Open Geospatial Consortium

Approval Date: 2013-01-18

Posted Date: 2013-01-02

Publication date: 2013-06-18

Reference number of this document: OGC 12-104r1

Reference URL for this document: http://www.openggis.net/def/doc-type/per/ows9-spegg

Category: Public Engineering Report

Editor(s): Panagiotis (Peter) A. Vretanos

OGC® OWS-9 Engineering Report - CCI - Single Point of Entry Global Gazetteer

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 Single Point of Entry Global Gazetteer (SPEGG) implemented for the OWS9 test bed. The SPEGG integrates two gazetteers – a copy of the USGS gazetteers containing domestic names (hosted by CubeWerx Inc.) and the NGA gazetteer containing foreign names (originally hosted at NGA but currently hosted by Intergraph Corp.). Both integrated gazetteers and the SPEGG implement the Web Feature Service (WFS) standard.

Keywords

ogcdoc, OGC document,ows9, ows-9, gazetteer, wfs

What is OGC Web Services 9 (OWS-9)?

OWS-9 builds on the outcomes of prior OGC interoperability initiatives and is organized around the following threads:

- **Aviation**: Develop and demonstrate the use of the Aeronautical Information Exchange Model (AIXM) and the Weather Exchange Model (WXXM) in an OGC Web Services environment, focusing on support for several Single European Sky ATM Research (SESAR) project requirements as well as FAA (US Federal Aviation Administration) Aeronautical Information Management (AIM) and Aircraft Access to SWIM (System Wide Information Management) (AAtS) requirements.

- **Cross-Community Interoperability (CCI)**: Build on the CCI work accomplished in OWS–8 by increasing interoperability within communities sharing geospatial data, focusing on semantic mediation, query results delivery, data provenance and quality and Single Point of Entry Global Gazetteer.

- **Security and Services Interoperability (SSI)**: Investigate 5 main activities: Security Management, OGC Geography Markup Language (GML) Encoding Standard Application Schema UGAS (UML to GML Application Schema) Updates, Web Services Façade, Reference Architecture Profiling, and Bulk Data Transfer.

- **OWS Innovations**: Explore topics that represent either new areas of work for the Consortium (such as GPS and Mobile Applications), a desire for new approaches to existing technologies to solve new challenges (such as the OGC Web Coverage Service (WCS) work), or some combination of the two.

- **Compliance & Interoperability Testing & Evaluation (CITE)**: Develop a suite of compliance test scripts for testing and validation of products with interfaces implementing the following OGC standards: Web Map Service (WMS) 1.3 Interface Standard, Web Feature Service (WFS) 2.0 Interface Standard, Geography Markup Language (GML) 3.2.1 Encoding Standard, OWS Context 1.0 (candidate encoding standard), Sensor Web Enablement (SWE) standards, Web Coverage Service for Earth

Observation (WCS-EO) 1.0 Interface Standard, and TEAM (Test, Evaluation, And Measurement) Engine Capabilities.

The OWS-9 sponsors are: AGC (Army Geospatial Center, US Army Corps of Engineers), CREAF-GeoViQua-EC, EUROCONTROL, FAA (US Federal Aviation Administration), GeoConnections - Natural Resources Canada, Lockheed Martin Corporation, NASA (US National Aeronautics and Space Administration), NGA (US National Geospatial-Intelligence Agency), USGS (US Geological Survey), UK DSTL (UK MoD Defence Science and Technology Laboratory).

License Agreement

Contents

Page

OGC® OWS-9 Engineering Report - CCI - Single Point of Entry Global Gazetteer

1 Introduction

1.1 Scope

This document provides a technical description of the Single Point of Entry Global Gazetteer (SPEGG) implemented for the OWS9 test bed. The SPEGG integrates two gazetteers – a copy of the USGS gazetteers containing domestic names (hosted by CubeWerx Inc.) and the NGA gazetteer containing foreign names (originally hosted at NGA but currently hosted by Intergraph Corp.). Both integrated gazetteers and the SPEGG implement the Web Feature Service (WFS) standard.

This document describes the information model (ISO19112) that is the common model used by the USGS and NGA gazetteers and is also the model served by the SPEGG.

This document describes extensions to the Filter Encoding Standard to add text search operators and fuzzy string-matching operators. For testing purposes these operators were implemented in the USGS gazetteer since it is a copy of the USGS gazetteer and can thus be safely destabilized in this manner during the OWS9 test bed.

This document discusses the technical details of a cascading WFS which is used to implement the SPEGG.

Finally this document provides a high level description of semantic mediation that is used to draw equivalence in terms used by the USGS gazetteer and the NGA gazetteer. For example the USGS uses the term “hill” which the equivalent NGA term is “elevated point”. A query directed at the SPEGG should be able to use either term – or even another equivalent term – and obtain reasonable results from the server.

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 Core of Engineers
Gobe Hobana	Envitia
Perry Peterson	Pyxis Innovation
Idan Shatz	Pyxis Innovation
Panagiotis (Peter) A. Vretanos	CubeWerx Inc.
David Wesloh	NGA

1.3 Future work

1.3.1 Introduction

This clause work items that might be considered for future test beds.

1.3.2 Replication

Sub-clause 8.3.2.2 discusses some of the challenges encountered implementing an SPEGG using a WFS-C. Several of the problems (e.g. applying advanced text searching operators on a server that does not support them) can be resolved by having the WFS-C maintain a local copy of the data from a cascaded server which would require some form of replication to be used. The WFS API has the necessary underpinnings to support this kind of replication but will require some tweaking to provide an efficient replication capability. Future work in this area would allow a WFS-C to thus, augment the capabilities of a weaker cascaded server by locally replicating its data and processing requests locally.

