Open Geospatial Consortium

Publication Date: 2016-01-28

Approval Date: 2015-10-26

Posted Date: 2015-09-08

Reference number of this document: OGC 15-011r2

Reference URL for this document: http://www.opengis.net/doc/PER/tb-11-WFS-T-IER

Category: Public Engineering Report

Editor: Panagiotis (Peter) A. Vretanos

OGC Testbed-11 Multiple WFS-T Interoperability

Warning

This document is not an OGC Standard. This document presents a discussion of technology issues considered in an initiative of the OGC Interoperability Program. This document does not represent an official position of the OGC. It is subject to change without notice and may not be referred to as an OGC Standard. However, the discussions in this document could very well lead to the definition of an OGC Standard. Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

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

License Agreement

Contents

Page

Figures

Page

Tables

Page

Abstract

This document describes the work done in the OGC Testbest-11 to support multiple Web Feature Service – Transactional (WFS-T) instance interoperability with particular focus on the enterprise-to-enterprise data synchronization use case.

This document describes the new capabilities added to the participating WFSs to support the transactional use cases implemented for the OGC Testbed-11.

This document describes the implementation of a Nearest Neighbor stored query to support the OGC Testbed-11 transactional use cases.

This document describes an extension the WFS standard to support inserting, updating, deleting and querying features with multimedia-valued properties. Multimedia-valued properties are properties whose value may be -- for example -- images, videos clips or audio clips. During the OGC Testbed-11, properties whose values were photographs were tested but the work is generally extensible for any multimedia value type (e.g. audio, video, images, documents, etc.).

This document describes a peer-to-peer data synchronization protocol that was used in the OGC Testbed-11 to synchronize data between web feature servers. Furthermore, the document described extensions to the WFS API to support this synchronization protocol.

Business Value

Parties wishing to deploy a WFS with a nearest neighbor store query can, using the information presented in this document, do so in an interoperable way.

Parties wishing to deploy a WFS that handles multimedia values can, using the information presented in this document, do so in an interoperable way.

Parties wishing deploy a federation of WFSs that synchronize data among themselves can, using the information presented in this document, the necessary extensions to the WFS to allow this to happen in a standard and interoperable way.

Keywords

ogcdocs, testbed-11, WFS-T, WFS, client, server, interoperability, GSS, GeoPackage, WPS, schema translation, simple feature GML

Testbed-11 Multiple WFS-T Interoperability

1 Introduction

1.1 Scope

This document describes the work done in the OGC Testbest-11 to support multiple WFS-T instance interoperability by way of a transaction scenario involving the interaction between clients and multiple WFS-T servers as well as the interaction between the servers themselves, especially in the use case of enterprise-to-enterprise data synchronization.

The document presents an overview of the transaction scenario, the components used to implement the scenario in the OGC Testbed-11 demo and the new capabilities added to the WFS-T server to support the scenario.

1.2 Document contributor contact points

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

Name	Organization
Panagiotis (Peter) A. Vretanos	CubeWerx Inc

1.3 Revision history

Date	Release	Editor	Primary clauses modified	Description
26-FEB-2015		PAV	All	Initial draft
08-SEP-2015		PAV	8	Respond to comments from Byron Cochrane

1.4 Future work

- Investigate integrating the enterprise-to-enterprise sync protocol in the GSS standard (see OGC 10-069r3).

1.5 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 02-069, Geography Markup Language

OGC 02-058, Web Feature Service

OGC 03-105r1, OpenGIS Geography Markup Language (GML) Encoding Standard

OGC 04-094, OpenGIS Web Feature Service (WFS) Implementation Specification

OGC 06-121r3, OGC Web Services Common Standard

OGC 07-036, OpenGIS Geography Markup Language (GML) Encoding Standard

OGC 09-025r2, OGC Web Feature Service 2.0 Interface Standard – With Corrigendum

OGC 10-069r3, OWS-7 Engineering Report – Geosynchronization Service

OGC 11-080r1, A REST binding for WFS 2.0 (Change request)

OGC 14-102, OGC Web Feature Service 2.5 Interface Standard (pending)

OGC 15-010, OGC Testbed-11 WFS-T Information Exchange Architecture

OGC 15-052, OGC Testbed-11 REST Engineering Report

OGC 15-053, OGC Testbed-11 JSON/GeoJSON in OGC Stds ER

OGC 15-068r2, Testbed-11 GeoPackaging Engineering Report

3 Terms and definitions

3.1

attribute <XML>
name-value pair contained in an **element** (4.6)

[ISO 19136:2007, definition 4.1.3]

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

3.2

client
software component that can invoke an **operation** (4.17) from a **server** (4.28)

[ISO 19128:2005, definition 4.1]

3.3

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

[ISO 19111:2007, definition 4.5]

3.4

coordinate reference system
coordinate system (4.5) that is related to an object by a datum

[ISO 19111:2007, definition 4.8]

3.5

coordinate system
set of mathematical rules for specifying how **coordinates** (4.3) are to be assigned to points

[ISO 19111:2007, definition 4.10]

3.6

element <XML>
basic information item of an XML document containing child elements, **attributes** (4.1) and character data

[ISO 19136:2007, definition 4.1.23]

3.7

feature
abstraction of real world phenomena

[ISO 19101:2002, definition 4.11]

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** (4.7) instance

3.9

filter expression

predicate expression encoded using XML

[OGC 09-026r2, definition 4.11]

3.10

interface

named set of **operations** (4.17) that characterize the behaviour of an entity

[ISO 19119:2005, definition 4.2]

3.11

join predicate

filter expression (4.9) that includes one or more clauses that constrain properties from two different entity types

[OGC 09-026r2, definition 4.16]

NOTE In this International Standard, the entity types will be **feature** (4.7) types.

3.12

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 2045:1996]

3.13

namespace <XML>

collection of names, identified by a URI reference which are used in XML documents as **element** (4.6) names and **attribute** (4.1) names

[W3C XML Namespaces:1999]

3.14

operation

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

[ISO 19119:2005, definition 4.3]

3.15
property
facet or attribute of an object, referenced by a name

[OGC 09-026r2, definition 4.21]

3.16
resource
asset or means that fulfils a requirement

[OGC 09-026r2, definition 4.23]

NOTE In this International Standard, the resource is a **feature** (4.7), or any identifiable component of a feature (e.g. a property of a feature)

3.17
request
invocation of an **operation** (4.17) by a **client** (4.2)

[ISO 19128:2005, definition 4.10]

3.18
response
result of an **operation** (4.17) returned from a **server** (4.28) to a **client** (4.2)

[ISO 19128:2005, definition 4.11]

3.19
schema
formal description of a model

[ISO 19101:2002, definition 4.25]

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** (4.1) and **elements** (4.6) of an XML object (for example, a document or a portion of a document).

