Open Geospatial Consortium Inc.

Date: 2010-01-20

Reference number of this document: OGC 09-104r1

Version: 0.4.0

Category: OpenGIS® Discussion Paper

Editors: Arne Schilling, Thomas H. Kolbe

Draft for Candidate OpenGIS® Web 3D Service Interface Standard

Warning

This document is not an OGC Standard. This document is an OGC Discussion Paper and is therefore not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, an OGC Discussion Paper should not be referenced as required or mandatory technology in procurements.

Document type: OpenGIS® Discussion Paper
Document subtype: Approved
Document stage: Discussion Paper version 0.4.0
Document language: English

Contents

Page

Figures

Page

Tables

Page

i. Preface

This document contains significant modifications and extensions to the original discussion paper OGC 05-019 and changes some parameter names to harmonize with the current OWS Common Standard [OGC 06-123r1].

Although this candidate specification defines a stand-alone web service for the 3D visualization of geographic data, the document strongly references OGC standards. This draft is based on the OWS Common Implementation Standard [OGC 06-121r3]. Concepts and definitions are derived from other OGC Implementation Standards.

Suggested additions, changes, and comments on this draft version are welcome and encouraged. Such suggestions may be submitted by email message or by making suggested changes in a well commented edited copy of this document.

ii. Document terms and definitions

This document uses the standard terms defined in Subclause 5.3 of [OGC 06-121r3], which is based on the ISO/IEC Directives, Part 2. Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this standard.

iii. Document contributor contact points

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

Name	Organization
Arne Schilling	University of Bonn, Department of Geography
Thomas H. Kolbe	Berlin University of Technology
Alexander Zipf	University of Heidelberg, Department of Geography
Steffen Neubauer	University of Bonn, Department of Geography

iv. Revision history

Date	Release	Editor	Primary clauses modified	Description
2003-10-25	0.1.0	Thomas H. Kolbe		First internal SIG 3D version
2004-07-03	0.2.0	Thomas H.		Mapping coordinate axes from real world to

		Kolbe		computer graphics GetScene: BBOX becomes mandatory GetScene: scene's displacement to local coordinates GetScene: new parameters TRANSFORM, ENVIRONMENT, MINHEIGHT, MAXHEIGHT
2004-12-15	0.3.0	Udo Quadt		Complete restructuring of the document using OGC template English translation
2009-05-15	0.4.0	Arne Schilling	all	Major revision. Document now based on OWS Common OGC 06-121r3. Added Operations GetFeatureInfo, GetLayerInfo, and GetTile. Added LOD handling to GetScene operation. Changed default 3D format from VRML to X3D.
2010-01-18	0.4.0	Carl Reed	Various	Prepare for publication as a DP

v. Changes to the OGC Abstract Specification

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

vi. Future work

Improvements to this document are desirable to improve the acceptance in the industry and scientific community.

Styled Layer Descriptors

The concept of Styled Layer Descriptors is described in the OGC Styled Layer Descriptor Implementation Specification. It enables the user to control the graphical style of 3D graphic elements. The current OGC Symbology Encoding has limited capability to support the requirements for 3D visualization. Therefore we need some extensions to the SLD standard for describing Materials, surfaces, 3D Icons etc. (SLD3D). These extensions could be either integrated into the current standard or implemented as an optional extension within a different namespace. The user styles can be submitted within a POST request.

Add SOAP Interface

A SOAP interface may be desirable, but it is not yet described in the OWS Common Implementation Standard. As soon as the OGC decides on a common framework for adding SOAP interfaces, this may be incorporated into this document.

Misc.

Explanation and use of geo extensions

Foreword

This version of the draft standard replaces all previous versions.

Due to the similarity of concepts, this document is partly derived from the Web Map Service Implementation Specification Version 1.3.0 [OGC 06-042] and shares many elements in the GetCapabilities response.

This document includes 4 annexes; Annexes A and B are normative, and Annexes C and D are informative.

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

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

Introduction

A Web 3D Service (W3DS) is a portrayal service for three-dimensional geodata, such as landscape models, city models, textured building models, vegetation objects, and street furniture. Geodata is delivered as scenes that are comprised of display elements, optimized for efficient real time rendering at high frame rates. 3D Scenes can be interactively displayed and explored by internet browsers with 3D plugins, or loaded into virtual globe applications. A W3DS is capable of handling data sets of a wide range of scales, from full globes down to smaller immobile objects such as lanterns which are still of geographic relevance. It can handle data sets consisting of multiple Levels of Detail for each object, thereby greatly increasing performance without sacrificing quality.

The representation of geographic objects may range from very detailed and textured models, to prototypic and generic models used for same types of objects, to abstract boxes or symbols. Scenes are retrieved by queries defining the geographic area, information layers, styles, and further parameters, very similar to the Web Map Service (WMS) interface. The formats used for encoding 3D scenes are designed for limited bandwidth networks like the internet and realistic and efficient real time rendering. They avoid overhead produced by e.g. XML tags, and exploit graphics technologies like display lists, re-using scene graph nodes by defining links, indexed geometry arrays, and real time shaders. Due to the formats used to deliver content, the W3DS is not restricted to static objects; it can also include animations and other visual effects, as well as pre-defined behaviors triggered by user interactions.

The role of a W3DS is similar to the role of a WMS. It does not necessarily provide the original geodata but a visual representation (view) of it. This view is called a “Scene” and consists of display elements representing the geometry, appearance, and behavior of geographic objects. It does not contain attributes and semantic information. In contrast to the WMS, the output of a W3DS is not images, but scene graphs consisting of a tree like structure of nodes, groups, transforms, shapes, materials, and geometries. Since attributes are not part of the scene graph, they must be accessed by additional service operations. The W3DS GetFeatureInfo operation is responsible for accessing additional information on objects and functions equivalent to the WMS.

The W3DS can have its own data store or it can retrieve all of the required data using an implementation instance of the OGC Web Feature Service interface standard (WFS), forwarding and processing all data on request. Another possibility is to locally cache the content provided by a WFS instance and then synchronize all data in order to save network bandwidth and processing time during the handling of requests, thus reducing the response times. In the case of using a WFS instance to access an original data store, we assume that the WFS instance supports the ability to provide an OGC CityGML encoding as the response encoding payload. Otherwise, clients would not benefit from an intermediate 3D portrayal service. In such a setup, the WFS implementation(s) play the role of a central data repository, that is maintained and updated by the owner of the data.

The W3DS then acts as middleware enabling efficient visualization and streaming to the interactive client.

The OGC Geography Markup Language (GML) standard already supports many features for describing three-dimensional objects. The GML information model is derived from the ISO 19100 series of international standards and contains elements for describing surfaces, solids, and aggregate types. CityGML adds semantic and structure to the GML schema and defines very precisely how elements of the urban environment can be modeled in a logical and interchangeable way. CityGML is defined as an application profile of GML and contains a spatial model, an appearance model, and a thematic model for various types of real world objects. In contrast to the usual 2D vector formats (e.g. SVG), CityGML can represent the geometrical properties of objects in a level of detail that is even sufficient for Virtual Reality applications. However, this specification draft was developed in order to leverage the usage of another set of standards that focus on efficient real time visualization techniques and 3D cartographic features. These include first and foremost X3D, which is a W3C standard, and KML which is an OGC standard. The advantage of using visualization centric exchange formats is on the one hand the more compact encoding allowing a higher throughput in limited bandwidth networks and on the other hand the support of sophisticated appearance, shading, behavior, and animation models. The W3DS shall support X3D as mandatory format.