1.3.3 Big data handling

The WFS 2.0 standard supports response paging which allows very large result sets to be accessed in a piecewise manner similar to how search engines present results one page at a time while offering links to fetch previous or subsequent pages. This is one example of an OGC service capability geared towards big data handling. A future work items in this area would exercise, enhance and generalize such capabilities across all OGC services especially those that inherently deal with large data sets (e.g. SOS, WMTS, WCS …). Among other things, big data handling includes providing sufficient metadata to properly characterise the scope of a response. For example, the following XML fragment shows a response for the USGS server which implements WFS 2.0:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- This Web Feature Server is a component of CubeWerx Suite 6.5.7. -->
<wfs:FeatureCollection
   timeStamp="2013-01-07T19:33:08-05:00"
   numberReturned="50000"
   numberMatched="2757145"
   xmlns="http://www.isotc211.org/19112"
   xmlns:iso19112="http://www.isotc211.org/19112"
   xmlns:gmdsf1="http://www.isotc211.org/2005/gmdsf1"
   xmlns:wfs="http://www.opengis.net/wfs/2.0"
   xmlns:gml="http://www.opengis.net/gml"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ...
```

The response shows that the entire result set contains 2757145 features and the current response document contains 50000 features – the default configured page size for the USGS server. Subsequent requests with startIndex=50001 would retrieve the next 50000 features and so on until the entire 2757145 records are retrieved. Finally, other aspects to be considered for big data handling include the use of binary encodings such as BXML and compression.

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, OpenGIS Web Feature Service 2.0 Interface Standard (also ISO 19142)

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

OGC 11-122r1, WFS Gazetteer Application Profile

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

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

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] and in OpenGIS® Abstract Specification Topic TBD: TBD 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
interface
named set of **operations** that characterize the behaviour of an entity

[ISO 19119:2005]

3.11
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]

3.12
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]

3.13
operation
specification of a transformation or query that an object may be called to execute

[ISO 19119:2005]

3.14
property
facet or attribute of an object, referenced by a name

[ISO 19143]

3.15
request
invocation of an **operation** by a **client**

[ISO 19128:2005]

3.16
response
result of an **operation** returned from a **server** to a **client**

[ISO 19128:2005]

3.17
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 (see Clause 9).

3.18
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).

3.19
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.

3.20
server
particular instance of a **service**

[ISO 19128:2005]

3.21
service
distinct part of the functionality that is provided by an entity through **interfaces**

[ISO 19119:2005]

3.22
service metadata
metadata describing the **operations** and geographic information available at a **server**

[ISO 19128:2005]

3.23
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>

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
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 Modelling Language (UML) static structure diagram, as described in Sub-clause 5.2 of [OGC 06-121r3].

5 Single point of entry global gazetteer overview

In the United States two agencies are responsible for officially maintaining gazetteers. The USGS is responsible from managing domestic place names while NGA is responsible for managing all non-domestic or foreign place names. The result is that anyone generally searching for a place name on the Earth must search two gazetteers each using their own internal organization, classification schemes, etc. The goal of the single point of entry global gazetteer is to provide a single WFS endpoint where someone can query both gazetteers in a seamless and integrated way.

This ER covers a number of aspects related to implementing such an end point. Topics discussed in this ER include a common schema for both of the gazetteers involved, semantic mediation of terms used to describe location types, cascading WFS and enhanced text-search Filter operators.

6 Gazetteer Schema

6.1 Introduction

The SPEGG is enabled by a common schema that is offered by both the USGS and NGA gazetteers. This common schema is an XML encoding of the data model found in ISO-19112, “Geographic information -- Spatial referencing by geographic identifiers”. This common schema allows the SPEGG to cascade both the USGS and NGA servers without the need to perform a schema translation.

This common schema, referred to as the “lite” ISO-19112 schema, was developed during previous work involving both USGS and NGA and resulted in the creation of OGC 11-122r1, Gazetteer Service – Application Profile of the Web Feature Service Best Practice.

6.2 Gazetteer UML model

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

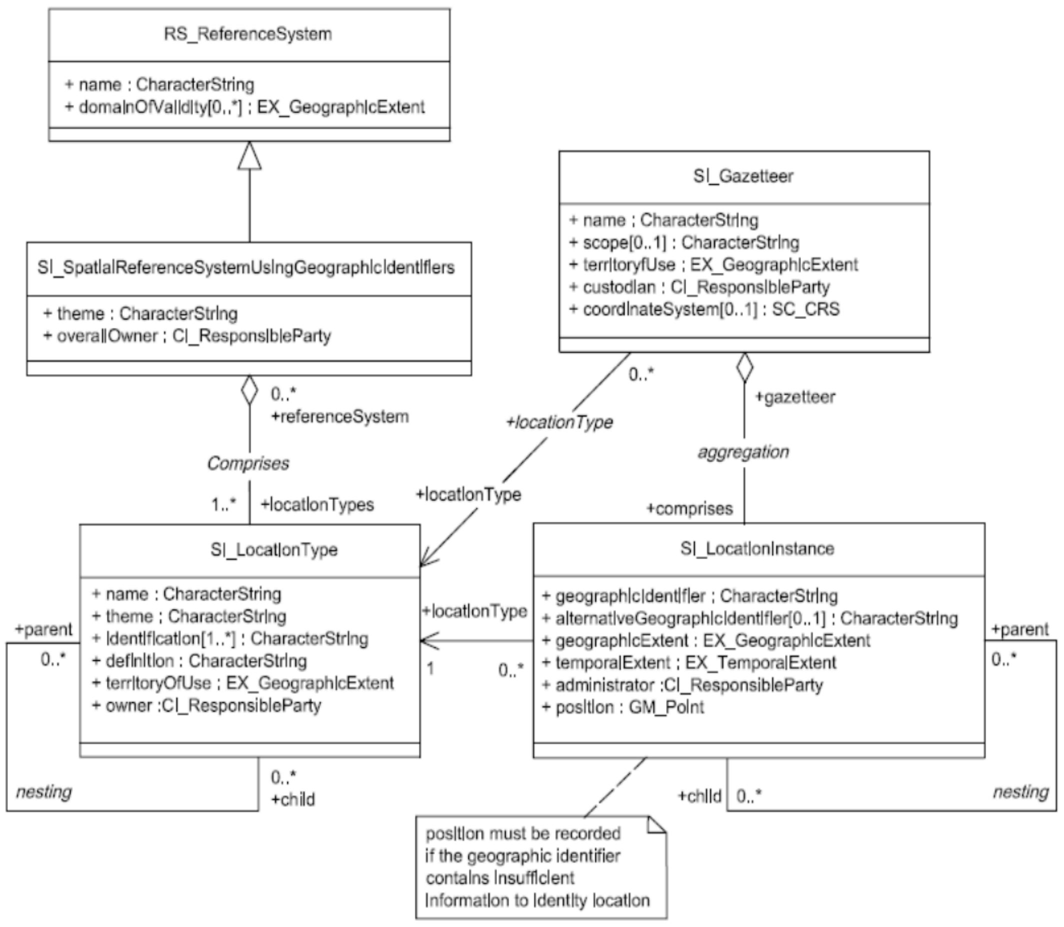
RS_ReferenceSystem
+ name : CharacterString
+ domainOfValidity[0..*] : EX_GeographicExtent
SI_SpatialReferenceSystemUsingGeographicIdentifiers
+ theme : CharacterString
+ overallOwner : CI_ResponsibleParty
SI_Gazetteer
+ name : CharacterString
+ scope[0..1] : CharacterString
+ territoryUse : EX_GeographicExtent
+ custodian : CI_ResponsibleParty
+ coordinateSystem[0..1] : SC_CRS
0..*
+referenceSystem
Comprises
1..*
+locationTypes
0..*
+locationType
+locationType
+gazetteer
aggregation
+comprises
SI_LocationType
+ name : CharacterString
+ theme : CharacterString
+ identification[1..*] : CharacterString
+ definition : CharacterString
+ territoryOfUse : EX_GeographicExtent
+ owner :CI_ResponsibleParty
SI_LocationInstance
+ geographicIdentifier : CharacterString
+ alternativeGeographicIdentifier[0..1] : CharacterString
+ geographicExtent : EX_GeographicExtent
+ temporalExtent : EX_TemporalExtent
+ administrator :CI_ResponsibleParty
+ position : GM_Point
+locationType
1
0..*
+parent
0..*
nesting
0..*
+child
+child
0..*
nesting
+parent
0..*
position must be recorded if the geographic identifier contains insufficient information to identity location

Figure 1 – ISO19112 UML Model

For the purposes of OWS-9, the important classes are the SI_LocationInstance and SI_LocationType. 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 (see 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 (see 6.1) both the USGS and NGA gazetteers implement the ISO19112 UML model (see 6.2). In addition, because of previous work (see 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 and used in OWS9 is encoded as a GML (see OGC 07-036) application schema using the level 1 of the Geography Markup Language (GML) simple features profile (see OGC 10-100r3).

The XML schema for ISO19112 can be found in ANNEX A. It was used unmodified in the OWS9.

7 Query requirements

7.1 Filter operators

7.1.1 Introduction

A goal of the OWS9 test bed was to implement and test advanced text searching and proximity searching.

Advanced text searching operators extended the OGC Filter Encoding Standard (see OGC 09-026r1) to support "starts with" searches, "ends with" searches, sub-string containment searches and fuzzy string matching searches.

Proximity searching methods extended server capabilities to support centre-point-radius searches and nearest neighbour searches.

7.1.2 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:ValueReference>
        <fes:Literal>Bost*</fes:Literal>
    </fes:PropertyIsLike>
</fes:Filter>
```

