Open Geospatial Consortium Inc.

Date: 2005-10-14

Reference number of this OGC® Project Document: **OGC 05-076**

Version: 1.0.0 (Corrigendum)

Category: OGC® Implementation Specification

Editor: John D. Evans

Web Coverage Service (WCS), Version 1.0.0 (Corrigendum)

Document type: OpenGIS® Implementation Specification
Document subtype: Corrigendum
Document stage: Approved
Document language: English

Contents

Page

1 Scope 1
2 Conformance 1
3 Normative references 1
4 Terms and definitions 2
5 Conventions 4
5.1 Symbols (and abbreviated terms) 4
5.2 UML notation 4
5.3 XML schema notation 6
6 Basic service elements 6
6.1 Introduction 6
6.2 Version numbering and negotiation 7
6.2.1 Version number form 7
6.2.2 Version changes 7
6.2.3 Appearance in requests and in service metadata 7
6.2.4 Version number negotiation 7
6.3 General HTTP request rules 8
6.3.1 Overview 8
6.3.2 Key-value pair encoding (GET or POST) 9
6.3.2.1 Overview 9
6.3.2.2 Parameter ordering and case 9
6.3.2.3 Parameter lists 9
6.3.3 XML encoding 10
6.4 General HTTP response rules 10
6.5 Service exceptions 10
7 GetCapabilities operation 11
7.1 Introduction 11
7.2 GetCapabilities request 11
7.2.1 Key-value pair encoding 11
7.2.2 XML encoding 12
7.3 GetCapabilities response: Capabilities XML document 13
7.3.1 Overview 13
7.3.2 Service 13
7.3.3 Capability 15
7.3.4 ContentMetadata and CoverageOfferingBrief 15
7.3.4.1 Overview 15
7.3.4.2 metadataLink 17
7.3.4.3 description 17
7.3.4.4 name 17
7.3.4.5 label 17
7.3.4.6 lonLatEnvelope 17

7.3.4.7 keywords 17
7.3.4.8 Additional coverage properties: DescribeCoverage 17
7.3.4.9 XLink pointer to external catalog 18
7.3.5 Exceptions 18

8 DescribeCoverage operation 18
8.1 Introduction 18
8.2 DescribeCoverage requests 18
8.2.1 Overview 18
8.2.2 Key-value pair encoding 18
8.2.3 XML encoding 19
8.3 DescribeCoverage response: CoverageDescription and CoverageOffering 20
8.3.1 Overview 20
8.3.2 domainSet 21
8.3.2.1 Overview 21
8.3.2.2 SpatialDomain 21
8.3.2.3 TemporalDomain 22
8.3.3 rangeSet 23
8.3.3.1 Overview 23
8.3.3.2 AxisDescription (for compound range sets) 24
8.3.3.2.1 Introduction 24
8.3.3.2.2 XML syntax 25
8.3.3.2.3 Compound range sets 26
8.3.3.3 NullValues 27
8.3.4 SupportedCRSs and coordinate reference systems (CRS) 28
8.3.4.1 Overview 28
8.3.4.2 requestResponseCRSs 29
8.3.4.3 requestCRSs 29
8.3.4.4 responseCRSs 29
8.3.4.4.1 nativeCRSs 29
8.3.5 SupportedFormats 29
8.3.6 SupportedInterpolations 30

9 GetCoverage operation 30
9.1 Introduction 30
9.2 GetCoverage requests 31
9.2.1 Overview 31
9.2.2 Key-value pair encoding 31
9.2.2.1 Overview 31
9.2.2.2 SERVICE=WCS / VERSION=version 33
9.2.2.3 REQUEST=GetCoverage 33
9.2.2.4 COVERAGE=*name* 33
9.2.2.5 CRS 33
9.2.2.6 RESPONSE_CRS 33
9.2.2.7 BBOX 34
9.2.2.8 TIME 34
9.2.2.9 PARAMETER 34
9.2.2.10 Grid size: WIDTH, HEIGHT, DEPTH 35
9.2.2.11 Grid resolution: RESX, RESY, RESZ 35

9.2.2.12 INTERPOLATION 35
9.2.2.13 FORMAT 36
9.2.2.14 EXCEPTIONS....36
9.2.3 XML encoding....36
9.2.3.1 Overview....36
9.2.3.2 sourceCoverage....36
9.2.3.3 DomainSubset....37
9.2.3.4 RangeSubset....38
9.2.3.5 InterpolationMethod 38
9.2.3.6 Output CRS and Format....38
9.3 GetCoverage response 38
9.3.1 Overview....38
9.3.2 Coverage encoding 39
9.3.3 Multi-file payloads....39
9.4 Exceptions....39
A.1 GetCapabilities request Schema 40
A.2 GetCapabilities response schema....40
A.3 DescribeCoverage request schema 40
A.4 DescribeCoverage response schema....40
A.5 GetCoverage request schema....40
A.6 Service exception schema....40
B.1 Introduction....42
B.2 Example GetCapabilities XML request....42
B.3 Example GetCapabilities XML response 42
B.4 Example DescribeCoverage XML request 42
B.5 Example DescribeCoverage XML response....42
B.6 Example GetCoverage XML request....42
B.7 Example Service Exception XML 42
C.1 Introduction....43
D.1 Introduction....44
D.2 UML packages....44
D.3 WCS package....47
D.4 Get Coverage package 48
D.5 Describe Coverage package....49
D.6 Range Set package 50
D.7 WCS Values package....51
D.8 WCS Get Capabilities package....52
D.9 Service package 53
D.10 WCS Capability package 55
D.11 Content Metadata package....56
D.12 OGC Web Service package 57

i. Preface

The original specification of the OGC Web Coverage Service (WCS) interface resulted from extensive design and testing in the OGC Interoperability Program. In response to

the WCS Discussion Paper (June 2002) and Request For Comments (December 2002), others in the geospatial community made helpful suggestions.

This Corrigendum provides several schema fixes and document clarifications, which do not significantly affect the design or implementation of compliant technology. ***Please see the revision history for more details on the corrigendum changes.***

ii. Submitting organizations

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

- BAE SYSTEMS Mission Solutions
- Commonwealth Scientific and Industrial Research Organisation (Australia)
- CubeWerx Inc.
- Deutschen Zentrum für Luft- und Raumfahrt (German Aerospace Center)
- Galdos Systems Inc.
- George Mason University
- IONIC SOFTWARE s.a.
- National Aeronautics and Space Administration (U.S.)
- National Imagery and Mapping Agency (U.S.)
- Oracle Corp.
- PCI Geomatics
- RasDaMan GmbH
- Raytheon Company

iii. Document Contributors

OGC's Web Coverage Service Revision Working Group made extensive revisions to the WCS draft between March and August 2003. Revision Working Group members are listed below.

iv. Revision history

Date	Release	Author	Paragraph modified	Description
2001-11-17	0.5	John Evans		Initial DIPR version
2001-11-29	0.5	Jeff Lansing		Added initial DescribeCoverage content.
2001-11-29	0.5	Stephane Fellah		Revised Format and Interpolation definitions; many substantive comments
2002-01-31	0.5.1	John Evans	(most)	Revisions & corrections; XML Schema
2002-02-20	0.6	Stephane Fellah	(most)	New schemas
2002-04-04	0.7	John Evans	(most)	Fixed and streamlined the 0.6 schema; added compound observations; documentation and integration.
2003-05-18	0.8	WCS Revision Working Group	(most)	Major revision; Responded to RFC comments and RWG change proposals
2003-08-01	0.8.5	John Evans, Arliss Whiteside, WCS RWG	(most)	Finalized content and format for submission to TC.

Date	Release	Author	Paragraph modified	Description
2003-08-26	0.8.6	John Evans (with input from Arliss Whiteside, Panagiotis Vretanos, Luc Donéa, Charles Roswell, and John Herring) *(These changes are tracked in the MS-Word version of this document)*	vi, 7.2.1, 7.2.2 3, 5.1 4.1 5.2 5.3 6.2.4 7.2.2 (Fig. 2) 7.3.1 7.3.4.1 8.3.1 8.3.2.1 8.3.5 9.3.2 A.6 B.7	Set version to "1.0.0" throughout Added GML 3 to references & abbrevs. Defined Bounding Box as spatial only Added notes on UML diagrams Mentioned Altova, Inc., and XML Spy ® version is optional only in GetCapabilities GetCapabilities/section not repeatable WCS_Capabilities/@version required Added role of AssociationAttributeGroup Added version and updateSequence Clarified discrete/continuous coverages Added URLs for coverage formats Referred back to 8.3.5 New Service Exception schema Removed Service Exception example
2003-08-26	1.0.0	Arliss Whiteside	D	Enhanced and corrected UML model
2005-09-01	Corri-gendum	John Evans (for the WCS 1.1 RWG)	Annex A 7.3.1 9.2.2.{1,12} 8.3.6, 9.2.3.5 9.3.3	Modified several XML schema definitions Clarified result of using Section parameter Reinstated KVP Interpolation parameter Explained need for interpolation Guidance for multi-file payloads
2006-01-20	Minor Edits	Carl Reed	Various	Copyright, general checkover

v. Changes to the OGC Abstract Specification

The OGC® Abstract Specification does not require changes to accommodate the technical contents of this document.

vi. Future work

Future versions of this WCS specification are expected to consider various expansions of the abilities specified herein, some adding abilities that were deliberately not included in this Version 1.0.0. Some of the possible expansions thus include:

a) Expand supported coverage types beyond grid coverages only.

b) Expand ability to retrieve desired parts of the full Capabilities document, by the Get-Capabilities operation.

c) Directly transfer in WCS interface more information describing the coverage range (or observable or value space), beyond pointing at an external description.

d) Expand ability to retrieve elevation subset of a coverage, beyond current regularly spaced (grid) elevations only.

e) Expand ability to retrieve spatial subset of a coverage, beyond current regularly spaced (grid) positions only.

f) Expand supported coverage range sets beyond current single homogeneous "Range Component".

Foreword

Some of the elements of this specification may be the subject of patent rights. The Open Geospatial Consortium Inc. shall not be held responsible for identifying any such patent rights.

This edition of the Web Coverage Service (WCS) Specification cancels and replaces previous drafts (OGC 01-018; 02-024, 03-065r6). Technical changes from the 02-024 versions included a substantially revised Capabilities schema; new schemas and syntax for operation requests (GetCoverage, DescribeCoverage); and integration with GML 3.0.

Introduction

The Web Coverage Service (WCS) supports electronic interchange of geospatial data as "coverages" – that is, digital geospatial information representing space-varying phenomena. Examples of coverages are satellite imagery, digital elevation models, and triangulated integrated networks (TINs).

A WCS provides access to potentially detailed and rich sets of geospatial information, in forms that are useful for client-side rendering, multi-valued coverages, and input into scientific models and other clients. The WCS may be compared to the OGC Web Map Service (WMS) and the Web Feature Service (WFS). Like thse standards, a WCS allows clients to choose portions of a server's information holdings based on spatial constraints and other criteria.

Unlike WMS (OGC document 01-068r3), which filters and portrays spatial data to return static maps (rendered as pictures by the server), the Web Coverage Service provides available data together with their detailed descriptions; allows complex queries against these data; and returns data with its original semantics (instead of pictures) which can be interpreted, extrapolated, etc. -- and not just portrayed.

Unlike WFS (OGC Document 02-058), which returns discrete geospatial features, the Web Coverage Service returns representations of space-varying phenomena that relate a spatio-temporal domain to a (possibly multidimensional) range of properties.

The Web Coverage Service provides three operations: *GetCapabilities, GetCoverage, and DescribeCoverage*. The *GetCapabilities* operation returns an XML document describing the service and brief descriptions of the data collections from which clients may request coverages. Clients would generally run the *GetCapabilities* operation and cache its result for use throughout a session, or reuse it for multiple sessions. If GetCapabilities cannot return descriptions of its available data, that information must be available from a separate source, such as an image catalog.

The *DescribeCoverage* operation lets clients request a full description of one or more coverages served by a particular WCS server. The server responds with an XML document that fully describes the identified coverages.

The *GetCoverage* operation of a Web Coverage Service is normally run after *GetCapabilities* and *DescribeCoverage* replies have shown what requests are allowed and what data are available. The *GetCoverage* operation returns a coverage (that is, values or properties of a set of geographic locations), bundled in a well-known coverage format. Its syntax and semantics bear some resemblance to the WMS *GetMap* and WFS *GetFeature* requests, but several extensions support the retrieval of coverages rather than static maps or discrete features.

OGC Web Coverage Service (WCS) Specification

Version 1.0.0 (Corrigendum) – OGC Document No. 05-076

1 Scope

This document specifies how a Web Coverage Service (WCS) serves to describe, request, and deliver multi-dimensional coverage data over the World Wide Web. This version of the Web Coverage Service is limited to describing and requesting grid (or "simple") coverages with homogeneous range sets.

Grid coverages have a domain comprised of regularly spaced locations along the 1, 2, or 3 axes of a spatial coordinate reference system. Their domain may also have a time component, which may be regularly or irregularly spaced. A coverage with a homogeneous range set defines, at each location in the domain, either a single (scalar) value (such as elevation), or a series (array / tensor) of values all defined in the same way (such as brightness values in different parts of the electromagnetic spectrum).

The WCS design, while limited in this version to simple, homogeneous coverages, is designed to extend in future versions to other coverage types defined in the OGC Abstract Specification (Topic 6, "The Coverage Type," OGC document 00-106).

2 Conformance

Conformance with this OGC Implementation Specification may be checked using all the relevant tests specified in Annex C (normative).

3 Normative references

The following normative documents contain provisions that, through reference in this text, constitute provisions of this specification. 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.

IETF RFC 2045 (November 1996), Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, Freed, N. and Borenstein N., eds., <http://www.ietf.org/rfc/rfc2045.txt>

IETF RFC 2616 (June 1999), Hypertext Transfer Protocol – HTTP/1.1, Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T., eds., <http://www.ietf.org/rfc/rfc2616.txt>

IETF RFC 2396 (August 1998), Uniform Resource Identifiers (URI): Generic Syntax, Berners-Lee, T., Fielding, N., and Masinter, L., eds., <http://www.ietf.org/rfc/rfc2396.txt>

ISO 19105: Geographic information — Conformance and Testing

ISO 19123, Geographic Information — Coverage Geometry and Functions

OGC 01-068r3, Web Map Service v. 1.1.1, <http://www.opengis.org/techno/specs/01-068r3.pdf>

OGC 02-023r4, OpenGIS Geography Markup Language (GML) Implementation Specification, v3.00 <http://www.opengis.org/techno/documents/02-023r4.pdf>

OGC AS 0, The OpenGIS Abstract Specification Topic 0: Overview, OGC document 99-100r1 <http://www.opengis.org/techno/abstract/99-100r1.pdf>

OGC AS 12, The OpenGIS Abstract Specification Topic 12: OpenGIS Service Architecture (Version 4.2), Kottman, C. (ed.), <http://www.opengis.org/techno/specs.htm>

XML 1.0, W3C Recommendation 6 October 2000, Extensible Markup Language (XML) 1.0 (2nd edition), World Wide Web Consortium Recommendation, Bray, T., Paoli, J., Sperberg-McQueen, C.M., and Maler, E., eds., <http://www.w3.org/TR/2000/REC-xml>

W3C Recommendation 2 May 2001: XML Schema Part 0: Primer, <http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/>

W3C Recommendation 2 May 2001: XML Schema Part 1: Structures, <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>

W3C Recommendation 2 May 2001: XML Schema Part 2: Datatypes, <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>

4 Terms and definitions

For the purposes of this document, the terms and definitions given in the above references apply, as do the following terms.

4.1
bounding box
a set of 2, 4, or 6 numbers indicating the upper and lower bounds of an interval (1D), rectangle (2D), or parallelepiped (3D) along each axis of a given spatial CRS

4.2
capabilities XML
service-level metadata, expressed in XML, describing the operations and content available at a **service instance**

4.3
client
software component that can invoke an **operation** from a **server**

4.4
georectified grid
a grid having regular spacing in a projected or geographic coordinate reference system (CRS)

NOTE A grid for which there is a linear relationship between the grid coordinates and those of a projected or geographic coordinate reference system.

4.5
georeferenced grid
a grid that is not georectified, but that is associated with (one or more) coordinate transformations which relate the image or engineering CRS to a projected or geographic CRS

NOTE These coordinate transformations are usually not affine or simple, and are usually empirically determined. (Synonym: *georeferenceable*).

4.6
interface
named set of **operations** that characterize the behavior of an entity [OGC AS 12]

4.7
operation
specification of a transformation or query that an object may be called to execute [OGC AS 12]

4.8
service
distinct part of the functionality that is provided by an entity through **interfaces** [OGC AS 12]

4.9
request
invocation of an **operation** by a **client**

4.10
response
result of an **operation,** returned from a **server** to a **client**

4.11
service instance
server
actual implementation of a **service –** a software component on which a **client** can invoke an **operation**

5 Conventions

5.1 Symbols (and abbreviated terms)

The following symbols and abbreviated terms are used in this document.

API	Application Program Interface
CRS	Coordinate Reference System
DCP	Distributed Computing Platform
GML	OGC Geography Markup Language, v3.00 (OGC 03-023r4)
ISO	International Organization for Standardization
OGC	Open Geospatial Consortium
OWS	OGC Web Service
UML	Unified Modeling Language
XML	Extensible Markup Language
1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
4D	Four Dimensional

5.2 UML notation

Certain diagrams that appear in this document are presented using static structure diagrams in the Unified Modeling Language (UML) [OMG]. The UML notations used in this document are described in the diagram below.

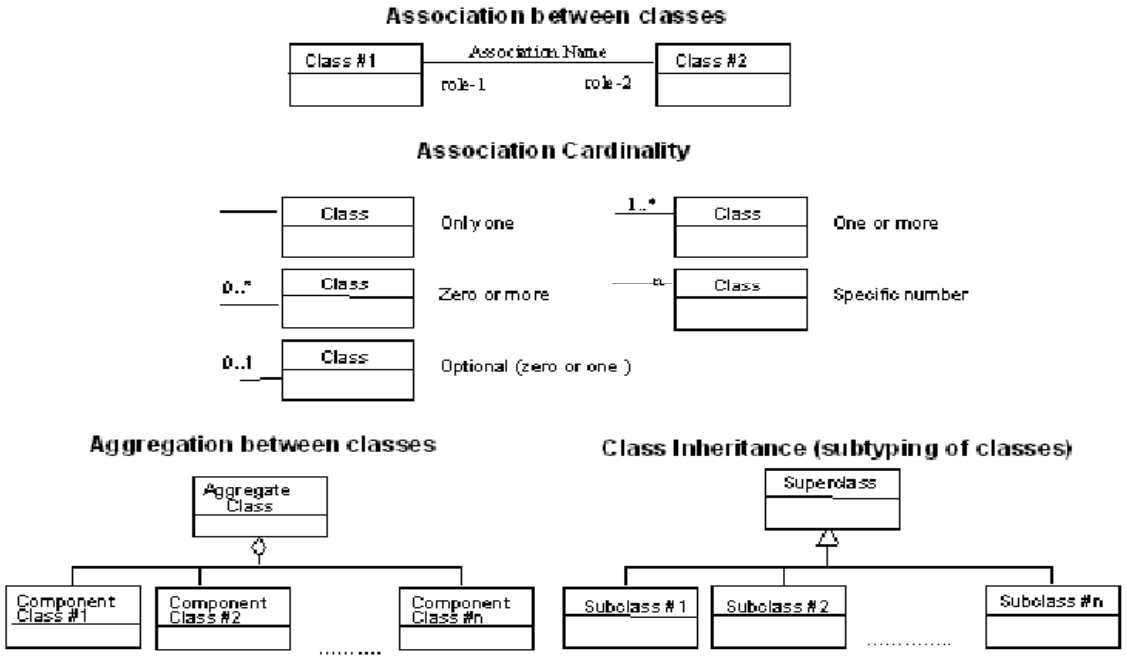

Figure 1 - UML Notation

In these UML class diagrams, the class boxes with three compartments and a light background are the primary classes being shown in this diagram, usually the classes from one UML package. The class boxes with a grey background are other classes used by these primary classes, usually classes from other packages. The class boxes without compartments do not show the class attributes, which are usually shown on another class diagram.