3.20
schema <XML Schema>
collection of **schema** (4.26) components within the same target **namespace** (4.16)

[ISO 19136:2007, definition 4.1.54]

EXAMPLE Schema components of W3C XML Schema are types, **elements** (4.16), **attributes** (4.1), groups, etc.

3.21
server
particular instance of a **service** (4.29)

[ISO 19128:2005, definition 4.12]

3.22
service
distinct part of the functionality that is provided by an entity through **interfaces** (4.10)

[ISO 19119:2005, definition 4.1]

3.23
service metadata
metadata describing the **operations** (4.17) and geographic information available at a **server** (4.28)

[ISO 19128:2005, definition 4.14]

3.24
Uniform Resource Identifier
unique identifier for a resource, structured in conformance with IETF RFC 2396

[ISO 19136:2007, definition 4.1.65]

NOTE The general syntax is <scheme>::<scheme-specified-part>. The hierarchical syntax with a **namespace** (4.16) is <scheme>://<authority><path>?<query>

3.25
filter capabilities XML
metadata, encoded in XML, that describes which **predicates** defined in this International Standard a system implements

3.26
function
rule that associates each **element** from a domain (source, or domain of the function) to a unique element in another domain (target, co-domain, or range)

[ISO 19107:2003, definition 4.41]

3.27
predicate
set of computational **operations** applied to a data instance which evaluate to true or false

3.28
predicate expression
formal syntax for describing a **predicate**

3.29
base URL
HTTP GET URL for a server's OGC capabilities document without the GetCapabilities request parameters attached

NOTE: this base URL must match the HTTP GET base URL reported in the Capabilities document of the service

NOTE: the base URL is used to identify the server in lieu of a service id which is not currently define in OWS common but has been posted a change request to OGC (see OGC 11-117)

3.30
category document
documents that describe the categories allowed in Collection

3.31
change feed
collection of ATOM entries that describe changes to a data store expressed using the WFS Transaction syntax (see OGC 04-094)

3.32
collection
resource that contains a set of member resources

NOTE In this candidate standard, collection are implemented as ATOM feeds (see IETF 4287)

3.33
collector
a person or entity that proposes changes to data

3.34
entry resource
members of a collection that are represented as ATOM entry documents (see IETF RFC 4287)

3.35
event
any detectable or discernable occurrence that has significance for the management of an SDI

3.36
follower
person or process that accesses or subscribes to the replication feed of a GSS for the purpose of data synchronization

3.37
integrator
person or process that reviews proposed data changes and then makes a determination (based on established criteria) if the proposed change is acceptable or not

3.38
member resource
resource whose IRI is listed in a Collection with a atom:link element with a relation of "edit" or "edit-media"

3.39
publisher
synonym for collector (see X.X)

3.40
replication feed
collection of ATOM entries containing a log of changes that have been applied to a data store that can be used for the purpose of replicating or synchronizing with that data store

3.41
representation
entity included with a request or response (see IETF RFC 2616)

3.42
resolution feed
collection of ATOM entries describing the disposition of proposed changes listed in a change feed

3.43
reviewer
synonym for integrator (see 4.15)

3.44
service document
XML document that describes the location and capabilities of one or more Collections grouped into Workspaces

3.45
topic
collection of ATOM entries that satisfy some query predicates

NOTE: this is also referred to as a filtered feed because a topic is generated by querying a base feed and applying some predicate; for example a topic could consist of all the entries that lie within some defined boundary

3.46
workspace
named group of collections

4 Conventions

4.1 Abbreviated terms

Some more frequently used abbreviated terms:

API	Application Program Interface
AtomPub	ATOM Publishing Protocol
CGDI	Canadian Geospatial Data Infrastructure
CGI	Common Gateway Interface
COTS	Commercial Off The Shelf
CRS	Coordinate reference system
DCE	Distributed Computing Environment
DCOM	Distributed Component Object Model
DCP	Distributed Computing Platform
EPSG	European Petroleum Survey Group
FES	Filter Encoding Specification
GML	Geography Markup Language
GSS	GeoSynchronization Service
HTTP	Hypertext Transfer Protocol
HTTPS	Secure Hypertext Transfer Protocol
IDL	Interface Definition Language
IETF	Internet Engineering Task Force
KVP	Keyword-value pairs
MIME	Multipurpose Internet Mail Extensions
OGC	Open Geospatial Consortium
OWS	OGC Web Service
REST	Representational State Transfer

SDI	Spatial Data Infrastructure
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
UCR	Urban Climate Resilience thread
UML	Unified Modelling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
VSP	Vendor Specific Parameter
WFS	Web Feature Service
WNS	Web Notification Service
WSDL	Web Services Description Language
XML	Extensible Markup Language

5 Transaction scenario

5.1 Scenario overview

The transaction scenario described in this Engineering Report is related to the treatment of vented manholes during a flood scenario.

Vented Manholes exist in waste water systems in order to ensure that gasses do not build up to dangerous levels. However, in flood events, water entering vented manholes may cause overflow of the waste water in locations where it is not planned or managed. Prior to an anticipated event engineers inspect and treat (types of treatment include cover, seal, etc.) known trouble spots.

Re-inspection by engineers or reporting of problems by others (emergency personnel, public, etc.) continues through the duration of the event. Post event, actions include inspection / repair / removal of treatments and an evaluation of the suitability of each treatment in light of then event.

Currently this inspection and treatment process is poorly supported by WFS-T because of its lack of handling of multimedia data (i.e. photos and videos) and the lack of a nearest neighbor search capability. OGC Testbed-11 addressed these shortcomings.

Furthermore, the transaction scenario requires the exchange of information between multiple agencies each exposing their data resources using the WFS-T servers. As a result, interoperability between multiple WFSs – usually via a synchronizing agent is crucial. The original intent in the Testbed was to deploy a Geosynchronization service (see OGC 10-069r3) but because of the synchronization requirement at multiple levels, starting from mobile devices out in the field and scaling up to enterprise level synchronization between WFS servers, it was decided in OGC Testbed-11 to develop a peer-to-peer synchronization protocol to satisfy the requirement. This Engineering Report describes the protocol as it was applied to the enterprise-to-enterprise sync use case.

In relation to the enterprise-to-enterprise synchronization use case, often data about physical items is stored in more than one database. In the transaction scenario, manhole data is stored in two enterprise systems at the city level. Both databases contain the same data to a large extent, but differ in some feature characteristics. Testbed-11 explored options to support dynamic schema mapping from one database schema to another during the synchronization process of the two databases. The solution makes use of an ebRIM-based catalog acting as a schema registry with the ability to associate schemas from two WFS-T servers and identify if a schema translation script (e.g. XSLT script) exists a source WFS-T's schemas to a target WFS-T's schema.