7.1.3 Ends with

Then “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:ValueReference>
        <fes:Literal>%York</fes:Literal>
    </fes:PropertyIsLike>
</fes:Filter>
```

7.1.4 Advanced text search extension

7.1.4.1 Introduction

This sub-clause defines a set of operators that extend the capabilities of OpenGIS’s Filter Encoding 2.0 Standard (see OGC 09-026r1) to support advanced text searching capabilities.

Rather than being presented as a change request for OGC 09-026r1, the operators defined in this sub-clause are presented as a standalone extension package that makes use of the extension points defined in the Filter Encoding 2.0 Standard (see 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

For OWS9 the operators described here were implemented by the USGS gazetteer offered by CubeWerx Inc.

The consolidated schemas for the advanced text search extension package can be found in ANNEX B.

7.1.4.2 PropertyContains operator

7.1.4.2.1 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".

7.1.4.2.2 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;"/>
         <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>
```

7.1.4.3 KVP encoding

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

Table 1 – KVP parameter for 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.

7.1.4.4 Parameter discussion

The ValueReference parameter shall reference a value to be tested (see 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>
```

7.1.4.5 Fuzzy string matching

7.1.4.5.1 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.

7.1.4.5.2 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>
```

7.1.4.6 KVP encoding

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

Table 2 – KVP parameter for 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.

7.1.4.6.1 Parameter discussion

The mandatory *ValueReference* parameter shall reference a value to be tested (see 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 (see http://www.levenshtein.net/) and the Jaro-Winkler algorithm (see http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance). Additional vendor-specific algorithms may also be specified using the pattern "vendor:{name}" but this standard does not describe what these methods might be.

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".

7.1.5 Proximity searches

7.1.5.1 Introduction

A common class of queries performed on gazetteers is proximity searches. In the OWS-9 test bed two types of proximity searches were implemented and tested; centre-point-radius searches and nearest neighbour searches.

7.1.5.2 Center-point-radius search

A centre-point-radius search involves the client specifying a centre point and a radius and the server finding all objects that spatially interact with the defined circle. The specific spatial interaction is specified by the client in the query predicate.

No changes are required to the XML-encoding of the FES (see 09-026r1) in order to support centre-point-radius searches. The only requirements are t:

- the server support circular geometries
- the server, in its capabilities document, advertise that the circular geometry that it understands is a valid operand for spatial operators.