In these class diagrams, the following five stereotypes of UML classes are used:

a) <<Interface>> A definition of a set of operations that is supported by objects having this interface. An Interface class cannot contain any attributes.

b) <<DataType>> A descriptor of a set of values that lack identity (independent existence and the possibility of side effects). A DataType is a class with no operations whose primary purpose is to hold the information.

c) <<Enumeration>> A data type whose instances form a list of alternative literal values. Enumeration means a short list of well-understood potential values within a class.

d) <<CodeList>> is a flexible enumeration that uses string values for expressing a long list of potential alternative values. If the list alternatives are completely known, an enumeration shall be used; if the only likely alternatives are known, a code list shall be used. Code lists are more likely to have their values exposed to the user.

e) <<Type>> A stereotyped class used for specification of a domain of instances (objects), together with the operations applicable to the objects. A Type class may have attributes and associations.

NOTE All the stereotypes listed above are adapted from Subclause 6.8 of ISO 19103.

In this document, the following standard data types are used:

a) CharacterString – A sequence of characters

b) Boolean – A value specifying TRUE or FALSE

c) URI – An identifier of a resource that provides more information about data

d) URL – An identifier of an on-line resource that can be electronically accessed

5.3 XML schema notation

Several diagrams in this document represent XML Schema constructs using the graphical symbols provided by the XML Spy software suite (Altova, Inc. / <http://www.xmlspy.com/>). These are depicted in Figure 2 below.

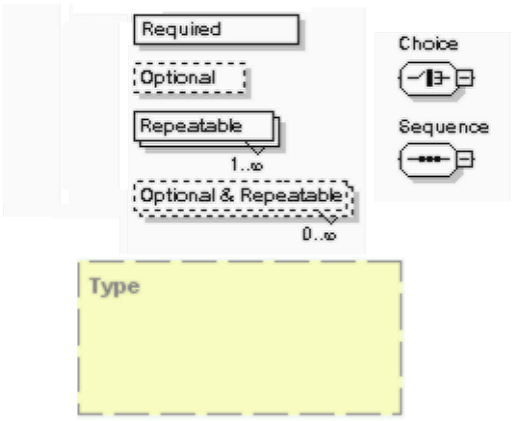

Figure 2. XML Schema graphic symbols

6 Basic service elements

6.1 Introduction

This clause describes aspects of Web Coverage Server behavior (more generally, of OGC Web Service behavior) that are independent of particular operations, or that are common to several operations or interfaces.

6.2 Version numbering and negotiation

6.2.1 Version number form

The published specification version number contains three positive integers, separated by decimal points, in the form "x.y.z". The numbers "y" and "z" will never exceed 99. Each OWS specification is numbered independently.

6.2.2 Version changes

A particular specification's version number **shall** be changed with each revision. The number **shall** increase monotonically and **shall** comprise no more than three integers separated by decimal points, with the first integer being the most significant. There may be gaps in the numerical sequence. Some numbers may denote experimental or interim versions. Service instances and their clients need not support all defined versions, but **must** obey the negotiation rules below.

6.2.3 Appearance in requests and in service metadata

The version number appears in at least two places: in the Capabilities XML describing a service, and in the parameter list of client requests to that service. The version number used in a client's request of a particular service instance **must** be equal to a version number which that instance has declared it supports (except during negotiation as described below). A service instance may support several versions, whose values clients may discover according to the negotiation rules.

6.2.4 Version number negotiation

A Client may negotiate with a Service Instance to determine a mutually agreeable specification version. Negotiation is performed using the **GetCapabilities** operation [see Clause 7] according to the following rules.

All Capabilities XML must include a protocol version number. In response to a **GetCapabilities** request containing a version number, an OGC Web Service **must** either respond with output that conforms to that version of the specification, **or** negotiate a mutually agreeable version if the requested version is not implemented on the server. If no version number is specified in the request, the server **must** respond with the highest version it understands and label the response accordingly.

Version number negotiation occurs as follows:

a) If the server implements the requested version number, the server **must** send that version.

b) If a version unknown to the server is requested, the server **must** send the highest version it knows that is less than the requested version.

c) If the client request is for a version lower than any of those known to the server, then the server **must** send the lowest version it knows.

d) If the client does not understand the new version number sent by the server, it **may** either cease communicating with the server **or** send a new request with a new version number that the client does understand but which is less than that sent by the server (if the server had responded with a lower version).

e) If the server had responded with a higher version (because the request was for a version lower than any known to the server), and the client does not understand the proposed higher version, then the client **may** send a new request with a version number higher than that sent by the server.

The process is repeated until a mutually understood version is reached, or until the client determines that it will not or cannot communicate with that particular server.

EXAMPLE 1 Server understands versions 1, 2, 4, 5 and 8. Client understands versions 1, 3, 4, 6, and 7. Client requests version 7. Server responds with version 5. Client requests version 4. Server responds with version 4, which the client understands, and the negotiation ends successfully.

EXAMPLE 2 Server understands versions 4, 5 and 8. Client understands version 3. Client requests version 3. Server responds with version 4. Client does not understand that version or any higher version, so negotiation fails and client ceases communication with that server.

The version parameter is mandatory in requests other than **GetCapabilities**.

6.3 General HTTP request rules

6.3.1 Overview

At present, the only distributed computing platform (DCP) explicitly supported by OGC Web Services is the World Wide Web itself, or more specifically Internet hosts implementing the Hypertext Transfer Protocol (HTTP). Thus the Online Resource of each operation supported by a service instance is an HTTP Uniform Resource Locator (URL). The URL may be different for each operation, or the same, at the discretion of the service provider. Each URL **must** conform to the description in [HTTP] but is otherwise implementation-dependent; only the parameters comprising the service request itself are mandated by the OGC Web Services specifications.

HTTP supports two request methods: GET and POST. One or both of these methods may be defined for a particular OGC Web Service type and offered by a service instance, and the use of the Online Resource URL differs in each case.

An Online Resource URL intended for HTTP GET requests is in fact only a URL prefix to which additional parameters must be appended in order to construct a valid Operation request. A URL prefix is defined as an opaque string including the protocol, hostname, optional port number, path, a question mark '?', and, **optionally**, one or more server-specific parameters ending in an ampersand '&'. The prefix uniquely identifies the particular service instance. For HTTP GET, the URL prefix **must** end in either a '?' (in the absence of additional server-specific parameters) or a '&'. In practice, however, Clients **should** be prepared to add a necessary trailing '?' or '&' before appending the Operation parameters defined in this specification in order to construct a valid request URL.

An Online Resource URL intended for HTTP POST requests is a complete and valid URL to which Clients transmit encoded requests in the body of the POST document. A WCS server **must not** require additional parameters to be appended to the URL in order to construct a valid target for the Operation request.

6.3.2 Key-value pair encoding (GET or POST)

6.3.2.1 Overview

Using Key-Value Pair encoding, a client composes the necessary request parameters as keyword/value pairs in the form "keyword=value", separated by ampersands (‘&’), with appropriate encoding [IETF RFC 2396] to protect special characters. The resulting query string may be transmitted to the server via HTTP GET or HTTP POST, as prescribed in the HTTP Common Gateway Interface (CGI) standard [IETF RFC 2616].

Table 1 summarizes the request parameters for HTTP GET and POST.

Table 1 –Parts of a Key-Value Pair OGC Web Service Request

URL Component	Description
http://host[:port]/path	URL of service operation. The URL is entirely at the discretion of the service provider.
{name[=value]&}	The query string, consisting of one or more standard request parameter name/value pairs defined by an OGC Web Service. The actual list of required and optional parameters is mandated for each operation by the appropriate OWS specification.

Notes: [ ] denotes 0 or 1 occurrence of an optional part; {} denotes 0 or more occurrences.

A request encoded using the HTTP GET method interposes a '?' character between the service operation URL and the query string, to form a valid URI which may be saved as a bookmark, embedded as a hyperlink, or referenced via Xlink in an XML document.

6.3.2.2 Parameter ordering and case

Parameter names **shall not** be case sensitive, but parameter values shall be case sensitive. In this document, parameter names are typically shown in uppercase for typographical clarity, not as a requirement.

Parameters in a request **may** be specified in any order.

An OGC Web Service **must** be prepared to encounter parameters that are not part of this specification. In terms of producing results per this specification, an OGC Web Service **shall** ignore such parameters.

6.3.2.3 Parameter lists

Parameters consisting of lists shall use the comma (",") as the delimiter between items in the list: e.g., `parameter=item1,item2,item3`. Multiple lists can be specified as the

value of a parameter by enclosing each list in parentheses ("(", ")"): e.g., `parameter=(item1a,item1b,item1c),(item2a,item2b,item2c)`. If a parameter name or value includes a space or comma, it shall be escaped using the URL encoding rules [IETF RFC 2396].

6.3.3 XML encoding

Clients may also encode requests in XML for transmission to the server using HTTP GET or (more often) HTTP POST. The XML request must conform to the schema corresponding to the chosen operation, and the client must send it to the URL listed for that operation in the server's Capabilities XML file, in accordance with HTTP POST [IETF RFC 2616]).

NOTE To support SOAP messaging, clients need only enclose this XML document in a SOAP envelope as follows:

```
<env:Envelope
 xmlns:env="http://www.w3.org/2001/09/soap-envelope">
  <env:Body>
    request document here
  </env:Body>
</env:Envelope>
```

6.4 General HTTP response rules

Upon receiving a valid request, the service **must** send a response corresponding exactly to the request as detailed in the appropriate specification. Only in the case of Version Negotiation (described above) may the server offer a differing result.

Upon receiving an invalid request, the service **must** issue a Service Exception as described in Subclause 6.5 below.

NOTE As a practical matter, in the WWW environment a client should be prepared to receive either a valid result, or nothing, or any other result. This is because the client may itself have formed a non-conforming request that inadvertently triggered a reply by something other than an OGC Web Service, because the Service itself may be non-conforming, etc.

6.5 Service exceptions

Upon receiving an invalid request, the service **must** issue a Service Exception XML message to describe to the client application or its human user the reason(s) that the request is invalid.

Service Exception XML **must** be valid according to the Service Exception XML Schema in Subclause A.7. In an HTTP environment, the MIME type of the returned XML **must** be "`application/vnd.ogc.se_xml`". Specific error messages can be included either as chunks of plain text or as XML-like text containing angle brackets ("<" and ">") if included in a character data (CDATA) section as shown in the example of Service Exception XML in Subclause A.7.

Service Exceptions **may** include exception codes as indicated in Subclause A.7. Servers **shall not** use these codes for meanings other than those specified. Clients **may** use these codes to automate responses to Service Exceptions.

7 GetCapabilities operation

7.1 Introduction

Each Web Coverage Server must describe its capabilities. This clause defines an XML document structure intended to convey general information about the service itself, and summary information about the available data collections from which coverages may be requested.

7.2 GetCapabilities request

7.2.1 Key-value pair encoding

The general form of a **GetCapabilities** request is defined in Clause 6, and summarized in Table 2 below.

Table 2. GetCapabilities request URL parameters

Request Parameter	Required/ Optional	Description
REQUEST=GetCapabilities	Required	Request name
VERSION=1.0.0	Optional	Request version
SERVICE=WCS	Required	Service type
SECTION=/ *or* /WCS_Capabilities/Service *or* /WCS_Capabilities/Capability *or* /WCS_Capabilities/ContentMetadata	Optional	Section of Capabilities document to be returned
UPDATESEQUENCE	Optional	Capabilities version

The VERSION and SERVICE parameters, respectively, denote the version number of the specification and the service it addresses. For WCS requests, the SERVICE parameter **must** have the value "WCS".

NOTE When making requests of a WCS server, which may offer other OGC Web Services as well, the SERVICE parameter indicates that the client seeks information about the WCS server in particular.

The SECTION parameter denotes which portion of the Capabilities XML document to return: **Service**, **Capability, or ContentMetadata**. (Figure 3 below depicts the Capabilities XML document structure.) If this parameter is not supplied, the request is for the entire Capabilities XML document.

The **optional** UPDATESEQUENCE parameter is for maintaining cache consistency. Its value can be an integer, a timestamp in [ISO 8601:1988] format, or any other number or string. The server **may** include an UpdateSequence value in its Capabilities XML. If present, this value **should** be increased when changes are made to the Capabilities (for ex-

ample, when new coverages are added to the service). The server is the sole judge of lexical ordering sequence. The client **may** include this parameter in its GetCapabilities request. The response of the server based on the presence and relative value of Update-Sequence in the client request and the server metadata **shall** be according to Table 3:

Table 3 — Use of UpdateSequence Parameter

Client Request UpdateSequence Value	Server Metadata UpdateSequence Value	Server Response
None	Any	most recent Capabilities XML
Any	None	most recent Capabilities XML
Equal	Equal	Exception: code=CurrentUpdateSequence
Lower	Higher	most recent Capabilities XML
Higher	Lower	Exception: code=InvalidUpdateSequence

7.2.2 XML encoding

The **GetCapabilities** XML request schema is depicted in Figure 2 below.

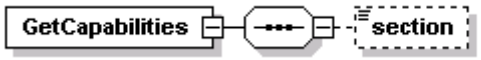

Figure 2. GetCapabilities request

The top-level XML element, **GetCapabilities**, has three attributes as defined in Table 4.

Table 4. GetCapabilities XML attributes

Attribute	Required / Optional	Description
version	Optional	Request version. Defaults to the latest available version, currently 1.0.0.
service	Required	Service type (needed when several services share a single access point)
update-Sequence	Optional	Used for cache management. A service provider must increase this value when adding new content.

The **GetCapabilities** element has one optional sub-element, **section**, defined like the SECTION parameter in Table 2 and Subclause 7.2.1; it denotes which portion(s) of the Capabilities XML document to return: **Service**, **Capability**, or **ContentMetadata**. If the **section** element is not supplied, the server must return the entire Capabilities XML document.

For example, a client may encode a **GetCapabilities** request in XML as follows:

```
<GetCapabilities version="1.0.0" service="WCS">
    <section>/WCS_Capabilities/Capability</section>
</GetCapabilities>
```

7.3 GetCapabilities response: Capabilities XML document

7.3.1 Overview

The response to a **GetCapabilities** request with no **Section** parameter is a Capabilities XML document conforming to the Schema given in Subclause A.3, composed of three main sections depicted in Figure 3:

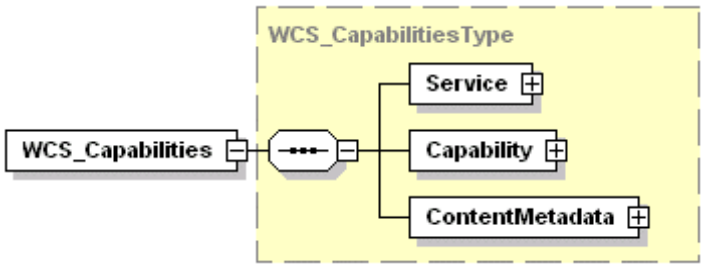

Figure 3 - WCS_Capabilities top-level element

The top-level element, **WCS_Capabilities**, has two attributes:

- **version** *(required)* indicates the WCS version to which the Capabilities XML document conforms.

- **updateSequence** *(optional)* indicates content or service updates. (A service provider must increase this value when adding new content or changing any aspects of the service.)

If the **GetCapabilities** request has a **Section** parameter, the reply is not a full **WCS_Capabilities** element, but instead one of its three child elements: **Service**, **Capability**, or **ContentMetadata**.

Subclauses 7.3.2- 7.3.4 below detail the various parts of the **WCS_Capabilities** XML schema.

7.3.2 Service

The first section, **Service**, contains service metadata elements shared by other OGC Web Services, that provide a minimal, human readable description of the service.

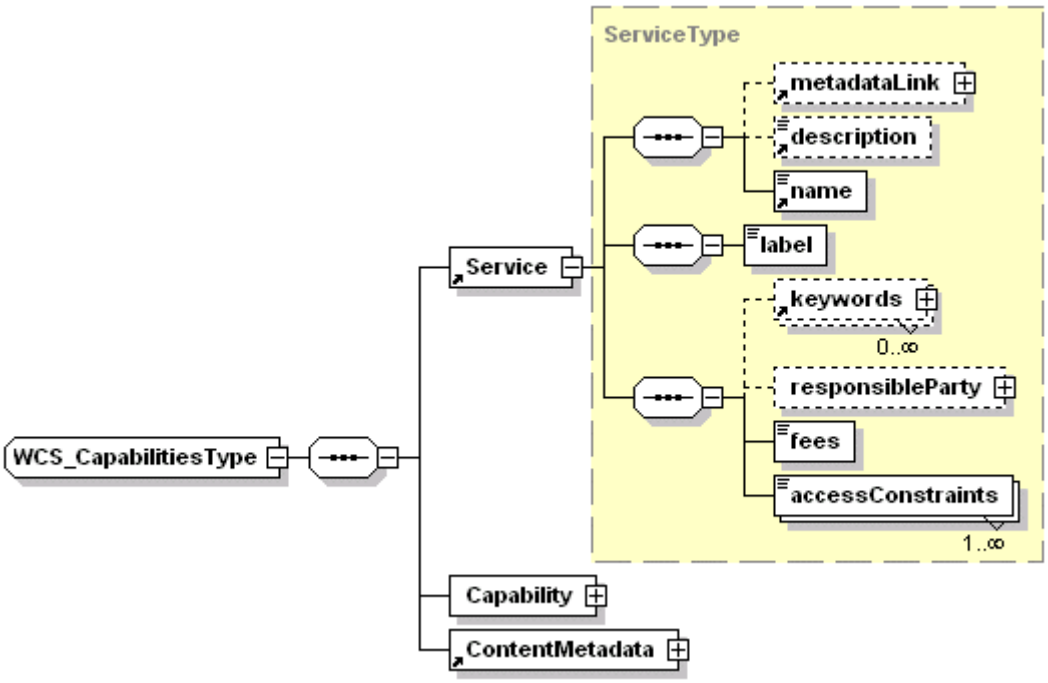

Figure 4. Service

This section is structured much like the WMS and WFS service descriptions, with the sub-elements defined in Table 5.

Table 5. Service section elements

Element Name	Required / Optional	Description
description	Optional	A description of the server.
name	Required	A name the service provider assigns to the server.
label	Required	A human-readable label for this server, for use in menus and other displays.
wcs:metadataLink	Optional	A link to external metadata (of type “FGDC”, “TC211”, or “other”)
keywords	Optional	Short words to aid catalog searching
responsibleParty	Optional	A tree of elements that identify the service provider and provide contact details.
fees	Required	A text string indicating any fees imposed by the service provider. The keyword NONE is reserved to mean no fees.
accessConstraints	Required	A list of codes describing any access constraints imposed by the service provider. The keyword NONE is reserved to mean no access constraints are imposed.

The **Service** element also has two optional attributes, **version** and **updateSequence**, both defined as for **WCS_Capabilities** (7.3.1 above). These attributes are used only when the **Service** section appears alone (in response to a GetCapabilities request qualified with a Section parameter). These attributes must be omitted in the context of the full

Capabilities XML document (where they appear on the parent element **WCS_Capabilities**).

7.3.3 Capability

The second section, **Capability**, of type **WCSCapabilityType**, also resembles that for WMS and WFS. It describes the requests that the service supports, the formats in which exceptions are returned, and any other vendor-specific service capabilities.