NOTE: The schema translation component of this testbed is described in document OGC 15-010, OGC Testbed-11 WFS-T Information Exchange architecture EE, and is noted here because of the likely need for schema translation in the enterprise-to-enterprise synchronization use case.

5.2 Scenario outline

5.2.1 WFS-T Image transactions using WFS-T (Manhole treatment)

Image or multimedia WFS-T transactions were illustrated during the OGC Testbed-11 using the scenario outlined here:

1. Engineer is instructed to go out in the field to check on manholes;

2. Engineer finds first manhole;

3. Engineer performs necessary treatment and takes a picture of the manhole;

4. Engineer uploads the picture to an enterprise database;

5. Engineer performs a nearest-neighbor query to find the next manhole;

6. Engineer discovers next manhole and goes there;

7. Engineer checks WFS if this manhole has been treated already (there are many engineers in the field);

8. Engineer finds out that the manhole has been treated but not photographed; and

9. Engineer takes photo and sends it to a WFS-T.

This scenario also illustrates the need for a nearest neighbor search which was manifested in the OGC Testbed-11 demonstration as a stored query offered by the CubeWerx Inc. WFS (see 6).

5.3 Components

Table 1 summarizes the components deployed during Testbed-11 to satisfy the transactions scenarios (i.e. flood transaction scenario and the enterprise-to-enterprise sync scenario).

Table 1 – Components used in OGC Testbed-11 transaction scenarios

Scenario Component	Vendor	Component version Version	Purpose in scenario
Engineer's central database	CubeWerx	CubeWerx Suite 8.1.2 for Oracle 12.1 on Linux f20 x64	1. Provides nearest neighbor search capability 2. Provide multimedia WFS-T support 3. Act as sync peer in enterprise-to-enterprise sync experiment.
City corporate database	GIS-FCU	Geoserver 2.7.1.1 with MySQL (Apache 2.0)	Second server providing multimedia WFS-T support.
	IBM Cloudant	Bespoke WFS façade sitting on top of CouchDB	Act as sync peer in enterprise-to-enterprise sync experiment.
Mobile client used by field engineers and inspectors	GIS-FCU	Custom built Android app	Act as mobile client to locate manholes, evaluate their status, take photo and upload to enterprise server.

Figure 1 graphically illustrates the components used in the transaction scenarios tested during the OGC Testbed-11. The green box in Figure 1 delimits the components used for the enterprise-to-enterprise sync scenario which is described in Clause 8 of this document. The blue box delimits the components used to demonstrate the transactional flood scenario.

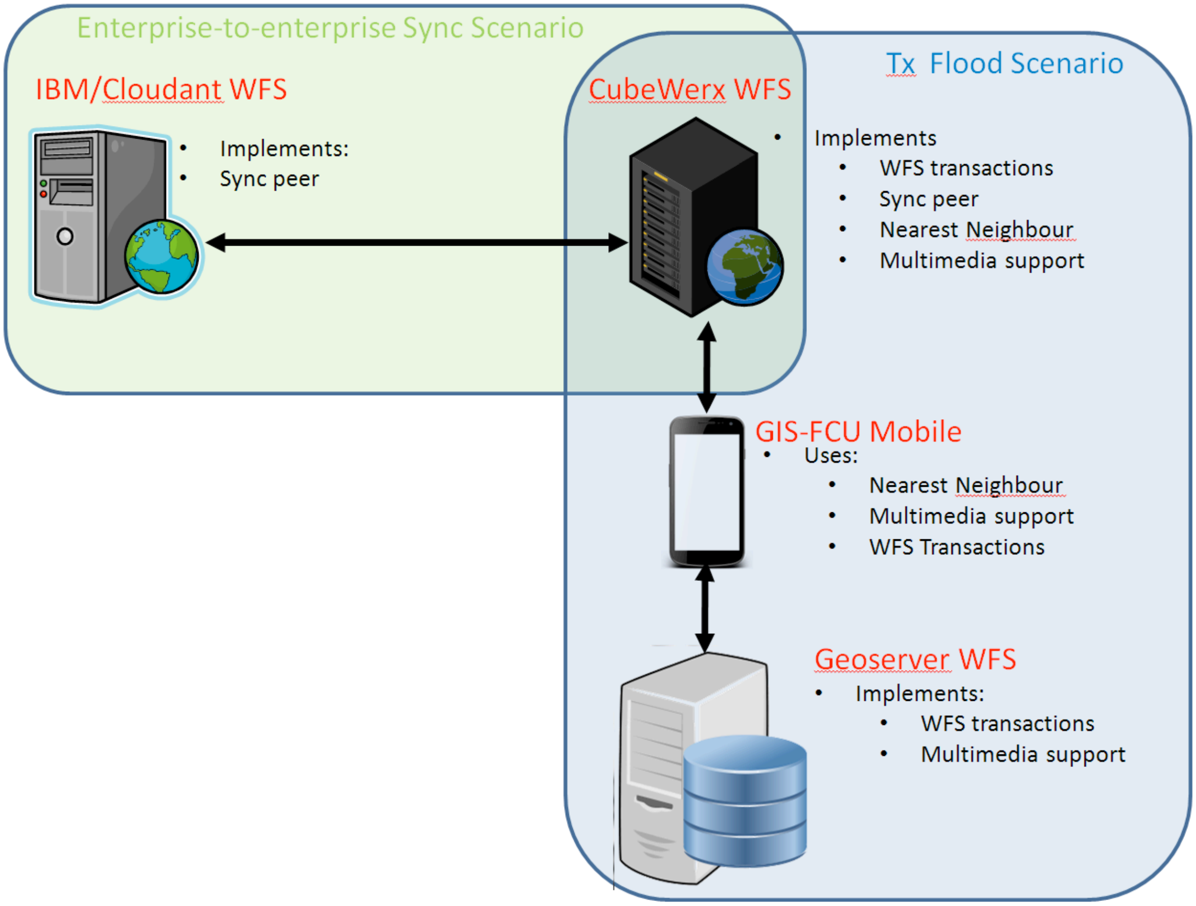


Figure 1 – Components deployed for transactional flood scenario and enterprise-to-enterprise sync scenario

5.4 Data

The feature type of interested for the flood transaction scenario is the wwAccess feature type which stores the list of manholes including vented manholes (i.e. AccessType='Vented Manhole').

6 Stored queries for OGC Testbed-11

6.1.1 Introduction

6.1.2 Nearest Neighbor Search

Part of the transaction scenario in the OGC Testbed-11 was to use a mobile client out in the field to check the status of manhole covers. The client would be used to find the nearest manholes to the location of the device which would then be treated (if required) and their status updated accordingly via WFS-T.