The following XML-fragment show part of the FilterCapabilities section of a capabilities document advertising that gml:CircleByCenterPoint is a valid spatial operator arguments:

```
 <fes:Spatial_Capabilities>
    <fes:GeometryOperands>
       <fes:GeometryOperand>gml:Point</fes:GeometryOperand>
       <fes:GeometryOperand>gml:LineString</fes:GeometryOperand>
       <fes:GeometryOperand>gml:CircleByCenterPoint</fes:GeometryOperand>
       <fes:GeometryOperand>gml:Polygon</fes:GeometryOperand>
       <fes:GeometryOperand>gml:Envelope</fes:GeometryOperand>
    </fes:GeometryOperands>
    …
</fes:Spatial_Capabilities>
```

The following XML-encoded request illustrate a GetRecords request using gml:CircleByCenterPoint:

```
<wfs:GetFeature
   service="WFS"
   version="2.0.0"
   outputFormat="application/gml+xml; version=3.2"
   xmlns:iso19112="http://www.isotc211.org/19112"
   xmlns:wfs="http://www.opengis.net/wfs/2.0"
   xmlns:fes="http://www.opengis.net/fes/2.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/wfs/2.0 ../../wfs.xsd">
   <wfs:Query typeNames="iso19112:SI_LocationInstance">
      <fes:Filter>
         <fes:Intersects>
            <fes:PropertyName>position</fes:PropertyName>
            <gml:Curve srsName="urn:ogc:def:EPSG::4326">
               <gml:segments>
                  <gml:CircleByCenterPoint numArc="1">
                     <gml:pos>38.895111 -77.036667</gml:pos>
                     <gml:radius uom="m">20000</gml:radius>
                  </gml:CircleByCenterPoint>
               </gml:segments>
            </gml:Curve>
         </fes:Intersects>
      </fes:Filter>
```

```
    </wfs:Query>
</wfs:GetFeature>
```

For KVP-encoded requests, Table 3, defined additional parameters that may be implemented in order to conveniently express centre-point-radius queries:

Table 3 – KVP parameters for centre-point-radius search

Keyword	Description	Data type and value	Optionality
LAT	Latitude expressed in WGS84.	Number	Zero or one (Optional)
LON	Longitude expression in WGS84.	Number	Zero or one (Optional)
RADIUS	Search radius expressed in meters along the surface of the Earth.	Number	Zero or one (Optional)

For KVP-encoded requests, the implied spatial operator is Intersects. The following is an example of a KVP-encoded center-point-radius search:

http://cubewerx.pvretano.com/usgs/gazetteer/cubeserv.cgi?config=lite&datastore=lite&service=WFS&request=GetFeature&typeName=SI_LocationInstance&lat=38.895111&lon=-77.036667&radius=500&maxFeatures=1000&srsName=urn:ogc:def:crs:EPSG::4326

7.1.5.3 Nearest Neighbour

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.

For the OWS-9 test bed, the WFS 2.0 standard was leveraged and the nearest neighbour algorithm was implemented as a stored query (see OGC 09-025r1, clause 7.9.3). The stored query was names "Nearest Neighbour By Location Type" and was assigned the identifier "urn:cw:def:query:OGC-WFS::NearestNeighbours:ByLocationTypeName". Table 4 defines the parameters for this stored query. All parameters are mandatory.

Table 4 – KVP parameters for Nearest Neighbour

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 5 nearest objects of type "Park":

http://cubewerx.pvretano.com/usgs/gazetteer/cubeserv.cgi?config=lite&datastore=lite&service=WFS&version=2.0&request=GetFeature&storedQuery_Id=urn:cw:def:query:OGC-WFS::NearestNeighbours:ByLocationTypeName&locationTypeName=Park&lat=38.889480&lon=-77.035149&srsName=urn:ogc:def:crs:EPSG::4326&count=5

7.2 Diacritic and Special Character handling

All servers in the OWS-9 test bed used the UTF-8 character set for handing diacritics, native scripts, special characters, etc.

8 Cascading WFS

8.1 Introduction

This sub-clause describes aspects of the design and implementation of a cascading WFS (WFS-C) that behaves as a single point of entry global gazetteer. The purpose of a WFS-C is to provide a single point of access to two or more other WFSs. The benefits of a WFS-C include:

- A WFS-C allows integrated access to but distributed ownership and management of feature data sets.

- A WFS-C can mediate differences between servers and the data they offer. Some examples include mediating differences in CRS support, differences in response formats, differences in semantics (see OGC 12-103r3), etc. thus simplifying client side development.

- The use of WFS-C promotes partnership and collaboration between organizations which stimulates the development and use of common and/or compatible schemas thus reducing complexity at the data model level.
    - Such community developed schemas are critical to the wide-spread deployment of WFS
    - Such community developed schemas also reduce the proliferation of schemas which hinders interoperability.
    - The ISO 19112 schema used in OWS9 was the result of such collaboration between USGS, NGA and other parties (see 11-122r1).

8.2 Architecture

Figure 2 illustrates the architecture of the SPEGG as deployed for the OWS9 test bed. The WFS-C appears, to clients, like any normal WFS but on the server side it behaves like WFS client accessing the USGS and NGA gazetteers.

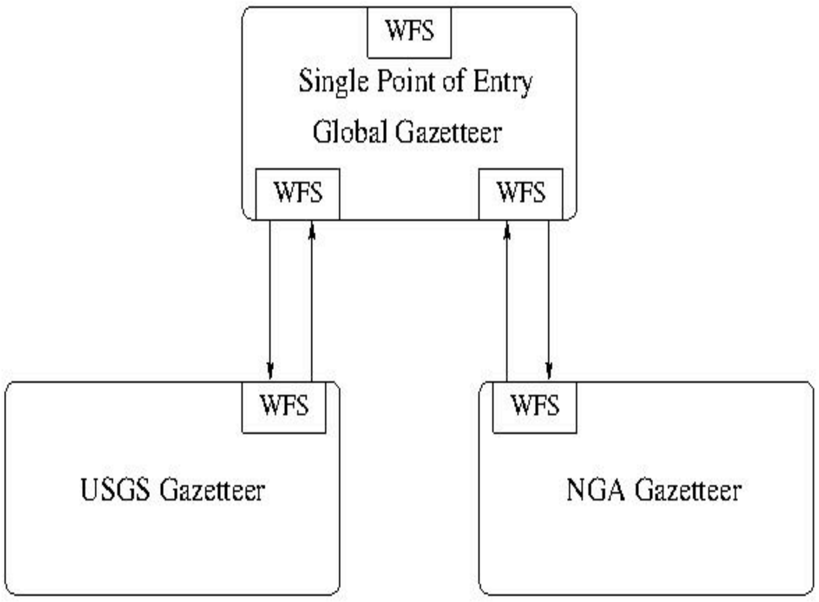

Figure 2 - Single Point of Entry Global Gazetteer Architecture

The SPEGG WFS supports up to version 2.0 of the WFS standard (see OGC 09-025r1). The USGS gazetteer, hosted by CubeWerx Inc., also supports up to version 2.0 of the WFS standard (see OGC 09-025r1) while the USGS gazetteer, hosted by Intergraph Corp., implements version 1.1.0 of the WFS standard (see OGC 04-025).