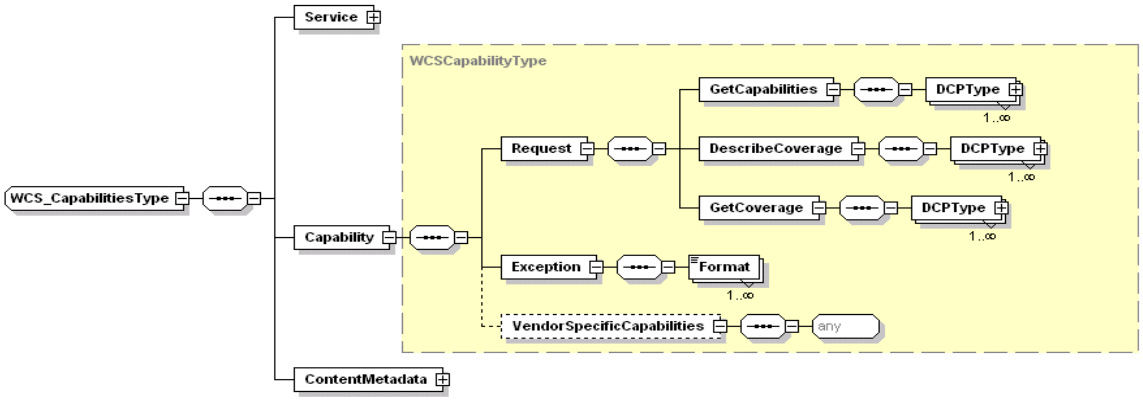

Figure 5. Capability

The **Request** sub-element has three required sub-elements, one for each WCS request (GetCapabilities, DescribeCoverage, and GetCoverage). Within each of these, the **DCPType** element lists the distributed computing platform(s) supported, and the corresponding network access points. (For the moment, HTTP is the only DCP defined; a server may list either a GET or POST access point, or both, for each operation.)

The **Capability** element also has two optional attributes, **version** and **updateSequence**, both defined as for **WCS_Capabilities** (7.3.1 above). These attributes are used only when the **Capability** section appears alone (in response to a GetCapabilities request qualified with a Section parameter). These attributes must be omitted in the context of the full Capabilities XML document (where they appear on the parent element **WCS_Capabilities**).

7.3.4 ContentMetadata and CoverageOfferingBrief

7.3.4.1 Overview

The third section of the Capabilities XML file (ContentMetadata) has Xlink attributes belonging to the GML **AssociationAttributeGroup**. These are used to refer to another source, such as an image catalog service, from which content metadata are available. (This is intended for servers with thousands or millions of coverage offerings, for which searching a catalog search is more feasible than fetching a long XML document.)

ContentMetadata also has the two optional attributes **version** and **updateSequence**, both defined as for **WCS_Capabilities** (7.3.1 above). These attributes are used only when the **ContentMetadata** section appears alone (in response to a **GetCapabilities** re-

quest qualified with a **Section** parameter). These attributes must be omitted in the context of the full Capabilities XML document (where they appear on the parent element **WCS_Capabilities**).

ContentMetadata has an optional and repeatable sub-element, **CoverageOfferingBrief**, depicted in Figure 6.

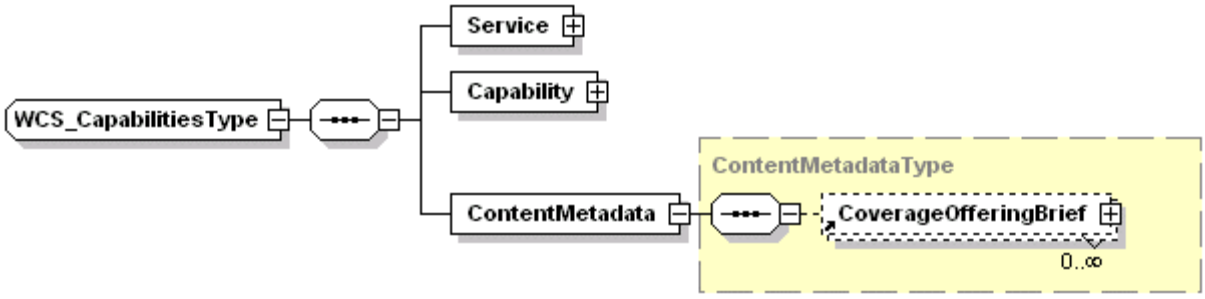

Figure 6. ContentMetadata

The **CoverageOfferingBrief** structure is depicted in Figure 7 below.

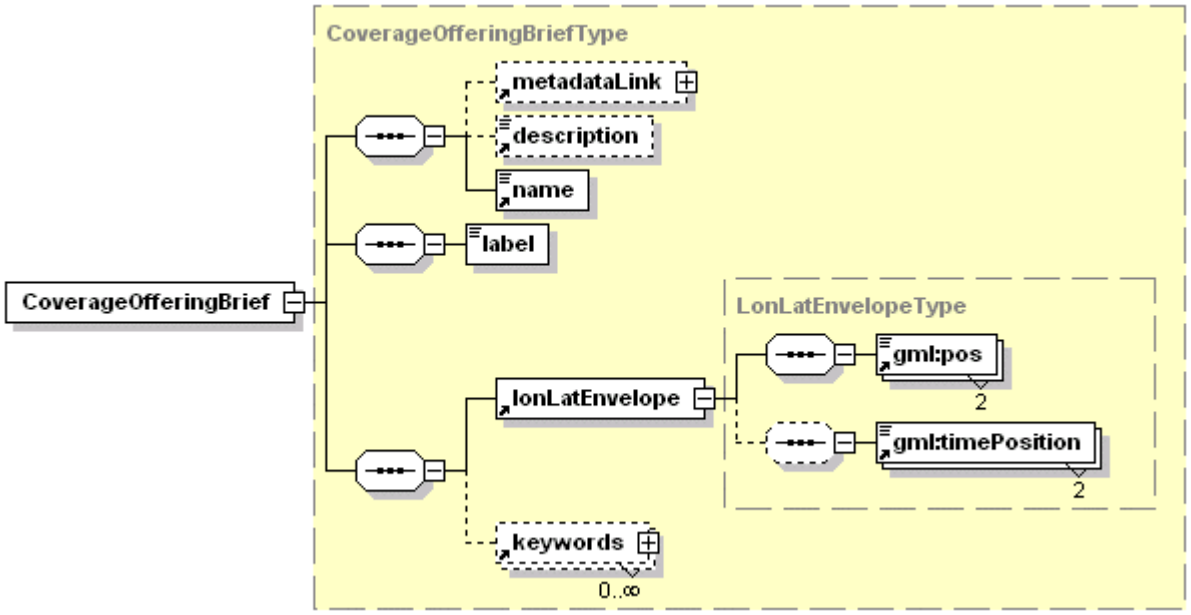

Figure 7. CoverageOfferingBrief

The following subclauses describe each sub-element of **CoverageOfferingBrief**, and two mechanisms for obtaining more detailed information about a server's coverage offerings.

NOTE The first four of these elements -- **description**, **name**, **label**, and **metadataLink** -- are used in a similar fashion by **RangeSet** and **AxisDescription** in Subclauses 8.3.3 and 8.3.3.2.2.

7.3.4.2 metadataLink

The *optional* **metadataLink** element is *recommended* for access to detailed, standardized metadata about the parent element (in this case, **CoverageOfferingBrief**). It has a series of Xlink attributes (GML's **AssociationAttributeGroup**) that allow it to point to an external source of metadata; its **type** attribute indicates the standard to which the metadata complies. Three types of metadata are defined: *'TC211'* (referring to ISO TC211's Geospatial Metadata Standard 19115); *'FGDC'* (referring to the US FGDC Content Standard for Digital Geospatial Metadata); and *'other'*.

7.3.4.3 description

The *optional* **description** element contains a narrative description of its parent element (in this case, **CoverageOfferingBrief**).

7.3.4.4 name

The *required* **name** unique identifies its parent element (in this case, **CoverageOfferingBrief**): that is, the same **name** value is not used for any siblings of that parent element on the same server.

7.3.4.5 label

The *required* **label** element contains a human-readable string describing its parent element (in this case, **CoverageOfferingBrief**), for presentation in client forms or menus.

7.3.4.6 lonLatEnvelope

This *required* element defines a bounding box that encloses all of the data available through the coverage offering. It expresses the corners of this bounding box using a pair of GML **pos** elements, in the WGS 84 geographic CRS with Longitude preceding Latitude and both using decimal degrees only. If included, height values are third and use metre units. These are followed by an optional pair of GML **timePosition** elements to express a time-span.

7.3.4.7 keywords

The *optional* **keywords** elements contain keywords that describe the coverage offering.

7.3.4.8 Additional coverage properties: DescribeCoverage

The elements defined in **CoverageOfferingBrief** provide a summary-level description of coverage data available from a given service. Clients may be able to formulate simple GetCoverage requests based only on this information. However, in order to make more finely tuned GetCoverage requests, clients will usually need to obtain further details about a particular coverage, using the **DescribeCoverage** operation (see Clause 8).

7.3.4.9 XLink pointer to external catalog

Some WCS servers may have thousands or millions of coverage offerings available, making it impractical to list them all under **ContentMetadata**. For this reason, **ContentMetadata** also has an attribute group, GML's **AssociationAttributeGroup**. This lets the **ContentMetadata** section point to an external catalog via Xlink, instead of (or in addition to) listing coverage descriptions inline. This attribute group includes the standard Xlink attributes (**type**, **href**, **role**, **arcrole**, **title**, **show**, and **actuate**), as well as GML's **remoteSchema** for stating the schema of the remote resource. All of these attributes are optional; but if the **ContentMetadata** element is empty (i.e., no **CoverageOfferingBrief** elements), then at least the Xlink **href** attribute must be present, and must list the URL of a catalog that clients can search for coverage descriptions in order to make appropriate **DescribeCoverage** or **GetCoverage** requests

7.3.5 Exceptions

In the event that the web coverage server encounters an error servicing a **GetCapabilities** request, it shall raise an exception as described in Sublause 6.5.

8 DescribeCoverage operation

8.1 Introduction

Once a client has obtained summary descriptions of the coverages available from a particular WCS server, it may be able to make simple GetCoverage requests immediately. But in most cases the client will need to issue a **DescribeCoverage** request to obtain a full description of one or more coverages available. The server responds to such a request with an XML document describing one or more coverages served by the WCS.

8.2 DescribeCoverage requests

8.2.1 Overview

A **DescribeCoverage** request lists the coverages to be described, identified by the **Coverage** parameter. A request that lists no coverages shall be interpreted as requesting descriptions of all coverages that a WCS can serve. (Server support for such a request is optional; if a server does not support it, it must return an exception rather than the requested list.)

8.2.2 Key-value pair encoding

Table 6 describes the complete **DescribeCoverage** request in its HTTP GET form.

Table 6. DescribeCoverage URL parameters

URL Component	Description
http://server_address/path/script?	URL of WCS server. *Required.*
REQUEST=DescribeCoverage	Request name. Must be "DescribeCoverage ". *Required.*
SERVICE=WCS	Service name. Must be "WCS". *Required.*
VERSION=1.0.0	Request protocol version. *Required.*
COVERAGE=name1, name2, …	A comma-separated list of coverages to describe (identified by their name values in the Capabilities response). *Optional.* Default is all coverages, if the server supports it.

The SERVICE and VERSION parameters are defined as for GetCapabilities and GetCoverage requests (see Subclause 7.2). The COVERAGE parameter specifies one or more coverages by their identifier. These identifiers must be among those listed in the **name** element of **CoverageOfferingBrief** element(s) in the Capabilities XML document.

8.2.3 XML encoding

Figure 8 depicts the **DescribeCoverage** Schema.

Figure 8. DescribeCoverage XML request

DescribeCoverage has two attributes, **service** *(required)* and **version** *(required)*, both defined as for **GetCapabilities** (Subclause 7.2).

DescribeCoverage has one optional and repeatable sub-element, **Coverage**, each of which designates a coverage offering identified by its **name** (obtained via a prior GetCapabilities request to the server, or possibly from a third-party source). If the **Coverage** element is absent, the server may return full descriptions of every coverage offering available, or return a service exception.

A simple example follows, for requesting descriptions of three different coverages.

```
<DescribeCoverage service="WCS" version="1.0.0">
    <Coverage>Landsat_TM_Mosaic</Coverage>
    <Coverage>WMO_Daily_Temps</Coverage>
    <Coverage>Census_population_tables</Coverage>
</DescribeCoverage>
```

8.3 DescribeCoverage response: CoverageDescription and CoverageOffering

8.3.1 Overview

In response to a **DescribeCoverage** request, a WCS shall return an XML document whose top-level element is a **CoverageDescription** containing **CoverageOffering** elements describing all (and only) the requested coverage offerings.

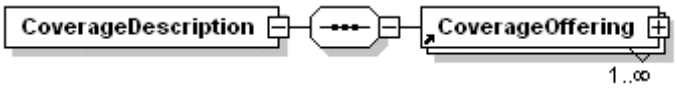

Figure 9. Coverage Description top-level structure

Each **CoverageOffering** element has the structure depicted in Figure 10:

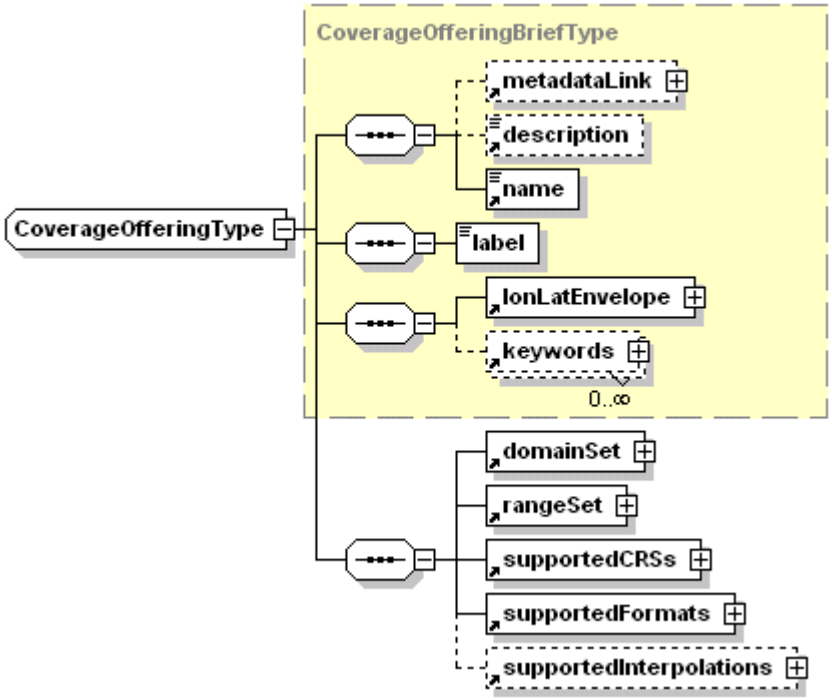

Figure 10. CoverageOffering

CoverageOffering has two attributes, **version** *(required)* and **updateSequence** *(optional)*, both defined as for **WCS_Capabilities** (7.3.1 above).

CoverageOffering extends **CoverageOfferingBrief** (Subclause 7.3.4), to provide additional details on the domain and range of a coverage offering. Clients may use these to assess the data's fitness for use, and to formulate fine-grained GetCoverage requests. Table 7 summarizes these additional elements:

Table 7. CoverageOffering: additional elements beyond CoverageOfferingBrief

Element name	Required / Optional	Description
domainSet	Required	The available coverage locations in space and/or time available from a coverage offering
rangeSet	Required	A description of coverage values available from a coverage offering
supportedCRSs	Required	The coordinate reference system(s) in which the server can accept requests against this coverage offering and produce coverages from it.
supportedFormats	Required	The formats (file encodings) in which the server can produce coverages from this coverage offering.
supportedInterpolations	Optional	The spatial interpolation methods available for resampling or generalizing coverage values when needed to fill a GetCoverage request.

Subclauses s 8.3.2-8.3.6 describe these five elements in more detail.

8.3.2 domainSet

8.3.2.1 Overview

The first of these elements, **domainSet**, describes the domain of the coverage offering – that is, the locations in space and/or time for which values or measures are available (whether by direct retrieval or by spatial interpolation). **GetCoverage** requests should retrieve meaningful data from the Coverage Offering if their spatial or temporal constraints (BBOX or **BoundingBox**, TIME, or **Time)** intersect the locations or times described in **domainSet**.

domainSet must include a **SpatialDomain** (describing the spatial locations – whether discrete or continuous – for which coverages may be requested), a **TemporalDomain** (describing the time instants or intervals for which coverages may be requested), or both.

8.3.2.2 SpatialDomain

Figure 11 depicts the structure of the **domainSet** and **spatialDomain** elements.

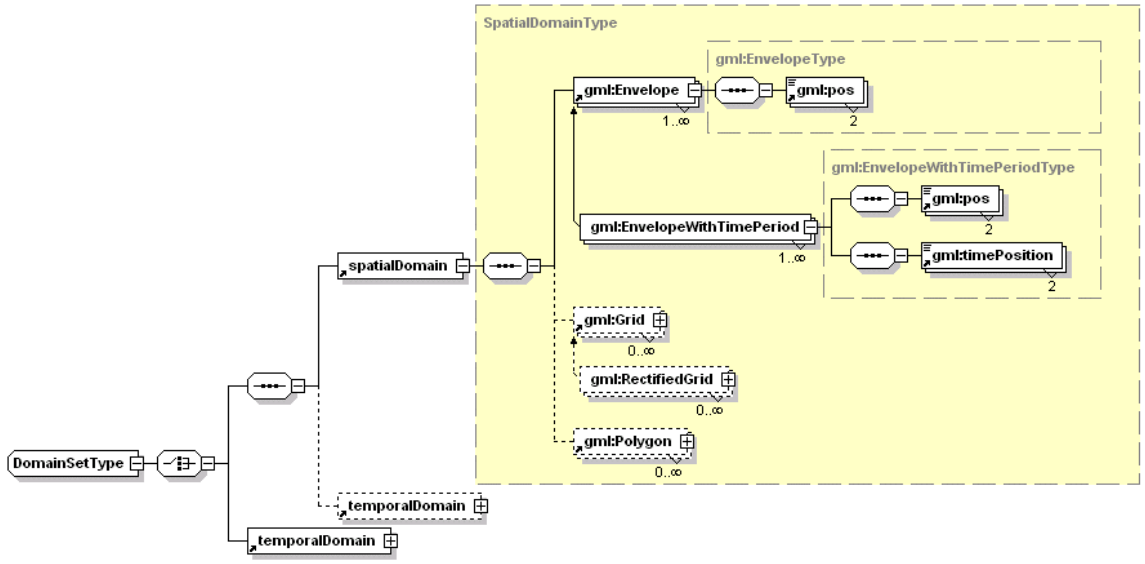

Figure 11. domainSet and spatialDomain

The **spatialDomain** element offers several options to service providers. First, a service provider must describe the spatial extent of the domainusing one or more GML **Envelope** elements. (The **GML EnvelopeWithTimePeriod** element may be used in place of **Envelope**, to add the time bounds of the coverage offering.) Each of these describes a bounding box defined by two points in space (or two positions in space and two in time). This bounding box may simply duplicate the information in the **lonLatEnvelope** of **CoverageOfferingBrief;** but the intent is to describe the locations in more detail (e.g., in several different CRSs, or several rectangular areas instead of one overall bounding box).

In addition, a service provider may describe the internal grid structure of a coverage offering, using a GML **Grid** or **RectifiedGrid** in addition to an **Envelope**. This element can help clients assess the fitness of the gridded data for their use (e.g. its native resolution, inferred from the **offsetVector** of a GML **RectifiedGrid**), and formulate grid coverage requests expressed in the internal grid coordinate reference system.

