Open Geospatial Consortium

Publication Date: 2014-07-16

Approved Date: 2014-06-14

Posted Date: 2014-05-15

Reference number of this document: OGC 14-029r2

Reference URL for this document: http://www.opengeospatial.net/doc/PER/testbed10/virtual-gaz

Category: Public Engineering Report

Editor: Martin Klopfer

OGC® Testbed 10 Virtual Global Gazetteer Engineering Report

Warning

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements.

Document type: OGC® Engineering Report
Document subtype: NA
Document stage: Approved for public release
Document language: English

Abstract

This document provides a technical description of the Virtual Global Gazetteer implemented for OGC Testbed 10.

The Virtual Global Gazetteer integrates two gazetteers – a copy of the USGS gazetteer containing domestic names and a copy of the NGA gazetteer containing non-domestic names (hosted by Interactive Instruments) and provides the capability to link to additional local gazetteers and linked data information, allowing a user to retrieve extended information on locations selected from either of the initial gazetteers. The access to linked data information provided by these gazetteers was achieved by GeoSPARQL enabling these gazetteers using semantic mapping components

Keywords

Ogcdoc, ogc documents, Testbed10, gazetteer, geosparql, wfs, linked data,semantic mediation

Preface

A significant part of the OGC standards development process is the Interoperability Program (IP), which conducts international interoperability initiatives such as Testbeds, Pilot Projects, Interoperability Experiments, and Interoperability Expert Services. These activities are designed to encourage rapid development, testing, validation, demonstration and adoption of open, consensus based standards and best practices.

The OGC Testbed 10 (Testbed-10) is a Testbed within the Interoperability Program. Within Testbed-10, The Cross-Community Interoperability (CCI) thread seeks to build on interoperability within communities sharing geospatial data and advance semantic mediation approaches for data discovery, access and use of heterogeneous data models and heterogeneous metadata models. This thread explored the creation of domain ontologies and tools to create, assemble, and disseminate geographic data provided voluntarily by individuals. One objective was to build integration across all OGC web services with the intent to provide a better understanding of service content and the relationships or associations that exist between OGC services and resources/content.

The Virtual Global Gazetteer effort within the CCI Thread extended the Single Point of Entry Global Gazetteer (SPEGG) work from OWS-9, building on the framework

established in the earlier testbed and expanding gazetteer functionality to include gazetteer conflation and semantic gazetteer linking.

License Agreement

Contents

Page

Figures

Page

Tables

Page

OGC® Testbed 10 Virtual Global Gazetteer Engineering Report

1 Introduction

1.1 Scope

This document provides a technical description of the Virtual Global Gazetteer implemented for the OGC Testbed 10.

The Virtual Global Gazetteer integrates two gazetteers – a copy of the USGS gazetteer containing domestic names (hosted by Compusult) and a copy of the NGA gazetteer containing non-domestic names (hosted by Interactive Instruments) and provides the capability to link to additional local gazetteers and linked data information, allowing a user to retrieve extended information on locations selected from either of the initial gazetteers. The access to linked data information provided by these gazetteers was achieved by GeoSPARQL enabling these gazetteers using semantic mapping components (provided by Image Matters LLC) mapping RDBMS and WFS data to knowledge representation (RDF) on the fly, which is described in clause 11.

The work addressed fictional real world scenarios described in clause 7, which define a number of query capabilities such query by name, feature description, country and spatial constraint. Encoding examples for the queries are provided in clause 8.

Since the Testbed-10 gazetteer work extended the achievements of the related OWS-9 thread, the previous cascading WFS architecture approach and the current requirements for enhancements are discussed in clause 9. The approach used by Image Matters to semantically-enabled existing data stores (Geonames stored in a PostGIS database) and services (USGS, NGA, Geobase WFS-Gs) by providing a GeoSPARQL interface on top of existing data APIs (SQL and OGC Query) was explored during Testbed-10 and can be considered as an alternative to the syntactic approach used in OWS9.

The information model (ISO19112) used as the common model by the USGS and NGA gazetteers and is also the model served by the Virtual Global Gazetteer. Semantic mediation ensures that queries defined in common language, e.g. a query for a <summit>, return the appropriate results from underlying services, in this case the USGS uses the term “hill” which the equivalent NGA term is “elevated point”. A high level description of semantic mediation that is used to draw equivalence in terms used by the USGS gazetteer and the NGA gazetteer is provided in clause 10, including a number of change

requests towards a future revision of the WFS-G BP. A detailed documentation of the underlying principles has been addressed in OWS-9 (cf. OGC 12-103r3).

Gazetteer conflation has been utilized to match entries from multiple names sources, sharing or replacing attribute information, and presenting the fused results to users. For the Testbed 10 scenario the NGA gazetteer populated place features were matched with the New Brunswick gazetteer populated place features, creating a table of links between the two data sets and offering the user information on the matching level. The process is described in clause 12, a detailed discussion of conflation is contained in the Testbed-10 CCI Provenance ER (cf. OGC 14-001r).

Issues experienced during the implementation are discussed in the respective clauses and resulted in a number of change requests and recommendations for future work. To ensure they receive appropriate attention in subsequent activities, these have been summarized in clause 13.

This ER concludes with a description of the utilized data sources.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization
Luiz Bermudez	Open Geospatial Consortium
Doug Caldwell	U.S. Army Corps of Engineers
Rob Cass	Compusult
Stephane Fellah	Image Matters LLC
Gobe Hobona	Envitia
Martin Klopfer	IGSI
David Wesloh	NGA

1.3 Future work

A number of issues were identified in the Virtual Global Gazetteer thread, which are discussed in the respective clauses of this report. To ensure they receive appropriate attention in subsequent activities, these have been summarized in clause 13.

1.4 Forward

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

OGC 09-025r1, OGC Web Feature Service 2.0 Interface Standard (also ISO 19142)

OGC 09-026r1, OGC Filter Encoding 2.0 Encoding Standard

OGC 11-122r1, WFS Gazetteer Application Profile

OGC 12-103r3, OWS-9 Engineering Report - CCI – Semantic Mediation

OGC 12-104, OWS-9 Engineering Report - CCI - Single Point of Entry Global Gazetteer

OGC 10-100r3, Geography Markup Language (GML) simple features profile

OGC 06-121r3, OGC® Web Services Common Standard

OGC 11-052r4 OGC® GeoSPARQL - A Geographic Query Language for RDF Data

OGC 14-001 OGC® Testbed 10 CCI Provenance Engineering Report

OGC 14-021r2 OGC® Testbed 10 CCI Profile Interoperability Engineering Report

OGC 14-049 OGC® Testbed 10 CCI Ontology Engineering Report

3 Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard [OGC 06-121r3] shall apply. In addition, the following terms and definitions apply.

3.1 Attribute <XML>

name-value pair contained in an **element**

[ISO 19136:2007]

NOTE In this document an attribute is an XML attribute unless otherwise specified.

3.2 client
software component that can invoke an **operation** from a **server**

[ISO 19128:2005]

3.3 coordinate
one of a sequence of n numbers designating the position of a point in n-dimensional space

[ISO 19111:2007]

3.4 coordinate reference system
coordinate system that is related to an object by a datum

[ISO 19111:2007]

3.5 coordinate system
set of mathematical rules for specifying how **coordinates** are to be assigned to points

[ISO 19111:2007]

3.6 element <XML>
basic information item of an XML document containing child elements, **attributes** and character data

[ISO 19136:2007]

3.7 feature
abstraction of real world phenomena

[ISO 19101:2002]

NOTE A feature can occur as a type or an instance. The term "feature type" or "feature instance" should be used when only one is meant.

3.8 feature identifier
identifier that uniquely designates a **feature** instance

3.9 filter expression
predicate expression encoded using XML

[ISO 19143]

3.10 GeoSPARQL
SPARQL with a standardized set of geospatial functions that are needed to manipulate geospatial information.

3.11 interface
named set of **operations** that characterize the behaviour of an entity

[ISO 19119:2005]

3.12 Linked Data
A pattern for hyperlinking machine-readable data sets to each other using Semantic Web techniques, especially via the use of RDF and URIs. Enables distributed SPARQL queries of the data sets and a browsing or discovery approach to finding information (as compared to a search strategy). Linked Data is intended for access by both humans and machines. Linked Data uses the RDF family of standards for data interchange (e.g., RDF/XML, RDFa, Turtle) and query (SPARQL). If Linked Data is published on the public Web, it is generally called Linked Open Data.

[W3C http://www.w3.org/TR/ld-glossary/]

3.13 Multipurpose Internet Mail Extensions (MIME) type
media type and subtype of data in the body of a message that designates the native representation (canonical form) of such data

[IETF RFC 2045]

3.14 namespace <XML>
collection of names, identified by a URI reference which are used in XML documents as **element** names and **attribute** names

[W3C XML Namespaces]

3.15 operation
specification of a transformation or query that an object may be called to execute

[ISO 19119:2005]

3.16 property
facet or attribute of an object, referenced by a name

[ISO 19143]

3.17 request
invocation of an **operation** by a **client**

[ISO 19128:2005]

3.18 response
result of an **operation** returned from a **server** to a **client**

[ISO 19128:2005]

3.19 response model
schema defining the properties of each **feature** type that can appear in the **response** to a query **operation**

NOTE This is the schema of feature types that a **client** can obtain using the DescribeFeatureType operation (cf. Clause 9).

3.20 schema
formal description of a model

[ISO 19101:2002]

NOTE In general, a schema is an abstract representation of an object's characteristics and relations to other objects. An XML schema represents the relationship between the **attributes** and **elements** of an XML object (for example, a document or a portion of a document).

3.21 schema <XML Schema>
collection of **schema** components within the same target **namespace**

[ISO 19136:2007]

EXAMPLE Schema components of W3C XML Schema are types, **elements**, **attributes**, groups, etc.

3.22 server
particular instance of a **service**

[ISO 19128:2005]

3.23 service
distinct part of the functionality that is provided by an entity through **interfaces**

[ISO 19119:2005]

3.24 service metadata
metadata describing the **operations** and geographic information available at a **server**

[ISO 19128:2005]

3.25 Uniform Resource Identifier
unique identifier for a resource, structured in conformance with IETF RFC 2396

[ISO 19136:2007]

NOTE

The general syntax is <scheme>::<scheme-specified-part>.

The hierarchical syntax with a **namespace** is <scheme>://<authority><path>?<query>

3.26 Symbol
A “symbol” is essentially a “bitmap or vector image that is used to represent a point.

3.27 Symbol Table
A “symbol table” or set of symbol metadata on the other hand denotes a “term referring to the storage of named objects, including line types, layers, text styles and blocks.

4 Conventions

4.1 Abbreviated terms

API	Application Program Interface
CRS	Coordinate Reference System
ER	Engineering Report
FES	Filter Encoding Standard
GML	Geography Markup Language
GMLSF	Simple Feature Geography Markup Language
HTTP	Hypertext Transfer Protocol
KVP	Keyword-Value Pair
MIME	Multipurpose Internet Mail Extensions
NGA	National Geospatial-Intelligence Agency
OWL	Ontology Web Language
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF Query Language
UML	Unified Modelling Language
USGS	United States Geological Survey
WFS	Web Feature Service
WFS-C	Cascading Web Feature Service
XML	Extensible Markup Language

4.2 UML notation

Most diagrams that appear in this standard are presented using the Unified Modeling Language (UML) static structure diagram, as described in Subclause 5.2 of [OGC 06-121r3].

5 Virtual Global Gazetteer overview

The Virtual Global Gazetteer effort extended the Single Point of Entry Global Gazetteer (SPEGG) work from OWS-9, building on the framework established in the earlier testbed and expanding gazetteer functionality to include gazetteer conflation and gazetteer linking.

The key task in this thread focused on the development of an enhanced Virtual Global Gazetteer Client, advancing fault-tolerant capabilities, and opening the service to the wider community for comment.

There are two agencies in the United States in charge of maintaining the official gazetteers:

- The USGS gazetteer manages all domestic place names, accessible through a WFS 2.0

- The NGA gazetteer manages all non-domestic place names, accessible through a WFS 1.1.0

Both gazetteers not only run on different WFS versions with different capabilities, but also utilize the proprietary data and classification schemes of each organisation. Hence a query, e.g. for a <summit>, needs to be mapped to a query for <elevated point> for the NGA service and <hill> for the USGS service to ensure the client receives all relevant matches from both underlying services.

The basic requirements and concepts from OWS-9 remains largely unchanged for the Virtual Global Gazetteer, with the notable difference that in Testbed-10 the Gazetteer is expected to access large scale data-sets, as opposed to a limited demo data-set in OWS-9. In addition the requirements were extended to include query filters for <country> and <feature type>.

Extended gazetteer linking capabilities support tapping into information services beyond the NGA and USGS gazetteers. Once the desired location is selected from either gazetteer, linked information such as DBPedia, LinkedGeoData (OpenStreet Map Data), Geonames (through Image Matters GeoSPARQL), Canadian Geobase can be accessed in a unified way using (Geo)SPARQL protocol and query language.

This ER addresses the implementation of the extended client and endpoints as well as the applied semantic mediator, conflation approaches and gazetteer linking to further information services using GeoSPARQL endpoints for semantic mapping components to data sources (Geonames stored in PostGIS or NGA, USGS, Canadian Geobase WFS-G) or by using SPARQL endpoint to other knowledge source (DBPedia, LinkedGeoData).

6 Gazetteer Schema

6.1 Introduction

The common schema used for the USGS and NGA gazetteers is the Document OGC 11-122r1 Gazetteer Service - Application Profile of the WFS Best Practice. Approval Date: 2012-01-30, available at: http://www.opengis.net/doc/wfs-gaz-ap.

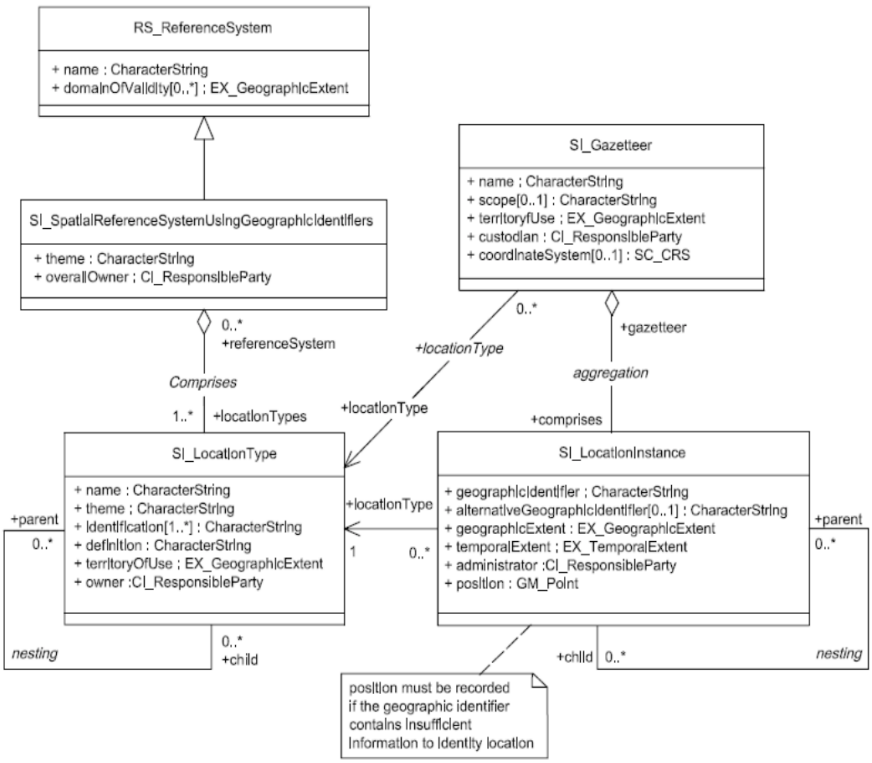
This schema is the result of previous OWS work and based on an XML encoding of the data model found in ISO-19112, "Geographic information -- Spatial referencing by geographic identifiers". The advantage of the WFS-G schema is that the USGS and NGA servers can be cascaded without the need to perform a schema translation.