In both cases, the USGS and NGA servers offer the GMLSFL1 encoding of the ISO 19112 model (see clause 6).

8.3 Data sources

The SPEGG cascades two gazetteers, one from USGS and one from NGA, both serve the ISO 19112 "lite" schema (see Clause 6) through a WFS API.

The USGS server was hosted by CubeWerx Inc. and implements the WFS 2.0 standard in addition to a number of extensions described in this document. The URL of the USGS server is:

http://cubewerx.pvretano.com/usgs/gazetteer/cubeserv.cgi?config=lite&datastore=lite&service=WFS&request=GetCapabilities

The NGA server was originally hosted at NGA but later, because of technical issues, was hosted by Intergraph Corp. Intergraph hosted the global gazetteer at:

http://namesweb.intergraphgovsolutions.com/nameswfsg/request.aspx?service=WFS&request=GetCapabilities

8.4 Implementation of SPEGG

8.4.1 Introduction

The single point of entry global gazetteer was implemented in the OWS-9 test bed using a WFS-C. Requests to the SPEGG WFS-C are redirected to the cascaded servers (i.e. the USGS gazetteer and the NGA gazetteer) and the results from the cascaded server are integrated into a single response to the client. For all intents and purposes the client believes that a single server is being accessed.

This section of the document discuses a number of technical issues associated with implementing the SPEGG WFS-C and encountered during the OWS9 test bed deployment. The technical issues include:

1. Merging capability documents
2. Version handling
3. Schema handling

4. Semantics mediation
5. Exception handling
6. maxFeatures parameter
7. Axis-order

8.4.2 Compatibility matrix

A WFS-C, like any WFS, must report a capabilities document. Unlike other WFS's however, a WFS-C does not generate a capabilities document based on intrinsic capabilities but rather 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 WFS used for the USGS gazetteer implements WFS 2.0 and supports a number of spatial operators while the WFS used for the NGA gazetteer implements WFS 1.1.0 and 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.

8.4.2.1 Merging capabilities document

8.4.2.1.1 Introduction

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, two approaches for merging capabilities document where tested -- a "minimum common capabilities" approach and a "maximum capabilities" approach.

8.4.2.1.2 Minimum common capabilities approach

The "minimum common capabilities" approach for merging capabilities documents reads all the capabilities documents of the cascaded WFSs and reports the minimum set of common operations in the OperationMetadata section, the set of common feature types reported in the feature type list and the minimum set of filter operators in the FilterCapabilities section of the WFS-C.

A benefit of the "minimum common capabilities" approach is that it does not burden the WFS-C with having to do extensive request rewrites to suite each incoming request for each cascaded server. Since the WFS-C capabilities document will be report the least common set of capabilities, client apps will read this and generate request that can be run on all cascaded server with possibly slight modifications.

The main problem encountered with the "minimum common capabilities" approach, at least during the OWS9 test bed, was that the minimum set of capabilities was too minimal. Because the NGA WFS only supported a very small set of WFS capabilities, the WFS-C – using the "minimum common capabilities" approach – also ended up reporting a small set of capabilities and thus "hiding" most of the new and interesting capabilities implemented in the USGS server for OWS9 (see Clause 7).

8.4.2.1.3 Maximum capabilities approach

To overcome the limitations of the "minimum common capabilities" approach, the WFS-C implemented by CubeWerx Inc. for OWS9 was modified to implement a "maximum capabilities" approach.

In a "maximum capabilities" approach, the WFS-C reports the maximum set of capabilities reported by any of the cascaded servers – in the case of OWS-9 this was the capabilities document from the USGS server. Incoming requests are then rewritten for each cascaded server based on its capabilities.

The problem with this approach is that the WFS-C can get into a situation where there is no suitable rewrite of an incoming request that can be executed on all cascaded servers.

8.4.2.2 Version mismatch handling

For OWS-9, the SPEGG integrated two servers that support different versions of the WFS standard; the USGS server which supports WFS 2.0 and the NGA server which supports V1.1.0.

Because WFS 2.0 is a superset of WFS 1.1, for many requests it is a simple matter for the SPEGG to rewrite a request from a WFS 2.0 to a WFS 1.1 request. In some cases it is as simple as changing the version number. However, in other situations the re-write resulting from the version mismatch is not nearly so trivial.

Consider the case of the Nearest Neighbour (NN) capability implemented in the USGS WFS which implemented V2.0 of the WFS standard. The NN capability was implemented in the USGS server as a stored query (see X.X). However, there is simply no way for the SPEGG WFS-C to rewrite a stored query to be executed on a V1.1 WFS because that version of the WFS standard does not support stored queries.

Another example is the advanced text search operators (see 7.1.4) implemented 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.

To overcome such problems, the SPEGG WFS-C was further modified to support asymmetric query execution. Simply put, asymmetric query execution means that the WFS-C evaluates whether a request can be executed, or re-written to be executed, on a particular cascaded server. If it can, the request is executed on that cascaded server and the results are integrated into the response generated by the SPEGG. If, however, the incoming request cannot be executed on the cascaded server, or be rewritten and executed on that cascaded server, then that cascaded server is simply ignored – in other words, it does not contributed any data to the response.

While asymmetric query execution allows version mismatch and other issues to be handled by a WFS-C it is probably only useful in contrived situations such as the OWS-9 test bed. In the end, the only reliable approach to handling version and capability mismatches is to implement a WFS-C using the "minimum common capabilities" approach. Because a WFS-C more-or-less reflects the capabilities of the server that it cascades, for the OWS-9 test-bed the SPEGG would appear as a V1.1 server to client and would only implement the BBOX spatial operator – since that is the only spatial operators the NGA server implements.

8.4.2.3 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.

For the OWS-9 test bed approach 2 was used because it was the one that could be implemented within the time and resource constraints of the project. However, the recommended approach for handing the maxFeatures parameter is approach 3.