Finally, a service provider may also describe the spatial domain by means of a (repeatable) GML **Polygon**, representing the polygon(s) covered by the coverage spatial domain. This is particularly useful for areas that are poorly approximated by a GML **Envelope** (such as satellite image swaths, island groups, other non-convex areas).

8.3.2.3 TemporalDomain

The **TemporalDomain** element, which may or may not accompany a **spatialDomain** element, describes the valid time constraints for **GetCoverage** requests (that is, the times for which valid data are available). Figure 12 depicts the structure of the **TemporalDomain** element.

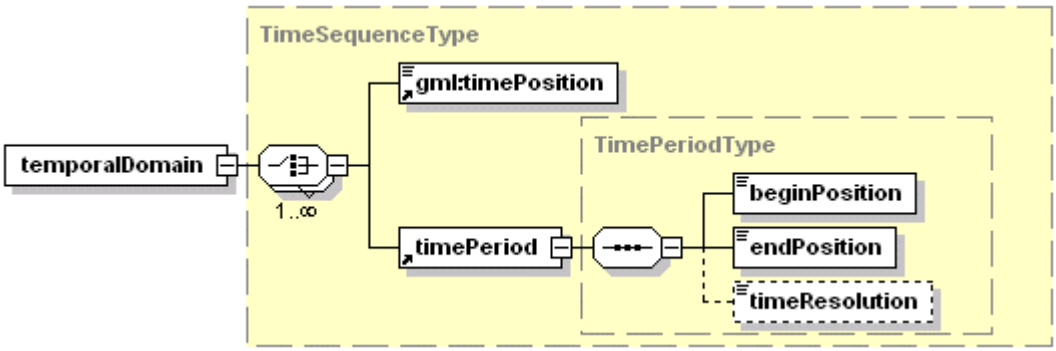

Figure 12. TemporalDomain

TemporalDomain is structured as any sequence of time instants (using GML's **TimePosition**) and / or time periods (using **timePeriod** with **beginPosition** and **endPosition**, both of the GML **TimePosition** type). These time periods may be regularly sampled (indicated by the optional **timeResolution** element) or continuous (when **timeResolution** is absent).

timePeriod, GML's **timePosition**, **beginPosition**, and **endPosition** have an optional attribute **frame** that denotes a reference time frame. The default frame is "ISO-8601", denoting the ISO 8601 frame. GML's **timePosition**, **beginPosition** and **endPosition** also have two optional attributes, **calendarEraName** and **indeterminatePosition**. These denote, respectively, the calendar era (e.g., "BC" or "AD") and indeterminate time values such as "now."

8.3.3 rangeSet

8.3.3.1 Overview

The second element of CoverageOffering is a **rangeSet**. This element defines the properties (categories, measures, or values) assigned to each location in the domain. Any such property may be a scalar (numeric or text) value, such as population density, or a compound (vector or tensor) value, such as incomes by race, or radiances by wavelength.

Figure 13 depicts the structure of the **rangeSet** element.

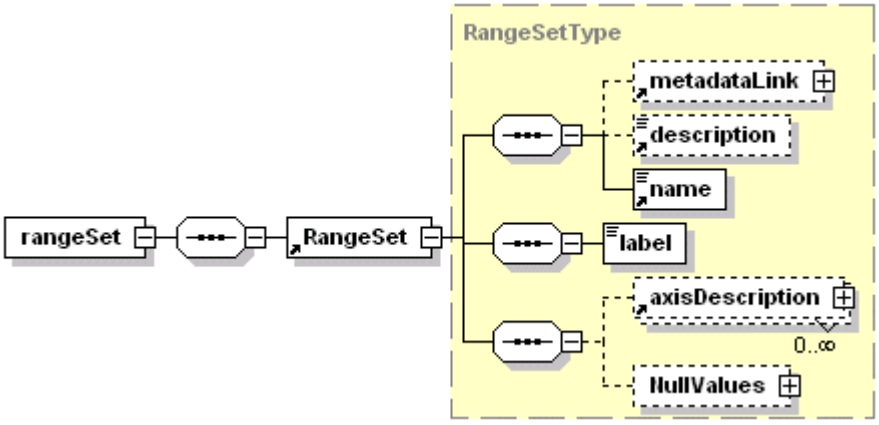

Figure 13 - rangeSet

The **rangeSet** property has one sub-element, **RangeSet**, with three attributes and six sub-elements. Its attributes are familiar (all are optional):

a) **semantic**: a pointer to the definition of the RangeSet values

f) **refSys**: a pointer to the reference system in which the RangeSet values are expressed

g) **refSysLabel**: a short label denoting the reference system, for onscreen display

RangeSet has the following sub-elements:

a) The first four (**metadataLink**, **description**, **name**, and **label**) are described in Subclauses 7.3.4.2 to 7.3.4.5.

h) The *optional and repeatable* **axisDescription/AxisDescription** element is for compound observations. It describes an additional parameter (that is, an independent variable besides space and time), and the valid values of this parameter, which GetCoverage requests can use to select subsets of a coverage offering. Subclause 8.3.3.2 provides further details.

i) The *optional* **nullValues** element is used when valid values are not available; see Subclause 8.3.3.3 for further details.

8.3.3.2 AxisDescription (for compound range sets)

8.3.3.2.1 Introduction

A range set may have either simple scalar values (such as terrain elevation, or yesterday's maximum temperature) or compound values. Compound values consist of a set of identically defined measurements or observations, reported for each of several values of a "control" variable, or aggregated into several "bins". The "bin" or "control" parameter may be any independent variable (besides those in the domain), which **GetCoverage** requests may use for constraints.

Examples of compound observations include a multispectral radiance (that is, brightness *by wavelength*, typical of satellite imagery), age distribution (counts of people *by age*

brackets, in a census table), or climate pattern (mean rainfall *by month of the year* in a climate database).

A compound range set may have more than one control parameter or set of "bins", for quantities related to values of several parameters (such as counts of wildlife tabulated both by size and by species).

8.3.3.2.2 XML syntax

For a compound-valued range set, in addition to describing the measured or observed quantities themselves, it's often useful to describe the control parameter(s) in some detail, and to list the "valid" parameter values – those for which measurements are available (or "by which" aggregate values are available). Such descriptions enable **GetCoverage** requests to retrieve meaningful subsets constrained along the values of the parameter. This is the intent of the optional **AxisDescription** element, structured as in Figure 14.

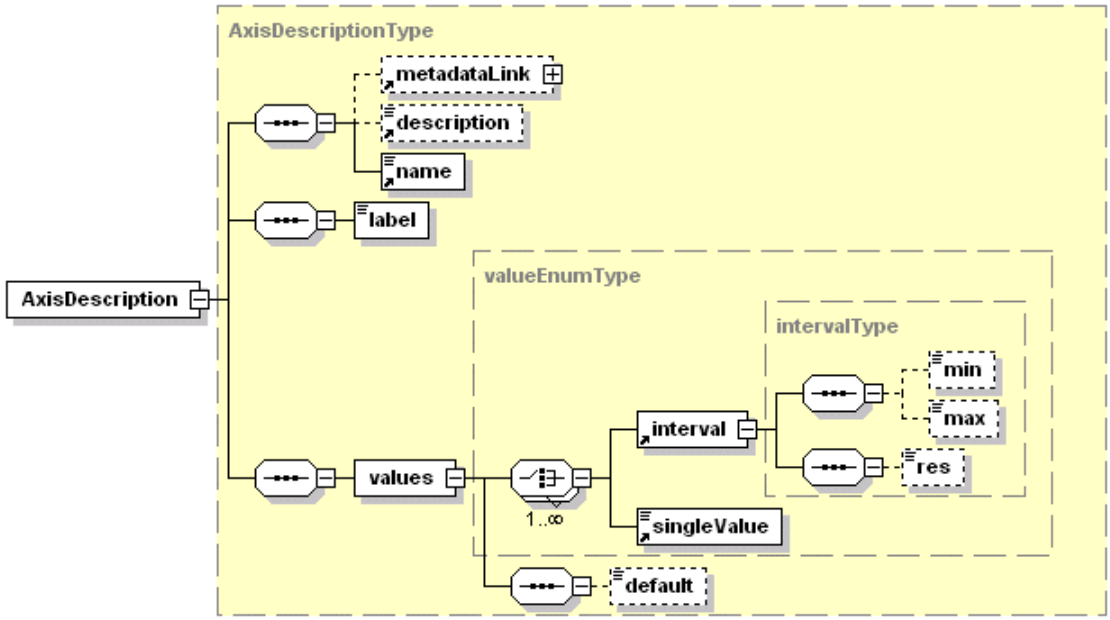

Figure 14 – AxisDescription

AxisDescription has three attributes (all optional):

a) **semantic** points to the definition of the parameter values

j) **refSys**: a pointer to the reference system in which the values are expressed

k) **refSysLabel**: a short label denoting the reference system, for onscreen display

AxisDescription has five sub-elements. The first four (**metadataLink, description**, **name**, and **label**) are described in Subclauses 7.3.4.2 to 7.3.4.5. In addition, the **values** element lists the parameter values or intervals for which data are available.

The **values** element has two optional attributes, **type** (denoting the element's datatype) and **semantic** (defined as for **AxisDescription**), and the following sub-elements:

a) **interval** denotes the time values between the values of its sub-elements **min** and **max**. It has three attributes and three sub-elements, all optional:

   1) The **type** attribute denotes the element's datatype. (This and the next attribute may override those on the parent **values** element.)
   2) The **semantic** attribute points to the definition of the parameter values.
   3) The **atomic** attribute indicates whether **GetCoverage** requests must use constraints that encompass the entire interval. (If false, then **GetCoverage** requests may use values in between **min** and **max** as constraints.)
   4) Sub-elements **min** and **max** list the lower and upper bounds of the interval for which coverage data are available. (Both values are expressed in the reference system denoted by the **refSys** attribute on **AxisDescription**). Both elements have an optional **closure** attribute denoting whether the interval is "closed" or "open" at each bound (i.e., includes or excludes the edge value itself). (The default value is "closed".)
   5) The interval may be a continuous set of parameter values between the **min** and **max** values, or a set of regularly spaced values. In the latter case, the **res** sub-element lists the spacing between adjacent parameter values. (If **res** is absent, the interval is a continuous set of parameter values: any value between **min** and **max** should produce a meaningful **GetCoverage** response.)

l) The **singleValue** element lists a single parameter value for which data are available. Its optional attributes, **type** and **semantic,** are defined as for **interval**.

NOTE The **values** element may have any sequence of **interval** and **singleValue** sub-elements.

m) The **default** element lists the parameter value that the server will use for **GetCoverage** requests that omit a constraint along this parameter. **GetCoverage** requests against a coverage offering whose **AxisDescription** has no **default** must specify a valid constraint for this parameter.

For example, the **AxisDescription** for a multispectral image might indicate that the coverage range reports brightness values for each of several wavelength "bands" expressed in nanometers.

8.3.3.2.3 Compound range sets

A compound valued range set is designed for observations that are identically defined – that report the same property, expressed in the same reference system. If a set of observations has any semantic variation, or any differences in the reference system, then the different kinds of observations belong in different coverages. For instance, a multispectral image in which some bands record emissive radiance, and others record reflective radiance, would require distinct coverages.

When several identically defined measurements are tied to an ordinal, interval, or ratio parameter, they may be structured either as several scalar-valued coverages or as a single compound-valued coverage. However, a compound-valued coverage is often preferable to multiple scalar-valued coverages. This is because the **AxisDescription** element lets clients retrieve subsets of the component observation by requesting intervals ("slices") along the parameter axis. For example, one may retrieve the near-infrared portions of a hyperspectral image by requesting that portion of the wavelength axis. Or, one may extract racial distribution among (only) the elderly from a table listing population counts by age and by race.

NOTE In a future version of this specification, compound values may also allow interpolation of measured observations along a range axis (where appropriate – e.g., for interval or ratio parameters, with values in narrow intervals separated by narrow gaps). For instance, hyperspectral imagery may allow interpolation of radiance values over wavelength if the spectral bands are narrow enough and close enough together. Similarly, population pyramids may allow interpolation of population over age if the age brackets are suitably defined.

The range axis construct also anticipates "virtual coverages" (that is, real-time coverage servers that can adjust measurement parameter values on request) – such as an imaging sensor whose wavelength bands can be remote controlled, or a census data server that tabulates raw questionnaire data into age brackets of the user's choice.

When several identically defined measurements are tied to a nominal (not ordinal) parameter (one for which intervals or "slices" are not defined, e.g., species, or landuse), a compound observable (such as counts by species) is functionally equivalent to multiple scalar-valued coverages (count of species1, count of species2, etc.). In either case, range subset requests are limited to lists of individual values (lions, tigers, bears, etc.). However, a compound observable does offer the notational convenience of describing the observable only once – a useful shorthand when the same observable is reported at many different values of a parameter.

8.3.3.3 NullValues

An important part of a range set description is the representation of null value(s) in the coverage. The coverage encoding itself may specify a fixed value for null (e.g. "–99999" or "N/A"), but more often the choice is up to the provider and must be communicated to the client outside of the coverage itself. This is the purpose of the optional **NullValues** element, whose structure is depicted in Figure 15.

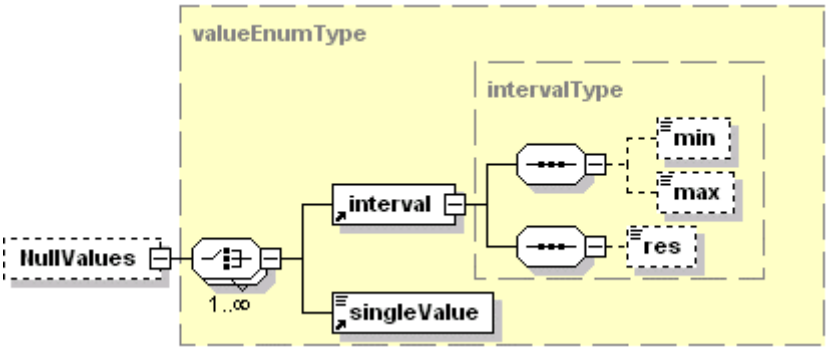

Figure 15. NullValues

NullValues is structured and interpreted exactly like the **values** element in **AxisDescription** (see Subclause 8.3.3.2.2).

8.3.4 SupportedCRSs and coordinate reference systems (CRS)

8.3.4.1 Overview

For each coverage offering, the **supportedCRSs** element lists the CRSs in which the server understands incoming **GetCoverage** requests; and those in which it can respond to **GetCoverage** requests. It may also list the native CRS of the data. Its structure is depicted in Figure 16.

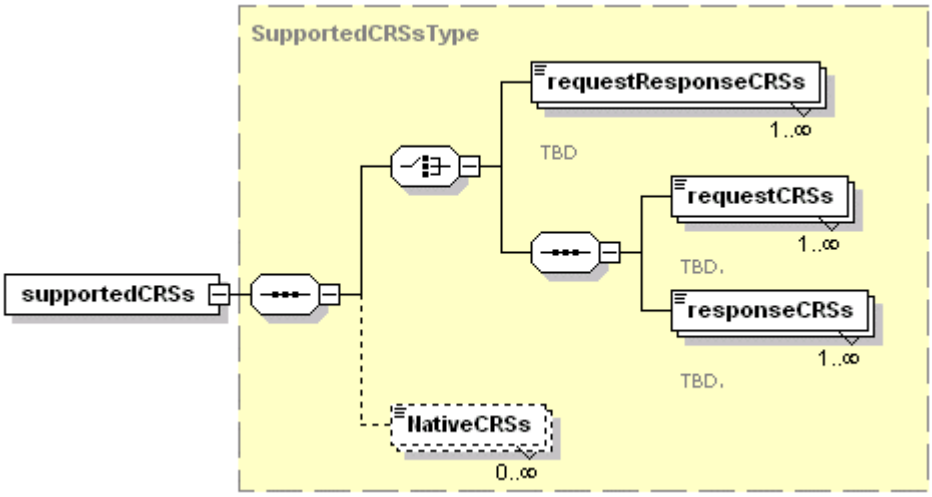

Figure 16. SupportedCRSs

The **supportedCRSs** element has *either* a **requestResponseCRSs** sub-element, *or both* a **requestCRSs** sub-element *and* a **responseCRSs** sub-element.It may also have a **NativeCRSs** element. These sub-elements all have the same content model, a list of CRS identifiers within a single code-space. These CRS identifiers may be any of the **EPSG:xyz**, **AUTO:xyz, or OGC:xyz** coordinate systems defined in the Web Map Service Implementation Specification (OGC Doc. 01-0685r3); or the strings "**Engineering**" or "**Image**" to denote an "engineering" or "image" CRS, whose relationship to earth coordinates may not be well defined. (The "image" CRS applies to images and grid coverages: it is a special kind of engineering CRS, defined as row and column offsets from the image origin.)

NOTE To infer the ground positions of locations expressed in the engineering or image coordinates of a coverage, a client needs additional information, such as control points or sensor metadata (e.g., the orbital model). WCS does not specify how to request, encode, or transmit this additional information: it may be embedded in the coverage response, available from a separate source, or otherwise known to the client.

For *georectified images or grids* , when this CRS references an EPSG or AUTO coordinate reference system, it designates the base ground CRS for the georectified image or grid -- not the internal image CRS. Each referenced CRS must be the same as referenced by the srsName attribute of one of the **RectifiedGrid** elements defined in Subclause 8.2.1.1.

8.3.4.2 requestResponseCRSs

A coverage offering must either advertise the CRSs in which it can both accept GetCoverage requests and deliver coverage responses, or detail its supported request and response CRSs separately. Thus, every Coverage *must* have *either* a **requestResponseCRSs** element, *or both* a **requestCRSs** *and* a **responseCRSs** element (each of which list one or more CRS identifiers). These CRSs *should* include the coverage offering's native CRS(s) as defined below**.**

8.3.4.3 requestCRSs

The **requestCRSs** element states the CRS(s) in which **GetCoverage** requests may be expressed against a coverage offering. These CRSs *should* include the coverage offering's native CRS(s) as defined below**.**

8.3.4.4 responseCRSs

The **responseCRSs** element states the CRS(s) in which coverage replies to GetCoverage requests may be expressed. These CRSs *should* include the coverage offering's native CRS(s) as defined below**.**

Servers that serve coverages in the special CRS codes "**Engineering**" or "**Image**" defined above **may** embed georeferencing information (e.g., sensor model or tie-points) in the coverage reply.

8.3.4.4.1 nativeCRSs

The *optional* **nativeCRSs** element states the native CRS(s) of a coverage – that is, the CRS(s) in which coverages can be obtained without any distortion or degradation of the data.

8.3.5 SupportedFormats

The *required* **supportedFormats** element advertises the output format(s) in which coverages may be requested from this coverage. Figure 17 depicts its structure.

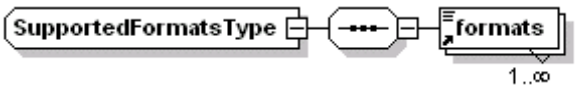

Figure 17. supportedFormats

These formats are identified by a simple string. Any format is acceptable, provided that at least one of the following formats is supported for each **coverageOffering**.

a) GeoTIFF <http://www.remotesensing.org/geotiff/geotiff.html>

b) HDF-EOS <http://heineken.gsfc.nasa.gov/>

c) DTED <http://www.nima.mil/publications/specs/printed/89020A/89020A_DTED.pdf>

d) NITF <http://www.ismc.nima.mil/ntb/baseline/1999.html>

e) GML <http://www.opengis.org/techno/documents/02-023r4.pdf>

Servers may serve coverages in other encodings as well. An individual server must indicate what encodings it supports on a coverage offering by listing them in the **supportedFormats** element in a **DescribeCoverage** response.

