Data and Service Discovery in Linked SDI and Linked VGI

Virtual Workshop on Geospatial Semantic Architectures

May 7, 2013

Todd Pehle
Chief Engineer
tpehle-at-orbistechnologies.com

Are you ready for the answers?
Agenda

• Introduce Example OGC Workflow
• Describe Analogous Linked Data Workflow:
  • Geographic Feature Types in Linked SDI/VGI
  • Data Discovery with GeoVoID
  • Service Capabilities with GeoSPARQL Service Descriptions
  • SPARQL-based Feature Collections
• Conclusion
Example OGC-based GIS Workflow

1. Discover OGC Catalog
2. Search Catalog by Feature Type/BBOX
3. Discover OGC WFS Service
4. GetCapabilities
5. GetFeature
6. DescribeFeatureType
7. Add WFS Layer(s) to Map
8. Get Feature By ID
What constitutes a feature type in Linked Data?

• Linked Data is described using RDFS and OWL ontologies giving data a formal semantics

• Consensus on a “core” *intensional* semantics for geographic phenomena remains elusive

• Option 1: Wait (a long time) for consensus

• Option 2: Minimize ontological commitment and apply definitions driven by *extensional* alignment (i.e. – no core)
Common LOD Feature Type Definitions

W3C Basic Geo (Linked VGI)  OGC GeoSPARQL (Linked SDI)

SpatialThing  SpatialObject

Point  ns:City

Relations = rdfs:subClassOf
Dataset Discovery with VoID and DCAT

- VoID Capabilities:
  - General metadata
  - Structural
  - Class/property partitions
  - Linksets
- DCAT Capabilities:
  - Interoperability of Catalogs
  - Non-RDF Catalogs
- Often stored in data portal like CKAN
  - Offers BBOX dataset queries
  - Has extension support for CSW
- Flexible discovery via centralized catalogs AND socialized links (VoID Repos, URI backlinks, etc.)

Source: http://docs.ckan.org/en/latest/geospatial.html
Geospatial Data Discovery with GeoVoID

Goals:

• Enable discovery of geographic feature data and services in LOD via:
  • Feature Type Discovery
  • Feature Type Spatial Extents
  • Dataset Spatial Extents
  • Thematic Attribution Schema Discovery (maybe)
  • GeoSPARQL Endpoint Discovery
• Reuse and extend existing LOD vocabs vs. reinvention adding additional heterogeneity
• GeoVoID serves *partially* as a WFS GetCapabilities and DescribeFeatureType for LOD
1. Describe datasets using VoID.
2. Then add some geo:

   georss:box (geometry)
   dc:coverage (placename)
   geosparql:rcc8-ntpp (topology)
   geovoid:crs

Simple enough?
Example GeoVoID Document Metadata

# rdf document metadata
<> a void:DatasetDescription;
dcterms:title "Test Dataset Description";
dcterms:description "This is a document containing VoID and GeoVoID descriptions of an example dataset."

dcterms:creator <http://example.org/bob>;
dcterms:created "04-01-2011"^^<xsd:date>;
# dublin core to assert spatial coverage of dataset
dcterms:coverage <http://example.org/the_earth>;
foaf:primaryTopic <http://example.org/ds1>;
foaf:topic <http://example.org/ds2>;
# bounding box of dataset
georss:box “-180 -90, 180 90”;

.
GeoVoID Access & Structural Metadata

# access metadata; No need to redefine a “GeoSPARQL” endpoint
void:sparqlEndpoint <http://example.org/ds1/sparql/url>;

# structural metadata
# can deref to get representative schema info for geo datasets
void:exampleResource <http://example.org/ds1/example/resource1>;
void:exampleResource <http://example.org/ds1/example/resource2>;

# can discover geovocabs used in geo datasets
void:vocabulary <http://example.org/vocab1>;
void:vocabulary <http://example.org/vocab2>;