The following subclauses have been adapted from OGC 12-104, OWS-9 Engineering Report - CCI - Single Point of Entry Global Gazetteer, with an added discussion on the implication of the extended query filter requirements.

6.2 Gazetteer UML model

Figure 1 illustrates the UML model – taken from ISO-19112 -- that defines the data model offered by the Virtual Global Gazetteer .

Figure 1 – ISO19112 UML Model



All place names offered by a gazetteer are instances of SI_LocationInstance. Each location instance may be classified as being of a specific type and each location type offered by a gazetteer is an instance of SI_LocationType.

A primary function of the semantic mediator (cf. OGC 12-102r3) is to map equivalent location types between the USGS and NGA gazetteers thus allowing clients to query by location type using either vocabulary.

6.3 ISO19112 XML Schema

As has already been mentioned (cf. 6.1) both the USGS and NGA gazetteers implement the ISO19112 UML model (cf. 6.2). In addition, because of previous work (cf. OGC 11-122r1) both servers implement the same XML realization of the ISO19112 model commonly referred to as ISO 19112 “lite”.

The ISO 19112 “lite” XML schema implemented by the USGS and NGA servers is encoded as a GML (cf. OGC 07-036) application schema using the level 1 of the Geography Markup Language (GML) simple features profile (cf. OGC 10-100r3).

The XML schema for ISO19112 can be found in ANNEX A.

6.4 Identified Issues

Three change requests to the current WFS-G Best Practices have been identified:

1. The parent property is not appropriate for locations due to its implication of inheritance (within Object Oriented modeling). Another reason is that the parent property implies a parent-child relationship whereas its role attribute may imply different relationships such as 'in_country', potentially contradicting the parent relationship. This could lead to incorrect interpretation of values due to ambiguity of the parent property or its role attribute.

   The resulting change request suggests replacing the parent property with a 'relation' property with tagged value for role.

2. The WFS-G BP specifies that the top level container should be an iso19112:SI_Collection element but then gives an example that uses a wfs:FeatureCollection element. This will lead to exceptions if clients do not know what top-level container (root element) to expect.

   The resulting change request suggests to constrain the WFS-G Response to providing wfs:FeatureCollection as the top-level container.

3. The xlink:href to LocationType in WFS-G uses a WFS query pointing to GML document. This is a problem when you just want to get the non-information resource URI (i.e. in GML context the namespace and the identifier (gml:id) of

the resource). To perform semantic mapping from XML to RDF, the current approach requires an expensive call to WFS, parsing a new GML document to extract its gml:id so reconstruction of the non-information resource URI can be done (namespace+gml:id). Xlink (and RDF references for that matter) will only work if you use a non-information resource URI and allow HTTP resolution to find the document form.

7 Use Case and Demo Scenarios for the Virtual Global Gazetteer

7.1 Obtaining information from linked data on a location

This use case and demo scenario for the Virtual Global Gazetteer in Testbed-10 revolves around a hypothetical delegation from delegation from Louisiana planning on attending the Tintamarre 2014 celebrations in St. John, New Brunswick. This is fitting, as many of the Acadians who were forced to migrate to Louisiana eventually returned to New Brunswick.

Tintamarre is an Acadian tradition of marching through one's community making noise with improvised instruments and other noisemakers, usually in celebration of National Acadian Day. The term originates from the Acadian French word meaning "clangour" or "din". The practice is intended to demonstrate the vitality and solidarity of Acadian society, and to remind others of the presence of Acadians. It originated in the mid-twentieth century, likely inspired by an ancient French folk custom.

The scenario demonstrates how the linking and conflation tools can be used to assist a Louisiana delegation in planning for the event in Saint Johns on August 15, 2014.

7.1.1 Gazetteer Linking

The Louisiana delegation is interested in obtaining basic information about Saint John, where they will be attending the Tintamarre 2014 celebrations. They would like a basic report with information about names and nicknames for the city, websites related to the city, the geography, typical weather, the type of government, and names of government officials. They would also like to learn more about the communities within Saint Johns. A quick look at the NGA gazetteer shows that this information is not available. They need to 'Get More Stuff' from linked data resources on the chosen location-

The purpose of the Gazetteer Linking demonstration is to show the value of linking for obtaining additional attribute and spatial information... information that can be used to find additional information.

7.1.1.1 Attribute Query

The initial step is to query the NGA database for the location of interest and to the additional information desired on this location:

1. View the NGA populated places in New Brunswick.

2. Click on Saint John (UFI -572890) and see returned information along with Get More Stuff

3. Using the connections through GeoNames (6138517) to dbPedia, (Saint_John,_New_Brunswick) to find out what information is available for Saint John in dbPedia.

4. Select the information to be reported, e.g.:

   Name

   foaf:name

   foaf:nick

   Websites

   dbbprop:website

   foaf:isPrimaryTopicOf

   Geography

   dbpedia-owl:isPartOf

   dbpedia-owl:populationMetro

   dbpedia-owl:populationMetroDensity

   dbpedia-owl: PopulatedPlace? /areaMetro

   dbpedia-owl:minimumElevation

   dbpedia-owl:maximumElevation

   dbpedia-owl:timeZone

   Weather

   dbpprop:augMeanC

   dbpprop:augLowC

dbpprop:augHighC

dbpprop:augHumidity

dbpprop:augRainDays

dbpprop:augRainMm

Politics

dbpedia-owl:governmentType

dbpedia-owl:leaderName

Subsequent queries on the returned results are also possible, in this case e.g. to retrieve further information on the mayor.

7.1.1.2 Spatial Query

The spatial query in the scenario is used to retrieve suburban community features from the New Brunswick Gazetteer that are within the area of interest.

1. View the NGA populated places in New Brunswick.
2. Click on Saint John (UFI -572890) and see returned information along with Get More Stuff.
3. Using the connections in the link spreadsheet to go to the OSM Way data (111878854).
4. Extract the polygon coordinates from the OSM Way data and store in format that can be reused.
5. Use the extracted polygon as a filter to find Suburban Community (GENERITERM field) features from the New Brunswick database that are inside the polygon.
6. Print a list of names (GEONAME field) of the suburban communities inside the Saint John polygon.

7.1.2 Conflation

After seeing the detail in the New Brunswick data set from the spatial query task, the Louisiana delegation wants to use this data set in their work. Unfortunately, there are no links from the New Brunswick data to the NGA data.

Links from the New Brunswick data to the NGA data are established using conflation. These links can be followed to recreate the results from the Gazetteer Linking demonstration, which include the Attribute Query and the Spatial Query.

The purpose of the Conflation demonstration is to show that conflation can be used to build links that can then be used to obtain additional attribute and geospatial information. This is a powerful capability!

7.1.2.1 Conflation Process

1. Run the gazetteer conflation WPS to establish links between the NGA features and the New Brunswick features.

   Parameters

   Source Gazetteer - WFS-G from NGA

   Target Gazetteer - WFS-G for New Brunswick gazetteer from NRCan

   Source Gazetteer Description Filter - PPL, PPLA, PPLA2, PPLA3, PPLA4, PPLC, PPLF, PPLH, PPLL, PPLQ, PPLR, PPLS, PPLW,PPLX, STLMT

   Target Gazetteer Description Filter - CITY, TOWN, UNP, VILG, MUN1

   Bounding Box Filter - -66.46 45.14 -65.33 45.88

   Search DIstance - 15 miles

   FuzzyWuzzy Threshold - > 80

2. View the connecting lines between the NGA and the New Brunswick names, which gives us an idea of the accuracies of the data sets.

3. Select Saint John from the New Brunswick data.

4. Use the link from Saint John in the New Brunswick data (FEATUREID 93f89999d05511d892e2080020a0f4c9) to Saint John in the NGA data (UFI - 572890) and execute the Attribute Query to add the information to Saint John in the New Brunswick data.

5. Use the link from Saint John in the New Brunswick data to Saint John in the NGA data and execute the Spatial Query from Gazetteer Linking to add the suburban community information to Saint John in the New Brunswick data.

7.2 Finding features with specific attributes

A generous donor has provided $1 million (USD) for to set up a broadcast antenna to share the Tintamarre 2014 celebration in Saint John with the widest possible audience. She has specified that the antenna be located on a mountain top within 150 miles of the city and that the mountain top could be in Canada or the United States. The goal for this task is to identify the tallest mountain within 150 miles of Saint John and find the closest airport to that mountain.

7.2.1 Finding Saint John - Fuzzy Search

1. The analyst accidently types in 'Saint Johns' and NGA feature designation PPL and an exact match to locate all the Saint Johns in New Brunswick. There are no Saint Johns in New Brunswick, (We could also try St. John or St. Johns to see if these might work as well)

2. The analyst turns on the fuzzy search with 'Saint Johns' and finds that there is a Saint John.

3. The analyst sees that there is only one Saint John in New Brunswick.

4. The analyst takes the coordinates from Saint John and uses them in the next process.

7.2.2 Tallest Mountain within 150 Miles of Saint John – Radial Search

1. The analyst initiates a 150 mile radial search query from Saint John (-66.095316, 45.230798) using the USGS feature type 'Summit' for features.

   The client application will show the user the feature types in the NGA gazetteer which match the USGS feature type 'Summit.' In other words,. the client is supposed to show the semantic mappings to the user as part of the query process, i.e., USGS feature type 'Summit' brings up NGA feature designations, Mountain, Hill, Knob, etc. Exposing the user to the semantic mapping is a key new capability in Testbed-10.

   The feature type term 'Summit' is entered and the location type from the New Brunswick gazetteer that is returned is mountain, or more specifically #SI_LOCATIONTYPE_MTN.

2. The application will locate all the mountain features within 150 miles and show the results in a table and on a map. All diacritics will be shown.

3. The analyst will intersect the query result with elevation data to obtain elevations for all the mountains.

4. The application will sort the results based on the elevation values in descending order, from highest to lowest. The tallest summit should be Mount Katahdin in Maine.

7.2.3 Closest Airport to Highest Mountain – Near Search

1. The analyst issues a near query from the location of the tallest mountain on NGA feature type Airfield (AIRF) to see what USGS feature types will be returned.

2. Curious with the result, the analyst changes the query to the USGS feature type 'Airport' to see what NGA features will be returned. He sees that this is more comprehensive and decides to use the USGS term 'Airport' in the query.

3. The analyst views the results in a table and on a map. The closest airport should be Millinocket Seaplane Base.

8 Virtual Global Gazetteer Query Requirements

8.1 Introduction

OGC Testbed 10 demonstrated a client that allows a user to formulate a query that includes the name, a name string filter for the name, a feature description, country, and a spatial constraint. The results are returned in tabular form, with the ability to search for additional results if a subset of the search results is returned by the query. By using gazetteer conflation (cf. clause 12) and linking (cf. clause 11), access to additional attribute and spatial information, e.g. through geonames.org, is enabled for returned results.

For the Testbed-10 demonstration the example of a Louisiana delegation is used, who would like a basic report with information about names and nicknames for the city, websites related to the city, the geography, typical weather, the type of government, and names of government officials. They would also like to learn more about the communities within Saint Johns.

8.2 Query by Name

Text searching operators are based on the OGC Filter Encoding Standard (cf. OGC 09-026r1) and support “starts with”, “ends with”, “sub-string containment” and “fuzzy string matching” searches.

All Testbed-10 servers used the UTF-8 character set for handing diacritics, native scripts, special characters, etc.

The user can enter a name, including diacritics, and select how the name is utilized in the query. Options include:

- Starts With (Saint *)
- Ends with (* John)
- Contains (**John**)
- Fuzzy Match (St. John ~ Saint John)

In our demo scenario, the source gazetteer is assumed to be WFS-G from NGA, which for the simple name query would return all stored features matching the above criteria. Given Saint John could not only be a town, but as well a church or a place name, the query should be narrowed down with one or more of the filters described in the following sub-clauses, which have been partly adapted from OGC 12-104, with an added discussion on the implication of the extended query filter requirements in Testbed-10.

8.2.1 Starts with

The “Starts with” operator is meant to match text strings that begin with a specified sequence of characters. For example, someone searching for records that contain the text string “Boston” might instead search for any string that starts with “Bost” in order to case a wider search net. The existing Filter Encoding standard already supports this type of predicate using the PropertyIsLike operator. The following example illustrates the use of the PropertyIsLike operator to match strings that start with a specified prefix:

```
<fes:Filter
   xmlns:fes="http://www.opengis.net/fes/2.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/fes/2.0
   ../../../../filter/2.0/filterAll.xsd">
   <fes:PropertyIsLike wildCard="*" singleChar="#" escapeChar="!">
   <fes:ValueReference>alternativeGeographicIdentifier</fes:ValueRefere
nce>
       <fes:Literal>Bost*</fes:Literal>
   </fes:PropertyIsLike>
</fes:Filter>
```


8.2.2 Ends with

The "Ends with" operator is meant to match test string that ends with a specified sequence of characters. For example, someone searching for records that contain the text string "New York" might instead search for any string that ends with "York". Like the "Start with" operator, the existing Filter Encoding standard already supports this type of predicate using the PropertyIsLike operator. The following example illustrates the use of the PropertyIsLike operator to match strings that end with a specified suffix:

```
<fes:Filter
    xmlns:fes="http://www.opengis.net/fes/2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/fes/2.0
    ../../../../filter/2.0/filterAll.xsd">
    <fes:PropertyIsLike wildCard="%" singleChar="#" escapeChar="!">
    <fes:ValueReference>alternativeGeographicIdentifier</fes:ValueRefere
nce>
        <fes:Literal>%York</fes:Literal>
    </fes:PropertyIsLike>
</fes:Filter>
```

8.2.3 PropertyContains

This sub-clause defines a set of operators that extend the capabilities of OpenGIS's Filter Encoding 2.0 Standard (cf. OGC 09-026r1) to support advanced text searching capabilities. The operators are a standalone extension package that makes use of the extension points defined in the Filter Encoding 2.0 Standard (cf. 7.12.3, OGC 09-026r1).