8.3.6 SupportedInterpolations

The *optional* **supportedInterpolations** element states whether and how the server will interpolate coverage values over the spatial domain when a request requires resampling, reprojection, or other generalization. Such interpolation will take effect with any GetCoverage request except when coverage data are retrieved exactly in their original resolution and native CRS. The interpolation method indicated shall be applied simultaneously to all spatial dimensions.

Using a *(repeatable)* **InterpolationMethod** sub-element, a coverage offering may list any of 6 spatial interpolation methods (Table 8).

Table 8. Interpolation methods

Interpolation Method	Description
nearest neighbor (default)	These are defined in ISO 19123 (Schema for Coverage Geometry and Functions), Annex B.
bilinear	
bicubic	
lost area	
barycentric	
none	No interpolation is available; requests must be for locations that are among the original domain locations.

supportedInterpolations has a **default** attribute that lists what interpolation method is used for GetCoverage requests that don't specify one. If **supportedInterpolations** is absent or empty with no **default** attribute, then clients should assume *nearest-neighbor* interpolation.

If the only interpolation method listed is 'none', clients may only retrieve (subsets of) this coverage in its native CRS and at its native resolution.

NOTE Currently no method is defined for interpolation over the time domain.

9 GetCoverage operation

9.1 Introduction

The **GetCoverage** operation allows retrieval of coverages from a coverage offering. A WCS server processes a **GetCoverage** request and returns a response to the client.

9.2 GetCoverage requests

9.2.1 Overview

A GetCoverage request may be encoded as key-value pairs, or as an XML document. The next two subclauses detail each of these encodings.

9.2.2 Key-value pair encoding

9.2.2.1 Overview

Table 9 specifies the complete GetCoverage Request.

Table 9 – The GetCoverage Request expressed as Key-Value Pairs.

URL Component	Description
http://server_address/path/script?	URL of WCS server. *Required.*
SERVICE=WCS	Service name: Must be "WCS". *Required*.
VERSION=1.0.0	Request protocol version. *Required*.
REQUEST=GetCoverage	Name of the request. Must be "GetCoverage". *Required*.
COVERAGE=*name*	Name of an available coverage. *Required.*
CRS=crs_identifier	Coordinate Reference System in which the request is expressed. *Required.*
RESPONSE_CRS= crs_identifier	Coordinate Reference System in which to express coverage responses. *Optional*; defaults to the request CRS.
BBOX=minx, miny, maxx, maxy, minz, maxz	Request a subset defined by the specified bounding box, with min/max coordinate pairs ordered according to the Coordinate Reference System identified by the CRS parameter. One of BBOX or TIME is *required*.
TIME= *time1,time2,…* *or* TIME= *min/max/res, …*	Request a subset corresponding to the specified time instants or intervals, expressed in an extended ISO 8601 syntax. *Optional* if a default time (or fixed time, or no time) is defined for the selected layer. One of BBOX or TIME is *required*.
PARAMETER= *val1,val2, …* *or* *PARAMETER*= *min/max/res*	*(Included only for range sets with compound values)* Request a range subset defined by constraining parameter *PARAMETER*. The *PARAMETER* key is a variable string; it must match the name of a parameter listed in the range set description of the selected coverage. For instance: *band=1,5,3 (e.g., radiance values in bands 1, 5, 3)* *age=0/18 (e.g., counts of people with ages under 18 yrs.)* *Optional* if the chosen range component has default values for the parameter.
WIDTH = w *(integer)* HEIGHT = h *(integer)* [DEPTH =d *(integer)*]	Request a grid of the specified width (w), height (h), and [for 3D grids] depth (d) (integer number of gridpoints). *Either* these or RESX, RESY, [for 3D grids] RESZ are *required*.
RESX=*x (double)* RESY=*y (double)* [RESZ=*z (double)*]	*[when requesting georectified grid coverages]* Request a coverage subset with a specific spatial resolution along each axis of the reply CRS. The values are given in the units appropriate to each axis of the CRS. *Either* these or WIDTH, HEIGHT, and [for 3D grids] DEPTH are *required*.
INTERPOLATION = *interpolation-method*	Requested spatial interpolation method for resampling coverage values into the desired output grid. *interpolation-method* must be one of the values listed in the **supportedInterpolations** element of the requested **CoverageOffering**. *Optional;* server-defined default as stated in 8.3.6.
FORMAT= *format*	Requested output format of Coverage. Must be one of those listed under the description of the selected coverage. *Required*.
EXCEPTIONS= application / vnd.ogc.se_xml	The format in which exceptions are to be reported by the Server. *Optional*.
(Vendor-specific parameters)	*Optional*.

9.2.2.2 SERVICE=WCS / VERSION=version

These parameters are defined as for GetCapabilities in Subclause 7.2.

9.2.2.3 REQUEST=GetCoverage

The Basic Service Elements clause defines this parameter. For GetCoverage, the value "GetCoverage" must be used.

9.2.2.4 COVERAGE=*name*

The **COVERAGE** parameter requests a single coverage, identified by a **name** under a **CoverageOfferingBrief** in the **ContentMetadata** section of the Capabilities XML document. If the Capabilities XML document does not have a **ContentMetadata** section, clients must obtain a valid coverage identifier from another source (such as a catalog service).

NOTE Future versions of this WCS specification may address ways to request multiple coverages, combining them according to mathematical or logical operators (Boolean or other rule-based overlay).

9.2.2.5 CRS

The CRS (Coordinate Reference System) parameter is defined in Subclause 8.3.4.

GetCoverage requests must use this parameter to specify the coordinate reference system in which the request domain constraints are expressed (**BBOX**). The values of this request parameter **must** be one of those defined in a **requestResponseCRSs** or **requestCRSs** element under the requested coverage.

If the Capabilities XML's **requestCRSs** element for a coverage offering lists only ***"Engineering"*** or ***"Image"*** (indicating a coverage that is not available in georectified form), then clients must request that coverage offering in its internal (local / pixel) coordinate reference system, by specifying **CRS=Engineering or CRS=Image** (case-insensitive) in the **GetCoverage** request.

Some WCS servers may support on-the-fly georectification of coverages that are georeferenced but not already georectified. Such servers accept requests expressed in a coverage's internal pixel / local coordinate system, but are able to express coverage replies in a ground coordinate system. Such servers may indicate this capability for a coverage by listing ***"Engineering"*** or ***"Image"***in their **requestCRSs** element, but also listing a ground coordinate system in a **ResponseCRSs** element for the same coverage offering. In such cases, GetCoverage requests may specify **CRS=Engineering** or **CRS=Image** (case-insensitive); but add a **RESPONSE_CRS** value corresponding to a ground coordinate reference system.

9.2.2.6 RESPONSE_CRS

This parameter specifies the coordinate system in which the coverage response should be referenced. This parameter is optional; its value defaults to that of **CRS** (described previ-

ously). Thus, omitting it requests a coverage response referenced in the same coordinate reference system as the request (like WMS and WFS).

The value of this request parameter **must** be one of those defined in a **requestResponseCRSs** or **responseCRSs** element under the requested coverage offering.

9.2.2.7 BBOX

A GetCoverage request may include a 1-D, 2-D, or 3-D spatial constraint expressed as a rectangle (or line, or parallelepiped) aligned with the axes of the spatial reference system given in the **CRS** parameter. Such a constraint is expressed as a **BBOX** parameter representing the coordinates of the southwest/lower and northeast/upper corners (in that order) as comma-separated numbers (e.g., **minx, miny, maxx, maxy**).

NOTE The order (southwest, northeast) often corresponds to (minimum x, minimum y, maximum x, maximum y) – but this is not always the case. For instance, when a Bounding Box expressed in longitude and latitude crosses the antimeridian (the meridian with longitude +/-180 degrees), its northeast corner's longitude is often less than that of its southwest corner.

Each corner's coordinate(s) must be expressed in the order and units given by the CRS.

For any part of the coverage domain that is partly or entirely contained in the Bounding Box defined by BBOX, the server must return coverage data in the requested format.

A GetCoverage request must include a valid BBOX, or TIME (below), or both.

9.2.2.8 TIME

If the **DescribeCoverage** XML reply defines a **TemporalDomain** on the selected coverage, GetCoverage requests may use a separate **TIME** parameter to constrain the request in time, thus supplementing a spatial (1D, 2D, or 3D) Bounding Box.

Time constraints and date / time values must be expressed using a time frame identified by the **frame** attribute on an element in the **TemporalDomain** of the requested coverage offering. The special keyword "***now***" may be used in lieu of a determinate time value, to request the most recent available data.

A GetCoverage request must include a valid BBOX (above), or TIME, or both.

9.2.2.9 PARAMETER

If the range set of the selected coverage consists of compound values, **GetCoverage** requests may include constraints defined on the parameter(s) of the compound range set. Such constraints are expressed as "*PARAMETER*=[value]", where

a) The variable string *PARAMETER* matches the **name** of an **AxisDescription** element defined on that range set, and

b) [value] is one of the acceptable values defined in the corresponding **AxisDescription** element.

For example, a coverage range set might consist of radiance values reported by wavelength intervals. For such a coverage offering, the **DescribeCoverage** reply might include an **AxisDescription** with the **name** "Wavelength", with units in nanometers. Given such a description, a **GetCoverage** request might limit the request to visible wavelengths by specifying

```
WAVELENGTH=650/700, 500/560, 430/500
```

This parameter constraint is optional if the selected range set has a **default** value on the corresponding **AxisDescription**.

9.2.2.10 Grid size: WIDTH, HEIGHT, DEPTH

GetCoverage requests may request coverage replies with a specific grid size. The parameters WIDTH, HEIGHT, DEPTH define the grid size (number of gridpoints or cells) along the three axes of the grid.

Either these parameters or RESX, RESY, RESZ are normally required. However, if the Capabilities XML reports *only* the Interpolation method "None" for the queried coverage, then GetCoverage requests must be for the full native resolution of the data; they may not use RESX, RESY, RESZ or WIDTH, HEIGHT, DEPTH to change the coverage resolution. In this case, BBOX alone is used for subsetting.

9.2.2.11 Grid resolution: RESX, RESY, RESZ

GetCoverage requests for gridded coverages may request coverage replies in specific grid resolutions. The parameters RESX and RESY define the grid-cell size along the first and second axes of the coordinate reference system given in CRS or RESPONSE_CRS.

If the RESPONSE_CRS is a 3D spatial reference system, then the additional RESZ parameter may be used to specify the desired resolution along the third axis of that coordinate reference system.

Either these parameters or WIDTH, HEIGHT, DEPTH are normally required when requesting grid coverages. However, if the DescribeCoverage XML reply reports *only* the Interpolation method "None" for the queried coverage, then GetCoverage requests must request the full native resolution of the data: they may not use RESX, RESY, RESZ or WIDTH, HEIGHT, DEPTH to change the coverage resolution. In this case, BBOX alone is used for subsetting.

9.2.2.12 INTERPOLATION

This parameter specifies what type of interpolation to use for resampling coverage values over the spatial domain. (Such an interpolation must always be expected if the request contains any of the parameters RESX, RESY, RESZ, or RESPONSE_CRS.) Its value must be one of those listed for this coverage offering in the DescribeCoverage XML response (Subclause 8.3.6). This parameter is optional; in its absence the server will apply a default interpolation method (which it may have declared in advance – cf. 8.3.6 above).

9.2.2.13 FORMAT

The value of this parameter must be one of those listed in a **supportedFormats/formats** element (see 8.3.5 above) under the selected coverage offering in the DescribeCoverage XML reply. In an HTTP environment, a "Content-type" entity header, containing an appropriate MIME type string for the chosen format, must precede the returned object.

9.2.2.14 EXCEPTIONS

A Web Coverage Service **must** offer the exception reporting format "**application/vnd.ogc.se_xml"** by listing it in a **Capability / Exceptions / Format** element in its Capabilities XML response. The entire MIME type string in **Capability / Exceptions / Format** is used as the value of the EXCEPTIONS parameter.

Errors are reported using Service Exception XML, as specified in Subclause A.3. This is the default exception format if none is specified in the request.

9.2.3 XML encoding

9.2.3.1 Overview

Figure 18 provides an overview of the **GetCoverage** XML request syntax.

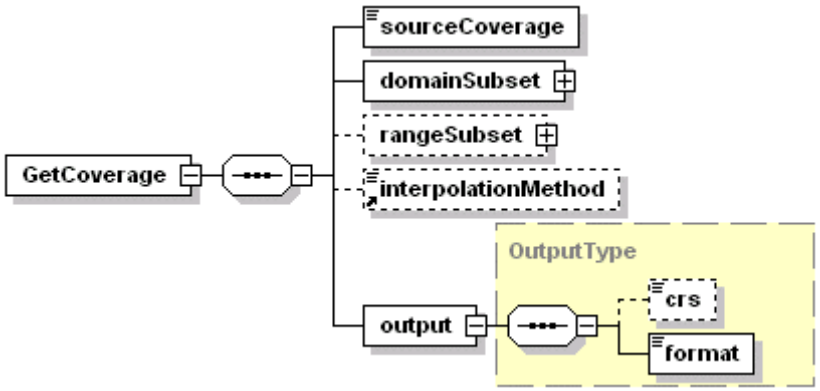

Figure 18 - GetCoverage

GetCoverage has two required attributes, **service** and **version** (defined as for GetCapabilities in Subclause 7.2). It also has 5 required sub-elements, described in Subclauses 9.2.3.2 to 9.2.3.6 below.

9.2.3.2 sourceCoverage

The **sourceCoverage** element specifies a single coverage available from the WCS server. Its value must match that of a **CoverageOfferingBrief / name** element obtained from the WCS server's Capabilities XML document, or from a third-party catalog describing the server's holdings. The type of the **sourceCoverage** element is `anyURI`.

9.2.3.3 DomainSubset

The **domainSubset** specifies what subset of the spatial and temporal domain to retrieve. Its syntax (shown in Figure 19 below) resembles that of **domainSet** in the coverage description (Subclause 8.3.2).

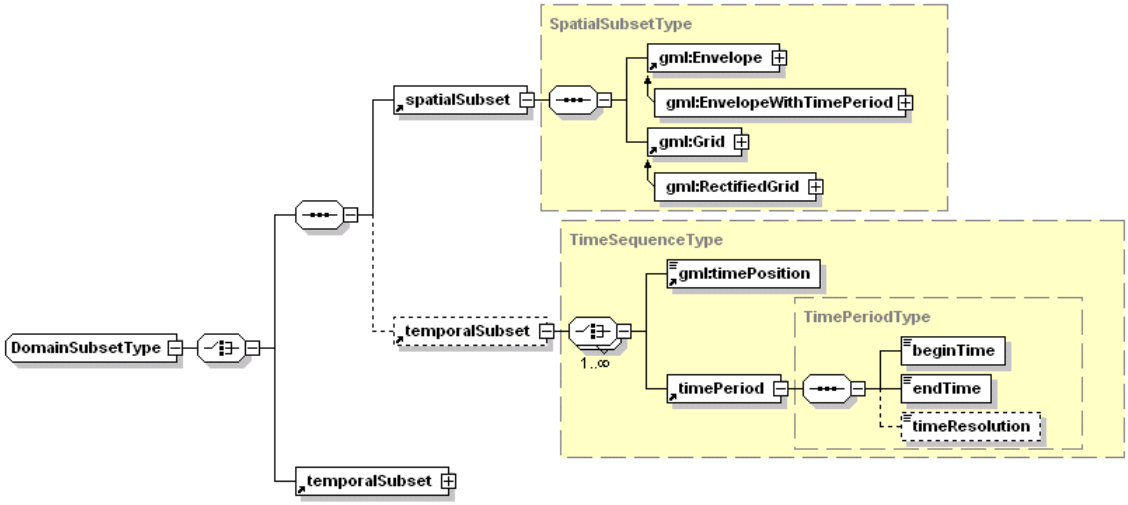

Figure 19. DomainSubset

The **domainSubset** must include either a **spatialSubset** (stating the spatial locations for the requested coverage), a **temporalSubset** (describing the time instant(s) or interval(s) for the requested coverage), or both.

The **spatialSubset** restricts the **spatialDomain** type described in Subclause 8.3.2.2. It requires both a single GML **Envelope** (to request data for an overall extent in space and time) and a single GML **Grid** (to specify the grid size (rows and columns) of the returned coverage).

NOTE In response to a **GetCoverage** request, a WCS server will return a grid of the requested size covering the requested area. This usually requires interpolating / resampling the coverage values stored on the server. To avoid any interpolation / resampling, clients should request the coverage in a native CRS stated by the server; and select a GML **Envelope** whose extent exactly matches that of the requested GML **Grid**. For such a request, if the chosen CRS is "Image" or "Engineering", the **Envelope** and **Grid** must both describe grids of the same size. For other CRSs, the **Envelope** and **Grid** must be related by the **offsetVector** values in the coverage description (if supplied in the coverage description).

Clients may substitute a GML **RectifiedGrid** for the GML **Grid**. This lets a client specify separate row and column offsets – e.g., when requesting a georectified grid whose rows and columns are not aligned with the axes of the chosen CRS.

NOTE A GML **RectifiedGrid** defines both a grid size (rows and columns) and a grid spacing in ground coordinates. Therefore it defines a particular spatial extent, which should match that of the GML **Envelope**.

The **temporalSubset** element has the same structure as **temporalDomain** (Subclause 8.3.2.3): it allows requests to specify a sequence of time instants and / or intervals.

9.2.3.4 RangeSubset

In the case of a compound range set, clients may request subsets by constraining the value of a range axis / parameter. **The rangeSubset** expresses what subset of the range to retrieve, using a repeatable **axisSubset** element, as depicted in Figure 20.

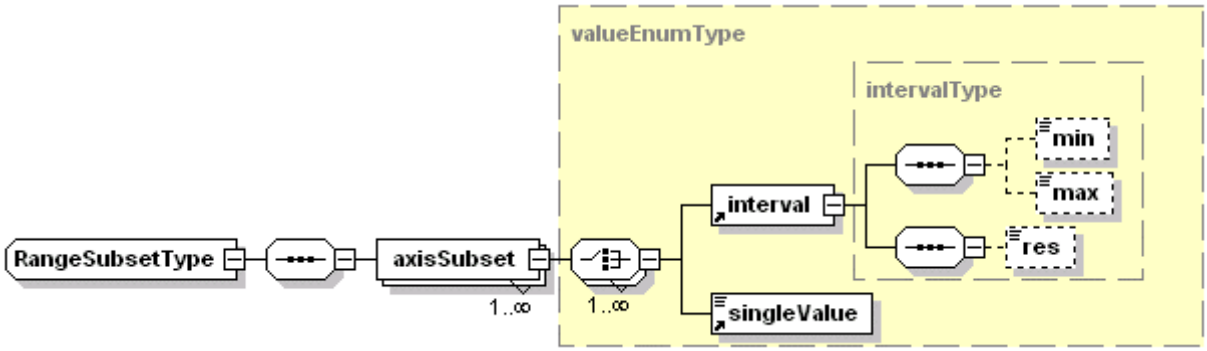

Figure 20. RangeSubset

The **axisSubset** element has a **name** attribute that must match that of an **AxisDescription** element defined on the given range component in the DescribeCoverage XML response (Subclause 8.3.3.2). The value(s) requested under **axisSubset** must be among those listed for the requested **AxisDescription** in the **DescribeCoverage** XML reply**.**

9.2.3.5 InterpolationMethod

The interpolationMethod specifies what type of interpolation to use for resampling coverage values over the spatial domain. This must be one of those listed for this coverage offering in the DescribeCoverage XML response (Subclause 8.3.6). Such an interpolation must always be expected if the request contains any of the components Grid (9.2.3.3) or crs (9.2.3.6). The **interpolationMethod** element is optional; in its absence the service will apply a default interpolation method (which it may have declared in advance – cf. 8.3.6 above).