In order to satisfy this part of the transaction scenario, a stored query was added to the CubeWerx WFS with the identifier "urn:cw:def:query:OGC-WFS::NearestNeighbours". The server's description of this stored query can be obtained using the following RESTful request to the CubeWerx server:

http://www.pvretano.com/cubewerx/cubeserv/default/wfs/2.5.0/ows11/query?storedQuery_Id=urn:cw:def:query:OGC-WFS::NearestNeighbours

The following table more conveniently presents the parameters of the nearest neighbors stored query:

Table 2 – Nearest Neighbor Stored Query Parameters

Parameter Name	Description
FEATURETYPE	The name of the WFS feature type to query.
LAT	Latitude of the search point
LON	Longitude of the search point
LATLONSRS	Coordinate reference system used for lat, lon parameters
PREDICATE	An additional non-spatial predicate for the nearest neighbors search

The stored query returns up to "count" features in the response sorted by their distance from the specified reference point. The following URL returns the 5 manhole covers nearest to the point (-43,172):

http://www.pvretano.com/cubewerx/cubeserv/default/wfs/2.5.0/ows11/query/urn:cw:def:query:OGC-WFS::NearestNeighbours?featureType=WWACCESS&lat=-43&lon=172&latlonSrs=urn:ogc:def:crs:EPSG::4326&count=5

In the OGC Testbed-11 transaction scenario, in addition to finding the nearest manholes, the mobile client had to find the nearest unchecked manholes. To satisfy this requirement, two changes were made.

First an additional property named "checked" was added to the wwAccess feature type. The following link gets the schema of the feature type:

http://www.pvretano.com/cubewerx/cubeserv/default/wfs/2.5.0/ows11/schema?typeName=wwAccess

to show the modification (which is highlighted in yellow):

```
<xs:schema
   targetNamespace="http://schemas.cubewerx.com/namespaces/null"
   xmlns:cw="http://schemas.cubewerx.com/namespaces/null"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:gml="http://www.opengis.net/gml/3.2"
   elementFormDefault="qualified"
   version="1.0">
   <xs:import namespace="http://www.opengis.net/gml/3.2"
      schemaLocation="http://www.pvretano.com/schemas/gml/3.2.1/gml.xsd"/>
   <xs:element name="wwAccess" type="cw:wwAccessType"
      substitutionGroup="gml:AbstractFeature"/>
   <xs:complexType name="wwAccessType">
      <xs:complexContent>
         <xs:extension base="gml:AbstractFeatureType">
            <xs:sequence>
               <xs:element name="Geometry" type="gml:PointPropertyType">
                  <xs:annotation>
                     <xs:documentation>
                        <cwmeta:Metadata/>
                     </xs:documentation>
                  </xs:annotation>
               </xs:element>
               <xs:element name="uid" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="AccessType" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="SourceId" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="LocationCe" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="NearestPru" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:integer">
                        <xs:totalDigits value="10"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="HansenId" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:integer">
                        <xs:totalDigits value="10"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="n" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:decimal">
                        <xs:totalDigits value="23"/>
                        <xs:fractionDigits value="15"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="e" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:decimal">
                        <xs:totalDigits value="23"/>
                        <xs:fractionDigits value="15"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="WwAccessID" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:integer">
                        <xs:totalDigits value="10"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="Compkey" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:integer">
                        <xs:totalDigits value="10"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="HeightAbov" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:decimal">
                        <xs:totalDigits value="23"/>
                        <xs:fractionDigits value="15"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="CreatedBy" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="CreatedDat" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="Constructi" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:integer">
                        <xs:totalDigits value="10"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="Depth" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:decimal">
                        <xs:totalDigits value="23"/>
                        <xs:fractionDigits value="15"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="Maintenanc" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="ServiceSta" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="Decommissi" type="xs:dateTime" minOccurs="0"/>
               <xs:element name="YearLaid" type="xs:dateTime" minOccurs="0"/>
               <xs:element name="SAPInterna" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="LastEditDa" type="xs:dateTime" minOccurs="0"/>
               <xs:element name="LastEditUs" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="254"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="WwAccessSt" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:integer">
                        <xs:totalDigits value="10"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
```

```
               <xs:element name="Photo" minOccurs="0">
                  <xs:complexType>
                     <xs:simpleContent>
                        <xs:extension base="xs:base64Binary">
                           <xs:attribute name="url" type="xs:anyURI"
                              use="optional"/>
                           <xs:attribute name="mimeType" type="xs:string"
                              use="required"/>
                           <xs:attribute name="role" type="xs:string"
                              use="optional"/>
                           <xs:attribute name="length"
type="xs:positiveInteger"
                              use="optional"/>
                        </xs:extension>
                     </xs:simpleContent>
                  </xs:complexType>
               </xs:element>
               <xs:element name="Checked" minOccurs="0">
                  <xs:simpleType>
                     <xs:restriction base="xs:integer">
                        <xs:totalDigits value="1"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
</xs:schema>
```

A value of 0 for the “Checked” property would indicate that the manhole was NOT checked and a value of 1 would mean the opposite.

Second, the PREDICATE parameter was defined for the nearest neighbor stored query so that additional constraints could be specified on the nearest neighbor search.

With these two elements in place, the following request – that the mobile client would send to the CubeWerx WFS -- finds the nearest unchecked manhole covers to the specified point:

http://www.pvretano.com/cubewerx/cubeserv/default/wfs/2.5.0/ows11/query/urn:cw:def:query:OGC-WFS::NearestNeighbours?featureType=WWACCESS&lat=-43&lon=172&latlonSrs=urn:ogc:def:crs:EPSG::4326&maxFeatures=5&predicate=Checked%3D0

Notice that the value of the predicate property is set to “Checked=0” so that only unchecked manholes are searched.

7 Multimedia support in WFS

7.1 Introduction

This section describes how multimedia-valued feature properties (i.e. image, video, audio, etc.) can be managed and manipulated within a WFS.

This section also describes the implementation of multimedia data handling implemented in the CubeWerx WFS for the OGC Testbed-11. Because of resource limitations (and the lack of RESTful clients) for Testbed-11, multimedia handling in the CubeWerx server has (for the most part) been implemented in the POX-with-POST binding.