OpenGIS® Web 3D Service Draft Implementation Standard

1 Scope

This OGC™ document specifies a new draft standard for web services delivering 3D Scenes. Scenes are collections of graphical 3D elements that can be explored interactively by the user.

This OGC™ document is applicable to servers with the purpose of publishing large and potentially very detailed city and landscape models in a distributed and heterogeneous environment.

2 Compliance

Compliance with this standard shall be checked using all the relevant tests specified in Annex A (normative).

3 Normative references

The following normative documents contain provisions that, through reference in this text, constitute provisions of this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

ISO 19105:2000, *Geographic information — Conformance and Testing*

OGC 06-121r3, *OpenGIS® Web Services Common Specification*, February 2007.

NOTE This OWS Common Specification contains a list of normative references that are also applicable to this Implementation Specification.

OGC 04-046r3, *The OpenGIS Abstract Specification, Topic 2: Spatial Referencing by Coordinates*, August 2004

OGC 08-007r1, *OpenGIS® City Geography Markup Language (CityGML) Encoding Standard*, August 2008.

OGC 04-095, *OpenGIS® Filter Encoding Implementation Specification*, May 2005.

OGC 05-077r4, *Symbology Encoding Implementation Specification*, July 2006.

OGC 05-078r4, *Styled Layer Descriptor profile of the Web Map Service Implementation Specification*, June 2007.

OGC 06-042, *OpenGIS® Web Map Server Implementation Specification*, March 2006.

ISO 8601:2004(E), *Data elements and interchange formats - Information interchange - Representation of dates and times,* December 2004.

ISO/IEC 19775, *Information technology — Computer graphics and image processing — Extensible 3D (X3D), Architecture and base components,* *http://www.web3d.org/x3d/specifications/*

ISO/IEC 19775, *Information technology — Computer graphics and image processing — Extensible 3D (X3D),* E*ncoding, Part1 – Part3, http://www.web3d.org/x3d/specifications/*

EPSG, *European Petroleum Survey Group Geodesy Parameter*s, Lott, R., Ravanas, B., Cain, J., Girbig, J.-P., and Nicolai, R., eds., http://www.epsg.org/

IETF RFC 2045 (November 1996), *Multipurpose Internet Mail Extensions (MIME) PartOne: Format of Internet Message Bodie*s, 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 Synta*x, Berners-Lee, T., Fielding, N., and Masinter, L., eds., http://www.ietf.org/rfc/rfc2396.txt

Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation 26 November 2008, http://www.w3.org/TR/xml/

In addition to this document, this standard includes several normative XML Schema Document files as specified in Annex B.

4 Terms and definitions

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

4.1
coordinate reference system
coordinate system that is related to the real world by a datum [ISO 19111]

4.2
coordinate system
set of mathematical rules for specifying how coordinates are to be assigned to points [ISO 19111]

4.3
layer
basic unit of geographic information that may be requested as a map from a server

4.4
map
portrayal of geographic information as a digital image file suitable for display on a computer screen

4.5
portrayal
presentation of information to humans [ISO 19117]

4.6
scene
Collection of graphical elements including geometries, materials, textures, behaviors, animations, lights, and viewpoints

5 Conventions

5.1 Abbreviated terms

Most of the abbreviated terms listed in Subclause 5.1 of the OWS Common Implementation Specification [OGC 06-121r3] apply to this document, plus the following abbreviated terms.

API	Application Program Interface
COM	Component Object Model
COTS	Commercial Off The Shelf
EPSG	European Petroleum Survey Group
GDI	Geodata infrastructure
GeoX3D	X3D with a extension for 3D geodata
GML	Geography Markup Language
HTTP	Hypertext Transfer Protocol
IDL	Interface Definition Language
IETF	Internet Engineering Task Force
LOD	Level of Detail
MIME	Multipurpose Internet Mail Extensions
NRW	North-Rhine Westphalia
POC	Point Of Camera

POI	Point Of Interest
RFC	Request for Comments
SIG	Special Interest Group
SLD	Styled Layer Descriptor
URL	Uniform Resource Locator
UTC	Universal Time Coordinated

5.2 UML notation

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

5.3 Used parts of other documents

This document uses significant parts of document [OGC 06-121r3]. To reduce the need to refer to that document, this document copies some of those parts with small modifications. To indicate those parts to readers of this document, the largely copied parts are shown with a light grey background (15%).

5.4 Platform-neutral and platform-specific standards

As specified in Clause 10 of OGC Abstract Specification Topic 12 “OpenGIS Service Architecture” (which contains ISO 19119), this document includes both Distributed Computing Platform-neutral and platform-specific standards. This document first specifies each operation request and response in platform-neutral fashion. This is done using a table for each data structure, which lists and defines the parameters and other data structures contained. These tables serve as data dictionaries for the UML model in Annex C, and thus specify the UML model data type and multiplicity of each listed item.

EXAMPLES 1 Platform-neutral standards are contained in Subclauses 8.2.1, and 9.2.1.

The specified platform-neutral data could be encoded in many alternative ways, each appropriate to one or more specific DCPs. This document now specifies encoding appropriate for use of HTTP GET transfer of operations requests (using KVP encoding), and for use of HTTP POST transfer of operations requests (using XML or KVP encoding). However, the same operation requests and responses (and other data) could be encoded for other specific computing platforms, including SOAP/WSDL.

EXAMPLES 2 Platform-specific standards for KVP encoding are contained in Subclauses 8.2.2, and 9.2.2.

EXAMPLES 3 Platform-specific standards for XML encoding are contained in Subclauses 8.2.3, and 9.2.3.

5.5 Data dictionary tables

The UML model data dictionary is specified herein in a series of tables. The contents of the columns in these tables are described in Table 1.

Table 1 — Contents of data dictionary tables

Column title	Column contents
Names (left column)	Two names for each included parameter or association (or data structure). The first name is the UML model attribute or association role name. The second name uses the XML encoding capitalization specified in Subclause 11.6.2 of [OGC 06-121r3]. The name capitalization rules used are specified in Subclause 11.6.2 of [OGC 06-121r3]. Some names in the tables may appear to contain spaces, but no names contain spaces.
Definition (second column)	Specifies the definition of this parameter (omitting un-necessary words such as "a", "the", and "is"). If the parameter value is the identifier of something, not a description or definition, the definition of this parameter should read something like "Identifier of TBD".
Data type and value (third column) or Data type (if are no second items are included in rows of table)	Normally contains two items: The mandatory first item is often the data type used for this parameter, using data types appropriate in a UML model, in which this parameter is a named attribute of a UML class. Alternately, the first item can identify the data structure (or class) referenced by this association, and references a separate table used to specify the contents of that class (or data structure). The optional second item in the third column of each table should indicate the source of values for this parameter, the alternative values, or other value information, unless the values are quite clear from other listed information.
Multiplicity and use (right or fourth column) or Multiplicity (if are no second items are included in rows of table)	Normally contains two items: The mandatory first item specifies the multiplicity and optionality of this parameter in this data structure, either "One (mandatory)", "One or more (mandatory)", "Zero or one (optional)", or "Zero or more (optional)". The second item in the right column of each table should specify how any multiplicity other than "One (mandatory)" shall be used. If that parameter is optional, under what condition(s) shall that parameter be included or not included? If that parameter can be repeated, for what is that parameter repeated?