This namespace for the advanced text search extension shall be:

http://www.opengis.net/fes/2.0/advstr/1.0

8.2.3.1 Introduction

The PropertyContains operator is similar to the PropertyIsLike operator, except that, unlike the PropertyIsLike operator which blindly compares sequences of characters, the PropertyContains operator interprets the words contained in a text field as individual, sequential units. You may thus specify one or more of these units as search criteria. In addition, the PropertyContains operator itself allows predicates to be specified allowing text fields to be searched for complex word relationships such as "word X is within 10 words of word Y".

8.2.3.2 XML encoding

The following schema fragment defines the PropertyContains operator.

```
   <xsd:element name="PropertyContains"
                type="advstr:PropertyContainsType"
                substitutionGroup="fes:extensionOps"/>
   <xsd:complexType name="PropertyContainsType">
      <xsd:complexContent>
         <xsd:extension base="fes:ExtensionOpsType">
            <xsd:sequence>
               <xsd:element ref="fes:expression" minOccurs="2"
maxOccurs="2"/>
               <xsd:element name="NearTerm" type="advstr:NearType"
                            minOccurs="0"/>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="NearType">
      <xsd:simpleContent>
         <xsd:extension base="xsd:string">
            <xsd:attribute name="within" type="xsd:positiveInteger"/>
         </xsd:extension>
      </xsd:simpleContent>
   </xsd:complexType>
   <xsd:element name="SearchTerms" type="advstr:SearchTermsType"
                substitutionGroup="fes:expression"/>
   <xsd:complexType name="SearchTermsType">
      <xsd:simpleContent>
         <xsd:extension base="xsd:string">
            <xsd:attribute name="andChar" type="xsd:string"
                           use="optional" default="&amp;amp;"/>
            <xsd:attribute name="orChar" type="xsd:string"
                           use="optional" default="|"/>
            <xsd:attribute name="notChar" type="xsd:string"
                           use="optional" default="!"/>
            <xsd:attribute name="eqChar" type="xsd:string"
                           use="optional" default="="/>
            <xsd:attribute name="escapeChar" type="xsd:string"
                           use="optional" default="\"/>
            <xsd:attribute name="matchCase" type="xsd:boolean"
                           use="optional" default="false"/>
         </xsd:extension>
      </xsd:simpleContent>
   </xsd:complexType>
```


8.2.3.3 KVP encoding

The following table defines the KVP parameters that encoding the PropertyContains operator:

Parameter	O/M	Default Value	Description
VALUEREFERENCE	M		A reference to a value to be tested by the operator. This can be the name of a property or an XPath expression pointing to a sub-field of a complex property.
SEARCHTERM	M		A string containing the search terms to be tested.
NEARTERM	O		If performing a proximity search, the value of this parameter is the proximal term.
WITHIN	O		If performing a proximity search, the value of this parameter defines the distance to be searched (e.g. WITHIN=10 specified that the NEARTERM should exist within 10 words of the SEARCHTERM)
ANDCHAR	O	&	The character used in the searchTerm parameter to indicate the logical AND connector
ORCHAR	O	|	The character used in the searchTerm parameter to indicate the logical OR connector
NOTCHAR	O	!	The character used in the searchTerm parameter to indicate the locical NOT connector
EQCHAR	O	=	The character used in the searchTerm parameter to indicate that two search terms may be considered equivalent
ESCAPECHAR	O	\	The character used in the searchTerm parameter to suspect the meaning of the andChar, orChar, notChar, eqChar and escapeChar and simply interpret them as characters that are part of the search terms
MATCHCASE	O	FALSE	A boolean indicating whether search terms should be tested taking case into account or not.

8.2.3.4 Parameter discussion

The ValueReference parameter shall reference a value to be tested (cf. OGC 09-026r1, clause 7.4.1).

The SearchTerms parameter contains one or more logically combined search terms that the value being tested shall/may contain. By default the value being tested must contain all listed terms in order for the PropertyContains operator to evaluate to true. In other words, the default logical connection between listed search terms is AND. The logical connection between search terms may be modified using the andChar, orChar, notChar and eqChar parameters. Each parameter defines the character that represents the corresponding logical operator. The defaults are andChar="&", orChar="|", notChar="!" and eqChar="=". Parentheses may be used to group terms together. The escapeChar may be used to suspect the meaning of the special andChar, orChar, notChar, eqChar and escapeChar characters and simply interpret them as being part of the search terms.

Example: The value being tested must contain the terms "cat" AND "dog" OR the term "fish".

<SearchTerms>(cat dog) | fish</SearchTerms>

The NearTerm parameter is used to specify a search term for proximity searching. The "within" parameter defines the search distance.

Example: Search for the term "Common" with 2 words of the term "Boston".

```
<PropertyContains>
   <ValueReference>alternativeGeographicIdentifier</ValueReference>
   <SearchTerms>Boston</SearchTerms>
   <NearTerm within="2">Common</NearTerm>
<PropertyContains>
```

8.2.4 Fuzzy string matching

8.2.4.1 Introduction

Fuzzy string match operators, unlike regular string matching operators, evaluate how "well" or to what extent two string match. This level of wellness is expressed as a percentage. For example, "string1" is a 86% match to "string2". Fuzzy string match operators can, for example, be used to allow meaningful searches to be executed with search terms that are misspelt.

The previous testbed introduced a new operator called PropertyMatches to support fuzzy string matching. The operator, which allows a client to specify the method and tolerance for the comparison, was provided through a typical WFS query filter.

8.2.4.2 XML encoding

The following XML Schema fragment defines the XML encoding for the PropertyMatches operator.

```
   <xsd:element name="PropertyMatches"
type="advstr:PropertyMatchesType"
                substitutionGroup="fes:extensionOps"/>
   <xsd:complexType name="PropertyMatchesType">
      <xsd:complexContent>
         <xsd:extension base="fes:ExtensionOpsType">
            <xsd:sequence>
               <xsd:element ref="fes:expression"
                            minOccurs="2" maxOccurs="2"/>
            </xsd:sequence>
            <xsd:attribute name="method" type="advstr:MatchMethodType"
                           use="optional" default="levenshtein"/>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:element name="MatchString" type="advstr:MatchStringType"
                substitutionGroup="fes:expression"/>
   <xsd:complexType name="MatchStringType">
      <xsd:simpleContent>
         <xsd:extension base="xsd:string">
            <xsd:attribute name="strength" type="advstr:PerCent"
                           use="optional" default="100"/>
         </xsd:extension>
      </xsd:simpleContent>
   </xsd:complexType>
   <xsd:simpleType name="PerCent">
      <xsd:restriction base="xsd:positiveInteger">
         <xsd:minInclusive value="1"/>
         <xsd:maxInclusive value="100"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:simpleType name="MatchMethodType">
      <xsd:union>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:enumeration value="levenshtein"/>
               <xsd:enumeration value="jaro-winkler"/>
            </xsd:restriction>
         </xsd:simpleType>
         <xsd:simpleType>
            <xsd:restriction base="xsd:string">
               <xsd:pattern value="vendor:\w{2,}"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:union>
   </xsd:simpleType>
```

8.2.4.3 KVP encoding

The following table defines the KVP encoding for the PropertyMatches operator:

Parameter	O/M	Default Value	Description
VALUEREFERENCE	M		A reference to a value to be tested by the operator. This can be the name of a property or an XPath expression pointing to a sub-field of a complex property.
MATCHSTRING	M		A string containing the search term to be matched.
STRENGTH	O	100	A number between 1 and 100 indicating the minimum matching strength. A value of 100 indicates that the two argument must be identical. A value of 90% means that the operator will evaluate to TRUE if the two arguments are at least a 90% match.
METHOD	O	levenshtien	The method to use to compute the strength of the match.

8.2.4.4 Parameter discussion

The mandatory *ValueReference* parameter shall reference a value to be tested (cf. OGC 09-026r1, clause 7.4.1).

The mandatory *MatchString* parameter shall contain the value against which the value referenced using the *ValueReference* parameter shall be tested.

The optional *method* parameter is used to identify the algorithm that shall be used to evaluate the reference value and the match string.

All servers shall implement the Levenshtein algorithm (cf. http://www.levenshtein.net) and the Jaro-Winkler algorithm (cf. http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance).

Whereas in the previous testbed the fuzzy matching capability was provided through a jaro-winkler distance calculation, in this current testbed the fuzzy matching was implemented through a Levenshtein distance calculation.

Additional vendor-specific algorithms may also be specified using the pattern "vendor:{name}" but this standard does not describe what these methods might be.

Since typicalls only one method is required and the implementation of multiple methods might slow down uptake, a choice of one of them should be mandated.

The optional *strength* parameter is used to indicate a matching threshold beyond which the PropertyMatches operator shall evaluate to true. The value of *strength* parameter shall be a value between 1 and 100 with 100 indicating that the arguments must be identical in order for the PropertyMatches operator evaluates to true. Lower values allow for increasingly "fuzzier" matches.

Example: The following example searches for alternative geographic identifiers that are a 90% match for the string "Albeon" using the Jaro-Winkler algorithm.

```
   <ogc:PropertyMatches method="jaro-winkler">

<ogc:PropertyName>alternativeGeographicIdentifier</ogc:PropertyName>
      <ogc:MatchString strength=”90”>Albeon</ogc:MatchString>
   </ogc:PropertyMatches>
```

This predicate would, for example, match the string "Ablion".

The successful use of the PropertyMatches operator confirmed its utility and potential role in future WFS-G usage. It is therefore recommended that this operator be included in future revisions of OGC standards.

8.3 Query by Feature Description

8.3.1.1 Introduction

The user should be able to filter queries by the feature description (also known as the feature designation). This description could reflect the terminology of either agency, i.e., the user should be able to query on USGS feature descriptions or NGA feature designations. These should be expanded to common language descriptions rather than codes. As an example, a user should be able to select USGS feature descriptions and pick a term like 'summit.' This term would access information from related NGA feature classes, such as 'mountain', 'hill', 'peak', 'rock', etc. The user should also be able to filter features based on the use of NGA terms. In this case, picking a term like 'hill' would return USGS 'summit' features. These mappings should be displayed to the user on the query form.

For the demo scenario the delegation would look for a town, i.e. a populated place. The selection in the client thus applies a source gazetteer description filter, narrowing the query down to features complying to the following NGA LocationTypes:

PPL, PPLA, PPLA2, PPLA3, PPLA4, PPLC, PPLF, PPLH, PPLL, PPLQ, PPLR, PPLS, PPLW,PPLX, STLMT

8.3.1.2 XML encoding

The PropertyIsSemanticallyRelatedTo operator, proposed in OWS-9, was reused for Testbed-10. A minor modification was made in constraining queries by placetype rather than URI. This modification was made because the URIs (hyperlinks in SI_LocationType instances) used in Testbed-10 were found to be different from those used OWS-9; the placetypes however remain the same between different implementations.

```
<ogc:PropertyIsSemanticallyRelatedTo>

<ogc:PropertyName>iso19112:locationType/@xlink:title</ogc:PropertyName>

<ogc:Literal>locale</ogc:Literal>

</ogc:PropertyIsSemanticallyRelatedTo>
```

8.3.1.3 KVP encoding

The PropertyIsSemanticallyRelatedTo operator is used within the Filter submitted with the request.

8.3.1.4 Parameter discussion

The PropertyIsSemanticallyRelatedTo operator, as used in Testbed-10, accepts only two parameters (an XPath through the PropertyName parameter and a locationType name through Literal property.

8.4 Query by Country

8.4.1.1 Introduction

Since queries for features on a global scale not only return a fast amount of (most likely unwanted) results, but also come with a performance limitations, the user would be expected to narrow down a search by selecting the country of interest first. Country names are expanded to common language descriptions rather than codes.
The testbed participants found that the “parent” property used in previous testbeds to indicate the country within which a place is located was not appropriate for such use; the key reason being that, from an object-orientation perspective, locations cannot be considered to have parent-child relationships. It was therefore recommended that the WFS-G Best Practice specification should be updated to introduce a property that can indicate administrative associations between places.

8.5 Query by Spatial Constraint

The user should be able to filter the query using a bounding box, radial search, or near query that will sort the results from closest to furthest away from a given coordinate.

8.5.1 Radial search

OGC web services offer a number of approaches for implementing radial search, e.g.:

- the intersection of a location property with a CircleByCenterPoint geometry, which was applied in OWS-9 and is described in OGC 12-104, or
- use of the DWithin operator which allows features to be tested within a distance of a geometry (typically a point).

In the case of radial search, the effect of these two approaches is the same as they both test for intersection with a buffer around a point. It should be noted however, that DWithin has the potential to support geometries other than a point.

Testbed-10 examined both approaches and found that the NGA, USGS and NB WFS-Gs supported radial search through DWithin, meaning that requests could be cascaded within minimal transformation. In contrast, since neither of the services supported CircleByCentrePoint, the transformation of a CircleByCentrePoint geometry into a supported geometry such as a polygon would be required to support the alternative approach for implementing radial search.

Based on this experience, it is recommended that WFS-G shall by default support radial search through DWithin operators.

For example:

```
<ogc:DWithin>
<ogc:PropertyName>iso19112:position</ogc:PropertyName>
<gml:Point srsName="urn:ogc:def:crs:EPSG::4326">
<gml:pos>44.90 -66.95</gml:pos>
</gml:Point>
<ogc:Distance units="m">10000</ogc:Distance>
</ogc:DWithin>
```

Similarly, for WFS 2.0:

```
<fes:DWithin>
<fes:ValueReference>iso19112:position</fes:ValueReference>
<gml:Point srsName="urn:ogc:def:crs:EPSG::4326">
<gml:pos>44.90 -66.95</gml:pos>
</gml:Point>
<fes:Distance uom="m">10000</fes:Distance>
</fes:DWithin>
```

8.5.1.1 Introduction

8.5.2 Nearest Neighbour

8.5.2.1 Introduction

A nearest neighbour search finds objects near a centre point and orders the features in the response according to the distance from that search point. Unlike a centre-point-radius search, a nearest neighbour search will always return a result – regardless of how far away from the centre point the closest object is – as long as the database is not empty.

8.5.2.2 Implementation

The nearest neighbor search relies on a spatial index being available in order to calculate distances to all locations in the supporting databases. This meant that the virtual global gazetteer would have to rely on the NGA WFS-G (based on WFS 1.1) and the USGS/NB WFS-G (based on WFS 2.0). The testbed participants found that currently the Filter 2.0 standard prevents WFS 2.0 from being backwards compatible with WFS 1.1. This is discussed in the Testbed-10 CCI Profile Interoperability Engineering Report (OGC 14-021), clause 10. Consequently, the nearest neighbour algorithm was implemented on the client application.