7.2 Schema

The CubeWerx WFS has been enhanced to handle feature properties with multimedia values (i.e. audio, video. images, etc.). The handling of multimedia values relies on the Geography Markup Language (GML) simple features profile (with Corrigendum) (2.0) standard which defines a schema pattern for encoding in-line (as base64Binary) or referencing binary data. The relevant schema pattern from the standard (see OGC 10-100r3, clause 8.4.4.10) is:

```
<xsd:element name="propertyName" [minOccurs="0|1"] [maxOccurs="1"]>
   <xsd:complexType>
      <xsd:simpleContent>
         <xsd:extension base="xsd:base64Binary|xsd:hexBinary">
            <xsd:attribute name="url" type="xsd:anyURI" use="optional"/>
            <xsd:attribute name="mimeType" type="xsd:string" use="required"/>
            <xsd:attribute name="role" type="xsd:string" use="optional"/>
            <xsd:attribute name="length" type="xsd:positiveInteger"
                           use="optional"/>
         </xsd:extension>
      </xsd:simpleContent>
   </xsd:complexType>
</xsd:element>
```

Using this schema pattern, the manhole feature type wwAccess was modified to add a "Photo" property. Here is the schema of the modified feature type: http://www.pvretano.com/cubewerx/cubeserv/default/wfs/2.5/ows11/schema

7.3 Create a new feature with a multimedia-valued property

Creating a new feature with an image value for the Photo property is simply a matter of either encoding the value in-line within the Insert action as a base64 value or referencing the value by URL as the value of the href attribute. In the latter case, the server retrieves the multimedia value before creating the feature in the WFS's repository. Here is an example of an insert statement with in-line value encoding (as is currently implemented in the CubeWerx WFS):

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:Transaction
```

```
   service="WFS"
   version="2.5.0"
   xmlns="http://www.someserver.com/myns"
   xmlns:cw="http://schemas.cubewerx.com/namespaces/null"
   xmlns:fes="http://www.opengis.net/fes/2.5"
   xmlns:wfs="http://www.opengis.net/wfs/2.5"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/wfs/2.5
                       http://www.pvretano.com/schemas/wfs/2.5/wfs.xsd
                       http://schemas.cubewerx.com/namespaces/null

http://www.pvretano.com/cubewerx/cubeserv.cgi?datastore=ows11&amp;service=WFS&a
mp;version=2.5&amp;request=DescribeFeatureType&amp;typeName=wwAccess">
   <wfs:Insert>
      <wwAccess>
         <Geometry>
            <gml:Point gml:id="GID1" srsName="urn:ogc:def:crs:EPSG::4326">
               <gml:pos>-43.603 172.574</gml:pos>
            </gml:Point>
         </Geometry>
         <uid>WWMH-4078</uid>
         <AccessType>Standard Manhole</AccessType>
         <SourceId>2604S00330</SourceId>
         <LocationCe>Non Verified</LocationCe>
         <NearestPru>1</NearestPru>
         <HansenId>63945</HansenId>
         <n>5734056.64</n>
         <e>2475523.95</e>
         <WwAccessID>4078</WwAccessID>
         <Compkey>8148</Compkey>
         <HeightAbov>30.8</HeightAbov>
         <Constructi>1</Constructi>
         <Depth>0.9</Depth>
         <Maintenanc>City Water and Waste</Maintenanc>
         <ServiceSta>In Service</ServiceSta>
         <YearLaid>1983-01-01</YearLaid>
         <SAPInterna>IE000000000010754107</SAPInterna>
         <LastEditDa>2013-10-18</LastEditDa>
         <LastEditUs>CCITY\TredinnickC</LastEditUs>
         <Photo
mimeType="image/jpeg">/9j/4AAQSkZJRgABAQEAZABkAAD/2wBDAAIBAQIBAQICAgICAgICAwUDA
wMDAwYEBAMFBwYHBwcG
BwcICQsJCAgKCAcHCg0KCgsMDAwMBwkODw0MDgsMDAz/2wBDAQICAgMDAwYDAwYMCAcIDAwMDAwM
DAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAz/wAARCAISArwDAREA
AhEBAxEB/8QAHQAAAAcBAQEAAAAAAAAAAAAAAAQIDBAUGBwAICf/EAEsQAAECBAQFAgQEBAQFAgUB
CQECEQADBCEFEjFBBhMiUWEHcQgUMoEjQpGhUrHB0RUz4fAJFiRi8UNyFyY0U5IlNXMYGYKTY6Ky
<!-- more base64 data here removed for the sake of brevity -->
/a/j/wCjieof9yhCapM5wpQLjfwI9A39pjg4/Jh4JywtIC1AZhvG1PFQ58m/YEVU3nJ/EmWB/MYX
P90pfAlKaarInqVqN4vD/ckb/wANhzTVU3lpPMmODrmMasv93+IDx+A+kVk3kp/Fmf8A5GNXD+En
Lz/2pH+G1c0LR+LM+k/mMWv9oPN/bglZ9XN58r8WZd/zGG/EQ3kISK2cx/Fm/X/EYv8A4Yseyayd
zk/izNFfmPaLx/EgjPrp7S/xpv0D85h2T+438f8ASAJ7n//Z</Photo>
      </wwAccess>
   </wfs:Insert>
</wfs:Transaction>
```

Simply POSTing this message to the server's transaction controller resource will create a new instance of the wwAccess feature that also includes a photo valued property.

7.4 Update a multimedia-valued property

Updating a feature with a multimedia value proceeds in the usual manner; a Transaction message is posted to the transaction handler of the server. Multimedia values can either be encoded in-line or referenced in which case the server shall retrieve the multimedia value prior to updating the feature. Here is an example of an update message that updates the value of the Photo property in the wwAccess feature:

```
<?xml version="1.0" ?>
<wfs:Transaction
   version="2.5.0"
   service="WFS"
   xmlns="http://www.someserver.com/myns"
   xmlns:cw="http://schemas.cubewerx.com/namespaces/null"
   xmlns:fes="http://www.opengis.net/fes/2.5"
   xmlns:wfs="http://www.opengis.net/wfs/2.5"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/wfs/2.5
       http://www.pvretano.com/schemas/wfs/2.5/wfs.xsd
       http://schemas.cubewerx.com/namespaces/null