When the data type used for this parameter, in the third column of such a table, is an enumeration or code list, all the values specified by a specific OWS shall be listed, together with the meaning of each value. When this information is extensive, these values and meanings should be specified in a separate table that is referenced in the third column of this table row.

The data type of many parameters, in the third table column, is specified as "Character String type, not empty". In the XML Schema Documents specified herein, these parameters are encoded with the xsd:string type, which does NOT require that these strings not be empty.

The contents of these data dictionary tables are normative, including any table footnotes.

6 W3DS overview

A Web 3D Service (W3DS) creates 3D scenes of landscape and city models that can be explored interactively on the client. It delivers graphical elements for displaying a

complete 3D map or parts of it. The client, which is equipped with modern 3D graphics acceleration hardware, can decide how to visualize and explore the scene and is not confined to certain viewpoints (like e.g. in panoramic images). The W3DS is suitable for a Medium Server Medium Client concept, which means that the Server collects the necessary geo data, and generates display elements which are streamed to the Client. The Client is responsible for rendering the display elements on the screen using the rendering techniques of his choice.

The W3DS draft standard is designed as a Portrayal Service. It does not provide the raw geo data but a 3D representation of the data. The difference is that the geodata itself is organized in features and object with additional attributes, meta data, and semantics, and the result of a Portrayal Service is something that can be viewed in real time. There is no guarantee that the internal structure of the resulting scenes and related attribute data is not typically missing due to the current 3D internet formats lacking support for some structures and semantics (e.g. X3D, COLLADA). It is even advisable to re-organize the scene graph structure for a more efficient rendering. For retrieving fully GML compatible and attribute rich geo data, a WFS should be used. The advantage of using visualization-centric formats is that they support a wide range of features for controlling the visual appearance (e.g. textures, surface properties, animations, lighting, atmosphere) and that they can be more efficiently transmitted and encoded.

6.1 Definition of Scene

Geodata is delivered as Scenes that are comprised of display elements, optimized for efficient real time rendering at high frame rates. 3D Scenes can be interactively displayed and explored by internet browsers with 3D plugins, or loaded into virtual globe applications.

The following items try to provide a definition of the term Scene as used in this document

a) A Scene is composed of data from one or multiple layers.
b) A Scene may also contain map elements (title, compass, scale bar, legend etc.). With all these elements the (although not existing) definition of a “3D map” could be fulfilled. It can also contain predefined viewpoints.
c) A Scene must be provided in a CRS that can be used for visualization. Graphics hardware is designed for handling single precision coordinates, which is often not sufficient for global CRSs like WGS-84 and UTM. Also, coordinate values must be provided in Meters or Inches, not angular values since normal graphics APIs cannot handle polar CRSs. If WGS-84 must be used, then the “geo”-extensions (e.g. Geo-X3D) should be implemented.
d) A Scene is composed of “Display Elements” (geometries, triangles, materials, animations, lights, fog).
e) Scenes are usually represented as scene graphs in modern 3D graphics APIs (e.g. OpenSG, Java3D) in the memory. Scene graphs define the hierarchy and structure of the Scene.
f) The structure of a Scene is not necessarily the same as in the underlying data repository. Parts of the Scene may be restructured in order to reduce the number

of geometries, hierarchy levels, materials, and textures and thus accelerate the rendering of display elements.

g) Semantics is usually missing since the various visualization languages (X3D, KML) do not support semantics.

6.2 Free space versus tiled space

The usual way to access map data is by defining a rectangular bounding box which is used as spatial selection filter. The bounding box can be of arbitrary size which allows very flexible client configurations. This capability is provided by the GetScene operation, which contains parameters for selecting layers, styles, time, CRS, LODs, and a rectangular, axis parallel bounding box.

In many cases it is useful to spatially partition all data in a layer into smaller chunks of data which can be prepared prior to be served by the W3DS server. This applies mostly for terrain data or other data representing 2,5D surfaces. Like in image pyramids, surface data can be made available as a set of adjacent tiles aligned on a grid or raster. For this kind of data, the preferred way of accessing individual tiles is not by defining a bounding box as spatial selection filter, but by using row and column indices referring to the position within the grid.

The W3DS includes a GetTile operation which facilitates the access to tiled data. A TileSet definition provides information on the spatial alignment of the tiles in a hierarchical grid structure (Figure 1).

The general use case for the GetTile operation is that a smart client is dynamically assembling the displayed scene graph from multiple tiles that are downloaded from a W3DS server. Tiles may be available in multiple sizes and resolutions or accuracies, referring for instance to the triangle density of the data. Tile sizes are related by powers of two and share the same origin. This ensures that tiles of several levels seamlessly fit together in a multi-resolution scene, which is the preferred way to achieve perspective views on complex landscapes.

The concept of multiple tile levels is loosely coupled to the concept of Levels of Detail (LODs). In this document the concept of LOD refers to the presence of multiple representations for a single object. Aggregations of objects are also allowed for lower LODs. This is also the concept used in CityGML. Tile levels refer to the strictly hierarchical organization of spatial subsets of the data of a layer. Dividing a rectangular tile into four quarters defines the next higher level. This higher level shall contain data of higher accuracy, or triangle count. The relation of the data amount (features, textures, triangles etc.) between tile levels should be approximately 1 to 4, to be consistent with already established tiling schemas and image pyramids.

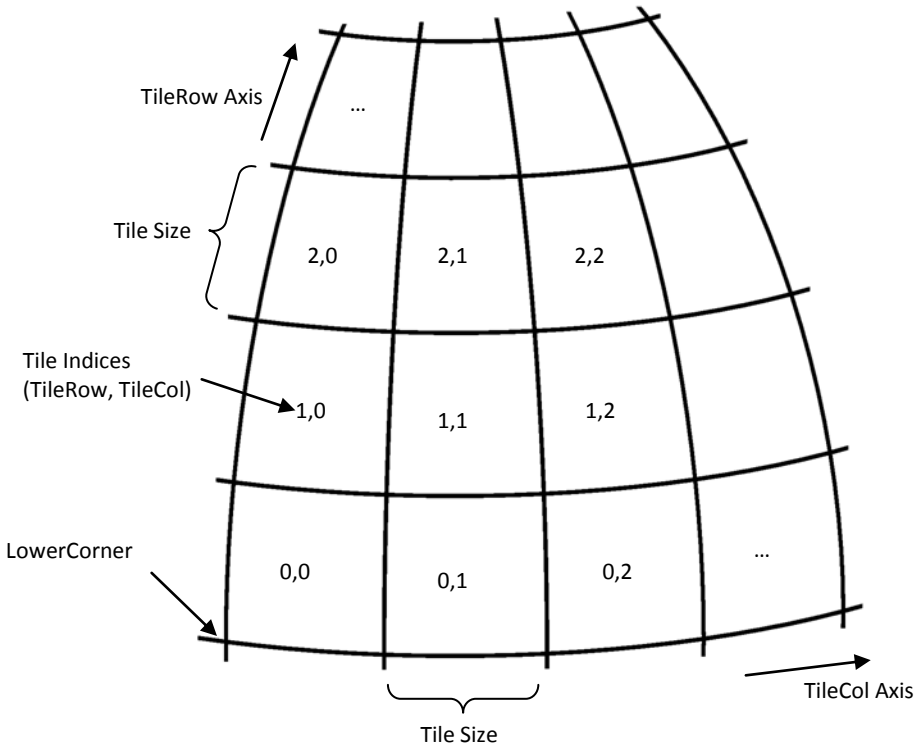

Figure 1: A tile level in the TileSet definition can be described as grid with origin at LowerCorner.

