Open Geospatial Consortium

Submission Date: 2015-08-10

Approval Date: 2016-03-08

Publication Date: <yyyy-dd-mm>

External identifier of this OGC® document: <http://www.opengis.net/doc/spec/WQS/1.0>

Reference number of this Document: OGC 14-121

Version: 1.0

Category: OpenGIS© Discussion Paper

Editor: Peter Baumann

OGC® Web Query Service

Warning

This document is not an OGC Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as 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: OGC Discussion Paper
Document subtype: Interface
Document stage: Final
Document language: English

Contents

Page

Tables

Page

i. Abstract

This OGC *Web Query Service* (WQS) defines a service interface for retrieving any kind of information from some information offering provided by the server addressed. WQS is completely agnostic of any semantics and, therefore, not bound to any predefined structures, such as coordinates, features, coverages, or metadata. This makes WQS particularly suitable for retrieval from heterogeneous data offerings combining features, coverages, and catalog information in some application defined way. A second use case is selective retrieval from a Capabilities document to avoid downloading large such documents and performing extraction on client side.

To this end, the *Query* request type is defined which, based on an XPath expression as input, extracts the matching information from the service's offering and returns it (currently: as an XML document).

ii. Keywords

ogcdoc, Web Query Service, query, xpath

iii. Preface

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.

iv. Submitters

The following organizations have submitted this Interface Specification to the Open Geospatial Consortium, Inc.:

- Jacobs University Bremen

v. Document Contributor Contact Points

Name	Organization
Peter Baumann	Jacobs University Bremen, rasdaman GmbH

vi. Revision history

Date	Release	Author	Paragraph modified	Description
2014-12-01	0.0.1	Peter Baumann	All	Created

2016-04-05	1.0.0	Peter Baumann	Several	Reformatted for publication

vii. Future Work

Extensions to this concept might address the following aspects:

- Extending model capabilities from hierarchical structures (as supported now) to more general structures (such as semantic graphs)
- Extending query capabilities from XPath to further, more powerful paradigms (such as SPARQL)
- Adding further encodings, such as JSON
- Adopting this functionality as part of OWS Common, given its general, overarching relevance

1 Scope

This OGC Web Query Service (WQS) specification defines how to selectively retrieve data from a server, without making any assumption about the data offered.

2 Conformance

This document establishes the following requirements and conformance classes:

- *query*, of URI http://www.opengis.net/spec/WQS/1.0/req/xpath; the corresponding conformance class is *xpath*, with URI http://www.opengis.net/spec/WQS/1.0/conf/xpath.

  This is the mandatory conformance class of this specification.

Standardisation target are WQS implementations (currently: servers).

Requirements URIs defined in this document are given by

http://www.opengis.net/spec/WQS/1.0/req/req{reqname},

conformance test URIs are given by

http://www.opengis.net/spec/WQS/1.0/conf/req{reqname}.

whereby {reqname} in the numbering scheme is to be substituted by the requirement identifier provided in the text.

Annex A of this document lists the conformance tests which shall be exercised on any software artefact claiming to implement WQS.

3 Terms and definitions

For the purposes of this document, the terms and definitions given in the above references apply. In addition, the following terms and definitions apply. An arrow "→" indicates that the following term is defined in this Clause.

3.1 Offering [of a service]

The complete information which a service provides for retrieval by clients, conceptually represented by a single XML document

4 Class *Query*

4.1 Overview

This Clause 4 defines the mandatory core requirements class, *query*. Clients and servers supporting this *query* requirements class shall support XPath-based selection from a WQS server's coverage offerings through a dedicated request type, *Query*, operating on the information offering of the WQS service, seen as a single XML document.

4.2 *GetCapabilities* request

A server announces support of the *query* requirements class to a client by adding the URL identifying this extension to the list of supported extensions delivered in the Capabilities document.

Requirement 1 – profile:
A WQS service implementing requirements class *query* **shall** include the following URI in the `Profile` element of the `ServiceIdentification` in a *GetCapabilities* response:
`http://www.opengis.net/spec/WQS/1.0/conf/query`

4.3 *Query* request

4.3.1 *Query* request

This request assumes an XML document on the server which is of some structure not specified further; in particular, no specific underlying schema is assumed. Part of this offering may be the Capabilities document, but this is at the discretion of the service.