http://www.pvretano.com/cubewerx/cubeserv.cgi?datastore=ows11&amp;servi
ce=WFS&amp;version=2.5&amp;request=DescribeFeatureType&amp;typeName=wwA
ccess">
   <wfs:Update typeName="wwAccess">
      <wfs:Property>
         <wfs:ValueReference>Photo</wfs:ValueReference>
         <wfs:Value>
            <Photo
mimeType="image/jpeg">/9j/4AAQSkZJRgABAQAAAQABAAD//gA7Q1JFQVRPUjogZ2Qta
nBlZyB2MS4wICh1c2luZyBJSkcg
SlBFRyB2NjIpLCBxdWFsaXR5ID0gNjUK/9sAQwALCAgKCAcLCgkKDQwLDREcEhEPDxEiGRo
UHCkk
KyooJCcnLTJANy0wPTAnJzhMOT1DRUhJSCs2T1VORlRAR0hF/9sAQwEMDQ0RDxEhEhIhRS4
nLkVF
RUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVFRUVF/8AAEQg
BgAHg
<!... more base64 encoded image data here -->
8ZjG51B92FZwY2aCTc4IqyjVTjli7yx/99CpvOiH/LaP/vsVpcklJy2Kg1L/AI8v+BClWaL
JJlj/
AO+xUd/LE1kwEsbHI4DCk9io7mRmkpN6/wB5fzpN6/3h+dc50C0lJvX+8Pzo3r/eH50hC5o
Jpu5f
7w/Ok3r/AHh+dADqKbuX+8Pzo3L/AHhQA6k60m5f7w/Ok3D1H50wFpabuHqPzo3D+8PzpAK
aa33T
RuH94fnSMw2nkdPWgD//2Q==</Photo>
         </wfs:Value>
      </wfs:Property>
      <fes:Filter>
         <fes:ResourceId
rid="CWFID.WWACCESS.0.34723.BA89DFD04C0B73161F20020000"/>
      </fes:Filter>
   </wfs:Update>
</wfs:Transaction>
```

7.5 Query

7.5.1 Feature query

In a query response, the CubeWerx server encodes multimedia property values by reference rather than encoding the values in-line in order to keep the bulk of the response down. Here is an example of a REST URL that returns a wwAccess feature with a photo: http://www.pvretano.com/cubewerx/cubeserv//default/wfs/2.5.0/ows11/wwAccess/CWFID.WWACCESS.0.34729.BA89DFD04C0B73161F20020000. Resolving the href URL that is the value of the Photo property will retrieve the image.

Although not yet implemented in the CubeWerx server, in the future a client would be able to use the resolve parameters (see OGC 09-025r2 clause 7.6.4) to control whether the multimedia data is encoded in-line or not [resolve=none|local means that the value is referenced; resolve=remote|all means that the value is encoded in-line in the response].

NOTE 1: If you follow the above link in your browser, the response may be HTML as the browser sets the Accept header to prefer HTML output. Adding the 'f' parameter allows GML to be returned: http://www.pvretano.com/cubewerx/cubeserv//default/wfs/2.5.0/ows11/wwAccess/CWFID.WWACCESS.0.34729.BA89DFD04C0B73161F20020000?f=application/gml%2Bxml;%20version=3.2

NOTE 2: The editor is still working of the GeoJSON generation of multimedia values but the pattern will be that established for GML (i.e. the value shall be encoded either in line or by reference depending on the value of the resolve query parameter).

7.5.2 Property Value query

Getting the multimedia value of a property is simply a matter of using a GetPropertyValue operation or referencing the property value's URL. Here is an example of the latter: http://www.pvretano.com/cubewerx/cubeserv/default/wfs/2.5.0/ows11/wwAccess/CWFID.WWACCESS.0.34709.BA89DFD04C0B73161F20020000/photo?f=application/gml%2Bxml;+version=3.2.

This is very nice but it would be better to be able to stream the "actual" or "native" binary value without the response container. In this way a client, say a browser, can handle the multimedia value in whatever manner it has been configured to do so (e.g. a browser could display a photo, or play the audio or video clip).

The initial idea for streaming multimedia values from a WFS was to define a new operation, similar to the one found in the CSW-ebRIM profile of the catalogue, called GetRepositoryItem. However, a better approach is to use what is already defined in the WFS; that is, the GetPropertyValue operation.

The GetPropertyValue operation (or accessing a property value's URL) does exactly as its name implies -- it fetches the value of a property of a feature. Unfortunately it returns the value encoded in-line as a base 64 value or referenced by URL within a ValueCollection container. There needs to be a way to signal to the server that a multimedia property value should be streamed in its native encoding and not as part of a standard response container. To satisfy this requirement a special token named "**native**" was defined and implemented in the CubeWerx server. This token can be used in the Accept header or as the value of the outputFormat|f parameter to let the server know that a multimedia value should be streamed in its native binary encoding tagged with its native MIME type. Here is an example: http://www.pvretano.com/cubewerx/cubeserv/default/wfs/2.5.0/ows11/wwAccess/CWFID.WWACCESS.0.34709.BA89DFD04C0B73161F20020000/photo?f=native

8 Enterprise-to-enterprise WFS-T synchronization

8.1 Introduction

The original intent in the testbed was to deploy a Geosynchronization service (see OGC 10-069r3) to support enterprise-to-enterprise synchronization. However, because of the synchronization requirement appearing at multiple levels, starting from mobile devices out in the field and scaling up to enterprise level synchronization between WFS servers, it was decided in the OGC Testbed-11 to develop a peer-to-peer synchronization protocol to satisfy the requirement.

This clause describes the peer-to-peer sync protocol implemented in the testbed to satisfy the enterprise-to-enterprise sync use case from OGC Testbed 11.