# a subset combined with a spatial extent = spatial partition
void:subset <http://example.org/ds1/part1>;
void:subset <http://example.org/ds1/part2>;

# number of geo features in geo dataset
void:entities 33123;
void:triples 10500444;
GeoVoID Class & Property Partitions

void:classPartition[
  void:class <http://example.org/ont#Road>; # Road = Feature Type
  void:entities 95; # Number of Road features
  # schema partitions for Road feature type
  void:propertyPartition [ void:property ogc:disjoint; ];
  void:propertyPartition [ void:property rdfs:label; ];
  # geographic feature type partitions can have geospatial extents
  georss:box “-180 -90, 180 90”;
  geosparql:rcc8-ntpp <http://example.org/the_whole_wide_world>;
  # geometry partitions for Road feature type
  void:classPartition[
    void:class <http://www.opengis.net/rdf#LineString>;
    void:entities 95; ];
  void:classPartition[
    void:class <http://www.opengis.net/rdf#Polygon>;
    void:entities 29; ]; ];
SPARQL Service Descriptions

• SPARQL Service Description is a vocabulary for describing features of a SPARQL service
  • Endpoint URIs
  • Supported formats
  • Entailment regimes

• GeoSPARQL Service Descriptions could help describe the capabilities of a given GeoSPARQL endpoint (analogous to GetCapabilities for WFS)
  • Spatial Functions
  • Logical Operators, etc.
GeoSPARQL Service Description

[] a sd:Service ;
  sd:endpoint <http://example.org/geosparql/> ;
  sd:supportedLanguage sd:SPARQL11Query ;
  # assert geosparql as supported language
  sd:supportedLanguage gsd:GeoSPARQLQuery ;
  # assert RDF and GIS formats supported
  sd:resultFormat <http://www.w3.org/ns/formats/GeoJSON>,
                  <http://www.w3.org/ns/formats/Turtle> ;
  # assert spatial functions supported by GeoSPARQL service
  sd:extensionFunction <http://www.opengis.net/def/queryLanguage/OGC-GeoSPARQL/1.0/function/buffer> ;
  sd:feature sd:DereferencesURIs ;
  sd:defaultEntailmentRegime ent:RDFS ;
  # define spatial functions
SPARQL-based Feature Collections

- OGC WFS GetFeature returns Feature Collections
- SPARQL SELECT “Feature Collections”:
  - Each Query Solution represents a feature
  - Query Solution “may” contain bound geometry variable
  - Solution Sequence represents a Feature Collection
- SPARQL CONSTRUCT “Feature Collections”:
  - Features in returned RDF can be asserted or inferred
  - Geometry may be part of a Query Solution
  - Solution Sequence represents a Feature Collection
- SPARQL-based FeatureCollection Envelopes?
  - Geometry can also be dereferenced and features co-referenced which yields additional capability
Mapping Between OGC and LOD GIS Workflow

**OGC**
1. Discover Catalog
2. Search Catalog by Feature Type/BBOX
3. Discover WFS Service
4. WFS Get Capabilities
5. WFS GetFeature
6. DescribeFeatureType
7. Add FeatureCollection to Map
8. GetFeatureByID

**LOD**
1. Discover DCAT or GeoVoID Repo
2. Search Repo by Type/BBOX
3. Get GeoSPARQL Endpoint
4. GeoSPARQL Service Description
5. GeoSPARQL Query
6. VoID partitions + SPARQL Describe + VoID example URI?
7. Add GeoSPARQL Results to Map
8. Dereference URI; Content negotiation for RDF/GIS
Conclusions

To “start” discovering geospatial data, services and feature types in Linked Data we need:

• Alignment axioms for Feature Type definitions
• GeoVoID methodology for dataset descriptions
• GeoSPARQL Service Description for service capabilities
• Common Feature Collection representations for SPARQL results
• Future work needs to address spatiotemporal

Thanks!