6.3 Geo-Visualization Pipeline

Regarding the portrayal of spatial data, the OGC employs a four level visualization pipeline, which describes visualization as a multi-level process starting from non-graphical object representations stored in a repository (e.g. a database) and ending with the final presentation of graphical entities on a display device (see Doyle and Cuthbert 1998 [1]). The first level in this pipeline - the data repository – contains all the spatial data, attributes, and relationships between entities in a form that can be used for various applications. TIn a selection step the objects of interest are retrieved. The second step transforms the selected spatial objects to a graphical representation, i.e. the spatial objects are mapped to display elements. In the third step the generated display elements are rendered to an image, which in the final step is displayed to the user by an appropriate output device.

The components of the portrayal pipeline don't have to reside on the same system; they can be distributed over the internet. However, in client-server applications the lower level components are typically installed on one or more servers while the remaining visualization tasks are handled by the client. According to their complexity clients are classified into thick, medium, and thin clients (Figure 2). Thick clients communicate on the feature level with the server. The advantage is that the client is free to realize any – including very complex - interaction schemes and analytic functions. The drawback, at least regarding web applications, is the high computing, memory, and bandwidth

requirements for processing GML 3D data. Medium clients download display elements that can be easily processed by the graphics hardware. Geometries usually consist of triangle arrays, indexed triangles, or "stripified" triangles. 3D browser plug-ins are available for X3D. KML can be used by virtual globe applications. Thin clients only have to cope with rendered images. Hence they are very easy to implement and require no special hardware. In this scheme, a perspective view service could be used for rendering images based on the camera position and viewing direction as specified in the request. Interaction on the client side is strongly limited but on the other side, sophisticated rendering techniques including ray tracing, ray casting, caustics etc. can be used.

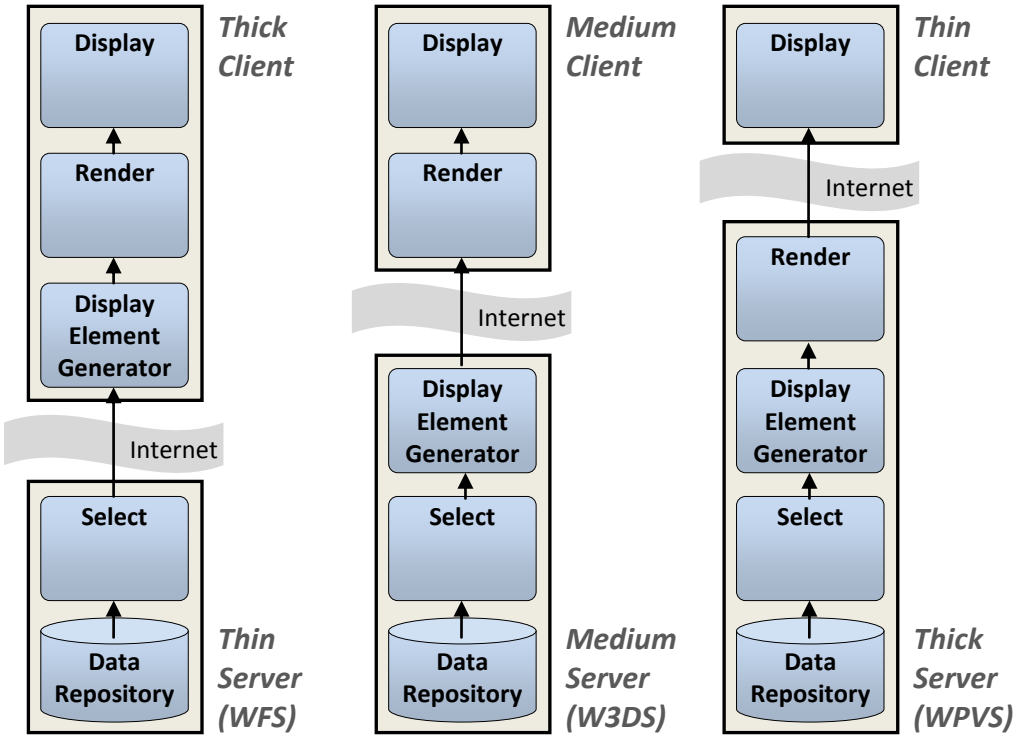

Figure 2 — Different balancing schemes between client and server

The decision as to which client model is appropriate for what application depends on the specific application scenario on the one hand and the availability of appropriate data and services on the other hand. If 3D visualization plays only a minor role (for example, when a 3D view of a sports arena has to be shown in an online ticket service application) a thin client should be used. If the application focus is instead on interactive exploration, at least a medium client has to be used. By focusing on medium clients, the W3DS fills the gap between thin and thick servers (Figure 2). The use of standard formats and standard plug-ins allows the widespread and easy use of data visualization including interaction by the user.

6.4 Historical Background

The idea for a W3DS was developed in the context of the initiative Geodata Infrastructure North Rhine-Westphalia (GDI-NRW), resulting in a prototype named Pilot

3D. The project, titled “Pilot 3D”, prototyped interoperable geo-visualization of 3D city and regional models (see [7]). Several W3DS implementations are already running and experiences made during the test phase are reflected in this draft specification.

The W3DS discussion paper was picked up in the GDI-3D project carried out at the University of Bonn as a central service for providing city and landscape models in a distributed Spatial Data Infrastructure (SDI-3D), see [8]. Concepts on how to integrate a range of other OGC services including OpenLS (Geocoder, Routing Service, Directory Service), WFS, CS-W, WPS, and SOS in such a 3D infrastructure were developed and shown in an integrated client. The W3DS was also included in the OGC OWS-6 initiative as experimental interface for providing LOD4 indoor models.

6.5 W3DS Interface

The W3DS interface (currently) specifies 5 operations that can be requested by a client and performed by a W3DS server. Those operations are:

a) GetCapabilities – This operation allows a client to request and receive back service metadata (or Capabilities) documents that describe the abilities of the specific server implementation. This operation also supports negotiation of the standard version being used for client-server interactions.

b) GetScene – This operation allows a client to retrieve 3D Scenes.

c) GetFeatureInfo – This operation allows a client to retrieve attribute data of selected features.

d) GetLayerInfo – This operation allows a client to retrieve information on available attribute names and values of a selected layer.

e) GetTile – This operation allows a client to retrieve single tiles using indices.

These operations have many similarities to other OGC Web Services, including the WMS, WFS, and WCS. Many of these interface aspects that are common with other OWSs are thus specified in the OpenGIS® Web Services Common Implementation Specification [OGC 06-121r3]. Many of these common aspects are normatively referenced herein, instead of being repeated in this standard.

Figure 3 is a simple UML diagram summarizing the W3DS interface. This class diagram shows that the W3DS interface class inherits the getCapabilities operation from the OGCWebService interface class, and adds the GetScene, GetFeatureInfo, GetLayerInfo, and GetTile operations. (This capitalization of names uses the OGC/ISO profile of UML.) A more complete UML model of the W3DS interface is provided in Annex C (informative).