8.2 Components

Figure 2 illustrates the components involved in the OGC Testbed-11 enterprise-to-enterprise sync experiment.

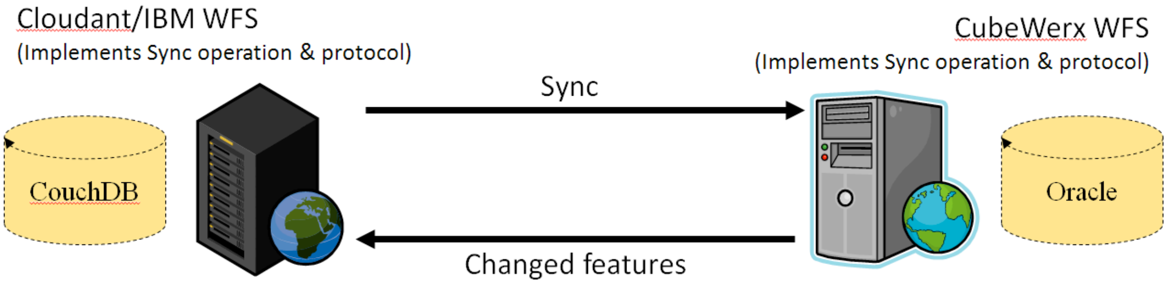

Cloudant/IBM WFS
(Implements Sync operation & protocol)
CubeWerx WFS
(Implements Sync operation & protocol)
Sync
CouchDB
Oracle
Changed features

Figure 2 - Components of the Sync experiment

The experiment was performed between two WFSs; the Cloudant/IBM WFS and the CubeWerx WFS. Table 3 lists the product versions and endpoint of the servers.

Table 3 – WFS product versions and end points

Vendor	Product	Supported WFS Versions	Endpoints
CubeWerx	CubeWerx Suite 8.1.1	2.5, 2.0, 1.1.0, 1.0.0	CubeWerx server
IBM Cloudant	RESTful JSON WFS	2.5	Cloudant server

Both servers implement version 2.5 of the WFS standard with the REST binding (see OGC 11-080r1). Furthermore, each server implemented the newly specified Sync resource (see 8.3) which allows the servers to synchronize the state of their features.

Table 4 lists the specific capabilities of each server used in the experiment. Cross-walking the capabilities resulted in the experiment being performed using features encoded with GeoJSON. In other words, when the server exchange features through the "Sync" resource, those features are encoded using GeoJSON. It should be noted the GML would work just as well if both server supported that output format.

Table 4 – Server capabilities

	CubeWerx	IBM Cloudant
Versions	2.5 2.0 1.1.0 1.0.0[2]	2.5
Operations	GetCapabilities DescribeFeatureType GetFeature ListStoredQueries DescribeStoredQueries GetPropertyValue Transaction Sync (new of Testbned-11) GET PUT POST DELETE	GET POST PUT DELETE
Output Formats	GML v3.2 GML v3.1.1 GML v2.1.2	GeoJSON

	GeoJSON KML SHAPE ATOM RSS HTML	
Spatial operators	Disjoint Equals Intersects Touches Crosses Contains Overlaps BBOX Within	BBOX
Spatial operands	gml:Envelope gml:Point gml:LineString gml:Polygon gml:CircleByCenterPoint	
Scalar operators	PropertyIsBetween PropertyIsEqualTo PropertyIsGreaterThan PropertyIsGreaterThanOrEqualTo PropertyIsLessThan PropertyIsLessThanOrEqualTo PropertyIsLike PropertyIsNotEqualTo PropertyIsNull	
Logical	And, Or, Not	
Available Stored Queries	GetFeatureById NearestNeighbours	
Number of CRSs	>10	<10
Support for GML SF	Yes	No
REST API	Yes	Yes
GeoJSON	Yes	Yes
ATOM	Yes	No
XSLT vendor	Yes	No

extension[1]		

8.3 The sync protocol

8.3.1 Introduction

This sub-clause describes the sync protocol that two enterprise servers implementing the “Sync” resource shall use to synchronize the state of their features.

In the simplest terms, the protocol involves each synchronization peer accessing the other’s “Sync” resource to get the set of changed features. The requesting peer then compares that list of changed features with the identically identified features in its data store and performs any necessary changes so that the feature states match.

This discussion assumes the existence of two hypothetical WFS sync peers labeled “**alpha**” and “**beta**.”

8.3.2 No prior synchronization between the peers

In the case where the two synchronization peers have never synchronized, the protocol proceeds as follows.

- **alpha** sends a “sync” request to **beta**
    - **REQUEST:**

      ```
      GET /sync?typename=%featuretype%&
                include_features=true&
                serviceid=%serviceid%&
                outputFormat=application/vnd.geo+json
      ```

    - **RESPONSE:**
        - In response to an initial sync request, all features in **beta** of the requested feature type shall be returned
        - The format of the response shall be as specified by the value of the outputFormat
            - For OGC Testbed-11 the response payload was a GeoJSON FeatureCollection consisting of GeoJSON Features
        - As is currently the case for WFS, all features in the response shall have a unique identifier. New or modified features shall overwrite existing ones by comparing the identifier value

        - Deleted features shall be identified by means of a special flag. In the OGC Testbed-11, features included a "_deleted" property which was set to true if the feature was deleted. This, however, is not a good general solution to this problem and more reflection is required.
        - The response shall include two custom HTTP headers
            - Checkpoint – whose values is a unique identifier
            - ServiceId which is a unique identifier for **beta**
    - Optional: peer-to-peer sync
        - System **beta** repeats the process by sending a sync request to **alpha**

8.3.3 Subsequent synchronization between peers

In the case where the synchronization peers **alpha** and **beta** has previously synchronized, the sync protocol proceeds as follows.

- Either the initial synchronization described above has occurred, or the two systems have been seeded with the same data set and a 'Checkpoint' identifier that identifies the data set's state at the time it was exported from the enterprise database has been distributed with the seed data.
- **alpha** sends sync request to **beta**
    - **REQUEST:**