8.5.2.3 KVP encoding

A stored query with the name “Nearest Neighbour By Location Type” and assigned the identifier “urn:cw:def:query:OGC-WFS::NearestNeighbours:ByLocationTypeName” can be implemented to provide the nearest neighbor capability on WFS-Gs supplying the virtual global gazetteer. The following table defines the parameters for this stored query:

Parameter Name	Expected Type
Lat	Number
Lon	Number
srsName	URI
locationTypeName	String

The following is an example invocation of the nearest neighbour stored query that returns the nearest objects of type “Mountain”:

http://www.opengeospatial.org/server?service=WFS&version=2.0&request=GetFeature&storedQuery_Id=urn:cw:def:query:OGC-WFS::NearestNeighbours:ByLocationTypeName&locationTypeName=mountain&lat=45.288278&lon=-66.062351&srsName=urn:ogc:def:crs:EPSG::4326&count=10

8.5.2.4 Parameter discussion

All parameters are mandatory.

8.5.3 Bounding-box search

8.5.3.1 Introduction

A bounding-box (BBOX) search finds objects within a coordinate rectangle. It applies to all feature types listed in the request.

8.5.3.2 XML encoding

The BBOX operator was used for this capability, for example:

```
<ogc:BBOX>
        <ogc:PropertyName>iso19112:position</ogc:PropertyName>
                <gml:Envelope srsName="urn:ogc:def:crs:EPSG::4326">
                        <gml:lowerCorner>44.8755955883576 -
66.8650672925836</gml:lowerCorner>
                        <gml:upperCorner>45.6188750089766 -
65.3198488862554</gml:upperCorner>
                </gml:Envelope>
</ogc:BBOX>
```

8.5.3.3 KVP encoding

The following table defines the KVP encoding for the BBox operator:

Parameter Name	Expected Type
LowerCorner longitude	Number
LowerCorner latitude	Number
UpperCorner longitude	Number
UpperCorner latitude	Number
crs URI (optional)	URI

8.5.3.4 Parameter discussion

This encoding allows N coordinates for each corner listed in the order of the optional crsuri. If the crsuri is not specified then the 2-D coordinates shall be specified using decimal degrees and WGS84

9 Architecture Enhancements

Based on the feedback from OWS-9, the basic architecture underlying the Virtual Global Gazetteer shall be improved and where necessary developed.

9.1 Cascading WFS accessing NGA and USGS Gazetteer

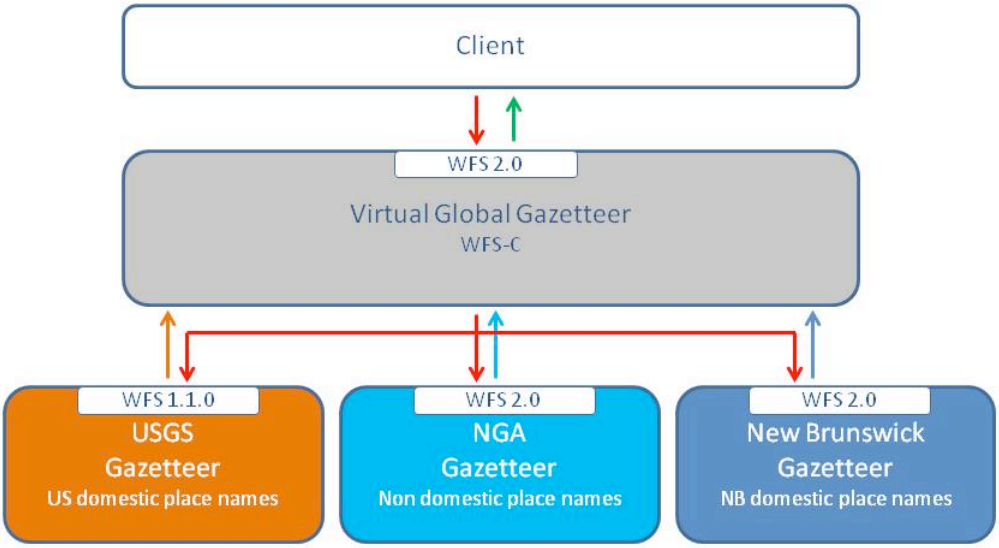
As in OWS-9, the Virtual Global Gazetteer has been implemented as a cascading WFS (WFS-C). The WFS-C basically behaves towards a client like a standard WFS. Instead of holding data itself, it cascades the client request to two or more subsequent WFS, which provide the actual data requested.

The major benefit of this approach is integrated access to distributed feature data sets, without the need for a client to explicitly know the underlying service endpoints and capabilities.

9.2 Architecture

The architecture of the Virtual Global Gazetteer is illustrated in figure 1 below:

Figure 2 - Virtual Global Gazetteer Architecture



On the client side the Virtual Global Gazetteer behaves like any WFS 2.0 service, as specified in OGC 09-025r1.

On the server side the Virtual Global Gazetteer behaves like a WFS client accessing the USGS and NGA gazetteers by cascading the original request to the underlying services.

The initial getCapabilities request from the client is cascaded down to the USGS and NGA WFS. To accommodate the different WFS capabilities, the Virtual Global Gazetteer creates a compatibility matrix from the cascaded responses. Based on this, subsequent requests can be re-written according to each gazetteers capabilities (cf. clause 9.1.2).

The USGS and NGA servers offer the GMLSFL1 encoding of the ISO 19112 model (cf. clause 6).

9.3 WFS-C Implementation Issues

This section of the document discuses a number of technical issues associated with the implementation and deployment of the WFS-C approach. Where applicable, reappearing issues encountered in OWS-9 have been adapted from OGC 12-104.

9.3.1 Compatibility matrix

A WFS-C reports a capabilities document which is based on the capabilities of the servers that it is cascading.

In general, a WFS-C will cascade child servers that support different versions of the WFS standard and that implement different sets of capabilities from the version of the standard they support. For example, the USGS Gazetteer WFS 2.0 supports a number of spatial operators while the NGA Gazetteer WFS 1.1.0 only supports the BBOX spatial operator.

For this reason, a WFS-C must internally create a compatibility matrix so that it has the necessary information to report a merged capabilities document and has the necessary information to rewrite input requests to suit each cascaded WFS.

9.3.2 Merging capabilities document

An important function that a WFS-C must perform is to decide how to merge the capabilities documents reported by all cascading servers in order to report a single capabilities document for the WFS-C.

During the OWS-9 test bed a “minimum common capabilities” approach and a “maximum capabilities” approach were tested. Testbed-10 followed the OWS-9 approach, which is discussed in OGC 12-104.

9.3.3 maxFeatures/count handling

The maxFeatures parameter on a GetFeature request is used to set the maximum number of features returned by WFS in a response document.

There are three ways of handling this parameter in a cascading server.

1. The cascading server applies the maxFeatures value to the entire result set – that is assembled from the responses from each cascaded server.

2. The cascading server passes along the maxFeatures value to each cascaded server and then concatenates all the results returning N x maxFeatures features in the response (where N is the number of cascaded servers).

3. The cascading server processes the responses from each server in a round-robin manner, including one feature in the response for each cascaded server response, until maxFeatures is reached.

The problem with approach (1) is that in most cases only records from the first cascaded server will be included in the response. Approach (2) violates the WFS standard because it actually returns more features than the client requested. Approach (3) seems to strike an acceptable balance between standard compliance and including records from each cascaded server in the response.

The approach used by Image Matters to semantically-enabled existing data stores (Geonames stored in a PostGIS database) and services (USGS, NGA, Geobase WFS-Gs) was explored during Testbed-10 and can be considered as an alternative to the syntactic approach used in OWS9. The Semantic mapping components provided a GeoSPARQL interface on top of existing data APIs (SQL and OGC Query). The benefit of this approach was to provide a unified knowledge representation (based on RDF Model), query language and access protocol (GeoSPARQL) to the data and leverage the existing infrastructure already in place.

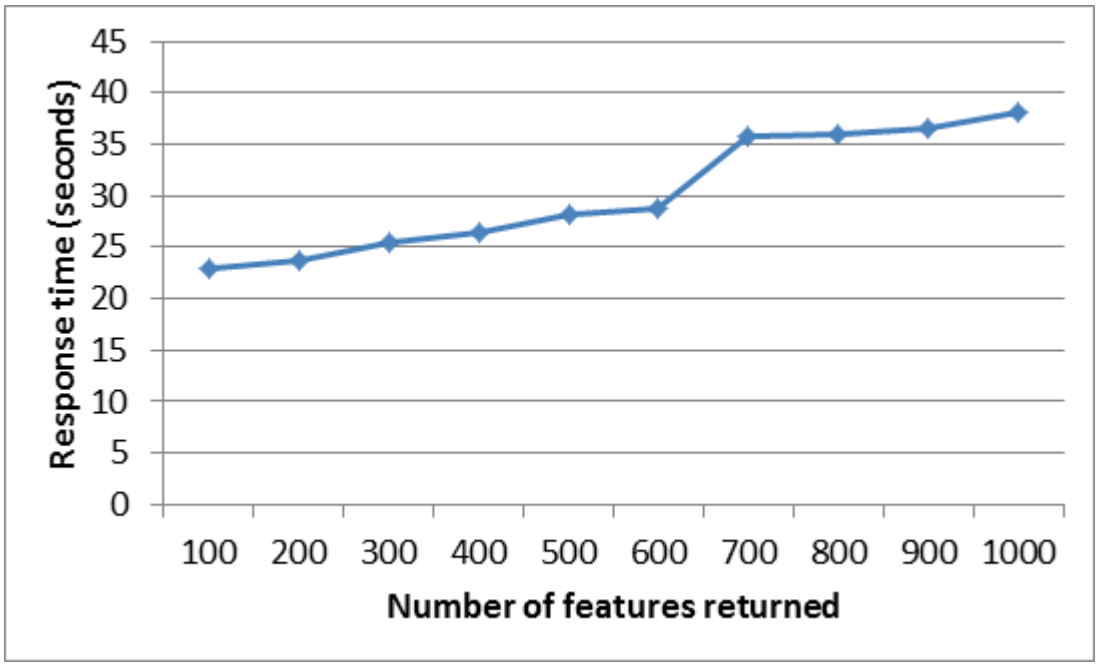
9.4 Performance and effort

The Virtual Global Gazetteer should be designed to access and deliver results from the complete NGA and USGS Gazeteer services.

9.4.1 Performance Tests

The previous testbed did not collect performance metrics. However, one clear difference is that the NGA WFS-G in OWS-9 only included data covering Mexico. In contrast, the NGA WFS-G in Testbed-10 covered the entire Globe (apart from the US). For each semantic query, the Virtual Global Gazetteer transforms a query and sends it to the different WFS-Gs providing data; thereafter, each WFS-G returns data to the Virtual Global Gazetteer which then returns compiles the responses and returns the compiled responses to the client application. The response times of the Virtual Global Gazetteer therefore include the response times of the services that supply data to the Virtual Global Gazetteer. Running a semantic query through the Virtual Global Gazetteer WFS-G, in Testbed-10 and with responses encoded in GML, resulted in the following response times.

Figure 3 – Response times vs. returned features



9.4.2 Caching options

Areas where caching could help to reduce the response times include:

- Caching of data retrieved from the other WFS-Gs such as the NGA, USGS and New Brunswick services.
- Caching of semantic mappings retrieved from the SPARQL Server.

It should be noted however that the low response times (indicating quick responses), imply that there is little value in caching data retrieved from other WFS-Gs.

9.4.3 Fault Tolerance

Due to lack of capabilities and potential problems in service, the addition of fault tolerant functionality needs to be addressed to assure consistent service and provide the user with an understanding of the results returned.

A distributed service environment always comes with the risk of one or more services not being available, or requests being beyond the capabilities of a service. In case a of a cascaded WFS this basically leaves two options to deal with a request, for which a subsequent service fails to respond:

1. Either the entire request fails and raises an exception which is reported to the client,

2. or only the exception from a subsequent server is handed on to the client, along inline with the valid responses from other servers cascaded servers.

Since option 1 isn't a desirable approach, the WFS 2.0 implementation of the Virtual Global Gazetteer in Testbed-10 followed option 2. Exceptions are reported to the client by assembling a response from the successful responses of the cascaded WFSs and including any exceptions inline in the response document:

- Exceptions for WFS 1.1.0 requests are included in-line with the response as XML comments since the WFS 1.1.0 standard does not address the issue of generating in-line exception reports.

- Exceptions for WFS 2.0 requests return an OWS exception report as specified in the WFS 2.0 standard (cf. OGC 09-025r1, clause 10.3), which is included in-line with the response.

10 Semantic Mediation in Testbed-10

10.1 Introduction

Semantic mediation is an approach to overcome differences between servers and the data they offer, such as differences in CRS support, response formats, or semantics. This has been addressed in OWS-9 (cf. OGC 12-103r3).

Since WFS 2.0 is a superset of WFS 1.1, many WFS2.0 requests can be directly rewritten to a WFS 1.1 request, as long as common capabilities are concerned and implementations follow the specifications. It might only require a change in the version number, whilst the request syntax can be kept intact. Where a WFS 2.0 requests a capability which isn't available from a WFS 1.1, this would cause an exception (cf. 9.7.3) that needs to be handled. However, a number of requests can also rewritten based on the capability matrix (cf. 9.1).

10.2 Testbed-10 Approach

Within Testbed-10 a 'thick mediator' approach was chosen to address the differences in capabilities and versions of the implemented WFS. The generic request transformation process is illustrated in the following diagram:

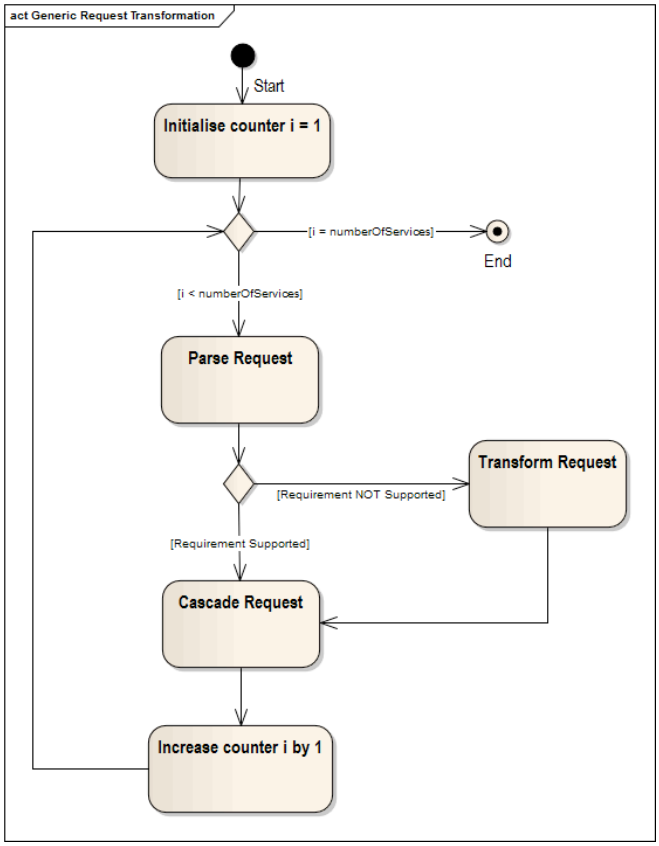


Figure 4 - Request transformation process

10.2.1 Request transformation

The WFS-G Best Practice only specifies an application schema for GML 3.1.1. Whereas the NGA gazetteer services was found to support GML 3.1.1, the USGS and NB gazetteer services were found to support GML 3.2.1 and not GML 3.1.1. The impact of this difference was that requests received by the mediator could not be simply forwarded to all three foundation gazetteer services without modification to specify the GML version supported by the target gazetteer service. The mediator was therefore configured to check the GML version specified in a request and then depending on the target gazetteer service, the mediator either replaced or retained the GML version specified. The following table explains how the request is transformed within the request transformation process presented earlier in this section:

Requirement	**NGA WFS-G**	**USGS WFS-G**	**NB WFS-G**
GML 3.1.1	Cascade	Transform to GML 3.2.1	Transform to GML 3.2.1
GML 3.2.1	Transform to GML 3.1.1	Cascade	Cascade

10.2.2 XPath references

Another issue encountered was that WFS allows a feature property to contain its value as content encoded inline or to reference its value through a simple XLink. It was observed that the USGS and NB services supported queries on properties in line, whereas the NGA service supported queries on properties both inline and by reference. This made it necessary for the mediator to check the XPath expression used to name the locationType and then depending on the target gazetteer service specified, the mediator either replaced or retained the XPath reference with an appropriate one.