The XPath expression submitted is evaluated against this single conceptual XML document, and the result is returned to the client.

Requirement 2 – request:
A *Query* request **shall** adhere to Figure 1 and Table 1.

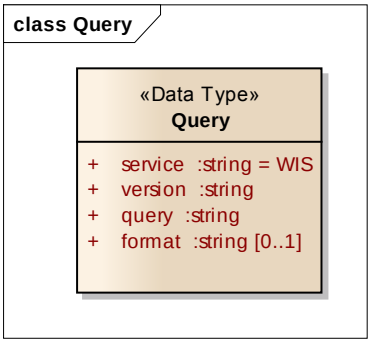

Figure 1 — `Query` request UML diagram

Table 1 — Components of `WQS::Query` request structure

Name	Definition	Data type	Multiplicity
`query`	XPath expression to be evaluated by the server	`string`	one (mandatory)
`Format`	Identifier of the output format, expressed as MIME type	`string`	Zero or one (optional)

Requirement 3 - xpath:
The `query` parameter in a *Query* request **shall** contain a syntactically valid XPath expression as per W3C XPath [1].

Example The following examples are valid expressions which may yield nonempty results on a Capabilities document; specifically, it extracts all data formats supported by this server:

```
//formatSupported
```

4.3.2 *Query* response

The response to a successful *Query* request is a document (which may contain XML tags) containing the information extracted from the server's offering in some appropriate encoding.

Requirement 4 – response:
The response to a successful *Query* request **shall** be given by the evaluation of the `query` argument against the offering of the WQS server.

Note: A server may reject requests generating foreseeably excessive amounts of data, such as retrieving an image encoded in GML.

Requirement 5 – encoding:
The response to a successful *Query* request containing a `format` parameter **shall** be encoded in the format specified by the `format` parameter.

Note 1: If no format parameter is provided in a request then the server may choose some default encoding on its own.

Note 2: Container formats like GMLJP2, zip, etc. are particular amenable to heterogeneous information retrieval.

4.4 Request Encodings

6.8.1 Overview

This Subclause specifies the encoding of a *Query* operation for each WQS protocol binding that a client and server support.

6.8.1 GET/KVP Encoding

Requirement 6 – get-kvp:
In a *Query* request using the GET/KVP protocol, a `query` parameter with value `x` **shall** be represented by an http key/value pair as follows, with `x` properly using http entities where required:

```
QUERY=x
```

Example The following is a complete Q*uery* request in GET/KVP notation; it delivers a list of all coverage identifiers:

```
http://www.acme.com/ows?
        SERVICE=WQS &
        VERSION=1.0 &
```



```
REQUEST=Query &
QUERY=/Capabilities/Contents/CoverageSummary/CoverageId &
FORMAT=application/gml+xml
```

6.8.2 XML/POST Encoding

Requirement 7 – xml-post:
A *Query* request using the XML/POST protocol **shall** be encoded as an `wis:query` element as defined in the XML Schema accompanying this specification.

Example The following is a complete *Query* request plus a response (assuming success) in XML/POST encoding:

```
<?xml version="1.0" encoding="UTF-8"?>
<wis:query xmlns:wis="http://www.opengis.net/wis/1.0">
    <wis:query>
        /Capabilities/Contents/CoverageSummary/CoverageId
    </wis:query>
</wis:Query>
```

6.8.3 SOAP Encoding

Requirement 8 – soap:
A *Query* request using the SOAP protocol **shall** be encoded as a `wis:query` element as defined in the XML Schema accompanying this specification.

4.5 Exceptions

Requirement 9 – exceptions:
When a WQS server encounters an error while evaluating a *Query* operation the server **shall** return an exception report message from the list in Table 2 with a `locator` parameter value as specified in the right column of Table 2 for each `exceptionCode` listed.

Table 2 — *Query* exception codes

`exceptionCode` value	**HTTP code**	**Meaning of exception code**	**`locator` value**
`InvalidQuery`	404	QUERY parameter does not represent a valid XPath expression	Position of violating element / parameter
`InvalidFormat`	404	FORMAT parameter does not specify a known MIME type, or result cannot be encoded in the format requested	FORMAT parameter
`NoMatch`	404	XPath expression in QUERY parameter does not address any element defined in the server offering	Position of violating element / parameter

ExcessiveResultVolume	404	Query would return an excessive amount of data (e.g., when requesting coverage range sets)	n.a.