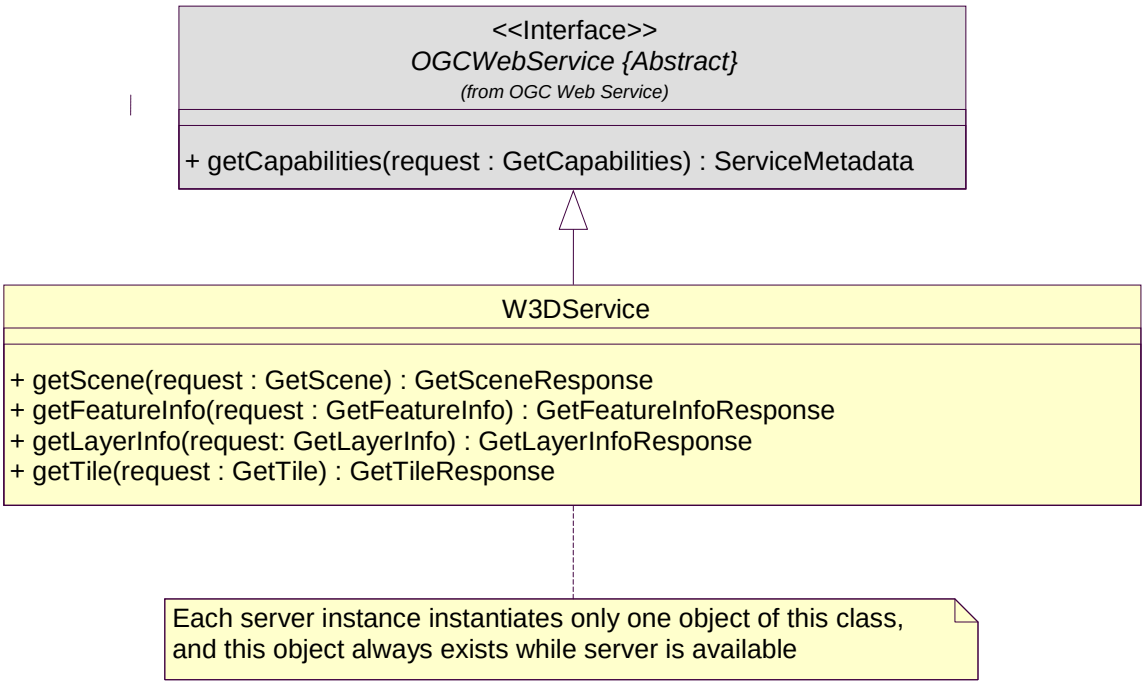

Figure 3 — W3DS interface UML diagram

NOTE In this UML diagram, the request and response for each operation is shown as a single parameter that is a data structure containing multiple lower-level parameters, which are discussed in subsequent clauses. The UML classes modeling these data structures are included in the complete UML model in Annex C.

Each of the W3DS operations is described in more detail in subsequent clauses.

6.6 Operation request encoding

The encoding of operation requests shall use HTTP GET with KVP encoding and HTTP POST with XML encoding as specified in Clause 11 of [OGC 06-121r3]. Table 2 summarizes the W3DS Service operations and their encoding methods defined in this standard.

Table 2 — Operation request encoding

Operation name	Request encoding
GetCapabilities (required)	KVP and optional XML
GetScene	XML and optional KVP
GetFeatureInfo	XML and optional KVP
GetLayerInfo	XML and optional KVP
GetTile	XML and optional KVP

7 GetCapabilities operation (mandatory)

7.1 Introduction

The mandatory GetCapabilities operation allows clients to retrieve service metadata from a server. The response to a GetCapabilities request shall be an XML document containing service metadata about the server, including specific information about layer properties and how to access data from the server. This clause specifies the XML document that a W3DS server shall return to describe its capabilities.

7.2 Operation request

The GetCapabilities operation request shall be as specified in Subclauses 7.2 and 7.3 of [OGC 06-121r3]. The value of the "service" parameter shall be "W3DS". The allowed set of service metadata (or Capabilities) XML document section names and meanings shall be as specified in Tables 3 and 7 of [OGC 06-121r3].

The "Multiplicity and use" column in Table 1 of [OGC 06-121r3] specifies the optionality of each listed parameter in the GetCapabilities operation request. Table 3 specifies the implementation of those parameters by W3DS clients and servers.

Table 3 — Implementation of parameters in GetCapabilities operation request

Names	Multiplicity	Client implementation	Server implementation
service service	One (mandatory)	Each parameter shall be implemented by all clients, using specified value	Each parameter shall be implemented by all servers, checking that each parameter is received with specified value
request request	One (mandatory)		
Accept Accept	Zero or one (optional)	Should be implemented by all software clients, using specified values	Shall be implemented by all servers, checking if parameter is received with specified value(s)
Sections Sections	Zero or one (optional) [b]	Each parameter may be implemented by each client [b] If parameter not provided, shall expect default response If parameter provided, shall allow default or specified response	Each parameter may be implemented by each server [a] If parameter not implemented or not received, shall provide default response If parameter implemented and received, shall provide specified response
update update	Zero or one (optional) b		
Accept Accept	Zero or one (optional) b		

All W3DS servers shall implement HTTP GET transfer of the GetCapabilities operation request, using KVP encoding. Servers may also implement HTTP POST transfer of the GetCapabilities operation request, using XML encoding only.

EXAMPLE 1 To request a W3DS capabilities document, a client could issue the following KVP encoded GetCapabilities operation request with near-minimum contents:


```
http://hostname:port/path?SERVICE=W3DS&REQUEST=GetCapabilities&ACCEPTVE
RSIONS=0.3.0,0.4.0&SECTIONS=Contents&UPDATESEQUENCE=XYZ123&
ACCEPTFORMATS=text/xml
```

EXAMPLE 2 The corresponding GetCapabilities operation request XML encoded for HTTP POST is:

```
<?xml version="1.0" encoding="UTF-8"?>
<w3ds:GetCapabilities service="W3DS" request="GetCapabilities"
updateSequence="XYZ123">
    <w3ds:AcceptVersions>
        <ows:Version>0.3.0</ows:Version>
        <ows:Version>0.4.0</ows:Version>
    </w3ds:AcceptVersions>
    <w3ds:Sections>
        <ows:Section>Contents</ows:Section>
    </w3ds:Sections>
    <w3ds:AcceptFormats>
        <ows:OutputFormat>text/xml</ows:OutputFormat>
    </w3ds:AcceptFormats>
</w3ds:GetCapabilities>
```

NOTE an example for a complete GetCapabilities request can be found in [OGC 06-121r3] clause 7.2

7.3 GetCapabilities operation response

7.3.1 Normal response

The service metadata document shall contain the W3DS sections specified in Table 4. Depending on the values in the Sections parameter of the GetCapabilities operation request, any combination of these sections can be requested and shall be returned when requested.

Table 4— Section name values and contents

Section name	Contents
ServiceIdentification	Metadata about this specific server. The schema of this section shall be the same as for all OWSs, as specified in Subclause 7.4.3 and owsServiceIdentification.xsd of [OGC 06-121r3].
ServiceProvider	Metadata about the organization operating this server. The schema of this section shall be the same for all OWSs, as specified in Subclause 7.4.4 and owsServiceProvider.xsd of [OGC 06-121r3].
OperationsMetadata	Metadata about the operations specified by this service and implemented by this server, including the URLs for operation requests. The basic contents and organization of this section shall be the same as for all OWSs, as specified in Subclause 7.4.5 and owsOperationsMetadata.xsd of [OGC 06-121r3].
Contents	Metadata about the data served by this server. For the W3DS, this section shall contain data about the layers, as specified in Subclause 7.3.4 below.

In addition to these sections, each service metadata document shall include the mandatory "version" and optional updateSequence parameters specified in Table 6 in Subclause 7.4.1 of [OGC 06-121r3].