Requirement	**NGA WFS-G**	**USGS WFS-G**	**NB WFS-G**
Cascade	Cascade	Transform to locationType inline	Transform to locationType inline
locationType inline	Cascade	Cascade	Cascade

10.2.3 PropertyIsSemanticallyRelatedTo Operator

In OWS-9, an operator was designed to support the use of semantic filters in WFS queries. The new operator, named PropertyIsSemanticallyRelatedTo allowed a client application to filter features by property values that are semantically related to the search term.

Whereas in OWS-9 the location type hyperlinks were used to select relevant mappings, in Testbed-10 the actual place names were used.

The reason for the change of approach is that the WFS-G services used in the current testbed adopted a different URI scheme from the services used in the previous testbed. The impact is that an update is required to the PropertyIsSemanticallyRelatedTo operator proposed in the previous testbed to allow it to accept a choice of hyperlinks or literal values.

10.3 Implementation issues

10.3.1 Stored queries

Stored queries retrieve data based on a predefined filtering. These are supported by WFS 2.0, but not by WFS 1.0.

An example is the Nearest Neighbour capability, which was implemented in the USGS WFS 2.0 server as a stored query. Whilst the WFS-C recognizes the capability for the USGS service, it cannot rewrite that stored query to be executed on the NGA V1.1 WFS.

The issue is further discussed in the Testbed-10 CCI Profile Engineering Report, clause 10.

10.3.2 Advanced filtering operators

The advanced text search operators (cf. 7.1.4) implemented were implemented as an extension in the USGS gazetteer but not the NGA gazetteer. There is no efficient or reliable way that a request containing these operators can be rewritten so that it can be executed on the NGA server.

10.4 Conclusions and future work requirements

Semantic mediation in the general sense is the ability to transform data expressed in one ontology to another one at query time, so end users can retrieve heterogeneous data using their own vocabularies. This transformation is not always isomorphic.

The use of a Simple Knowledge Organization System (SKOS) for semantic mediation is appropriate for transforming taxonomies (which is a lightweight ontology). For example

taxonomies of feature types in gazetteers can be aligned using purely SKOS constructs (exactMatch, narrowerMatch,..). There is nothing wrong with that and it is the right approach to take. This is a very common mechanism used in the library community.

However, in the more general use case, transformations from one ontology to another require a more complex process that involves the use of rules. The hydro WG is trying to tackle this approach. The ImageMatters in Testbed-10 approach for semantic mapping is very similar to the one used by TopQuadrant with SpinMap <http://composing-the-semantic-web.blogspot.com/2011/04/spinmap-sparql-based-ontology-mapping.html>. SPARQL and declarative mapping expressed in RDF are used to perform class and property mapping.

A recommendation is to standardize the vocabulary for expressing semantic mapping, so it can be shared and be processed by machine. There is no consensus in the industry today how to proceed with semantic mediation. Some advocate the use of Rule Interchange Format (RIF), others advocate the use of a unified logic language, others the use of SPARQL. Time will tell what is the best approach.

OGC is tackling a problem that is a very advanced use case while it has not addressed the most fundamental problem, which is how to publish geographic linked data and define the foundational geospatial ontologies (a task we are addressing during this testbed).

Hence the following recommendations for future requirements should be considered in subsequent testbeds and future standardisation work:

10.4.1 Standardization of the core geospatial ontologies

Standardize the core geospatial ontologies, so people can use them to expressed their geographic data as linked data.

10.4.2 Best practices to publish geospatial linked data

Define set of best practices to publish geospatial linked data (use of GeoSPARQL, RDF, OWL, Linked Data Best practices)

10.4.3 Cleanup of the GeoSPARQL standard

Based on feedback from the OGC Geospatial Semantic WG and lessons learned from Testbed-10, there is a need to modularize and simplify GeoSPARQL specification. The Testbed-10 geospatial ontologies address many of these aspects (for example modularization of spatial relations), and could be used as a starting point. GeoSPARQL provides a geospatial extension of SPARQL by defining geospatial function extensions and data types (in the same way Spatial SQL extends SQL). GeoSPARQL could be simplified by clearly defining geospatial functions using SPARQL Service Description vocabulary standard and the geospatial datatypes (WKT Literal and GML Literal). The query language should be independent of the ontology describing geospatial concepts (the same way Spatial SQL is independent of relational models). We believe that this

simplification and modularization of the specification will foster the adoption of the specification to a larger community.

There is also a need to describe the capabilities of GeoSPARQL. These descriptions provide a mechanism by which a client or end user can discover information about the SPARQL service such as supported extension functions and details about the available geospatial dataset, supported CRSs, inferences supported. There are a number of standards that already exists to describe SPARQL endpoint such as the W3C SPARQL 1.1 Service Description, VoiD and DCAT. These standards will be adapted to accommodate description of GeoSPARQL endpoints, by defining profiles and best practices. The next testbed should demonstrate the feasibility and robustness of the approach by implementing the specifications.

We believe that this simplification and modularization of the specification will foster the adoption of the specification to a larger community.

10.4.4 Definition of vertical ontologies

Define vertical ontologies based on the core geospatial ontologies (example E&DM, Hydro, Gazetteer)

10.4.5 Migration of OGC services to REST-based semantic enabled web services

Migrate all OGC services to REST-based semantic enabled web services. This implies expressing their capabilities in RDF based on existing standards (DCAT, VOID, ADSM, ...) and descriptions of processes in RDF (parameters, constraints etc). I would start with Gazetteer, CSW, WFS and WPS.

11 Gazetteer Linking

Gazetteer Linking is based on the premise that features in a gazetteer and other sources of information have already been matched and the match between identifiers is stored in a concordance or is embedded in a data source. The former is known as a concordance link and the latter as an embedded link.

To take full advantage of the semantic web and ability to quickly move across links, the data sets should be encoded in an RDF. The goal of this task was to encode information in RDF from multiple gazetteers by leveraging existing infrastructure (WFS-Gs, RDBMS) using semantic mapping components, demonstrate a capability to list new information available from related resources (obtaining information from sources at least two sources distant from the original source), query and select the information of interest, and return the information in a query. This was done using open linked data standards (RDF,OWL,SPARQL) and the OGC GeoSPARQL query.

Going back to the Testbed-10 demo scenario, a quick look at the NGA gazetteer shows that extended information such as websites related to the city, the geography, typical weather etc. is not readily available.

They need to '***Get More Stuff***.' The purpose of the Gazetteer Linking demonstration is to show the value of linking for obtaining additional attribute and spatial information, i.e. information that can be used to find additional information.

11.1 Gazetteer Linking Concept

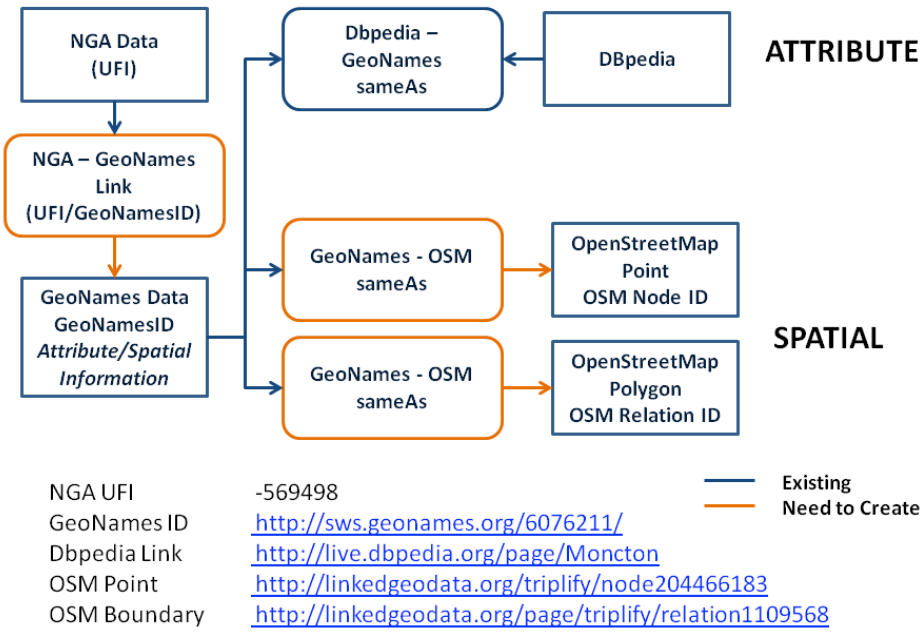


Figure 5 – Gazetteer linking concept

The general concept is to select a name from the NGA data store (which will be in RDF) - if there is more information available from other sources, a 'Get More Stuff' button will be displayed

The link to GeoNames will be done from the information provided in the NGA - Geonames.org ID Links file (cf. 12.1.3), which contains the NGA - GeoNames links for features which were extracted from NGA. This is useful for more global coverage, but unfortunately doesn't cover the New Brunswick names. These are found in the New Brunswick Populated Place Links Excel spreadsheet (cf. 12.1.5), which includes information on New Brunswick populated places, including NGA Unique Feature Identifier (UFI), GeoNames ID, and OpenStreetMap (OSM) IDs for representations as Nodes (for points) as well as Ways or Relations (for polygons).

The goal in Testbed-10 was to show that from the NGA data, you can 'Get More Stuff", including crowdsourced attribute information as well as spatial information.

11.2 Interface Concept

The graphic below illustrades the basic concept - once the results from the NGA Gazetteer are retrieved, you can pick the result of interest and see the NGA data

Moncton, New Brunswick (NGA UFI -569498) Get More Stuff

GNS Extended View	
Name (Gazetteer Order/No Diacritics):	Moncton
Name (Reading Order/No Diacritics):	Moncton
Name Rank:	NULL
Name Link:	NULL
Short Name (No Diacritics):	NULL
Generic (No Diacritics):	NULL
UFI:	-569498
UNI:	-804391
DD Lat:	46.09652
DD Long:	-64.79757
MGRS Coordinate:	20TLS6105306342
JOG Reference:	NL20-04
Feature Designation Code:	PPL
Feature Class:	P
Population:	NULL
Elevation (meters):	NULL
Script Name:	NULL
Transliteration Code:	NULL
Dialect Name:	NULL
Creation Date:	1993/09/01
Modify Date:	2012/11/05
Effective Date:	NULL
Termination Date:	NULL

Figure 6 – interface concept

Click the Get More Stuff button displays a list of related resources:

If you select a resource, you get a list of attributes and/or spatial information. You select the ones you are interested in to build a template. This could be messy in the beginning, as some of these sources have a lot of attributes.

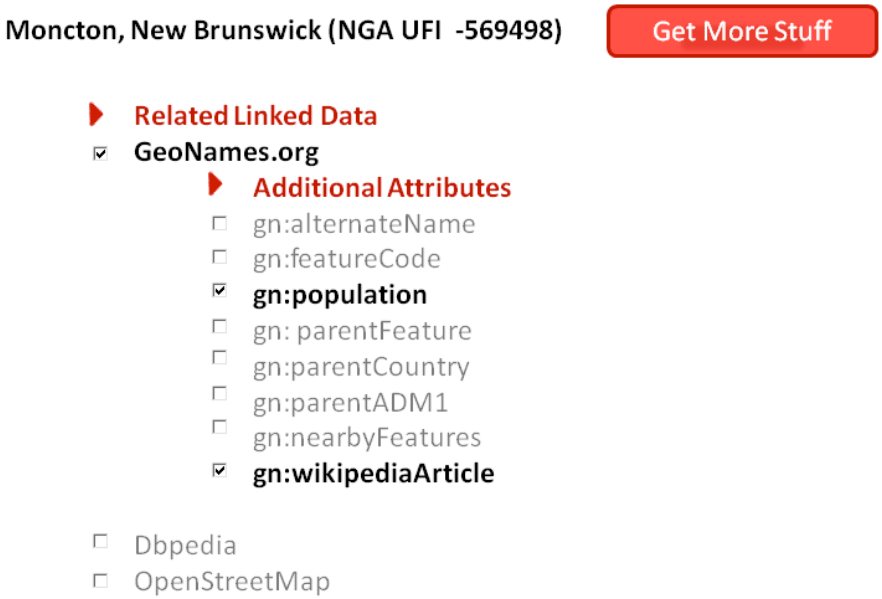


Figure 7 – concept of selecting resources

11.3 Semantic Mapping components

Until today, data integration has been accomplished using a single layer approach by writing data product translator from one format to another. For example, it is common practice today to use XSLT to transform one XML document to another XML format. The problem with this approach is that it mixes the structural and semantic transformation together. Also it does not scale, because it is based on a N-to-N mapping approach, and is error-prone due to reliance on human interpretation of data products.

The rules, which carry out the complete transformation process in one shot, have proven to be very complex. This causes serious problems in implementing and maintaining the rules of transformation. These problems arise due to the mixture of several different aspects of the overall transformation process, such terminology, granularity representation and structural and syntactic alignment. For this reason, any re-use of such rules is practically impossible.

To overcome this bottleneck a multi-layered framework should be used, which separates different aspects of the transformation process. The approach used in Image Matters Knowledge Mapping Service (KMS) is able to transform a complex programming task into a simple plug-and-play process where straightforward rule patterns are selected, instantiated, and combined. KMS uses a methodology for data integration based on a three-layer model, as presented in the figure below. The model contains a Data Product layer, a Data Model layer, and an Ontology layer.