9.2.3.6 Output CRS and Format

The **output** element asks for coverage responses to be expressed in a particular Coordinate Reference System (**crs**) and encoded in a particular **format**. Values for these elements must be among those listed under **supportedCRSs** and **supportedFormats**, respectively, in the **DescribeCoverage** XML reply (Subclauses 8.3.4 and 8.3.5).

9.3 GetCoverage response

9.3.1 Overview

The response to a valid GetCoverage request **must** be a coverage extracted from the coverage requested, with the specified spatial reference system, bounding box, size, and format.

An invalid GetCoverage request **must** yield an error output in the requested Exceptions format (or a network protocol error response in extreme cases).

In an HTTP environment, the returned value must have a Content-type entity header that matches the format of the return value.

9.3.2 Coverage encoding

A WCS server shall serve coverages in any of the formats listed in **supportedFormats** for the requested coverage offering

a) GeoTIFF <http://www.remotesensing.org/geotiff/geotiff.html>

b) HDF-EOS <http://heineken.gsfc.nasa.gov/>

c) DTED <http://www.nima.mil/publications/specs/printed/89020A/89020A_DTED.pdf>

d) NITF <http://www.ismc.nima.mil/ntb/baseline/1999.html>

e) GML <http://www.opengis.org/techno/documents/02-023r4.pdf>

(see 8.3.5 above).

9.3.3 Multi-file payloads

Several well-known coverage formats encode data and metadata into multiple files. When returning a multi-file payload to the client, WCS 1.0.0 servers should use multi-part MIME encoding [IETF RFC 2045].

NOTE A multi-part MIME bundle provides no guidance to clients as to the role of each file: the client must be able to recognize each piece by its MIME type or suggested filename. Other methods for bundling files exist, such as SOAP XML messages with binary attachments; but these will require additional consensus across the OGC suite of specifications.

9.4 Exceptions

For WCS, the **Exceptions** tag in the Capabilities response (and its counterpart EXCEPTIONS parameter in a GetCoverage request) is optional; if present, it must have a valid MIME type string as its value.

A Web Coverage Server throwing an exception shall adhere to the value of the EXCEPTIONS parameter. Nonetheless, a Web Coverage server may, due to circumstances beyond its control, return nothing (this might result from the HTTP server's behavior caused by a malformed request, by an invalid HTTP request, by access violations, or any of several other conditions). Web Coverage Service clients should be prepared for this eventuality.

Annex A
(normative)

WCS XML Schemas

A.1 GetCapabilities request Schema

See file wcsCapabilities.xsd

A.2 GetCapabilities response schema

See file wcsCapabilities.xsd

A.3 DescribeCoverage request schema

See file describeCoverage.xsd

A.4 DescribeCoverage response schema

See file describeCoverage.xsd

A.5 GetCoverage request schema

See file getCoverage.xsd

A.6 Service exception schema

This subclause contains the Service Exception Schema corresponding to this version of the WCS specification. This subclause also summarizes the defined exception codes and their meanings.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.opengis.net/ogc"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:xs="http://www.w3.org/2001/XMLSchema" ele-
mentFormDefault="qualified"  attributeFormDefault="unqualified">
    <xs:element name="ServiceExceptionReport">
        <xs:annotation>
            <xs:documentation> The ServiceExceptionReport element contains one or more Ser-
viceException elements that describe a service exception. </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="ServiceException" type="ogc:ServiceExceptionType"
```

```
                    minOccurs="0" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation> The Service exception element is used to describe a
service exception. </xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="version" type="xs:string" fixed="1.2.0"/>
    </xs:complexType>
</xs:element>

<xs:complexType name="ServiceExceptionType">
    <xs:annotation>
        <xs:documentation> The ServiceExceptionType type defines the ServiceException
element. The content of the element is an exception message that the service wished to convey
to the client application. </xs:documentation>
    </xs:annotation>    <xs:simpleContent>
        <xs:extension base="xs:string">
            <xs:attribute name="code" type="xs:string">

                <xs:annotation>
                    <xs:documentation> A service may associate a code with an exception by
using the code attribute. </xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="locator" type="xs:string" use="optional">
                <xs:annotation>
                    <xs:documentation> The locator attribute may be used by a service to indi-
cate to a client where in the client's request an exception was encountered. If the request in-
cluded a 'handle' attribute, this may be used to identify the offending component of the request.
Otherwise the service may try to use other means to locate the exception such as line numbers
or byte offset from the begining of the request, etc ... </xs:documentation>
                </xs:annotation>          </xs:attribute>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>