7.3.2 OperationsMetadata section contents

For the W3DS, the OperationsMetadata section shall contain all parameters as specified in Subclause 7.4.5 and owsOperationsMetadata.xsd of [OGC 06-121r3].

Table 5 — Parts of OperationsMetadata section

Names	Definition	Multiplicity and use
operation Operation	Metadata for one operation that this server interface implements	One or more (mandatory) One for each implemented operation
parameter Parameter	Parameter valid domain that applies to one or more operations which this server implements [a]	Zero or more (optional) One for each such parameter with limited domain
constraint Constraint	Constraint on valid domain of a non-parameter quantity that applies to this server	Zero or more (optional) One for each such quantity with limited domain
extendedCapabilities ExtendedCapabilities	Metadata about server and software additional abilities	Zero or one (optional) Included when server provides additional capabilities
a This parameter may be an input and/or output parameter of these operations.		

Table 6 — Parts of Operation data structure

Names	Definition	Data type	Multiplicity and use
Name name	Name of this operation (request) (for example, GetCapabilities)	Character string type, not empty	One (mandatory)
DCP DCP	Information for a Distributed Computing Platform (DCP) supported for this operation	DCP data structure, See [OGC 06-121r3] Table 15	One or more (mandatory) One for each supported DCP for this operation request [a]
parameter Parameter	Parameter valid domain that applies to this operation which this server implements [b]	ows:DomainType, See [OGC 06-121r3] Table 36	Zero or more (optional) One for each such parameter with limited domain
constraint Constraint	Constraint on valid domain of a non-parameter quantity that applies to this operation which this server implements [c]	ows:DomainType, See [OGC 06-121r3] Table 36	Zero or more (optional) One for each such quantity with limited domain
metadata Metadata	Metadata about this operation and its implementation [d]	Metadata contents or reference to metadata	Zero or more (optional) One for each such

Names	Definition	Data type	Multiplicity and use
			metadata object

a At present, only the HTTP DCP is defined, so the Operation subsection only includes one DCP subsection.

b This parameter may be an input and/or output parameter of this operation. If one of these Parameter data structures has the same parameter "name" as a Parameter subsection in the OperationsMetadata subsection, this Parameter subsection shall override the other one for this operation.

c If one of these Constraints has the same quantity "name" as a Constraint subsection in the OperationsMetadata section, this Constraint subsection shall override the other one for this operation.

d Each operation that uses some form of query or filtering should include metadata describing the query or filter languages and associated capabilities implemented by this server. The schema of this query languages metadata is (currently) specific to each OWS type, as defined by that Implementation Specification.

The mandatory values of various (XML) attributes shall be as specified in Table 7. Similarly, the optional attribute values listed in Table 8 shall be included or not depending on whether that operation is implemented by that server. In Table 7 and Table 8, the "Attribute name" column uses dot-separator notation to identify parts of a parent item. The "Attribute value**"** column references an operation parameter, in this case an operation name, and the meaning of including that value is listed in the right column.

Table 7 — Required values of OperationsMetadata section attributes

Attribute name	Attribute value	Meaning of attribute value
Operation.name	GetCapabilities	The GetCapabilities operation is implemented by this server.
	GetScene	The GetScene operation is implemented by this server.

Table 8 — Optional values of OperationsMetadata section attributes

Attribute name	Attribute value	Meaning of attribute value
Operation.name	GetFeatureInfo	The GetFeatureInfo operation is implemented by this server.
	GetLayerInfo	The GetLayerInfo operation is implemented by this server.
	GetTile	The GetTile operation is implemented by this server.

In addition to the values listed in Table 7 and Table 8, there are many optional values of the "name" attributes and "value" elements in the OperationsMetadata section, which may be included when considered useful. Most of these attributes and elements are for recording the domains of various parameters and quantities.

EXAMPLE 1 The domain of the exceptionCode parameter could record all the codes implemented for each operation by that specific server. Similarly, each of the GetCapabilities operation optional request parameters might have its domain recorded.

EXAMPLE 2 The domain of the Sections parameter in the GetCapabilities operation request could record all the sections implemented by that specific server.

7.3.2.1 HTTP Encoding

All W3DS servers shall specify the encodings that may be sent using HTTP GET or HTTP POST transfer of operation requests. Specifically, an ows:Constraint element shall

be included, with either “GetEncoding” or “PostEncoding” as the value of the “name” attribute and specifying different allowed values for each allowed encoding:

a) The value “SOAP” shall indicate that SOAP encoding is allowed when using HTTP POST transfer. The value “SOAP” is only valid for “PostEncoding”.

b) The value “XML” shall indicate that XML encoding is allowed (without SOAP message encapsulation), when using HTTP POST transfer. The value “XML” is only valid for “PostEncoding”.

c) The value “KVP” shall indicate that KVP encoding is allowed. The value “KVP” is valid for “GetEncoding” and “PostEncoding”.

If the HTTP connect point URL is different for different encodings of the operation requests, this ows:Constraint element shall be included in each Post element. If the connect point URL is the same for all encodings of all operation requests, this ows:Constraint element shall be included in the OperationsMetadata element.

7.3.2.2 Parameter values

All W3DS servers shall specify the possible encodings of the responses for each supported operation as specified in Table 5 and Table 6 if additional encodings to the default encoding are available. Specifically, ows:Parameter elements shall be included, with the parameter name in the “name” attribute and the allowed encodings as list of ows:Value elements. If ows:Parameter elements are included, the default encodings shall be included in the list.

An ows:Parameter element with name “ContentEncoding” shall specify whether the content has an additional encoding beyond that present in the media if the operation was successful. Most commonly this indicates that the content can be compressed with gzip or other compression methods. The default value for all operations is plain, meaning that no compression is performed. Although the content negotiation is usually implemented in the HTTP layer using request headers, this information is repeated in the service meta data in order to configure clients or inform users about available compression prior to the actual requests being made.

An ows:Parameter element with name “ExceptionFormat” shall specify the possible MIME types of the response if the operation failed. The default MIME type for all operations is “text/xml”. The special format “blank” may be applied to the GetScene and the GetTile operation.

Example for Parameter elements of the GetScene operation:

```
<ows:Parameter name="ContentEncoding">
   <ows:AllowedValues>
      <ows:Value>plain</ows:Value>
      <ows:Value>gzip</ows:Value>
   </ows:AllowedValues>
```



```
</ows:Parameter>
<ows:Parameter name="ExceptionFormat">
    <ows:AllowedValues>
        <ows:Value>text/xml</ows:Value>
        <ows:Value>text/plain</ows:Value>
        <ows:Value>blank</ows:Value>
    </ows:AllowedValues>
</ows:Parameter>
```

7.3.3 X3D MIME types

In order to correctly interpret the bytes of the stream generated as response to a GetScene or GetTile request, it is important to know the media type, which shall be specified as MIME type. The MIME type must be set by the server prior to writing to the output stream. X3D knows 3 different encodings. The base MIME type for X3D files is model/x3d. Each encoding has its own modifier. XML is +xml, Classic VRML is +vrml.

Table 9 — X3D Encodings and MIME Types

X3D Encoding	File Extension	MIME Type
Classic VRML	.x3dv	model/x3d+vrml
XML	.x3d	model/x3d or model/x3d+xml
Binary	.x3db	model/x3d+binary

The default X3D encoding refers only to the XML encoding. If Classic VRML and/or Binary is supported by the server, this shall be advertised by the data set's OuputFormat element.