Bibliography

[1] XML Path Language (XPath) Version 1.0. W3C Recommendation 16 November 1999, http://www.w3.org/TR/xpath/

[2] P. Baumann: OGC Web Coverage Service (WCS) 2.0 – Core. Version 2.0.1

Annex A

Abstract test suite

A WQS implementation must satisfy the following system characteristics to be conformant with this specification.

Test identifiers are relative to http://www.opengis.net/spec/WQS/1.0/query/conf. The identifier of each test consists of this path, a "/" (slash) character, and the name of the corresponding requirement.

A.1 Conformance Test Class: *query*

The OGC URI identifier of this conformance class is:
http://www.opengis.net/spec/WQS/1.0/conf/query.

Test Purpose:	Requirement 1
Test method:	Send valid *GetCapabilities* request to system under test. Check Capabilities document returned whether it contains the required element in the proper position. Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 2
Test method:	Send *Query* requests to system under test. Verify that the structures referenced by the requirement are accepted by the server (and returned in responses, respectively), and only those. To this end, send both valid and violating requests; in case of automatically verifiable definitions (such as XML Schema), verify through appropriate tools; otherwise (such as with UML), implement according tests manually. Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 3
Test method:	Send *Query* requests to system under test containing correct and incorrect XPath expressions in the `query` parameter. Check responses to contain an exception exactly for the incorrect parameters. Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 4
Test method:	Send valid *Query* requests to system under test. Check that request was successful and returned the appropriate result.

	Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 5
Test method:	Send valid *Query* requests to system under test containing a format parameter with a valid MIME type identifying a format that allows representing the result. Check that the response is encoded in the format requested. Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 6
Test method:	Send a valid *Query* request using the GET/KVP protocol to system under test following this encoding specification. Check that request was successful. Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 7
Test method:	Send a valid *Query* request using the POST/XML protocol to system under test following this encoding specification. Check that request was successful. Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 8
Test method:	Send a valid *Query* request using the SOAP protocol to system under test following this encoding specification. Check that request was successful. Test passes if all conditions are fulfilled.

Test Purpose:	Requirement 9
Test method:	For each exception situation defined, send an invalid *Query* request resembling such a situations. Check that the appropriate exception is returned. Test passes if all conditions are fulfilled.

-- end of ATS –

Annex B
(non-normative)

Examples

This Annex contains examples of *Query* requests and responses assuming the conceptual model of a Web Coverage Service (WCS) [2], see Figure 2. Note that the actual responses depend on the server's concrete, individual offerings and will normally be at least in part be different from the results displayed.