8.4.2.4 Exception handling

It is inevitable that a WFS-C will have to deal with exceptions from one or more cascaded servers while processing a request. In the case of exceptions from downstream servers a WFS-C can:

1. Cause the entire request to fail and raise an exception.
2. Somehow report the exception from a downstream server inline with the good responses from the other servers.

For the OWS-9 test bed, the SPEGG implemented the latter approach assembling a response from the successful responses of the cascaded WFSs and including any exceptions inline in the response document.

Because the SPEGG support all WFS versions up to 2.0 it must be prepared to deal with exceptions regardless of the request version.

For WFS 1.1.0 requests, exceptions 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.

For WFS 2.0 requests, the actual OWS exception report generated by a cascaded server is included in-line with the response. This is possible because the WFS 2.0 standard supports this kind of exception reporting (see OGC 09-025r1, clause 10.3).

8.4.2.5 Axis order issue

Once again, the axis order issue has reared its ugly head! The issue arises because of the ambiguous definition of the notation EPSG:XXXX and specifically EPSG:4326. It was proposed that for OWS-9 all coordinate reference system specifications should be expressed using URN notation.

In the longer term, OGC should consider having servers throw an exception whenever the EPSG:XXXX notation is used!

8.4.3 Schema handling

Although it is conceivable that a cascading WFS could implement a schema mapping capability – thus being able to cascade servers offering different schemas – for the OWS-9 test bed a common gazetteer schema, called the ISO 19112 lite schema (see 6.3), was used. The common schema was a fortuitous result of work done previously in another project involving USGS and NGA (see OGC 11-122r1).

8.4.4 Semantic mediation

The SPEGG integrates the USGS (domestic names) and the NGA (foreign names) gazetteers as a cascading server. The cascading server handles syntactic integration of the two servers – i.e. capabilities mismatches, protocol mismatches, schema mismatches, etc. -- but the cascading server does not implement any semantic mediation.

Simply put, semantic mismatches occur when different vocabularies are used to describe similar real-word objects. The ISO 19112 schema used in the OWS9 test bed includes a property called *locationType* that classifies each location instance. The semantic mismatch arises because the USGS and NGA gazetteers use different vocabularies for the *locationType* property. Thus similar objects are described using different terms and in some cases terms in one vocabulary do not exist in the other. For this reason, a semantic mediator has been implemented that analyses incoming requests to the SPEGG and rewrites them accordingly so that the correct records are returned.

The details of the semantic mediator can be found in the engineering report "OWS-9 CCI Semantic Mediation Engineering Report" (see 12-103r3).

ANNEX A

ISO-19112 model encoded as a GMLSFL1 application schema

This ANNEX contains the schema files for the ISO19112 "lite" schema used for the OWS-9 testbed. The schemas are currently hosted by CubeWerx Inc. at:

http://www.pvretano.com/schemas/gazetteer/1.0.0/gmlsf1/lite

iso19112.xsd:

```
 <xsd:schema
   targetNamespace="http://www.isotc211.org/19112"
   xmlns:iso19112="http://www.isotc211.org/19112"
   xmlns:gmdsf1="http://www.isotc211.org/2005/gmdsf1"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:gml="http://www.opengis.net/gml"
   xmlns:gmlsf="http://www.opengis.net/gmlsf"
   elementFormDefault="qualified"
   version="19112:2003">

   <xsd:annotation>
      <xsd:appinfo
source="../../../../gml/3.1.1/profiles/gmlsfProfile/1.0.0/gmlsfLevels.xsd">
         <gmlsf:ComplianceLevel>1</gmlsf:ComplianceLevel>

<gmlsf:GMLProfileSchema>../../../../gml/3.1.1/profiles/gmlsfProfile/1.0.0/gmlsf
[2].xsd</gmlsf:GMLProfileSchema>
      </xsd:appinfo>
   </xsd:annotation>

   <xsd:import namespace="http://www.opengis.net/gml"
               schemaLocation="../../../../gml/3.1.1/base/gml.xsd"/>

   <xsd:import namespace="http://www.opengis.net/gmlsf"

schemaLocation="../../../../gml/3.1.1/profiles/gmlsfProfile/1.0.0/gmlsfLevels.x
sd"/>

   <xsd:import namespace="http://www.isotc211.org/2005/gmdsf1"
schemaLocation="./gmd.xsd"/>

   <!-- ========================================================== -->
   <!-- SI_SpatialReferenceSystemUsingGeographicIdentifiers         -->
   <!-- ========================================================== -->
   <xsd:element name="SI_SpatialReferenceSystemUsingGeographicIdentifiers"
        type="iso19112:SI_SpatialReferenceSystemUsingGeographicIdentifiersType"
        substitutionGroup="gml:_Feature"/>
   <xsd:complexType
        name="SI_SpatialReferenceSystemUsingGeographicIdentifiersType">
      <xsd:complexContent>
         <xsd:extension base="gml:AbstractFeatureType">
            <xsd:sequence>
               <xsd:element name="name" type="xsd:string"/>
               <xsd:element name="domainOfValidity"
                            type="gmdsf1:EX_GeographicExtentPropertyType"
                            minOccurs="0"/>
               <xsd:element name="theme" type="xsd:string"/>
               <xsd:element name="overallOwner"
                            type="gmdsf1:CI_ResponsiblePartyPropertyType"/>
               <xsd:element name="position"
                            type="gml:PointPropertyType" minOccurs="0"/>
               <xsd:element name="locationTypes"
                            type="gml:ReferenceType"
                            minOccurs="1" maxOccurs="unbounded">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_LocationType/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>

            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- =================================================================== -->
   <!-- SI_LocationType                                                     -->
   <!-- =================================================================== -->
   <xsd:element name="SI_LocationType"
                type="iso19112:SI_LocationTypeType"
                substitutionGroup="gml:_Feature"/>
   <xsd:complexType name="SI_LocationTypeType">
      <xsd:complexContent>
         <xsd:extension base="gml:AbstractFeatureType">
            <xsd:sequence>
               <xsd:element name="name"
                            type="xsd:string"/>
               <xsd:element name="theme"
                            type="xsd:string"/>
               <xsd:element name="identification"
                            type="xsd:string" maxOccurs="unbounded"/>
               <xsd:element name="definition"
                            type="xsd:string"/>
               <xsd:element name="territoryOfUse"
                            type="gmdsf1:EX_GeographicExtentPropertyType"/>
               <xsd:element name="owner"
                            type="gmdsf1:CI_ResponsiblePartyPropertyType"/>
               <xsd:element name="parent"
                            type="gml:ReferenceType"
                            minOccurs="0" maxOccurs="unbounded">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_LocationType/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="child"
```

```
                            type="gml:ReferenceType"
                            minOccurs="0" maxOccurs="unbounded">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_LocationType/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- =============================================================== -->
   <!-- SI_Gazetteer                                                    -->
   <!-- =============================================================== -->
   <xsd:element name="SI_Gazetteer"
                type="iso19112:SI_GazetteerType"
                substitutionGroup="gml:_Feature"/>
   <xsd:complexType name="SI_GazetteerType">
      <xsd:complexContent>
         <xsd:extension base="gml:AbstractFeatureType">
            <xsd:sequence>
               <xsd:element name="isGlobal" type="xsd:boolean"/>
               <xsd:element name="name" type="xsd:string"/>
               <xsd:element name="scope" type="xsd:string" minOccurs="0"/>
               <xsd:element name="territoryOfUse"
                            type="gmdsf1:EX_GeographicExtentPropertyType"/>
               <xsd:element name="custodian"
                            type="gmdsf1:CI_ResponsiblePartyPropertyType"/>
               <xsd:element name="coordinateSystem"
                            type="xsd:anyURI" minOccurs="0"/>
               <xsd:element name="locationType"
                            type="gml:ReferenceType"
                            minOccurs="1" maxOccurs="unbounded">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_LocationType/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>

   <!-- =============================================================== -->
   <!-- SI_LocationInstance                                             -->
   <!-- =============================================================== -->
   <xsd:element name="SI_LocationInstance"
                type="iso19112:SI_LocationInstanceType"
                substitutionGroup="gml:_Feature"/>
   <xsd:complexType name="SI_LocationInstanceType">
      <xsd:complexContent>
```

```
         <xsd:extension base="gml:AbstractFeatureType">
            <xsd:sequence>
               <xsd:element name="geographicIdentifier"
                            type="xsd:string"/>
               <xsd:element name="alternativeGeographicIdentifier"
                            type="xsd:string"
                            minOccurs="0" maxOccurs="unbounded"/>
               <xsd:element name="geographicExtent"
                            type="gmdsf1:EX_GeographicExtentPropertyType"/>
               <xsd:element name="temporalExtent"
                            type="xsd:dateTime" minOccurs="0"/>
               <xsd:element name="administrator"
                            type="gmdsf1:CI_ResponsiblePartyPropertyType"/>
               <xsd:element name="position"
                            type="gml:PointPropertyType" minOccurs="0"/>
               <xsd:element name="gazetteer"
                            type="gml:ReferenceType">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_Gazetteer/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="locationType"
                            type="gml:ReferenceType">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_LocationType/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="parent"
                            type="gml:ReferenceType"
                            minOccurs="0" maxOccurs="unbounded">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_LocationInstance/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="child"
                            type="gml:ReferenceType"
                            minOccurs="0" maxOccurs="unbounded">
                  <xsd:annotation>
                     <xsd:appinfo source="urn:x-
gml:targetElement">iso19112:SI_LocationInstance/@gml:id</xsd:appinfo>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
</xsd:schema>
```

gmd.xsd:

```
<?xml version="1.0"?>
<xsd:schema
   targetNamespace="http://www.isotc211.org/2005/gmdsf1"
   xmlns:gmdsf1="http://www.isotc211.org/2005/gmdsf1"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:gml="http://www.opengis.net/gml"
   xmlns:gmlsf="http://www.opengis.net/gmlsf"
   elementFormDefault="qualified"
   version="2005">

   <xsd:annotation>
      <xsd:appinfo
source="../../../../gml/3.1.1/profiles/gmlsfProfile/1.0.0/gmlsfLevels.xsd">
         <gmlsf:ComplianceLevel>1</gmlsf:ComplianceLevel>

<gmlsf:GMLProfileSchema>../../../../gml/3.1.1/profiles/gmlsfProfile/1.0.0/gmlsf
.xsd</gmlsf:GMLProfileSchema>
      </xsd:appinfo>
   </xsd:annotation>

   <xsd:import namespace="http://www.opengis.net/gml"
               schemaLocation="../../../../gml/3.1.1/base/gml.xsd"/>

   <xsd:import namespace="http://www.opengis.net/gmlsf"

schemaLocation="../../../../gml/3.1.1/profiles/gmlsfProfile/1.0.0/gmlsfLevels.x
sd"/>

   <xsd:complexType name="CI_ResponsiblePartyPropertyType">
      <xsd:sequence>
         <xsd:element ref="gmdsf1:CI_ResponsibleParty"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="CI_ResponsibleParty">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="individualName" type="xsd:string"
                         minOccurs="0"/>
            <xsd:element name="organizationName" type="xsd:string"
                         minOccurs="0"/>
            <xsd:element name="positionName" type="xsd:string"
                         minOccurs="0"/>
                  <xsd:element name="voice" type="xsd:string"
                               minOccurs="0" maxOccurs="unbounded"/>
                  <xsd:element name="facsimile" type="xsd:string"
                               minOccurs="0" maxOccurs="unbounded"/>
                  <xsd:element name="deliveryPoint" type="xsd:string"
                               minOccurs="0"/>
                  <xsd:element name="city" type="xsd:string"
                               minOccurs="0"/>
                  <xsd:element name="administrativeArea" type="xsd:string"
                               minOccurs="0"/>
                  <xsd:element name="postalCode" type="xsd:string"
                               minOccurs="0"/>
                  <xsd:element name="country" type="xsd:string"
                               minOccurs="0"/>
                  <xsd:element name="electronicMailAddress" type="xsd:string"
                               minOccurs="0" maxOccurs="unbounded"/>
```

```
                <xsd:element name="linkage"
                             type="xsd:anyURI" minOccurs="0"/>
                <xsd:element name="protocol"
                             type="xsd:string" minOccurs="0"/>
                <xsd:element name="applicationProfile"
                             type="xsd:string" minOccurs="0"/>
                <xsd:element name="name"
                             type="xsd:string" minOccurs="0"/>
                <xsd:element name="description"
                             type="xsd:string" minOccurs="0"/>
                <xsd:element name="function" minOccurs="0">
                   <xsd:simpleType>
                      <xsd:restriction base="xsd:string">
                         <xsd:enumeration value="download"/>
                         <xsd:enumeration value="information"/>
                         <xsd:enumeration value="offlineAccess"/>
                         <xsd:enumeration value="order"/>
                         <xsd:enumeration value="search"/>
                      </xsd:restriction>
                   </xsd:simpleType>
                </xsd:element>
             <xsd:element name="hoursOfService" type="xsd:string"
                          minOccurs="0"/>
             <xsd:element name="contactInstructions" type="xsd:string"
                          minOccurs="0"/>
          <xsd:element name="role" minOccurs="0">
             <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                   <xsd:enumeration value="resourceProvider"/>
                   <xsd:enumeration value="custodian"/>
                   <xsd:enumeration value="owner"/>
                   <xsd:enumeration value="user"/>
                   <xsd:enumeration value="distributor"/>
                   <xsd:enumeration value="originator"/>
                   <xsd:enumeration value="pointOfContact"/>
                   <xsd:enumeration value="principalInvestigator"/>
                   <xsd:enumeration value="processor"/>
                   <xsd:enumeration value="publisher"/>
                   <xsd:enumeration value="author"/>
                </xsd:restriction>
             </xsd:simpleType>
          </xsd:element>
       </xsd:sequence>
    </xsd:complexType>
 </xsd:element>

 <xsd:complexType name="EX_GeographicExtentPropertyType">
    <xsd:sequence>
       <xsd:element ref="gmdsf1:EX_GeographicExtent"/>
    </xsd:sequence>
 </xsd:complexType>
 <xsd:element name="EX_GeographicExtent">
    <xsd:complexType>
       <xsd:sequence>
          <xsd:element name="extentTypeCode"
                       type="xsd:boolean"
                       minOccurs="0" default="1"/>
          <xsd:element name="boundingPolygon"
                       type="gml:SurfacePropertyType"
```

```
                         minOccurs="0"/>
            <xsd:element ref="gml:Envelope" minOccurs="0"/>
            <xsd:element name="geographicDescription"
                         type="gml:CodeType" minOccurs="0"/>

         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>

</xsd:schema>
```