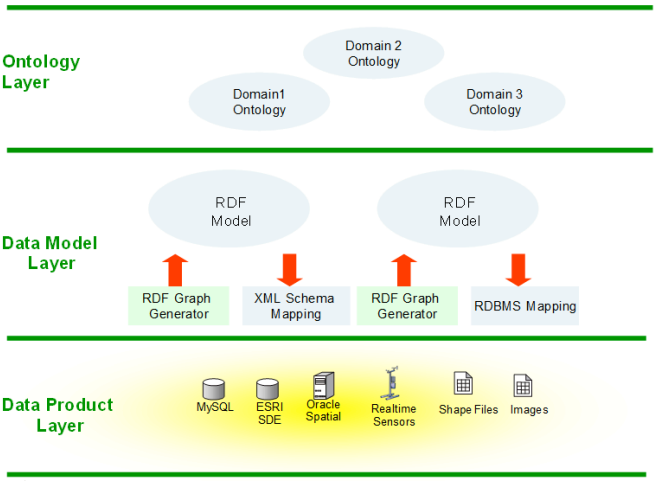


Figure 9 – KMS layer model

KMS provides the ability to map 'legacy' (geospatial or not) data stores and formats to a RDF knowledge representation using a unified declarative mapping expressed in RDF. KMS uses this mapping to translate semantic query (graph query, SPARQL,...) to native query language (such as Spatial SQL, XPath/XQuery, OGC Filter) or API calls. This framework allows the virtualization of the data into a semantic graph representation and provides real-time access to data into a unified semantic representation, which could be leveraged by other knowledge-centric service components (reasoners, query engine, (Geo)SPARQL endpoints, semantic mediation, visualizations).

For this tesbed, Image Matters investigated the semantic mapping of the database dump from Geonames.org database and USGS, NGA and New Brunswick WFS-G services. The semantic Mapping component was used to offer virtual GeoSPARQL endpoints over the mapped database and services. This approach provided a unified knowledge representation, query language and protocol to access existing gazetteer data infrastructure as illustrated in the figure below:

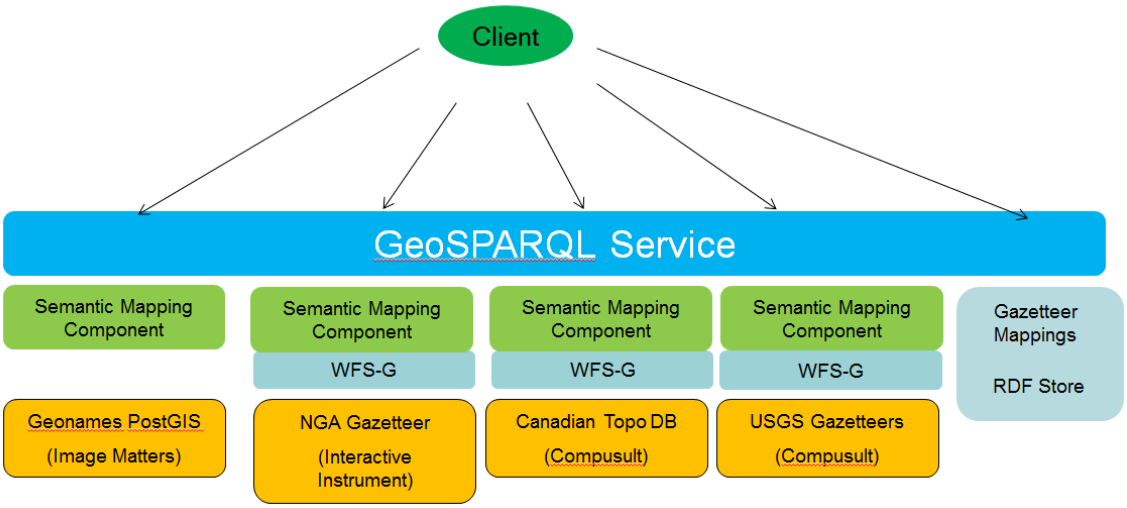


Figure 10 – semantic mapping approach

11.3.1 Geoname semantic mapping

For this project, a database dump of Geonames was installed and indexed in a PostGIS database instance on Image Matters Server. KMS Semantic mappings from relational database to RDF dataset were defined. Such mappings provide the ability to view existing relational data in the RDF data model, expressed in a structure and target vocabulary (ontology) aligned with the ISO 19112 model. The mappings are themselves RDF graphs and written down in Turtle syntax. The KMS processor was adapted to support directly geospatial functions defined in GeoSPARQL specification. The database was made accessible through a GeoSPARQL endpoint at the following address: http://ows10.usersmarts.com/ows10/gazetteers/geonames/sparql. GeoSPARQL queries sent to the server were translated to one or more spatial SQL queries and results were converted on the fly in RDF form. Using this approach performance of the system was similar to the native query as the overhead consists mainly to query rewriting and serialization in RDF form for sending final results to the client.

11.3.2 WFS-G Mapping

For this project, Image Matters upgraded its existing KMS plugin for WFS 1.0 to WFS 1.1. The plugin uses a semantic mapping from the GML schema to the same target ontology used for geonames (aligned with ISO 19112). The KMS processor is capable to convert GeoSPARQL query to one or more OGC Filter Query automatically based on this mapping information. The mapping approach used by KMS could be seen as generalization of W3C standard R2RML that performs mapping from RDF to RDBMS model.

For this testbed, Image Matters integrated three instances of WFS-G implementations. The same schema mapping was used for all three instances, as the WFS-G uses the same GML schema to represent locations in the gazetteers.

The first WFS-G instance was provided by Interactive Instrument hosted at http://services.interactive-instruments.de/xsprojects/ows10/service/gazetteer-simple/wfs?SERVICE=WFS&REQUEST=GetCapabilities providing NGA data.

http://ows10.usersmarts.com/ows10/gazetteers/usgs
This GeoSPARQL endpoint maps to the Compusult WFS-G
http://ows-10.compusult.net/wfs/services/?SERVICE=WFS&VERSION=1.1.0&REQUEST=GetCapabilities
providing USGS data

http://ows10.usersmarts.com/ows10/gazetteers/newbrunswick

This GeoSPARQL endpoint maps to the Compusult WFS-G
http://ows-10.compusult.net/wfs/services/?SERVICE=WFS&VERSION=1.1.0&REQUEST=GetCapabilities
providing New Brunswick data

The knowledge mapping to WFS-G required two steps. The first step consists of defining a simple ontology for representing Location. The second was to build a bridge that converts GeoSPARQL queries into one or more OGC Filter queries. To build the capabilities, a number of Java open sources were investigated (GeoServer, GeoToolkit, Deegree, Geotools). While most of these libraries are capable to build WFS server, very few were supporting WFS client side API. We found no open source capable to handle GML complex features response from WFS. Geotools seemed to be the only robust library capable to support simple feature response from WFS. The lack of support of complex features on client side rise the question whether or not WFS clients are too complex to be implemented to support any GML complex schemas served by WFS in a generic way. Due to limited time and budget, we used Geotools to implement a KMS plugin for WFS supporting only simple features.

The performance of the GeoSPARQL endpoints on simple features were similar to the performance of the WFS, but not as good as the direct mapping to RDBMS used in Geonames. This is due to the level of indirection and deserialization of the GML results to be converted to RDF. The OGC Query also do not provide fine grained query results as in GeoSPARQL or SQL.

Another issue found during the semantic mapping of WFS was the issue with reference to other Feature using XLink technique. The xlink:href to LocationType in WFS-G uses a WFS query pointing to GML document. This is a problem when you just want to get the non-information resource URI (i.e. in GML context the namespace and the identifier

(gml:id) of the resource). To perform semantic mapping from XML to RDF, the current approach requires an expensive call to WFS, parsing a new GML document to extract its gml:id so reconstruction of the non-information resource URI can be done (namespace+gml:id). Xlink (and RDF references for that matter) will only work if you use a non-information resource URI and allow HTTP resolution to find the document form. This is issue that would require future resolution as it may be a major blocker for performing semantic mapping for complex features.

11.4 Best Practices for Gazetteer Data in RDF

We recommend that existing gazetteers remains in the original storage (typically RDBMS) and use a semantic mapping approach that expose the data using GeoSPARQL protocol but also Linked Data REST API. This simplifies not only the access to geospatial data by using W3C Linked Data standards and best practices but also leverages existing optimization for spatial queries already in place in databases.

The idea here is to take a look at both standards and various implementations of linked data to develop the best practices for gazetteer data, such as:

- GeoNames.org ontology
- LinkedGeoData (OpenStreetMap)
- Ordnance Survey Linked Data Platform

11.5 Process

11.5.1 Preparation:

- Make WFS-G NGA data available in RDF
- SPARQL (**KMS** ImageMatters) will convert the GML to RDF

11.5.2 Sequence:

1. client presents NGA Gaz data for WFSG
2. user request getting links to crowdsourced data (**find more**) for a particular feature
3. client invokes a GeoSPARQL to KMS to get all links related to a feature
4. KMS returns an RDF - Maybe based on Geonames model or a GeoSpatial ontology?

5. Client gets RDF - present human readable RDF to the user
6. The user picks properties to customize a report -> template
7. Client generates a report based on the template.
8. Also the client is able to get geometries from other sources such as ehydro,OSM or OS, CSIRO

11.6 Recommendations

We propose to semantically-enable existing gazetteers by defining a new linked data REST API and GeoSPARQL based on the Testbed-10 Geospatial Ontology. The Geospatial Ontology will provide a solid foundation for defining a gazetteer ontology describing places of interest, toponyms and geospatial-temporal location. The gazetteer ontology should accommodate historical gazetteers, multiple geometries, multilingual requirements and different taxonomies for place types. The model will be designed to meet minimum-essential place information exchange requirements, while accommodating custom extensions using built-in extension mechanisms provided by RDFS and OWL. Places and Locations will be returned in RDF-compatible Linked Data formats (RDF/XML, Turtle, JSON-LD, NTriples). This will also support the linking of place instances to other relevant/related information (DBPedia, Geonames, Social Media, etc.) by leveraging standard Semantic Web technologies (RDF, HTTP, URLS).

The testbed should demonstrate the feasibility and robustness of the resulting Semantic Gazetteer and candidate specification by testing one or more implementations.

12 Gazetteer Conflation

Gazetteer conflation is the process of matching entries from multiple names sources, sharing or replacing attribute information, and presenting the fused results to users. This task is becoming more important with the proliferation of international, national, state, and crowdsourced gazetteers. This matching process enables a gazetteer producer to identify common features across sources as well as update and enhance existing sources. Gazetteer Conflation uses point-to-point conflation of data sets with limited attribution - basically a name and feature description.

12.1 Automated Gazetteer Conflation

In the first use case, the goal is to take a WFS-G gazetteer service (referred to as A) and match it with a data from another service (WFS-G or WFS) (referred to as B) that contains more accurate and/or more current information, displaying the conflated results. The assumption is that the information in A is inferior to the data in B and will be replaced by the information from B in cases where a match is found. The process follows the same basic steps as outlined for the transactional gazetteer conflation in the following section, except for steps 6 and 7 which are performed based on rules

12.2 Transactional Gazetteer Conflation

Automated Gazetteer Conflation is an optimistic approach, assuming that one data service is superior to another in every respect and can be used to replace the information without inspection. A more realistic scenario, Transactional Gazetteer Conflation, evaluates names one at a time, puts an analyst in the loop, and lets the analyst determine which positional and attribute information is transferred from the target service to the source service. In this use case, the analyst extracts a series of records for conflation and then steps through the set of records.

12.2.1 Conflation Process

1. The user searches for: feature types in a constrained area (e.g. all stations in Canada).
2. The client invokes the Global Gazetteer - that will return data from multiple gazetteers.
3. The client presents a map to the user. For example black and red dots for each different gazetteer.
4. The client presents the option to the user to conflate.
5. Client connects to a WPS "Synchronously" - and provides the getfeature requests of the gazetteers to be conflated.

6. WPS - ranks based on spelling an proximity for a possible match - returns a table with matches.
7. The client presents a map of the NGA feature and the matches with other gazetteer features and allows the user to select a new name or new position for a particular feature.
8. The Client presents an updated map.
9. The Client updates the information doing a WFS-T to the NGA WFS-G. (See more about Transaction Requirements)

12.3 Implementation

This scenario matches the NGA gazetteer populated place features with the New Brunswick gazetteer populated place features, creating a table of links between the two data sets where there are matches.

Using the proposed scenario will produce quite good, although not perfect, results. Names listed as matching will almost certainly be matches. The approach won't catch cases where the names are completely or significantly different, i.e., Moncton (Source) and Blair (Target) or even Fairfax (Source) and Fairfax Station (Target). This latter case is particularly difficult as names which include parts which are identical may or may not actually be matches, i.e., Fairfax (Source) and Fairfax South (Target).

12.3.1 User Inputs

The user must enter the following information to start the conflation process ...

- **Source Gazetteer**
  names from the target gazetteer will be matched against the source
- **Target Gazetteer**
  names from the target gazetteer will be matched against the source
- **Source Gazetteer Description Filter**
  feature descriptions or types to be used for the analysis
- **Target Gazetteer Description Filter**
  feature descriptions or types to be used for the analysis
- **Bounding Box Filters**
  the bounding box used to select source features for the analysis
- **Search DIstance**
  distance to search for target features around source feature

- **FuzzyWuzzy Threshold**
  name matching threshold to determine a match

- **Output File**
  name of the sameAs output RDF file containing the links

12.3.2 Source Parameters

- **Source Gazetteer**
  assumed to be WFS-G from NGA

- **Target Gazetteer**
  assumed to be WFS-G for New Brunswick gazetteer from NRCan

- **Source Gazetteer Description Filter**
  PPL, PPLA, PPLA2, PPLA3, PPLA4, PPLC, PPLF, PPLH, PPLL, PPLQ, PPLR, PPLS, PPLW,PPLX, STLMT

- **Target Gazetteer Description Filter**
  CITY, TOWN, UNP, VILG, MUN1

- **Bounding Box Filter**
  user-defined (needs to be in New Brunswick)

- **Search DIstance**
  15 miles

- **FuzzyWuzzy Threshold**
  > 80

- **Output File**
  user-defined

12.3.3 Source Data Preparation (Filtering)

The Source Gazetteer (NGA) features are filtered first by the Bounding Box Filter and then by the Source Gazetteer Description Filter.

The Target Gazetteer (New Brunswick) features are filtered by the Target Gazetteer Description Filter. They are not filtered by the Bounding Box Filter in order to prevent edge effects at the corner of the bounding box.

The filtering produces the data that will be used during matching. Source Gazetteer features are processed sequentially, one at a time.

12.3.4 Feature Level Processing - Select Target Features Within N Miles of Source Feature

The feature level processing applies to each feature in the Source Gazetteer. In the example illustrated below, the Source Gazetteer feature is Welshpool (NGA UFI - 575731). All features from the Target Gazetteer within the Search Distance are used to identify the best match.