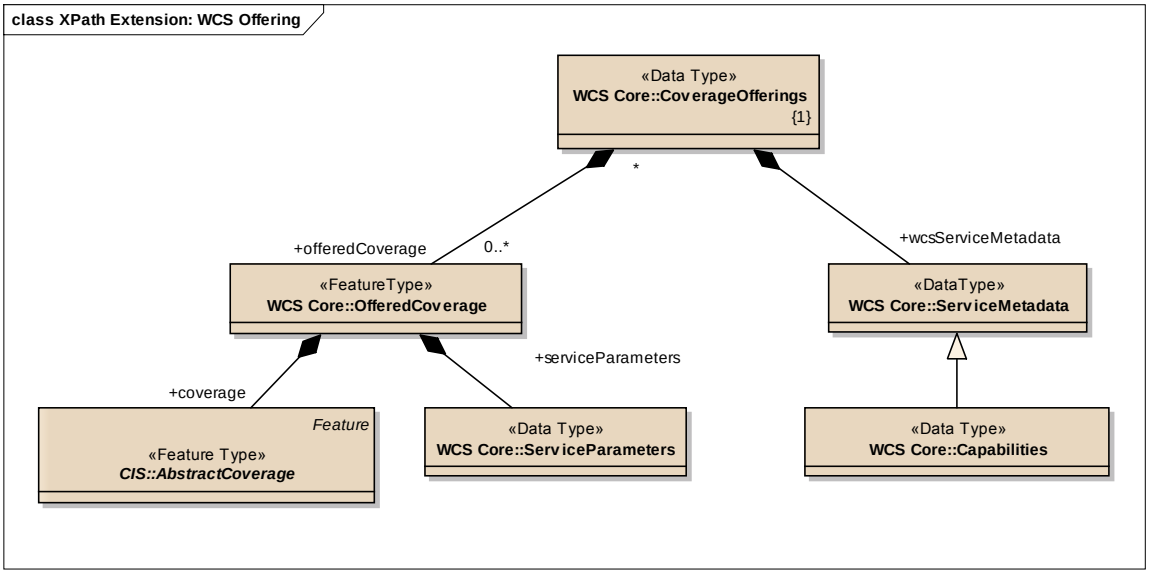

Figure 2 — WCS service offering UML diagram, based on OGC WCS [2]

- "The complete Capabilities document"

XPath request:

```
/CoverageOfferings/Capabilities
```

Response: a standard Capabilities document.

- "All WCS Extensions supported by this server"

XPath request:

```
/CoverageOffering/Capabilities/ServiceIdentification/Profile/text()
```

Shorthand version: `//Profile/text()`

Sample response:

```
http://www.opengis.net/spec/GMLCOV/1.0/conf/gml
http://www.opengis.net/spec/GMLCOV/1.0/conf/gml-coverage
http://www.opengis.net/spec/GMLJP2/2.0
http://www.opengis.net/spec/WCS_coverage-encoding_geotiff/1.0
```



```
http://www.opengis.net/spec/WCS_coverage-encoding_jpeg2000/1.0/
http://www.opengis.net/spec/WCS_coverage-encoding_netcdf/1.0
http://www.opengis.net/spec/WCS_protocol-binding_get-kvp/1.0/conf/get-kvp
http://www.opengis.net/spec/WCS_protocol-binding_post-xml/1.0
http://www.opengis.net/spec/WCS_protocol-binding_soap/1.0
http://www.opengis.net/spec/WCS_protocol-binding_get-rest/1.0/conf/get-rest
http://www.opengis.net/spec/WCS_service-
extension_processing/2.0/conf/processing
http://www.opengis.net/spec/WCS_service-extension_range-
subsetting/1.0/conf/record-subsetting
http://www.opengis.net/spec/WCS_service-
extension_transaction/2.0/conf/insert+delete
http://www.opengis.net/spec/WCS_service-extension_scaling/1.0/conf/scaling
http://www.opengis.net/spec/WCS_service-
extension_interpolation/1.0/conf/interpolation
http://www.opengis.net/spec/WCPS/1.0/conf/wcps-core
```

- "All data formats supported by this server"

XPath request:

```
/CoverageOfferings/Capabilities/ServiceIdentification/ServiceMeta
data/formatSupported/text()
```

Shorthand version: `//formatSupported/text()`

Sample response:

```
application/netcdf
image/jp2
image/tiff
image/png
application/gml+xml
```

- "Identifiers of all coverages offered"

XPath request:

```
/CoverageOffering/Capabilities/Contents/CoverageSummary/CoverageId/text()
```

Shorthand version: `//CoverageId/text()`

Sample response:

```
NASA_NIGHT_EARTH
NASA_NIGHT_EARTH_SCALED_SHALLOW_TOPO
```

- "spatial extent of coverage X"

XPath request:

```
//coverage[@id="X"]/boundedBy
```

Sample response:


```
<boundedBy>
    <Envelope
        srsName="http://www.opengis.net/def/crs/EPSG/0/4326"
        axisLabels="Lat Long"
        uomLabels="deg deg"
        srsDimension="2">
        <lowerCorner>-79 -0.01</lowerCorner>
        <upperCorner>0 59</upperCorner>
    </Envelope>
</boundedBy>
```

- "spatio-temporal locations of all 3-D coverages on this server".

XPath request:

```
//coverage/[@srsDimension=3]/boundedBy
```

Sample response:

(sequence of GML boundedBy elements)

- "Native CRS of coverage X"

XPath request:

```
//coverage[@id="X"]/boundedBy/Envelope/@srsName
```

Sample response:

```
http://www.opengis.net/def/crs/EPSG/0/4326
```

- "Pixel values of coverage X"

This is likely **not** supported to avoid returning excessively large documents.

For retrieval of coverages in GML use a GetCoverage request with
`FORMAT=application/gml+xml`
(if supported by the server, which can be checked in the in the Profile section of the Capabilities document).