ANNEX B
Consolidated Advanced Text Search Schemas

This ANNEX contains the consolidated schemas for the Advanced Text Search Extension for FES defined in this document and examples thereof.

B.1 Schema

```
<xsd:schema
   targetNamespace="http://www.opengis.net/fes/2.0/advstr/1.0"
   xmlns:advstr="http://www.opengis.net/fes/2.0/advstr/1.0"
   xmlns:fes="http://www.opengis.net/fes/2.0"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   elementFormDefault="qualified"
   version="1.0.0">

   <xsd:import namespace="http://www.opengis.net/fes/2.0"
               schemaLocation="../../filter/2.0/filterAll.xsd"/>

   <!-- ============================================================ -->
   <!-- EXTENDCED STRING MATHCIN OPERATORS                            -->
   <!--============================================================= -->
   <xsd:element name="PropertyMatches"
                type="advstr:PropertyMatchesType"
                substitutionGroup="fes:extensionOps"/>
   <xsd:element name="PropertyContains"
                type="advstr:PropertyContainsType"
                substitutionGroup="fes:extensionOps"/>

   <!--============================================================= -->
   <!-- TYPE DECLARATIONS                                             -->
   <!--============================================================= -->
   <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>
   <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;"/>
            <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="matchCase" type="xsd:boolean"
                           use="optional" default="false"/>
         </xsd:extension>
      </xsd:simpleContent>
   </xsd:complexType>
</xsd:schema>
```

B.2 Examples

B.2.1 Contains example 1

```
<?xml version="1.0"?>
<fes:Filter
   xmlns:fes="http://www.opengis.net/fes/2.0"
   xmlns:advstr="http://www.opengis.net/fes/2.0/advstr/1.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/fes/2.0
                       ../../../filter/2.0/filterAll.xsd
                       http://www.opengis.net/fes/2.0/advstr/1.0
                       ../advancedStr.xsd">
   <advstr:PropertyContains>
<fes:ValueReference>alternativeGeographicIdentifier</fes:ValueReference
>
      <advstr:SearchTerms>Boston</advstr:SearchTerms>
      <advstr:NearTerm within="2">Common</advstr:NearTerm>
   </advstr:PropertyContains>
</fes:Filter>
```

B.2.2. Contains example 2

```
<?xml version="1.0"?>
<fes:Filter
   xmlns:fes="http://www.opengis.net/fes/2.0"
   xmlns:advstr="http://www.opengis.net/fes/2.0/advstr/1.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/fes/2.0
                       ../../../filter/2.0/filterAll.xsd
                       http://www.opengis.net/fes/2.0/advstr/1.0
                       ../advancedStr.xsd">
   <advstr:PropertyContains>
<fes:ValueReference>alternativeGeographicIdentifier</fes:ValueReference
>
      <advstr:SearchTerms>Niagara</advstr:SearchTerms>
   </advstr:PropertyContains>
</fes:Filter>
```

B.2.3 Matches example 1

```
<?xml version="1.0"?>
<fes:Filter
   xmlns:fes="http://www.opengis.net/fes/2.0"
   xmlns:advstr="http://www.opengis.net/fes/2.0/advstr/1.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/fes/2.0
                       ../../../filter/2.0/filterAll.xsd
                       http://www.opengis.net/fes/2.0/advstr/1.0
                       ../advancedStr.xsd">
   <advstr:PropertyMatches method="jaro-winkler">
<fes:ValueReference>alternativeGeographicIdentifier</fes:ValueReference
>
      <advstr:MatchString strength="90">Albeon</advstr:MatchString>
   </advstr:PropertyMatches>
</fes:Filter>
```