It is also possible and recommended to compress the output stream. The standard compression method is gzip. For indicating gzipped output streams, a special header field called "Content-Encoding" shall be set by the server. This field indicates if the content has been encoded beyond that present in the media. This is the field that is used to indicate that the stream has been compressed with gzip. Clients must detect this field and if set to gzip, uncompress the stream prior to passing it to the X3D parser.

7.3.4 Contents section contents

The Contents section of a service metadata document contains metadata about the data served by this server. For the W3DS, this Contents section shall contain data about the name, title, bounding box, supported CRSs, and LODs of each layer, and information about available backgrounds.

The layer concept referred to in this document is not exactly the same as used in image processing and computer cartography. In WMS a layer can be considered as a transparent sheet with features drawn upon it. The order in which layers are drawn on the map is

important because the features may overlap. The order determines what will be visible and what will be hidden. In 3D scenes, the order in which features are drawn on the screen is irrelevant and the visibility is determined solely by the distance to the virtual viewpoint. Features overlap based on the spatial position of the triangles which make up the feature geometries. The layer concept is comparable to the FeatureType in WFS. However, we keep the term layer because it is commonly understandable and in order to make a distinction between WFS, which serves feature for being processed in GIS software and portrayal services, which serve images and display elements for end users.

The Contents section shall include the parameters specified in Table 10 through Table 16.

Table 10 — Parameters included in Contents section

Names	Definition	Data type and values	Multiplicity and use
Layer	Meta data describing a data set available from this server	Layer data structure, see Table 11	Zero or more (optional)
Background	Meta data describing a background available for a GetScene request	Background data structure, see Table 16	Zero or more (optional)

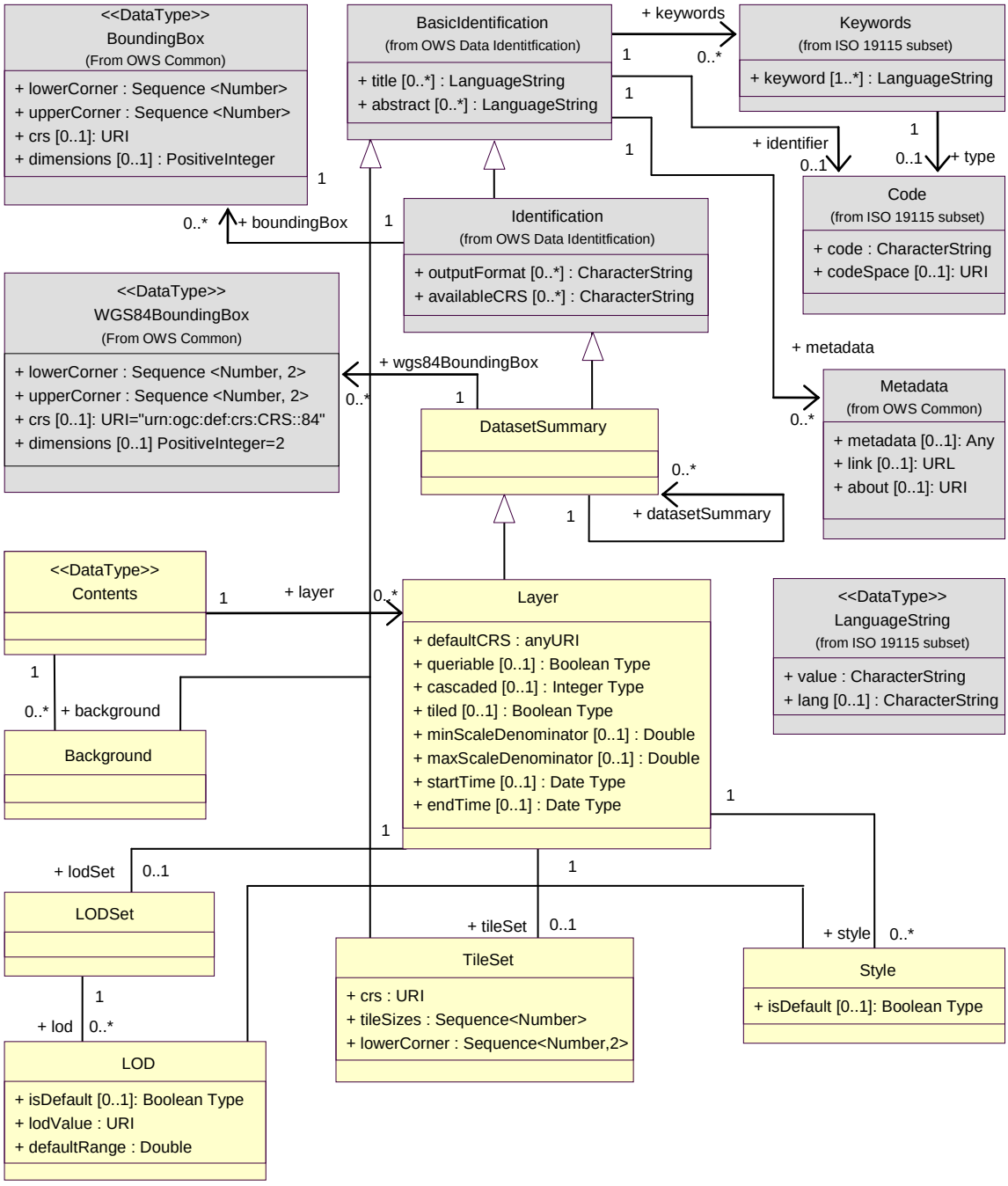

Figure 4 —UML model of W3DS contents section

Table 11 — Parts of Layer data structure

Names [a]	Definition	Data type and values	Multiplicity and use
title Title	Title of this layer, normally used for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Recommended and usually included [c] Include one for each

Names [a]	Definition	Data type and values	Multiplicity and use
			language represented
abstract Abstract	Brief narrative description of this layer, normally available for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Recommended and usually included Include one for each language represented
keywords Keywords	Unordered list of one or more commonly used or formalised word(s) or phrase(s) used to describe this dataset	See MD_Keywords class in ISO 19115	Zero or more (optional) One for each keyword authority used
identifier Identifier	Unambiguous identifier or name of this dataset, unique for this server	Character String type, not empty	One (mandatory)
wgs84BoundingBox WGS84BoundingBox	Minimum bounding rectangle surrounding dataset, using WGS 84 CRS with decimal degrees and longitude before latitude	WGS84BoundingBox data structure, see [OGC 06-121r3] Subclause 10.2	Zero or more (optional)
boundingBox [f] BoundingBox	Minimum bounding rectangle surrounding dataset, in available CRS.	BoundingBox data structure, see [OGC 06-121r3] Subclause 10.2	Zero or more (optional)
metadata Metadata	Reference to more metadata about this dataset	ows:Metadata, see [OGC 06-121r3] Table 32	Zero or more (optional)
datasetSummary DatasetSummary	Metadata describing one subsidiary dataset available from this server	Layer data structure, see this Table	Zero or more (optional)
outputFormat OutputFormat	Reference to a format in which output data from this server may be encoded	ows:MIMEType, see [OGC 06-121r3] clause 10.5	Zero or more (optional)
availableCRS AvailableCRS	Coordinate reference system in which data from this layer may be output by this server	URI [d]	Zero or more (optional)
defaultCRS DefaultCRS	Indicates which CRS is used by this layer for storing the data. If not explicitly stated in the GetScene request, this CRS will be used	URI [d]	One (mandatory)
style Style	Additional visual style available for this layer	Style data structure, see Table 15	Zero or more (optional)
queryable Queryable	Indicates whether the server supports the GetFeatureInfo operation on this layer	Boolean type, not empty	Zero or one (optional)