```
GET /sync?typename=%featuretype%&
           include_features=true&
           serviceid=%serviceid%&
           checkpoint=%checkpoint%
```

        - %checkpoint% is a unique identifier acquired from system **beta** in Step 1 or manually obtained by having the checkpoint distributed with the seed data set
    - **RESPONSE:**
        - The response is the same as that from Step 1 except that only those features that have changed since the last checkpoint are included in the response container.

8.4 Sync resource

8.4.1 Introduction

This clause describes the sync resource which implements the peer-to-peer synchronization protocol developed and tested during the OGC Testbed-11.

The basic function of the Sync resource is to allow one server to ask another peer server "which of your features has changed since the last time I accessed the sync resource?" With this information, the requesting server can update the state of its features to match the state of the feature offered by the sync peer. When both peers access each other's "Sync" resource, peer-to-peer synchronization is accomplished.

8.4.2 Request parameters

The following table defines the parameters of the sync resource – which in the REST binding manifests itself as the sync resource with the path "/sync".

Table 5 – Parameters for the sync resource

Parameter Name	Description	Default Value	Obligation
typeName	The name of the WFS feature type to synchronize	None	Mandatory
serviceId	A unique identifier for the entity making the request.	None	Mandatory
checkPoint	Identifies that last point of synchronization for the specified serviceId	None	Optional on first sync; Mandatory thereafter.
startIndex	The optional startIndex parameter indicates the index within the result set from which the server shall begin presenting results in the response document.	1	Optional
count	The optional count parameter limits the number of explicitly requested values (i.e. features or property values) that are presented in a response document.	10	Optional
outputFormat[1]	The optional outputFormat parameter specifies the format used to encode resources in the response document.	application/gml+xml; version=3.2	Optional
NOTE 1: The default output format listed here is GML 3.2 in anticipation of the fact that if this			

resource is incorporated into the WFS standard, the default output format would be GML. In the OGC Testbed-11, however, GeoJSON was used as the default output format.

8.4.3 Response

The canonical response to a sync request is a feature collection, encoded in GML 3.2, that contains the set of features created, modified or deleted since the last sync request from the specified serviceId.

In the OGC Testbed-11 experiment, however, the list of changed, modified or deleted features was encoded using GeoJSON which was an output format that both participating servers supported.

8.5 Relationship to GSS

This clause discusses the relationship between the sync protocol describe herein and the GeoSynchronization Service (GSS, see OGC 10-069r3).

The purpose of a GeoSynchronization service is to allow organizations to deploy a transactional WFS to a "crowd" without giving the crowd direct transactional access to the server. Rather the GSS mediates the transactional interaction between members of the crowd and the WFS allowing the organization to validate data before being committed to the server. In this sense a GSS is a workflow manager for crowdsourcing WFSs. A convenient way to think about GSS is as an OGC standards-based version of Open Street Map. Unlike Open Street Map, however, which replies on the crowd to verify data, the GSS enforces a more formal and mandatory approach to data validation.

In addition to acting as a crowdsourcing manager, the GSS specification describes a subscription subsystem that allows interested parties to be notified by a GSS whenever changes are committed to the WFS that the GSS is managing. This subscription subsystem can also be used synchronize interested third-party WFSs with the crowdsourced WFS that is managed by the GSS.

The current GSS synchronization subsystem is a subscription-based, push synchronization. This means that whenever a set of changes is committed to the crowdsourced WFS server managed by the GSS, the GSS will push those changes (via a WFS transactions) to whichever third-party servers have subscribed to the GSS for the purpose of synchronization (see7.2, OGC 15-010).

By contrast, the enterprise-to-enterprise sync capability tested in the OGC Testbed-11 describes the mechanism and protocol for synchronizing two servers but does not describe when synchronization occurs or how it is triggered. It is a more light-weight synchronization capability since no mediating service such as GSS is required. However, only WFSs that have implemented the necessary resources and protocol can participate in the sync thus preventing legacy WFSs for being synchronized. The GSS, on the other hand, does not require any changes to the WFSs participating in a sync operation and so legacy as well as current WFSs can be synchronization peers.

It seems reasonable, moving forward, that the enterprise-to-enterprise synchronization protocol be integrated into the GSS as part of its synchronization subsystem. However, time a resource limitation during the OGC Testbed-11 did not allow for this investigation to be performed and so this is flagged as a future work item.

Annex A

Alternative implementation for WFS-T multimedia handling

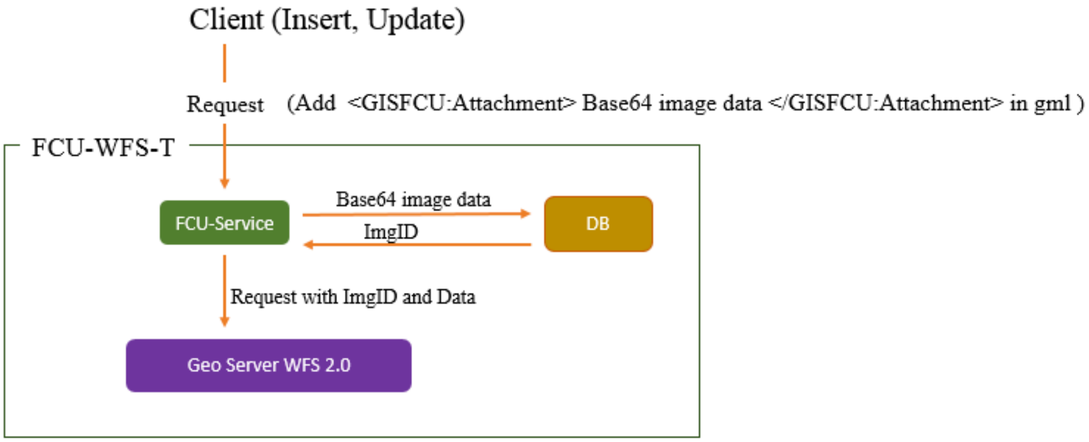
The following is a description of an alternative the mechanism that GISFCU implemented in their mobile client for handling multimedia content such as photos.

1. Mobile client generates request by following standard WFS request, but adds a customized tag, for instance <GISFCU:Attachment>, in order to include the base64 code of picture. For example: <GISFCU:Attachment>Base64 image data </GISFCU:Attachment>.

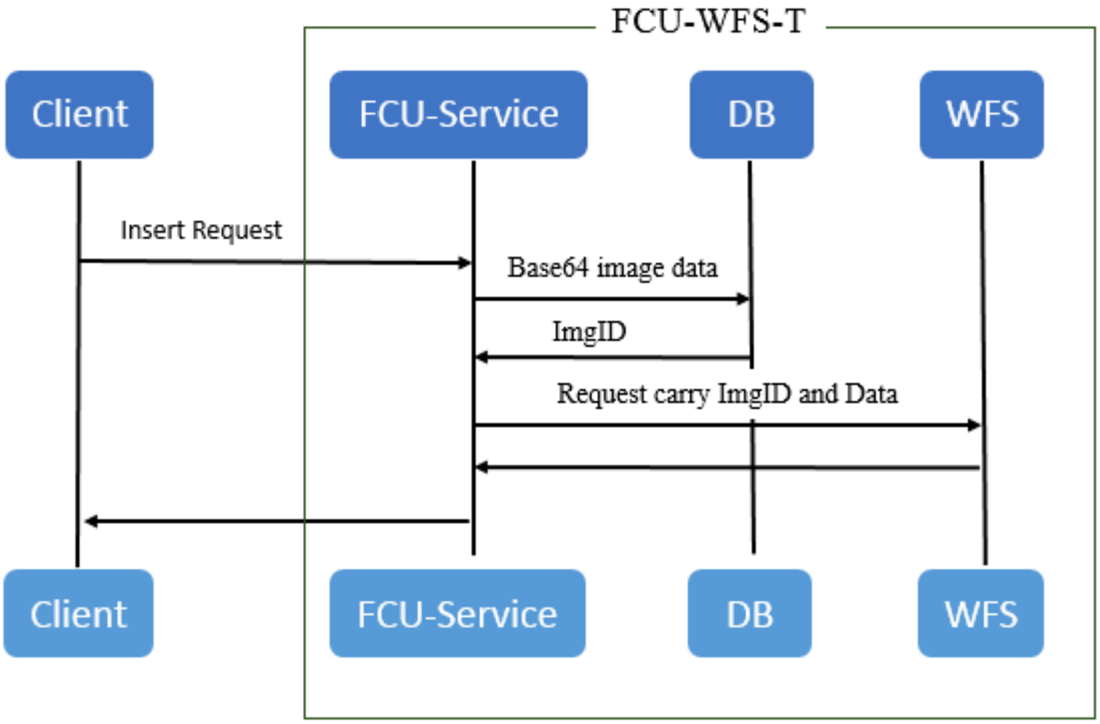
2. All features are stored as shapefile format in the backend of Geoserver. However, to improve the performance there is pre-process work to store the pictures (base64) into a database, and then replace the id (ImgID) of the pictures. The content of that tag will become <GISFCU:Attachment>ImgID </GISFCU:Attachment>.

3. The GISFCU service then resends the request to the WFS of Geoserver.

The figure below illustrates the process.


Client (Insert, Update)
Request
(Add <GISFCU:Attachment> Base64 image data </GISFCU:Attachment> in gml)
FCU-WFS-T
FCU-Service
Base64 image data
ImgID
DB
Request with ImgID and Data
Geo Server WFS 2.0

Sequence diagram.


FCU-WFS-T
Client
FCU-Service
DB
WFS
Insert Request
Base64 image data
ImgID
Request carry ImgID and Data
Client
FCU-Service
DB
WFS