</xs:schema>
```

Table A.1 — Exception codes defined by this specification

Exception Code	Meaning
InvalidFormat	Request contains a Format not offered by the service instance.
CoverageNotDefined	Request is for a Coverage not offered by the service instance.
CurrentUpdateSequence	Value of (optional) UpdateSequence parameter in GetCapabilities request is equal to current value of Capabilities XML update sequence number.
InvalidUpdateSequence	Value of (optional) UpdateSequence parameter in GetCapabilities request is greater than current value of Capabilities XML update sequence number.
MissingParameterValue	Request does not include a parameter value, and the service instance did not declare a default value for that parameter.
InvalidParameterValue	Request contains an invalid parameter value.

Annex B
(informative)

XML examples

B.1 Introduction

As an aid to understanding and a guide for implementation, this annex contains **example** XML which is valid according to the XML schemas in Annex A. Implementers should consult the main body of the specification document and the schemas to ensure compliance rather than editing this XML without full understanding.

B.2 Example GetCapabilities XML request

B.3 Example GetCapabilities XML response

B.4 Example DescribeCoverage XML request

B.5 Example DescribeCoverage XML response

B.6 Example GetCoverage XML request

B.7 Example Service Exception XML

Annex C
(normative)

Conformance

C.1 Introduction

Specific conformance tests for Web Coverage Service have not yet been determined and will be added in a future revision of this specification. At the moment, a WCS implementation must satisfy the following system characteristics to be minimally conformant with this specification:

a) *WCS Clients and servers* must support the *GetCapabilities*, *DescribeCoverage***,** and *GetCoverage* operations.

b) *WCS clients* must issue *GetCapabilities* requests in Key-Value Pair (KVP) or XML form. GetCapabilities KVP requests must conform to Subclause **7.2.2**. *GetCapabilities* XML requests must conform to Subclause **7.2.3**, and must be valid against the XML Schema definition in Subclause **A.1**.

c) *WCS servers* must respond to a *GetCapabilities* request with an XML document that conforms to Subclause ***7.3***, and is valid against the XML Schema definition in Subclause **A.2**.

d) *WCS clients* must issue *DescribeCoverage* requests in Key-Value Pair (KVP) or XML form. *DescribeCoverage* KVP requests must conform to Subclause **8.2.2.** DescribeCoverage XML requests must conform to Subclause **8.2.3**, and must be valid against the XML Schema definition in Subclause **A.3**.

e) *WCS servers* must respond to a *DescribeCoverage* request with an XML document that conforms to Subclause **8.3**, and is valid against the XML Schema definition in Subclause **A.4**.

f) *WCS clients* must issue *GetCoverage* requests in Key-Value Pair (KVP) or XML form. GetCapabilities KVP requests must conform to Subclause **9.2.2**. *GetCoverage* XML requests must conform to Subclause **9.2.3**, and must be valid against the XML Schema definition in Subclause **A.5**.

g) *WCS servers* must be able to respond to a *GetCoverage* operation with a coverage encoded in one of the output formats listed in Subclause **9.3.2**.

h) All clauses in the normative clauses of this specification that use the keywords "must", "must not", "required", "shall", and "shall not" must be satisfied.

Annex D
(informative)

UML model

D.1 Introduction

This annex provides a UML model of the WCS interface, using the OGC/ISO profile of UML summarized in Subclause 5.2.

Figure D.1 is a UML diagram summarizing the WCS interface. This class diagram shows that the WebCoverageService class inherits the getCapabilities operation from the abstract OGCWebService class, which is common to all OGC Web Services. The WebCoverageService class adds the getCoverage and describeCoverage operations. (The capitalization of class, operation, and data type names uses the OGC/ISO profile of UML.)

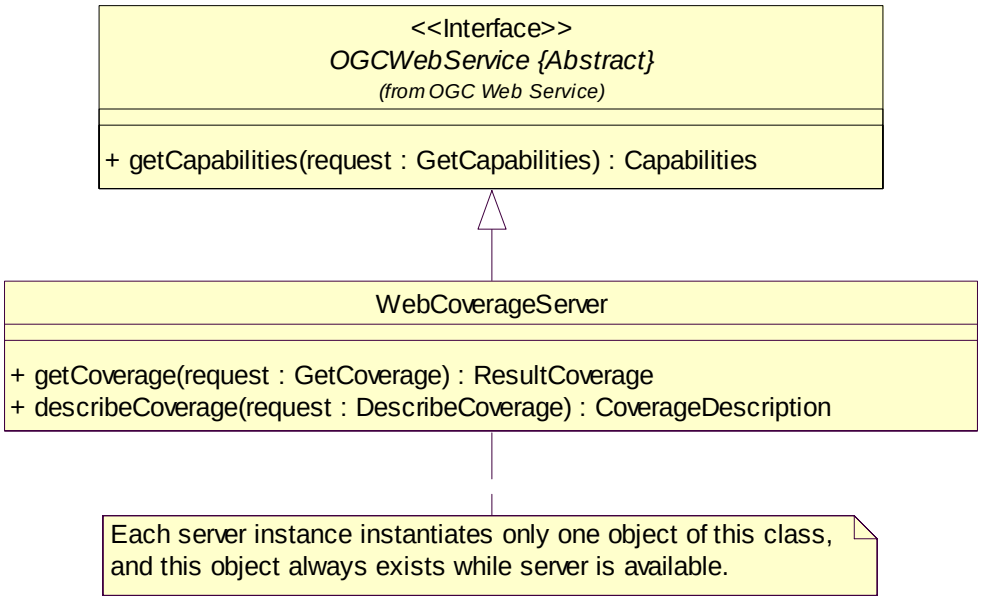

Figure D.1 — WCS interface UML diagram

Each of the three operations uses a request and a response data type, each of which can also be defined by one or more additional UML classes The following subclauses provide a more complete UML model of the WCS interface, adding UML classes defining the operation request and response data types.

D.2 UML packages

The WCS interface UML model is organized in nine packages, as shown in the package diagram in Figure D.2. These nine WCS-specific packages make use of three non-WCS-

specific packages, named OGC Web Service, ISO 19115 Subset, and GML Subset. This package diagram shows the dependencies among the various packages

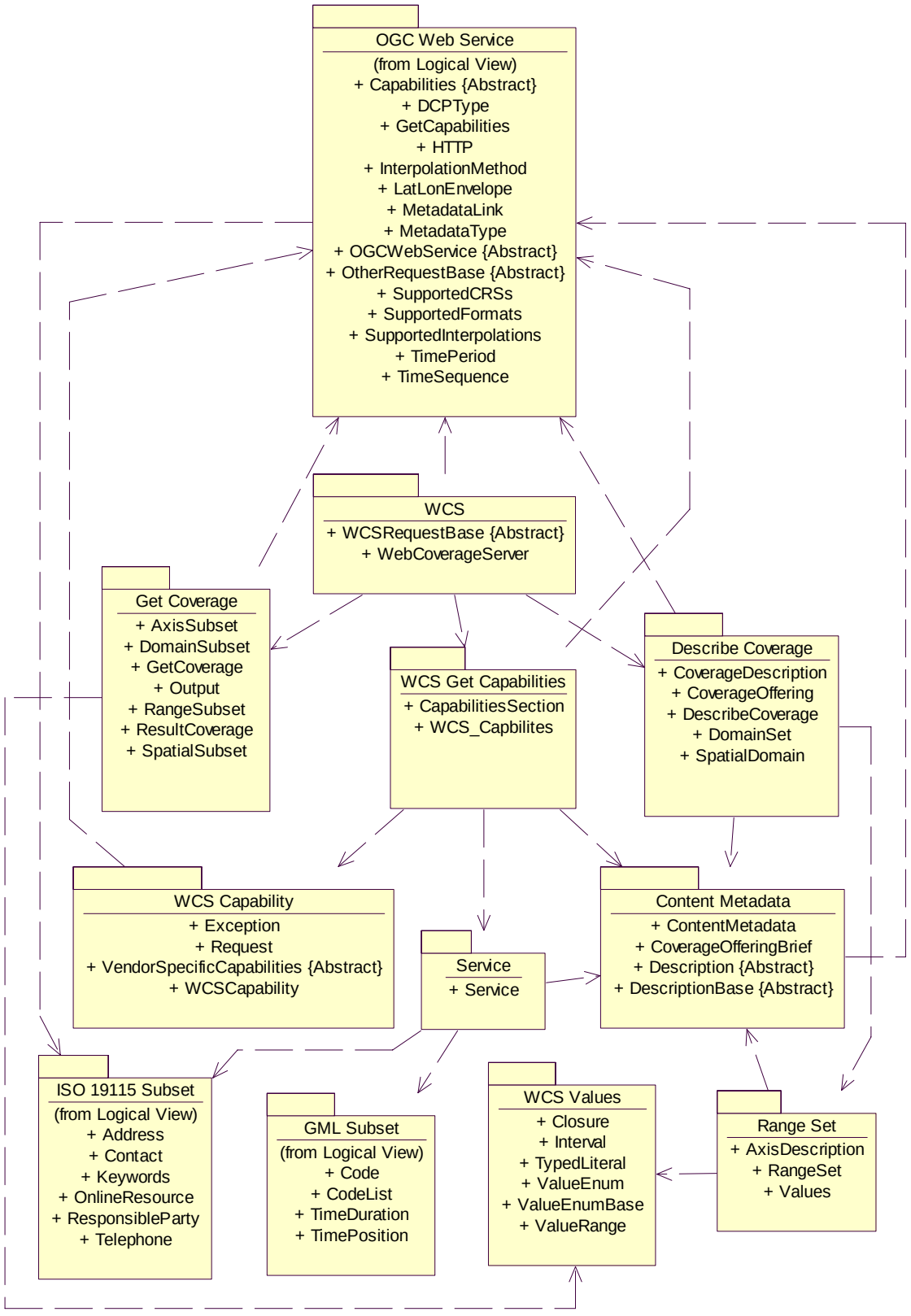

Figure D.2 — WCS interface package diagram

Each of the nine WCS-specific packages shown in Figure D.2 is described in the following subclauses, followed by the OGC Web Service package.

D.3 WCS package

