1

1999
- Ionic Software
- RedSpider
- Products

2001
- GML1 (co-author), GML2 (co-author), GML3 (RWG), WFS (co-author)

August 2007
- Implementation/support of
  - GML 2.1.2, 3.0.0, 3.0.1, 3.1.0, 3.1.1
  - GMLsf 1.0.0
  - EO GML (HMA)
  - GMLJP2

August 2011
- APOLLO Products

Ionic Software creation

INTERGRAPH

erdas
The Earth to Business Company
First and main product usage – WFS output format

- GML allows to describe features and feature collections

- Client applications needs to perform WFS requests to provide vector data styling and editing

- Issue: GML is really verbose

- It takes a lot of time to encode/transport/pars GML data

- **Performance bottleneck**: Makes client applications slow when lots of big geometries are involved

- Possible solution: GML compression, but does not help on encoding and parsing + adds compression/decompression time

- better idea: provide **vector data streaming protocol**, offer views with generalized geometries. But GML doesn’t lend itself to that solution.
Project implementations - 2004 – AIXM inclusion

• **Eurocontrol**: European Organization for the Safety of Air Navigation

• Use GML for the Aeronautical Information Exchange Model (AIXM), designed to enable the management and distribution of Aeronautical Information Services (AIS) data in digital format.

• Design of **custom GML Application schema** matching their needs
  - Compliant with the **GML3.1** specification
  - GML compliant software should be able to read/display the AIXM information as part of the GML data
  - Can be served through compliant WFS
  - include AIXM attributes related to the GML geometry

• GML schema flexibility came in handy
Project implementations - 2006 – OS GML schema

- **Ordnance Survey** - project Magnesium phase II

- Automatic system delivering Change Only Updates data to their clients using OS Master Map data (GML schema designed in Phase I)

- Automatic updates via an OpenGIS compliant WFS-T
  - Update is done by requesting the Master WFS with BBOX, result GML is then pushed to the client WFS-T to automatically update its content

- The OS Master Map GML schema was very complex

- Again, GML schema allowed the expression of a complex model, but this time with a drawback:
  - Some constructs (in particular xlink) are difficult to map to database schemas
  - As a result, some application schemas can turn out to be a nightmare at the time of implementation
Project implementations - 2006 – OS GML schema (cont’d)

- Second issue: the **need for versioning**. It was difficult to implement a versioning system for GML data

  → Versioning facilities should be included in GML specification
Project implementations - 2007 - GMLJP2

• **EUSC** Reference Facilities Project

• Partner with Luratech (JP2 compression)

• implementation of a **GMLJP2** coder/decoder (GML embedded in JP2 headers)

• read and serve GMLJP2 data through the **WCS** OGC web interface

• Goal: provide all info in one file:
  • aerial imagery
  • imagery metadata
  • annotations
  • related vector data

• WCS GMLJP2 results automatically comes with metadata + annotations!

• GML is a good fit for Imagery metadata and related information, but having it **embedded into the JP2** file can be an issue for metadata updates
Project implementations – Earth Observation

- **ESA HMA Projects**

  - Goal: provide standards to implement a European data management system. That system should aggregate and connect services allowing to find and access Satellite Imagery available in each country

  → definition of **EO GML Application schema** to describe sensor products (Radar, Optical, Atm…)

- **CS-W ebRIM EO EP to Catalog EO GML**

  - GML is flexible enough to model the complex metadata structure
Project implementations – Earth Observation (cont’d)

• Initially EO GML was based on gml:observation

• As gml:observation was likely to be deprecated in GML4, had to move to O&M to keep a model of observation (using GML3.2)

→ Watch out for backward compatibility issues from one version to another
Project implementations – Semantics related

• used GML in several projects involving semantic annotations
  • Past Swing european research project
  • Current SMAAD ESA project

• No specific construct is available to express semantic annotations at the feature type, feature instance or feature property levels
  • URIs can be expressed using xlink or anyURI types, but no strongly typed construct

→ OGC DP 08-167 : suggests new constructs for various levels of semantic annotations in several OGC standards, incl. GML

(an updated version of that document is to be released soon)
Summary of issues/recommandations

- Standard for **Vector Data Streaming** Protocol

- Provide **versioning** facilities in GML specification

- Watch out for **backward compatibility** issues

- Need for a strongly typed **semantic reference** construct

- GML schema flexibility and expressivity, a **double edged sword**:
  - really complex models can be mapped into GML
  - but resulting application schemas can be tricky to map into databases, if possible at all
Thank You!