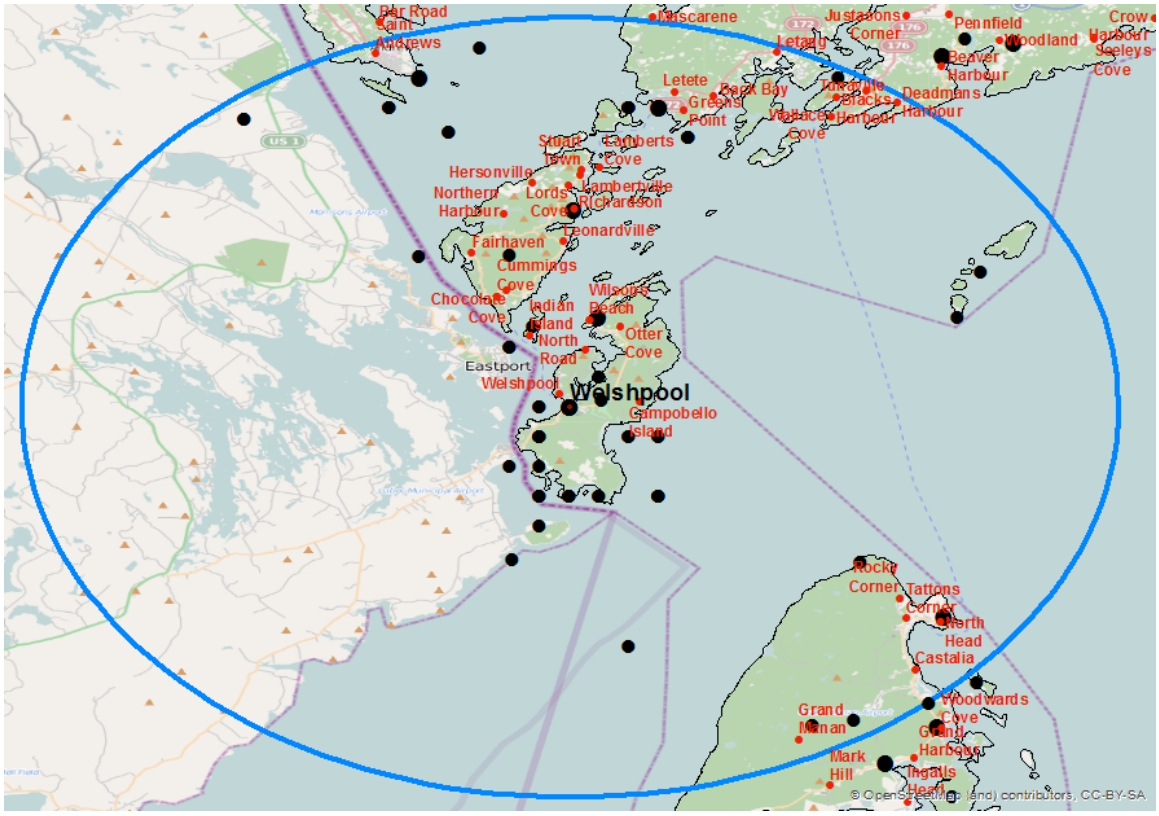


Figure 11 – spatial feature selection concept

12.3.5 Feature Level Processing - Calculate FuzzyWuzzy Score and Distance

For each Source Gazetteer feature, the FuzzyWuzzy score and distance is calculated for all the Target Gazetteer features within Search Distance miles.

The Source Gazetteer feature and the Target Gazetteer feature should be converted to an ASCII equivalent prior to performing the analysis. This will improve the string matching process.

FuzzyWuzzy is string matching code that is available and described at https://github.com/seatgeek/fuzzywuzzy. The Token Sort Function is used to calculate the string match code. It may have values from 0 to 100. 100 means a match according to the algorithm and is not necessarily equivalent to an exact match.

For more information about FuzzyWuzzy cf. http://seatgeek.com/blog/dev/fuzzywuzzy-fuzzy-string-matching-in-python

The distance (in miles) between the Source Gazetteer feature and each Target Gazetteer feature is calculated.

12.3.6 Feature Level Processing - Sort Results by FuzzyWuzzy Score (Descending) and Distance (Ascending)

The results of the previous step will be a list of all the Target Gazetteer features within the Search Distance of the Source Gazetteer feature. This list should be sorted from the highest FuzzyWuzzy score (FW Score in table) to the lowest, and then the closest distance to the furthest distance (Dist (MI) as illustrated int the following table:

FW Score	Dist (MI)	NGA_UFI	NB_ID	NGA_NAME	NB_NAME
100	0.568670301071	-575731	21f82ebdd05511d892e2080020a0f4c9	Welshpool	Welshpool Greens
38	117.960.536.757	-575731	0c7ee255849c20c3105bbb972007b17e	Welshpool	Point Wilsons
36	338.417.264.605	-575731	b4ed5f1dc6cd11d892e2080020a0f4c9	Welshpool	Beach Blacks
35	145.766.221.898	-575731	0c7f0397849c20c3ca332213f527ab7b	Welshpool	Harbour Chocolate
35	479.447.673.102	-575731	9e3c598ec6cd11d892e2080020a0f4c9	Welshpool	Cove Deadmans
32	147.292.093.006	-575731	0c7c3432849c20c327521a465995b2ca	Welshpool	Harbour
32	852.704.177.796	-575731	0c7df94e849c20c34977f27f1d1c05c3	Welshpool	Lords Cove Campobello
31	192.485.792.397	-575731	60a8b28a0a001204793f24c94b13bab6	Welshpool	Island
30	866.487.724.863	-575731	0c7c651f849c20c3615caac8f5a2e2f7	Welshpool	Hersonville
29	89.054.288.367	-575731	0c806c98849c20c3822b4d30093fbdba	Welshpool	Lambertville
29	636.899.012.498	-575731	ac95e69dc6cd11d892e2080020a0f4c9	Welshpool	Leonardville Lamberts
27	924.579.133.528	-575731	0c7dd423849c20c31daf052ab225d320	Welshpool	Cove
21	130.864.650.527	-575731	934d8c7ad05511d892e2080020a0f4c9	Welshpool	North Head
21	759.067.800.756	-575731	b7e47e57c6cd11d892e2080020a0f4c9	Welshpool	Richardson
21	337.869.271.395	-575731	0c804116849c20c3711edeb1db0b3c6f	Welshpool	Otter Cove
21	224.341.926.014	-575731	0c7972f9849c20c3de2ca9f1cdde9da0	Welshpool	North Road Wallace
19	132.671.563.423	-575731	0c7da0a4849c20c3b68501eb48922138	Welshpool	Cove Rocky
19	116.536.626.255	-575731	8c927c28d05411d892e2080020a0f4c9	Welshpool	Corner Saint
18	145.762.755.497	-575731	0c7a13ad849c20c3067b1c808b2465b7	Welshpool	Andrews Cummings
18	467.501.843.818	-575731	af6fbad4c6cd11d892e2080020a0f4c9	Welshpool	Cove Tattons
17	122.892.026.008	-575731	98027e78d05411d892e2080020a0f4c9	Welshpool	Corner

13	147.722.289.848	-575731	aca094fec6cd11d892e2080020a0f4c9	Welshpool	Letang
13	124.421.381.365	-575731	acbe2f21c6cd11d892e2080020a0f4c9	Welshpool	Letete
12	138.397.946.329	-575731	8dc46c93d05411d892e2080020a0f4c9	Welshpool	Castalia
11	139.737.929.809	-575731	0c80014c849c20c3dde516368eb2beb8	Welshpool	Tunaville
11	647.770.827.227	-575731	ba4e79b3c6cd11d892e2080020a0f4c9	Welshpool	Fairhaven
10	910.436.846.834	-575731	0c7f76c9849c20c381c323c50355613d	Welshpool	Stuart Town
9	295.333.829.532	-575731	93c7938bd05411d892e2080020a0f4c9	Welshpool	Indian Island
8	763.806.787.206	-575731	98050227c6cd11d892e2080020a0f4c9	Welshpool	Northern Harbour
0	142.509.490.603	-575731	0ceed7c9849c20c3d4c6ccd30ccd87d9	Welshpool	Grand Manan
0	125.672.591.357	-575731	0c7a0b5a849c20c3fced6585220d83ba	Welshpool	Back Bay

12.3.7 Feature Level Processing - Select Results > FuzzyWuzzy Threshold

All results greater than or equal to the FuzzyWuzzy Threshold should be kept and the top result will be added to a list of all results. In this case the top result has a FuzzyWuzzy score of 100.

Because of the sorting by FuzzyWuzzy and then distance, any ties in FuzzyWuzzy score would be broken by the closer/lower distance value.

In the unlikely event that both the FuzzyWuzzy score and the distance are identical, either choice would be acceptable.

FW Score	Dist (MI)	NGA_UFI	NB_ID	NGA_NAME	NB_NAME
100	0.568670301071	-575731	21f82ebdd05511d892e2080020a0f4c9	Welshpool	Welshpool

12.3.8 Source Gazetteer Dataset Processing - Combine All Results

The individual Source Gazetteer results are combined into a new table of all the results.

As the example below shows, there are multiple entries for the NGA UFI -575731. These include the names Welshpool and Welchpool, which are represented by separate rows (with differing Unique Name Identifiers - UNIs) in the NGA gazetteer.

FW Score	Dist (MI)	NGA_UFI	NB_ID	NGA_NAME	NB_NAME
89	0.568670301071	-575731	u‘21f82ebdd05511d892e2080020a0f4c9	Welchpool	Welshpool
100	0.568670301071	-575731	u'21f82ebdd05511d892e2080020a0f4c9	Welshpool	Welshpool

12.3.9 Source Gazetteer Dataset Processing - Select Best Result > FuzzyWuzzy Threshold

For each UFI, the highest scoring result is kept, as shown below. If possible, the original name (with diacritics) would be kept in the output.

FW Score	Dist (MI)	NGA_UFI	NB_ID	NGA_NAME	NB_NAME
100	0.568670301071	-575731	21f82ebdd05511d892e2080020a0f4c9	Welshpool	Welshpool

12.3.10 Feature Level Processing - Export Results

The exported data can be provided in several ways.

12.3.10.1 RDF results

If there were a New Brunswick gazetteer in RDF form, a sameAs RDF would provide the link between the NGA and New Brunswick gazetteers. This would be all that is required for linking, but wouldn't maintain any of the link lineage information.

Discussion:

To denote that a place in a gazetteer is the 'same' as another one in another gazetteer, the intuitive way is to use the **owl:sameAs** relation. However owl:sameAs has been misused in many existing linked data due to misunderstanding of the rules of inference defined in OWL. The following paper discusses some of the issues with the misuse of owl:sameAs: http://www.w3.org/2009/12/rdf-ws/papers/ws21.A

Instead a separate property was proposed **gaz:sameLocationAs**. This property is transitive and symmetric, so it will infer the mapping on other instances.

However to be more precise in the nature of the similarity linking resulting from the conflation process, a reification of the relationship could be used to add more attributes such as score of similarity and details of score or provenance information. The concept of **SimilarityLink** was introduced, as illustrated in the following example.

WPS RDF Example:

```
@prefix id: <http://www.opengis.net/ont/identifier#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix gaz: <http://www.opengis.net/ont/gazetteer#> .
@prefix prov: <http://www.w3.org/ns/prov#>.
@prefix wps: <http://www.opengis.net/ont/wps/conflation#> .

:ConflationResult1 a wps:LinkSet;
 wps:numResults 25;
 prov:generatedBy :WPSProcessExecution1; # Provenance information
```

```
 wps:hasLink [
 a wps:SimilarityLink ;
 wps:entity1 <http://earth-info.nga.mil/gns#-575731> ;
 wps::entity2
<http://www.nrcan.gc.ca/resource/21f82ebdd05511d892e2080020a0f4c9> ;
 wps:score 0.8 ; # aggregated score
 wps:scoreDetails [
 a wps:ScoreDetail;
 wps:distanceInMiles 0.568670301071;
 wps:fuzzyWuzzy 100
 ],
 [
 a wps:SimilarityLink ;
 wps:entity1 <http://earth-info.nga.mil/gns#-560089> ;
 wps::entity2
<http://www.nrcan.gc.ca/resource/0c7ee255849c20c3105bbb972007b17e> ;
 wps:score 0.8 ; # aggregated score
 wps:scoreDetail [
 a wps:ScoreDetail;
 wps:distanceInMiles 11.7960536757
 wps:fuzzyWuzzy 36
 ],
 [
 a wps:SimilarityLink ;
 wps:entity1 <http://earth-info.nga.mil/gns#-560089> ;
 wps::entity2
<http://www.nrcan.gc.ca/resource/b4ed5f1dc6cd11d892e2080020a0f4c9> ;
 wps:score 0.8 ; # aggregated score
 wps:scoreDetail [
 a wps:ScoreDetail;
 wps:distanceInMiles 3.38417264605
 wps:fuzzyWuzzyScore 35
 ].


<http://earth-info.nga.mil/gns#-560089> a gaz:Location;
 rdfs:label 'Welshpool';
 id:identifier "-560089".


<http://www.nrcan.gc.ca/resource/21f82ebdd05511d892e2080020a0f4c9> a
gaz:Location;
 rdfs:label 'Welshpool';
 id:identifier "0c827a26849c20c3b46eac09a9a02702".

<http://www.nrcan.gc.ca/resource/0c7ee255849c20c3105bbb972007b17e> a
gaz:Location;
 rdfs:label 'Greens Point';
 id:identifier "0c7ee255849c20c3105bbb972007b17e".


<http://www.nrcan.gc.ca/resource/b4ed5f1dc6cd11d892e2080020a0f4c9>> a
gaz:Location;
 rdfs:label 'Wilsons Beach';
 id:identifier "b4ed5f1dc6cd11d892e2080020a0f4c9>".
```


12.3.10.2 CSV File

Alternatively, the final table could be used to create a CSV file that would include the scores, ids, and names that were used in the match.

Other options for CSV files would include NGA features that were not matched and NRCan features that were not matched. This would give a better idea of the overall matching process.

12.3.11 Result handling in the Virtual Global Gazetteer Client

In the Virtual Global Gazetteer Client the results are returned in a tabular format. If the number of records returned is fewer than the total number of records identified by the query, then the user has the option of seeing the total number of results and paging through the results until all the records are displayed.

The user has the ability to sort the results by name or feature description. In addition, the user has the option of sorting the records from nearest to furthest away, based on a near spatial query.

If any processing errors occur, e.g. the results of the query are incomplete due to lack of WFS-G support by any of the servers, the client shows an according message (cf. 9.4.3).

13 Conclusions and Recommendations

13.1.1 Managing potential failovers

Improvements are desirable to the aspect of managing potential failover when using GeoSPARQL endpoints. Whilst this offers extended semantic mediation capabilities, it leads to additional questions of whether an additional component will be required in future testbeds for managing the failover.

Perhaps a registry or WPS could keep track of all of the ontologies to ensure that they are all online; possibly sending out an alert when an ontology becomes unavailable.

13.1.2 WFS-G: PropertyMatches operator

The successful use of the PropertyMatches operator confirmed its utility and potential role in future WFS-G usage. It is therefore recommended that this operator be included in future revisions on OGC standards (cf. 8.2.4).