The WCS package is shown in the class diagram in Figure D.3. This diagram does not show the classes used by the three operation requests and responses, which are shown (with this package) in the Get Coverage, Describe Coverage, and WCS GetCapabilities packages. This diagram also shows two used classes from the OGC Web Service package, which is common to all OGC Web Services.

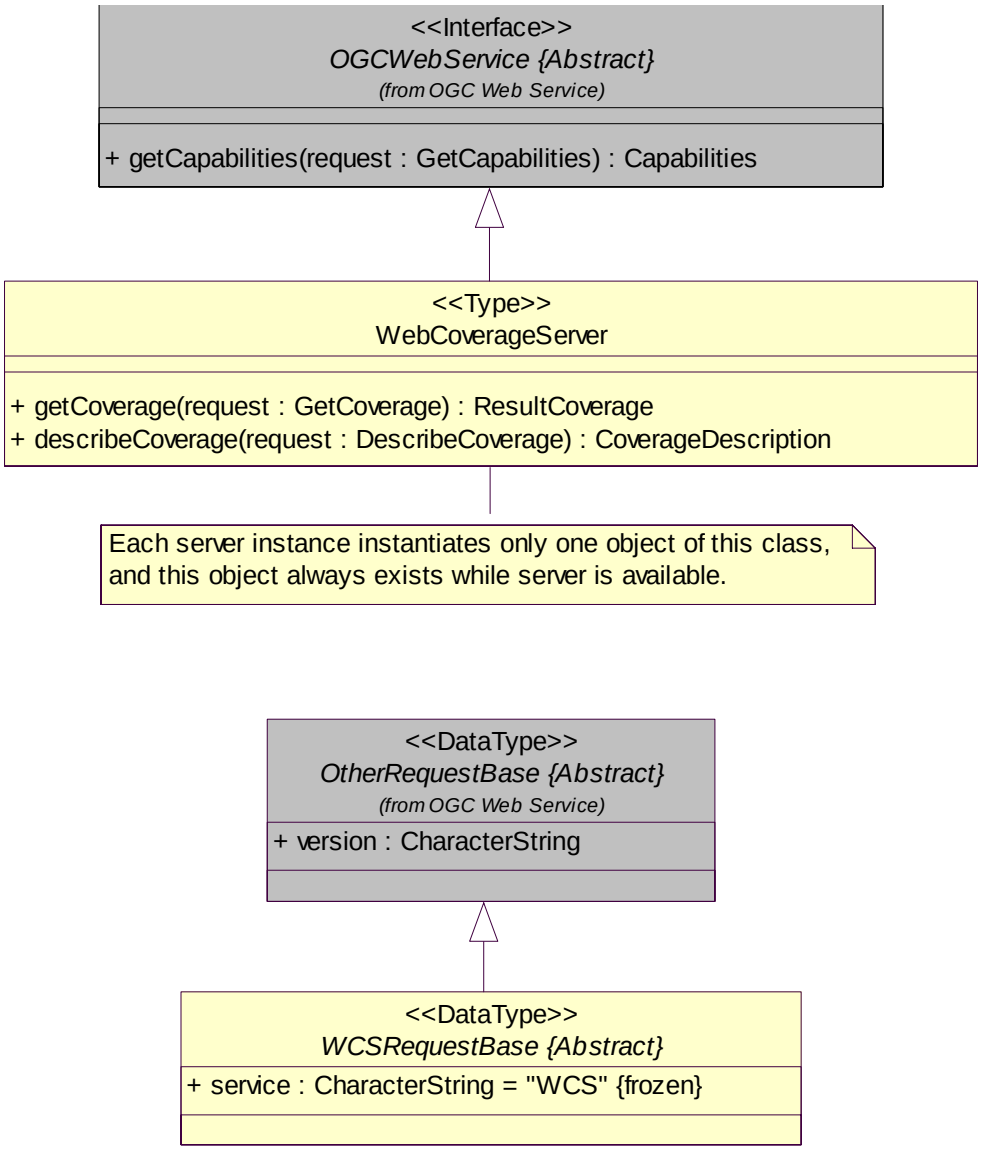

Figure D.3 — WCS package class diagram

D.4 Get Coverage package

The Get Coverage package is shown in the class diagram in Figure D.4. This diagram also shows the two classes of the WCS package plus several used classes from the OGC Web Service and WCS Values packages.

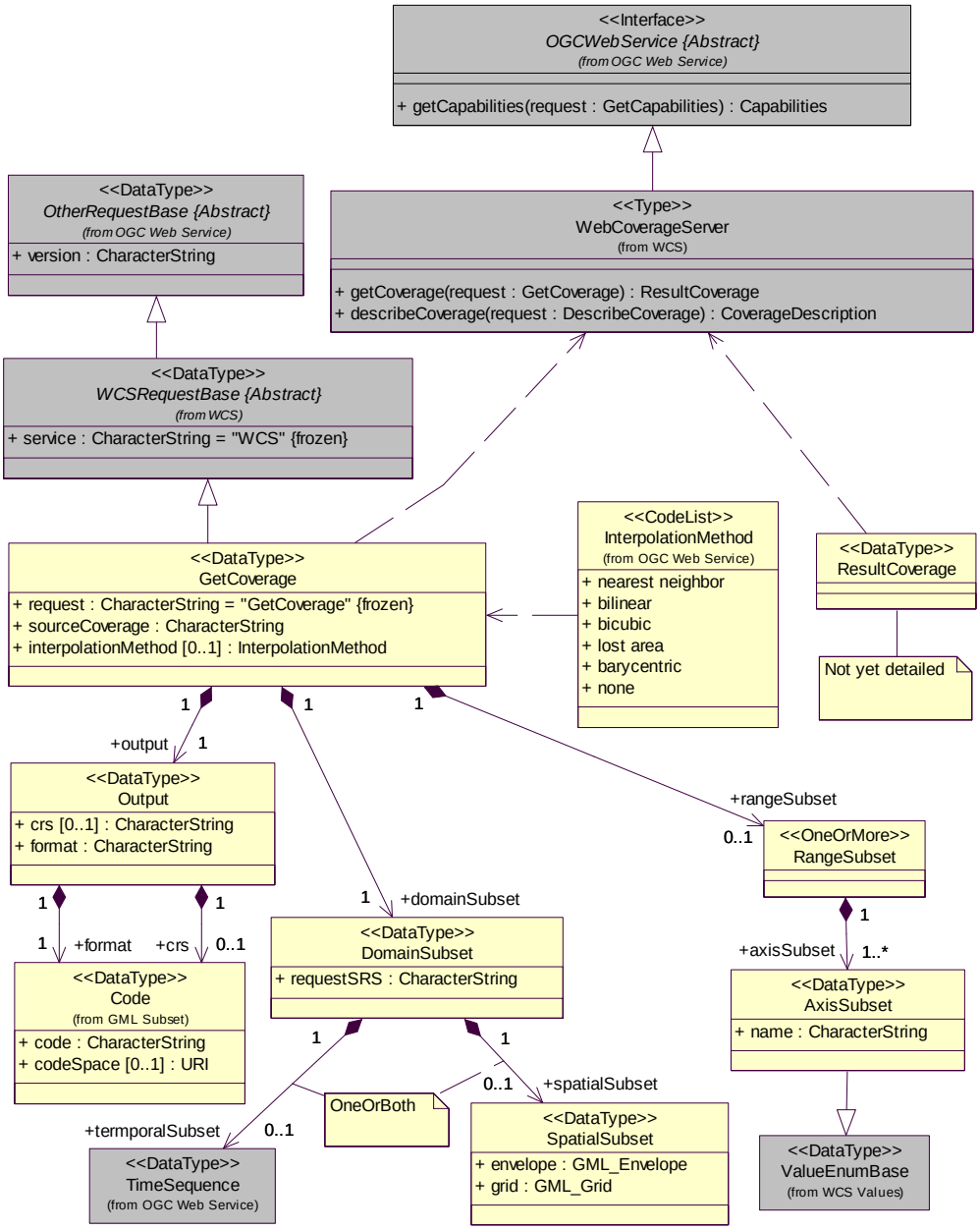

Figure D.4 —Get Coverage package class diagram

D.5 Describe Coverage package

The Describe Coverage package is shown in the class diagram in Figure D.5. This diagram does not show details of the RangeSet class, which is in the Range Set package that is detailed in the following subclause. This diagram also shows the two classes of the WCS package plus several used classes from the OGC Web Service and Content Metadata packages.

Notice that the SpatialDomain class uses three data types defined by GML, here named GML_Envelope, GML_Grid, and GML_Polygon, but not detailed in this Annex.

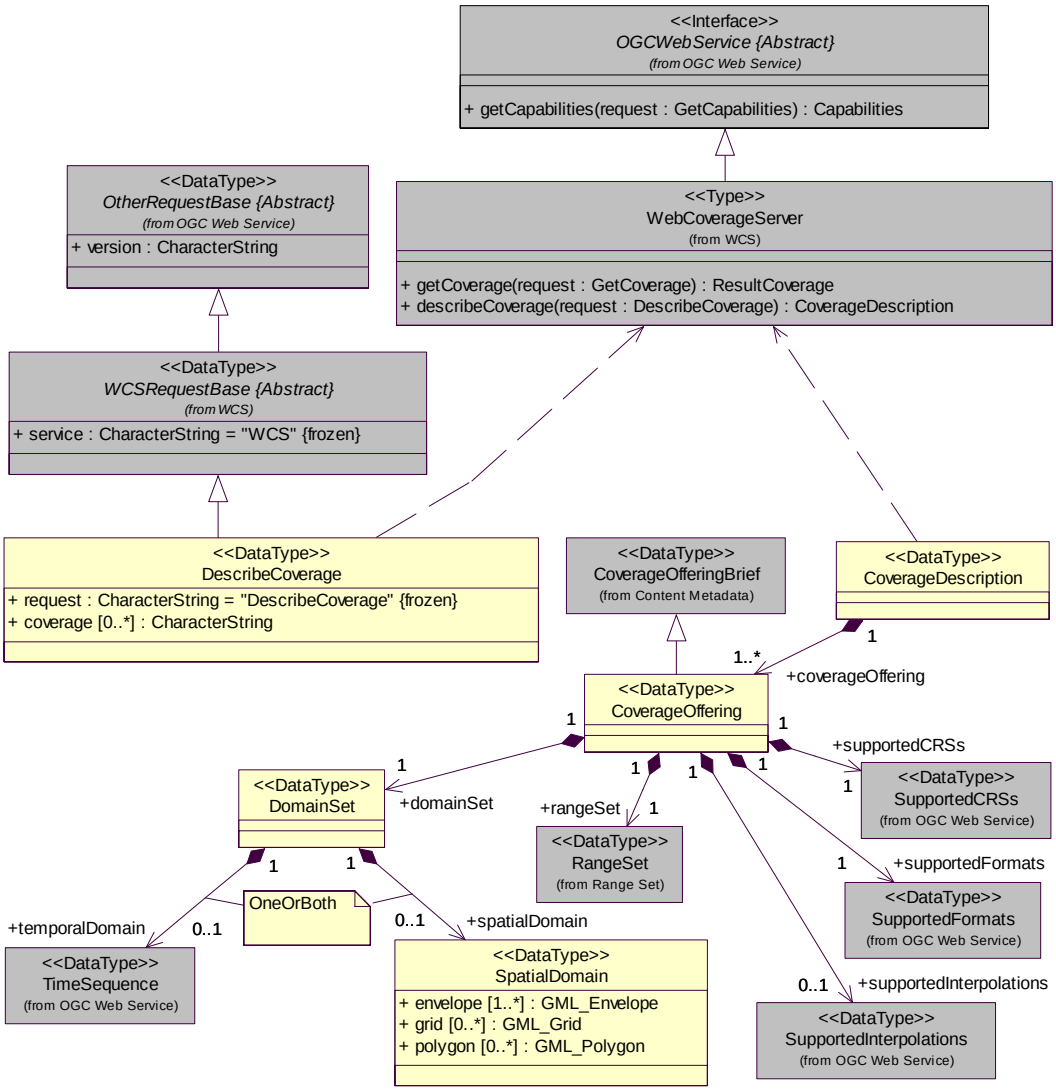

Figure D.5 — Describe Coverage package class diagram

D.6 Range Set package

The Range Set package is shown in the class diagram in Figure D.6. This diagram shows the two used classes of the WCS Values packages that is detailed in the following sub-clause, plus a used class from the Content Metadata package, detailed later.

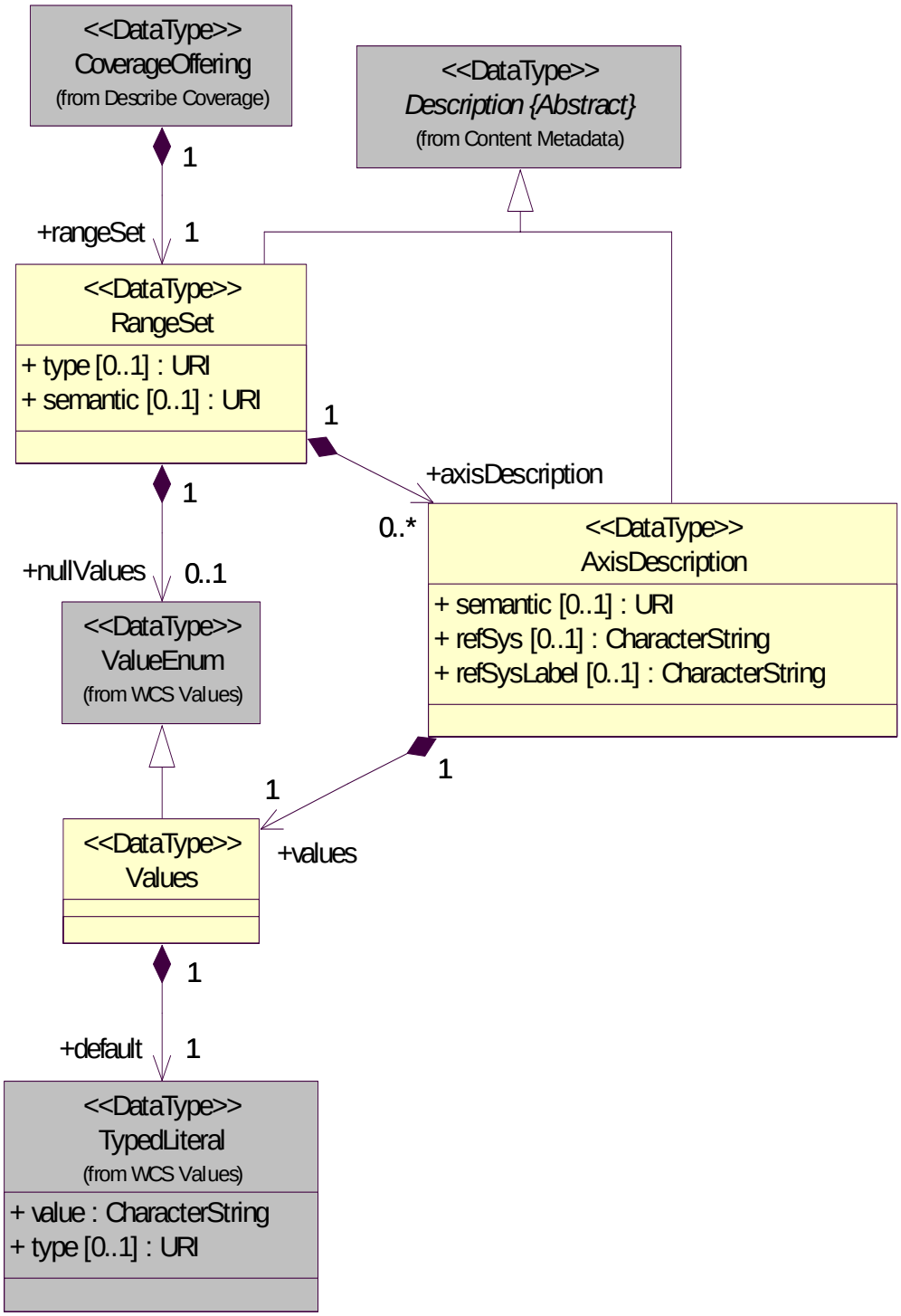

Figure D.6 — Range Set package class diagram

D.7 WCS Values package

The WCS Values package is shown in the class diagram in Figure D.7.

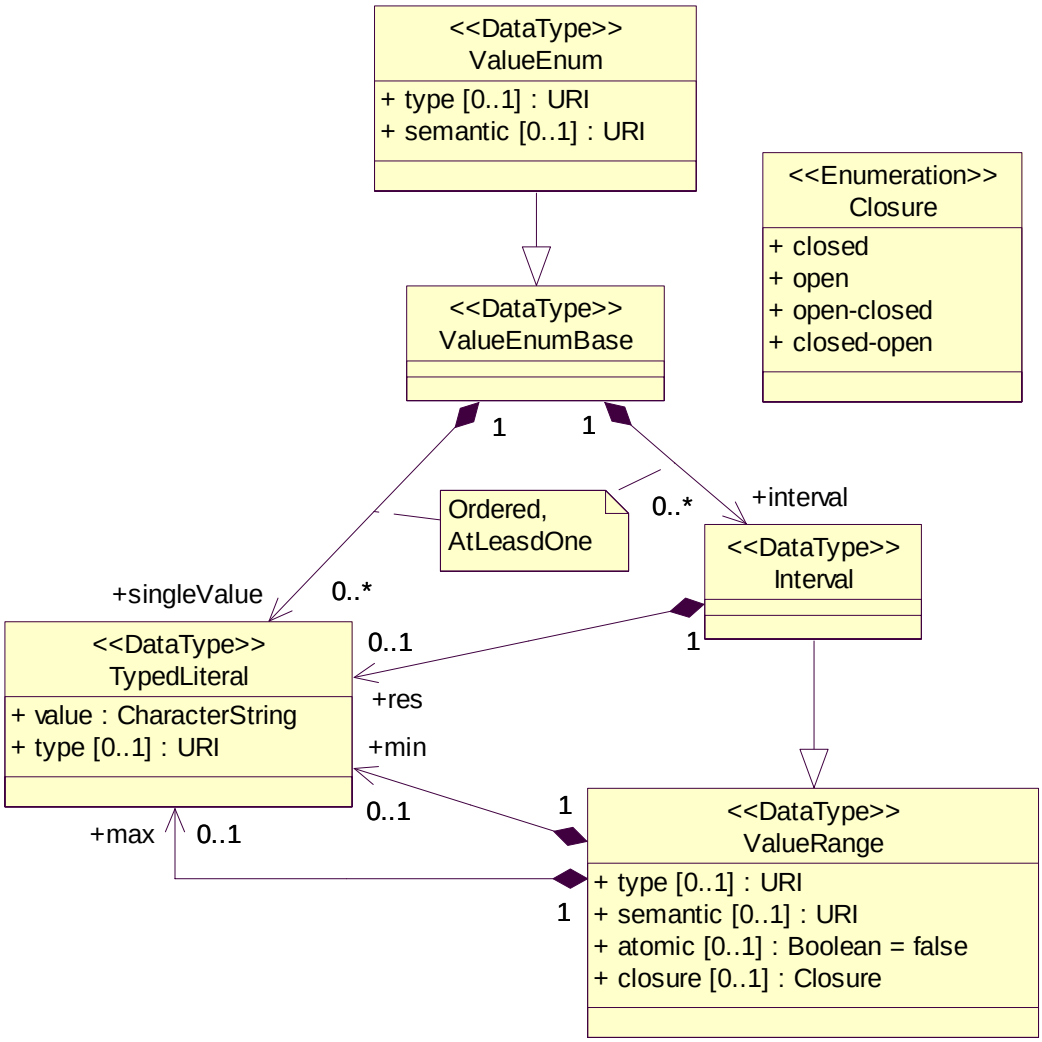

Figure D.7 — WCS Values package class diagram

D.8 WCS Get Capabilities package

The WCS Get Capabilities package is shown in the class diagram in Figure D.8. This diagram does not show details of the Service, WCSCapability, and ContentMetadata classes, which are in the Service, WCS Capability, and Content Metadata packages that are detailed in the following subclauses. This diagram also shows one class of the WCS package plus several used classes from the OGC Web Service package.

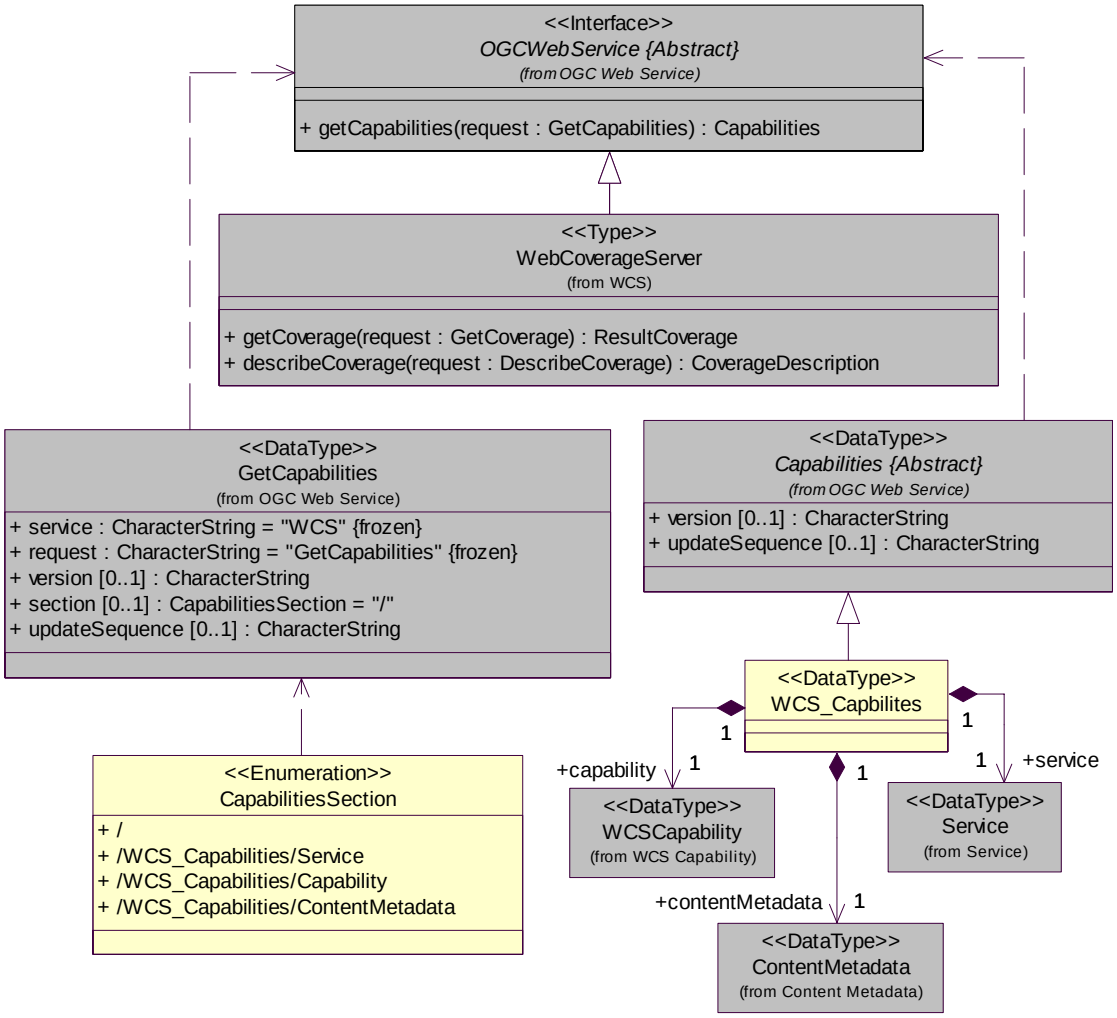

Figure D.8 — WCS Get Capabilities package class diagram

D.9 Service package

The Service package is shown in the class diagram in Figure D.9. This diagram also shows several used classes from the Content Metadata, ISO 19115 Subset, and GML Subset packages. (The ISO 19115 Subset and GML Subset packages are not detailed separately in this Annex.)

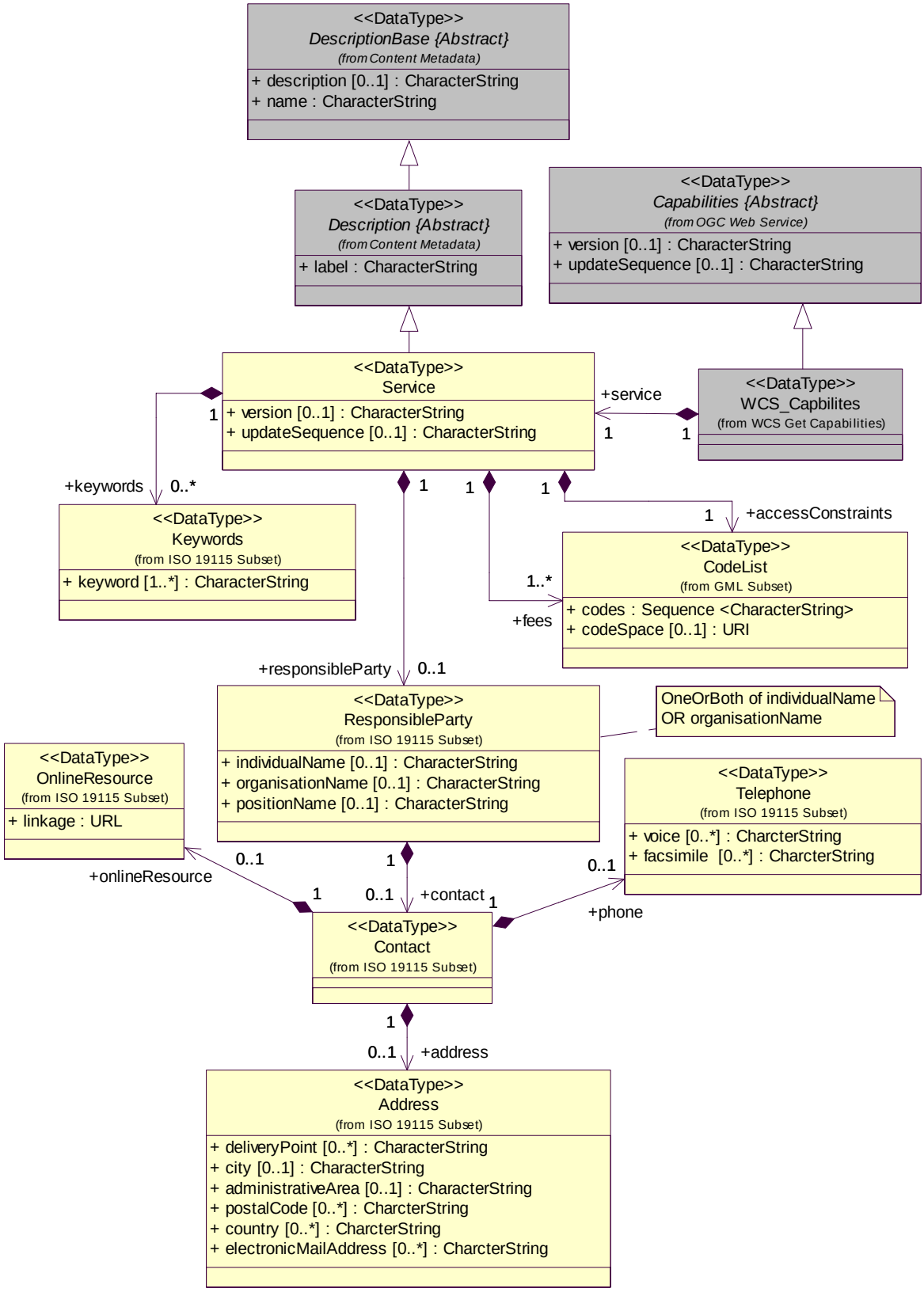

Figure D.9 — Service package class diagram

D.10 WCS Capability package

The WCS Capability package is shown in the class diagram in Figure D.10. This diagram also shows several used classes from the OGC Web Service and ISO 19115 Subset packages. (The ISO 19115 Subset package is not detailed separately in this Annex.)

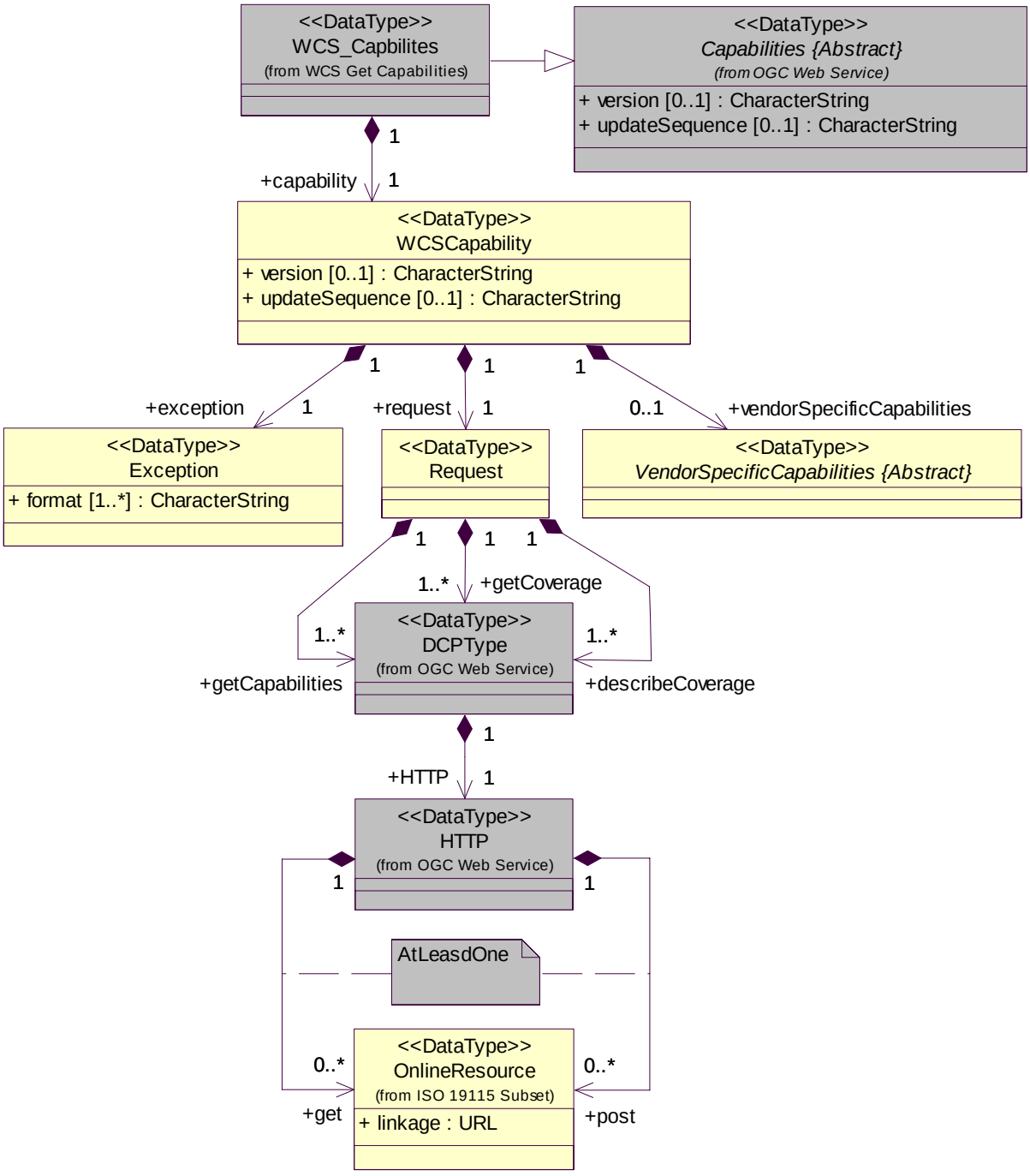

Figure D.10 — WCS Capability package class diagram

D.11 Content Metadata package

The Content Metadata package is shown in the class diagram in Figure D.11. This diagram also shows several used classes from the OGC Web Service and GML Subset packages. (The GML Subset package, with the TimePosition class in it, is not detailed separately in this Annex.)

Notice that the ContentMetadata class has an attached note which states that this class "Can reference other service providing content metadata, instead of or in addition to including CoverageOfferingBrief objects". This other service can be a catalog service. The association to the CoverageOfferingBrief class with the coverageOfferingBrief role is thus modeled as an aggregation (instead of a composition) association, since the equivalents of CoverageOfferingBrief objects can exist outside of ContentMetadata objects.

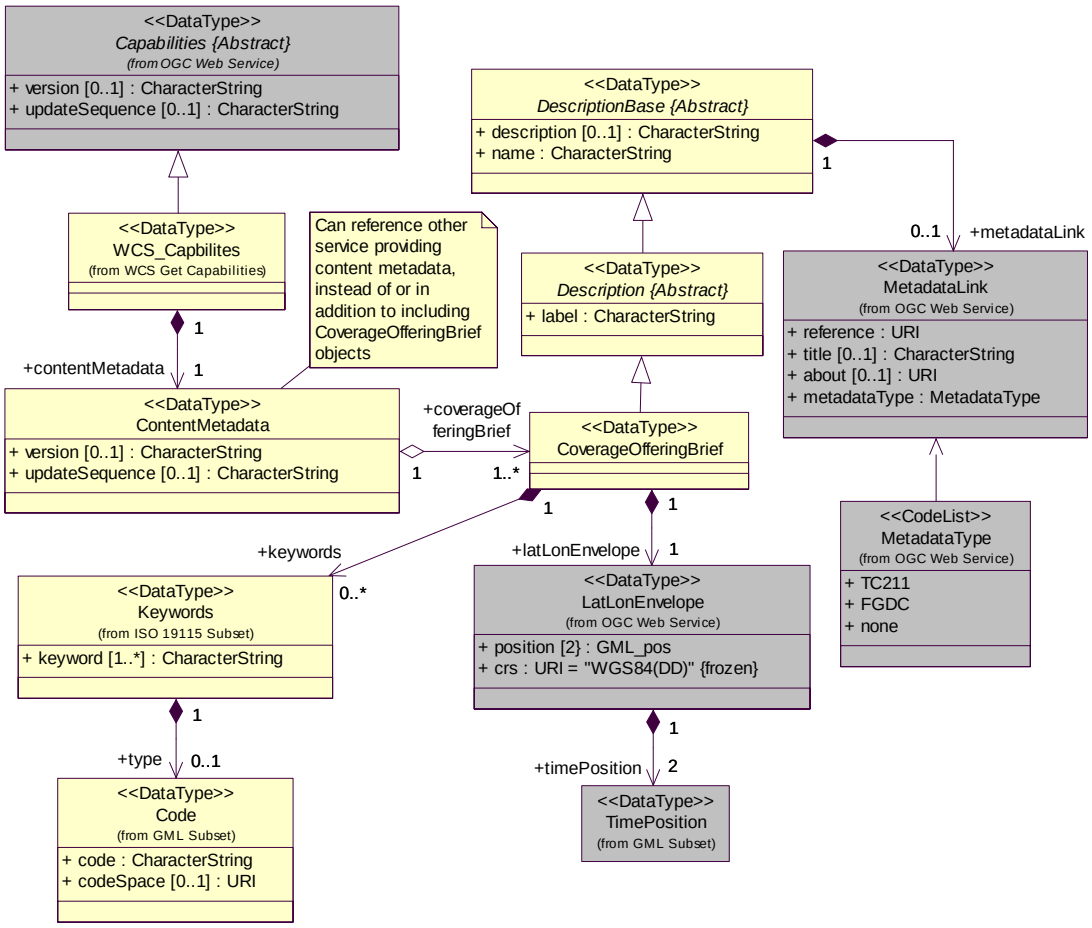

Figure D.11 — Content Metadata package class diagram

D.12 OGC Web Service package

The OGC Web Service package is shown in the class diagrams in Figures D.12 and D.13. These diagrams also show several used classes from the ISO 19115 Subset and GML Subset packages. (The ISO 19115 Subset and GML Subset packages are not detailed separately in this Annex. Also, the TimePosition and Timeduration classes in the GML Subset package are not detailed in these class diagrams.)

As shown, the OGC Web Service package contains several un-connected parts, which are separately used by the various WCS packages defined in the preceding subclauses. Notice that the LatLonEnvelope class uses a type from GML 3, here named GML_pos but not detailed in this Annex.

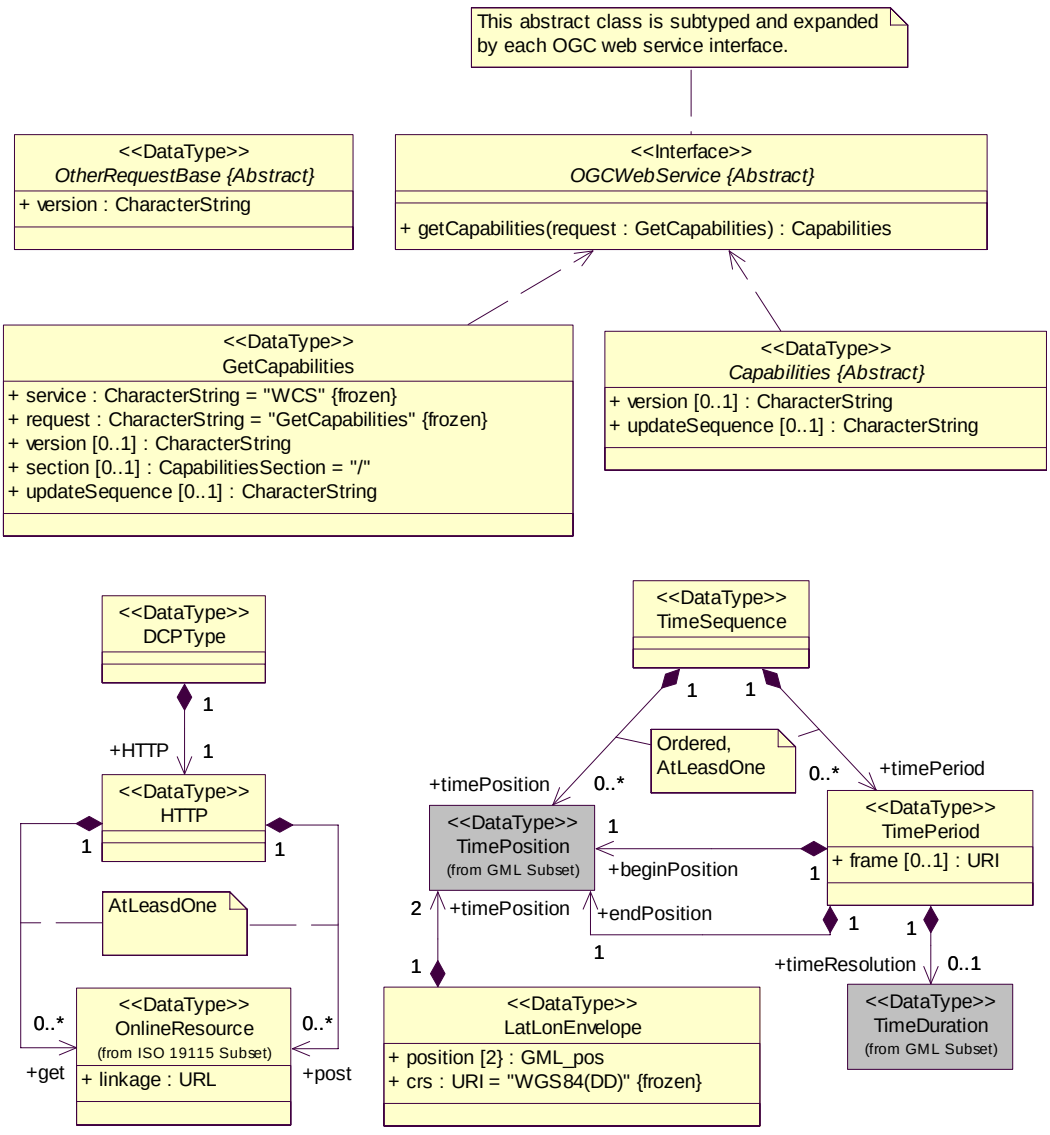

Figure D.12 — OGC Web Service package class diagram, page 1

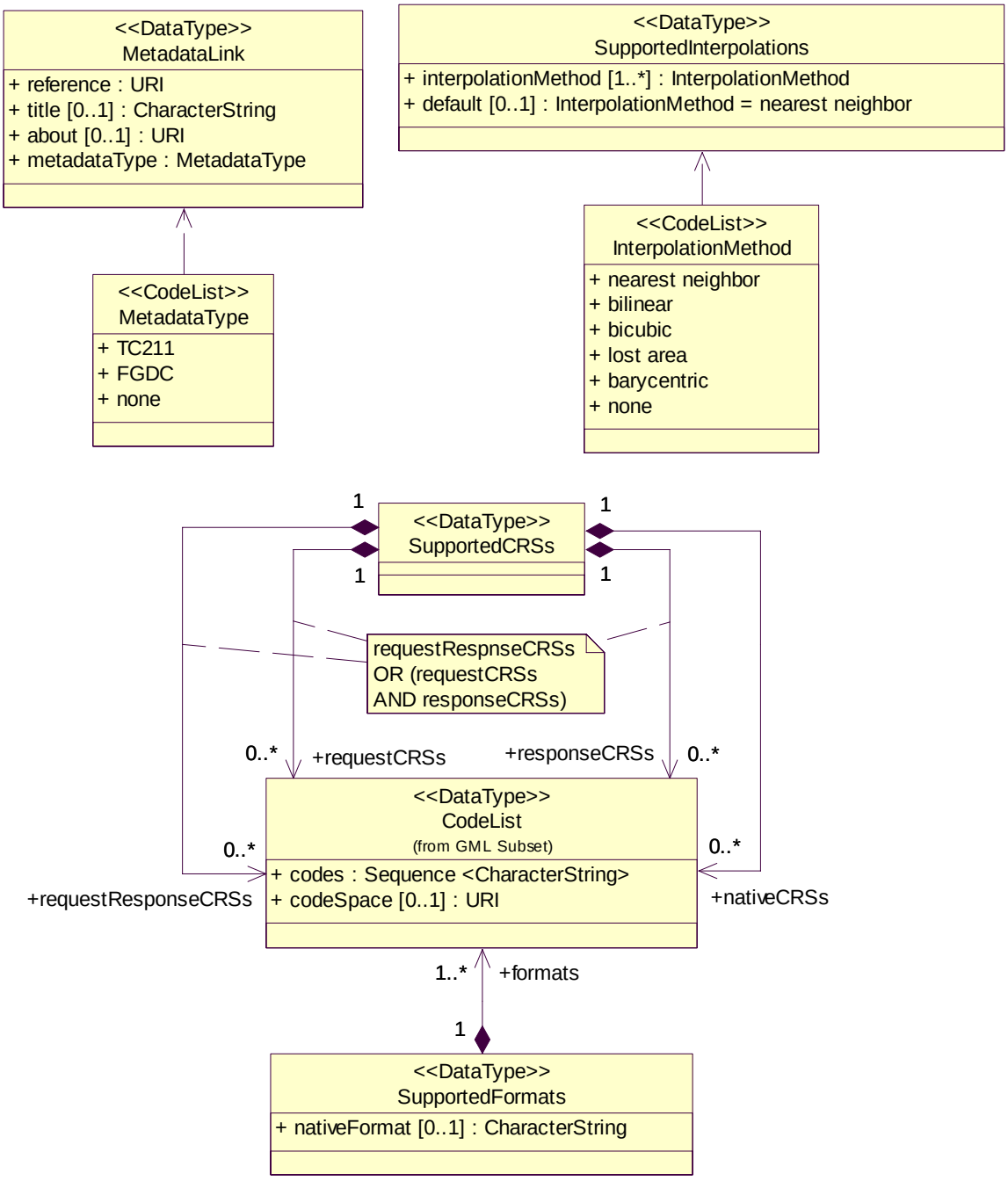

Figure D.13 — OGC Web Service package class diagram, page 2

Bibliography

[1] OGC 00-014r1, Guidelines for Successful OGC Interface Specifications

[2] ISO 19103, Geographic Information – Conceptual schema language

[3] OMG Unified Modeling Language Specification (UML), Version 1.5, March 2003, http://www.omg.org/docs/formal/03-03-01.pdf