13.1.3 WFS-G: Radial Search support

WFS-G shall by default support radial search through DWithin operators (cf. 8.5.1)

13.1.4 WFS-G: Use of Parent Property for locations

The parent property is not appropriate for locations due to its implication of inheritence (within Object Oriented modeling). Another reason is that the parent property implies a parent-child relationship whereas its role attribute may imply different relationships such as 'in_country', potentially contradicting the parent relationship. This could lead to incorrect intepretation of values due to ambiguity of the parent property or its role attribute.
The resulting change request suggests to replace the parent property with a 'relation' property with tagged value for role.´

13.1.5 WFS-G: Ambiguous top level container

The WFS-G BP specifies that the top level container should be an iso19112:SI_Collection element but then gives an example that uses a wfs:FeatureCollection element.This will lead to exceptions if clients do not know what top-level container (root element) to expect.
The resulting change request suggests to constrain the WFS-G Response to providing wfs:FeatureCollection as the top-level container.

13.1.6 WFS-G: Update WFS-G Best Practices for WFS 2.0

It is recommended that the WFS-G BP document is updated for WFS 2.0 and the Filter Encoding Specification 2.0, both of which were released in 2009. This would allow WFS-G services to take advantage of WFS 2.0 features such as stored queries. The PropertyContains operator described in 8.2.3 is an extension of the Filter Encoding Specification 2.0 and would not be available for WFS 1.1 services.

13.1.7 Semantic Gazetteer Ontology and API

We propose to semantically-enable existing gazetteers by defining a new linked data REST API and GeoSPARQL based on the Testbed-10 Geospatial Ontology. The Geospatial Ontology will provide a solid foundation for defining a gazetteer ontology describing places of interest, toponyms and geospatial-temporal location. The gazetteer ontology should accommodate historical gazetteers, multiple geometries, multilingual requirements and different taxonomies for place types. The model will be designed to meet minimum-essential place information exchange requirements, while accommodating custom extensions using built-in extension mechanisms provided by RDFS and OWL. Places and Locations will be returned in RDF-compatible Linked Data formats (RDF/XML, Turtle, JSON-LD, NTriples). This will also support the linking of place instances to other relevant/related information (DBPedia, Geonames, Social Media, etc.) by leveraging standard Semantic Web technologies (RDF, HTTP, URLS).

The testbed should demonstrate the feasibility and robustness of the resulting Semantic Gazetteer and candidate specification by testing one or more implementations.

13.1.8 Standardization of the core geospatial ontologies

Core geospatial ontologies should be standardized, so people can use them to expressed their geographic data as linked data.

13.1.9 Best practices to publish geospatial linked data

Best practices to publish geospatial linked data (use of GeoSPARQL, RDF, OWL, Linked Data Best practices) should be defined.

13.1.10 Cleanup of the GeoSPARQL standard

Based on feedback from the OGC Geospatial Semantic WG and lessons learned from Testbed-10, there is a need to modularize and simplify GeoSPARQL specification. The Testbed-10 geospatial ontologies address many of these aspects (for example modularization of spatial relations), and could be used as a starting point. GeoSPARQL provides a geospatial extension of SPARQL by defining geospatial function extensions and data types (in the same way Spatial SQL extends SQL). GeoSPARQL could be simplified by clearly defining geospatial functions using SPARQL Service Description

vocabulary standard and the geospatial datatypes (WKT Literal and GML Literal). The query language should be independent of the ontology describing geospatial concepts (the same way Spatial SQL is independent of relational models). We believe that this simplification and modularization of the specification will foster the adoption of the specification to a larger community.

There is also a need to describe the capabilities of GeoSPARQL. These descriptions provide a mechanism by which a client or end user can discover information about the SPARQL service such as supported extension functions and details about the available geospatial dataset, supported CRSs, inferences supported. There are a number of standards that already exists to describe SPARQL endpoint such as the W3C SPARQL 1.1 Service Description, VoiD and DCAT. These standards will be adapted to accommodate description of GeoSPARQL endpoints, by defining profiles and best practices. The next testbed should demonstrate the feasibility and robustness of the approach by implementing the specifications.

We believe that this simplification and modularization of the specification will foster the adoption of the specification to a larger community.

13.1.11 Definition of vertical ontologies

Vertical ontologies based on the core geospatial ontologies (example E&DM, Hydro, Gazetteer) should be defined.

13.1.12 Migration of OGC services to REST-based semantic enabled web services

All OGC services should be migrated to REST-based semantic enabled web services. This implies expressing their capabilities in RDF based on existing standards (DCAT, VOID, ADSM, ...) and descriptions of processes in RDF (parameters, constraints etc). It has been suggested to start with Gazetteer, CSW, WFS and WPS.

14 Data sources

14.1 WFS-G for USGS

The USGS Geographic Names Information System (GNIS) download page contains the geographic names for the United States.

USGS geographic names data is available in the Public Domain with no restrictions.

The WFS-G for the USGS is hosted by Compusult with the following capabilities:

GetCapabilities	http://ows-10.compusult.net/wfs/services/?SERVICE=WFS&VERSION=2.0.0&REQUEST=GetCapabilities
GetFeature for location name (example for 10 results)	http://ows-10.compusult.net/wfs/services/?service=WFS&version=2.0.0&request=GetFeature&typeName=SI_LocationType&count=10
GetFeature for location type (example for 10 results)	http://ows-10.compusult.net/wfs/services/?service=WFS&version=2.0.0&request=GetFeature&typeName=SI_LocationInstance&count=10
GetFeature for specific location type (using a stored query; this is the URL that is in the location name results)	http://ows-10.compusult.net/wfs/services/?SERVICE=WFS&VERSION=2.0.0&REQUEST=GetFeature&OUTPUTFORMAT=application%2Fgml%2Bxml%3B+version%3D3.2&STORED QUERY_ID=urn:ogc:def:query:OGC-WFS::GetFeatureById&ID=SI_LOCATIONTYPE_PARK#SI_LOCATIONTYPE_PARK

Note: The version 1.1.0 of this WFS-G service and output format GML 3.1.1 was used by Image Matters to perform the semantic mapping of this service and expose the service as a GeoSPARQL service.

14.2 WFS-G for NGA

NGA geographic names can be downloaded from their Country Files (GNS) web page. This page contains links to the individual country files as well as a consolidated file.

Two instances of a WFS-G containing NGA GNDB locations are provided by Interactive Instruments.

- The first complies with the latest WFS-G schema version 1.0.0.
- The second uses the "simple" WFS-G schema from OWS-9.

Both WFS-G instances have been altered regarding the contents of all xlink:href attributes. The WFS GetFeature request contained has been simplified and now only contains the absolutely necessary parameters.

14.2.1 WFS-G BP V1.0.0 compliant

The WFS-G BP V1.0.0 compliant WFS-G now uses <wfs:FeatureCollection> as root element instead of <iso19112:SI_Collection> formerly used. This issue was logged for a change request.

Service endpoint:

http://services.interactive-instruments.de/xsprojects/ows10/service/gazetteer/wfs

14.2.2 "simple" WFS-G schema (OWS-9) compliant

Service endpoint:

http://services.interactive-instruments.de/xsprojects/ows10/service/gazetteer-simple/wfs

Note: This version of WFS-G service was used by Image Matters to expose it as a GeoSPARQL service using their WFS Knowledge Mapping Service.

14.3 NGA-GeoNames.org Link File

The NGA - Geonames.org ID Links csv file contains links between NGA features and GeoNames.org features that were derived from those names.

Use and restrictions. The GeoNames.org data base is licensed under a Creative Commons Attribution 3.0 License, see http://creativecommons.org/licenses/by/3.0/ and it is assumed that the link file is similarly licensed, as not explicit statement is provided, The Data is provided "as is" without warranty or any representation of accuracy, timeliness or completeness.

The names in New Brunswick were not collected from NGA and are not in this list. A separate file has been created with these links (cf. 8.3.5).

14.4 WFS-G for Local WFS (New Brunswick)

This data can be downloaded from the GeoBase web site.

Use and restrictions: All distributed data should be accessed and used relatively to the GeoBase Unrestricted Use Licence Agreement. With this licence, users are granted a

non-exclusive, fully paid, royalty-free right and licence to exercise all intellectual property rights in the data. This includes the right to use, incorporate, sublicense (with further right of sublicensing), modify, improve, further develop, and distribute the data; and to manufacture and/or distribute Derivative Products. The Licensee shall identify GeoBase as a data source.

Service endpoint:

http://ows-svc1.compusult.net/nbgaz/services

GetCapabilities	http://ows-svc1.compusult.net/nbgaz/services/?SERVICE=WFS&VERSION=2.0.0&REQUEST=GetCapabilities
GetFeature for location name (10 results)	http://ows-svc1.compusult.net/nbgaz/services/?service=WFS&version=2.0.0&request=GetFeature&typeName=SI_LocationType&count=10
GetFeature for location type (10 results)	http://ows-svc1.compusult.net/nbgaz/services/?service=WFS&version=2.0.0&request=GetFeature&typeName=SI_LocationInstance&count=10
GetFeature for specific location type (using stored query; this is the URL that is in the location name results)	http://ows-svc1.compusult.net/nbgaz/services/?SERVICE=WFS&VERSION=2.0.0&REQUEST=GetFeature&OUTPUTFORMAT=application%2Fgml%2Bxml%3B+version%3D3.2&STOREDQUERY_ID=urn:ogc:def:query:OGC-WFS::GetFeatureById&ID=SI_LOCATIONTYPE_RIV#SI_LOCATIONTYPE_RIV

Note: The version 1.1.0 of this WFS-G service and output format GML 3.1.1 was used by Image Matters to perform the semantic mapping of this service and expose the service as a GeoSPARQL service.

14.5 New Brunswick Populated Place Link File

New Brunswick Populated Place Links is an Excel spreadsheet containing NGA, GeoNames.org, DBPedia and OpenStreetMap links for populated places in New Brunswick. The excel file was processed by Image Matters to be converted as Linked Data representation and made accessible through a SPARQL endpoint accessible at:

http://ows10.usersmarts.com/ows10/gazetteers/mappings/sparql

Not every feature is linked to every source. The links to OpenStreetMap are particularly relevant to the task of linking to spatial information, as they include links to multiple spatial representations for populated places, including Nodes (points) and Ways and Relations (polygons).

The OpenStreetMap OSM Semantic Network wiki page explains the encoding of OSM data.

Use and restrictions. The New Brunswick Populated Place Link File was created for the Testbed-10 effort and is available for project use without restriction.

14.6 Geonames database

The GeoNames database contains over 10,000,000 geographical names corresponding to over 7,500,000 unique features. All features are categorized into one out of nine feature classes and further subcategorized into one out of 645 feature codes. Beyond names of places in various languages, data stored include latitude, longitude, elevation, population, administrative subdivision and postal codes. All coordinates use the World Geodetic System 1984 (WGS84). Those data are accessible free of charge through a number of Web services and a daily database export.

For this tesbed, the database was installed and indexed in PostGIS on ImageMatters server and was made accessible through a GeoSPARQL endpoint using Image Matters Semantic Mapping Component, This GEOSPARQL endpoint maps directly to the Geonames database stored into a POSTGis instance. GeoSPARQL queries are rewritten into one or more (spatial or not) SQL queries and result sets are converted to RDF on the fly.

The GeoSPARQL endpoint for Geonames was deployed at:

http://ows10.usersmarts.com/ows10/gazetteers/geonames/sparql

14.7 DBPedia

DBpedia.org is a community effort to extract structured information from Wikipedia and to make this information available on the Web. DBpedia allows you to ask sophisticated queries against Wikipedia and to link other datasets on the Web to Wikipedia data.

The DBpedia knowledge base currently describes more than 3.64 million things, out of which 1.83 million are classified in a consistent Ontology, including 416,000 persons, 526,000 places, 106,000 music albums, 60,000 films, 17,500 video games, 169,000 organisations, 183,000 species and 5,400 diseases. The DBpedia data set features labels and abstracts for these 3.64 million things in up to 97 different languages; 2,724,000 links to images and 6,300,000 links to external web pages; 6,200,000 external links into other RDF datasets, 740,000 Wikipedia categories, and 2,900,000 YAGO categories. The DBpedia knowledge base altogether consists of over 1.2 billion pieces of information

(RDF triples) out of which 335 million were extracted from the English edition of Wikipedia and 865 million were extracted from other language editions.

For this testbed, DBPedia was used to access to additional information about places found in other gazetteers as demonstrated in Pyxis client demonstration.

The DBPedia information were accessible using the following SPARQL endpoint:

http://dbpedia.org/sparql

14.8 LinkedGeoData

LinkedGeoData is an effort to add a spatial dimension to the Web of Data / Semantic Web. LinkedGeoData uses the information collected by the OpenStreetMap project and makes it available as an RDF knowledge base according to the Linked Data principles. It interlinks this data with other knowledge bases in the Linking Open Data initiative. It consists of more than 1 billion nodes and 100 million ways and the resulting RDF data comprises approximately 20 billion triples. The data is available according to the Linked Data principles and interlinked with DBpedia and Geo Names. For this testbed, this database was used to perform semantic linking between gazetteers places related to New Brunswick area.

The data are accessible through Linked Data REST API or using the following SPARQL endpoint: http://linkedgeodata.org/sparql

14.9 Provenance SPARQL endpoint

Image Matters deployed a SPARQL endpoint with support of SPARQL 1.1 Update and Query protocol to store provenance information resulting from conflation results. The SPARQL endpoint use an instance of the open source Sesame to store the RDF information.The endpoint was deployed at:

For query: http://ows10.usersmarts.com/ows10/repositories/conflation

For update: http://ows10.usersmarts.com/ows10/repositories/conflation/statements

15 References

Geographical linked data: The administrative geography of great britain on the semantic web
Transactions in GIS 2008 Wiley Online Library

LinkedGeoData - Adding a Spatial Dimension to the Web of Data
The Semantic Web-ISWC 2009, 2009 - Springer

LinkedGeoData: A Core for a Web of Spatial Open Data
Semantic Web, 2012 IOS Press

Triplify - Light-Weight Linked Data Publication from Relational Databases
WWW 2009, April 20-24, 2009. Madrid, Spain. ACM 978-1-60558-487-4/09/04

Geographical Linked Data: a Spanish Use Case
I-SEMANTICS Triplification Challenge 2010. Graz Austria

Linked Data in SDI or How GML is not about Trees
13th AGILE International Conference on Geographic Information Science 2010, Guimarles, Portugal

A Survey of Current Approaches for Mapping of Relational Databases to RDF
W3C RDB2RDF? Incubator Group, January 8, 2009

From Geo-data to Linked Data: Automated Transformation from GML to RDF

15.1 Revision history

Date	Release	Editor	Primary clauses modified	Description
19.11.13	0.1	Klopfer	n/a	Initial Outline
10.02.14	0.2	Klopfer	n/a	Added Requirements & Discussions from Wiki
06.03.14	0.3	Klopfer	n/a	OGC doc no. assigned / formatting cleaned elevated semantic mediation discussion to top level
04.26.14	0.4	Fellah	n/a	Added section related to semantic mapping components and added GeoSPARQL related datasource
05.19.14	0.5	Klopfer	n/a	Consilidated inputs from Envitia / IM Included GeoSPARQL related issues from clause 10 in 1.4
06.02.14	0.6	Klopfer	n/a	Finalised document with input from Envitia and Compusult