Names [a]	Definition	Data type and values	Multiplicity and use
cascaded Cascaded	Indicates whether the data of this layer was obtained from another server	Integer type, not empty	Zero or one (optional)
tiled Tiled	Indicates whether the data of this layer is available as set of rectangular tiles	Boolean type, not empty	Zero or one (optional)
minScaleDenominator MinScaleDenominator	Minimum scale denominator at which it is useful to display data of this layer	Double type, not empty	Zero or one (optional)
maxScaleDenominator MaxScaleDenominator	Maximum scale denominator at which it is useful to display data of this layer	Double type, not empty	Zero or one (optional)
lodSet LODSet	Description of available discrete LODs of this layer	LODSet data structure, see Table 12	Zero or one (optional)
tileSet TileSet	Description of spatial partitioning into rectangular tiles	TileSet data structure, see Table 14	Zero or one (optional)
startTime StartTime	Earliest date and time at which features are available	Date type, not empty. Data type defined in ISO 8601:2004(E)	Zero or one (optional)
endTime EndTime	Latest date and time at which features are available	Date type, not empty. Data type defined in ISO 8601:2004(E)	Zero or one (optional)

a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.

c Software may display the "Identifier" value when the "Title" is absent

d The CRS shall be specified using either the European Petroleum Survey Group form "EPSG:<POSC Code>" or the URL format defined in OWS Common. Examples: "EPSG:31467", "urn:ogc:def:crs:EPSG:6.3:31467 ", see [OGC 06-121r3] clause 10.3 and [OGC 04-046r3]

e The LOD shall be specified using a combination of a prefix and a number, e.g. "CityGML:4" for indoor models

f There is no provision for describing disjoint bounding boxes like in WFS

7.3.4.1 OutputFormat

The element OutputFormat specifies a possible Format which can be used for encoding the contents of the layer. The Format is specified as MIME type.

7.3.4.2 AvailableCRS

The element AvailableCRS describes the possible CRSs that can be accepted if the GetScene operation or GetTile operation contains a CRS parameter. The values shall be specified using either the European Petroleum Survey Group form "EPSG:<POSC Code>" or the URI format defined in OWS Common (for instance "urn:ogc:def:crs:EPSG:6.3:31467", or simply "EPSG:31467") , see [OGC 06-121r3] clause 10.3 and [OGC 04-046r3].

7.3.4.3 Styles

Each layer may advertise additional styles (server styles) which modify the appearance and characteristics of the underlying data set. The style's Identifier is used in the GetScene request Styles parameter. If only a single style is available, that style is known as the “default” style and need not be advertised by the server. The data structure of Style is listed in Table 15. Each style consists of an Identifier, a Title which may be presented to the user, an Abstract and a list of Keywords. The Abstract should give a brief narrative description of how the visualization is influenced by the style.

Styles usually affect the symbolization of features, for example, the materials can be replaced by another material, including diffuse color, reflection properties, and transparency as defined in the style. Styling can also take the feature attribute values as input for distinguishing features of different categories by color. Styling may also apply different symbols to point and line features including geometric primitives, billboards, textures, and complex 3D proto types. The size of these symbols may be scaled according on feature attribute values. It is recommended to use the Symbology Encoding (SE) and Filter Encoding (FE) as basis for defining server styles (see [OGC 04-095] and [OGC 05-077r4]), and extend the capabilities for styling 3D objects (see [OGC 09-042]).

7.3.4.4 Queryable

The boolean attribute Queryable indicates whether the server supports the GetFeatureInfo operation on that layer. A server may support GetFeatureInfo on some of its layers, but need not support it on all layers. A server shall issue a service exception (code="LayerNotQueryable") if GetFeatureInfo is requested on a Layer that is not queryable.

7.3.4.5 Cascaded

A layer is said to have been “cascaded” if it was obtained from an originating server and then included in the service metadata of a different server. The second server may simply offer an additional access point for the Layer, or may add value by offering additional output formats or re-projection to other CRSs. If a W3DS cascades the content of another W3DS, then it shall increment by 1 the value of the cascaded attribute for the affected layers. If that attribute is missing from the originating server’s service metadata, then the Cascading WMS shall insert the attribute and set it to 1.

7.3.4.6 Tiled

The boolean attribute Tiled indicates whether the server supports the GetTile operation on that layer. A server may support GetTile on some of its layers, but need not support it on all layers. A server shall issue a service exception (code="LayerNotTiled") if GetTile is requested on a layer that is not tiled. If the attribute Tiled is set to true, then also a TileSet shall be defined for this layer. A server shall issue a service exception (code="TileSetNotDefined") if GetTile is requested on a layer that is tiled, but has no TileSet definition.

7.3.4.7 Scale Denominators

The <MinScaleDenominator> and <MaxScaleDenominator> elements define the range of scales for which it is appropriate to generate a map of a Layer. Although the concept of scale originates from 2D map displays, it is not altogether wrong for 3D computer graphics. 3D features are also drawn on the screen with a certain scale. The scale is not explicitly set for a specific scene because the view parameters are defined by other values, however it may be calculated. Moreover, the scale is not uniformly distributed on the screen; it can vary by several magnitudes. It depends on many factors that make this calculation very difficult. The factors include the screen pixel distance, resolution, the projection transformation function (parallel, perspective, distorted or other projections), distance of the object to the virtual camera position, field of view, viewing angles, etc.

Therefore, the <MinScaleDenominator> and <MaxScaleDenominator> elements should be considered as hints for the client in order to determine whether it makes sense to display the layer within a particular scene, not as hard limits. For instance, a layer containing a very big but low detail landscape model can include very high values as minimum and maximum scale denominators in its service meta data.

7.3.4.8 LODSet

The LODSet child node describes a set of Levels of Detail that can be provided by the layer (see Table 12 and Table 13). A layer may contain objects in several representations to choose from. In this document, the term LOD always refers to the concept of Discrete LODs, meaning that multiple representations are available for an object. These representations do not share any vertices or other geometric elements. For instance a building may be represented as simple box geometry, as geometry with additional façade textures, as group containing elements for walls, roofs, windows, doors, or even containing the complete room interior. Each of these representations describes the same object and can be therefore stored in the same layer. However, it is not necessary that all LODs are consistently available for each object. A Client may assemble his scene graph from subsets of the same layer with different LODs and thus reduce the workload of the graphics pipeline.

Each LOD is described in a separate LOD node containing a unique identifier (unique to the containing layer, since LOD nodes must not be cross-referenced by other layers), title, description, and the actual value or magnitude of the LOD. This value is a URI consisting of a prefix and the actual numeric value, separated by a colon, e.g. "CityGML:1". The prefix indicates the spectrum of possible values and how these values must be interpreted. The LOD definitions of CityGML can be found in [08-007r1] clause 6.2. The prefix is optional; if it is omitted then also colon must be omitted and the value must be a floating point number. The numeric value indicates the actual “level” of detail on an ordinal scale. The scale values have a total order, which is connexive ($a > b$ or $a < b$ or $a = b$) and transitive ($a > b > c$ implies $a > c$), but no interval or metric may be derived from the values. For instance, “CityGML:4” is more accurate than “CityGML:2”, but not necessarily twice as accurate. The order of the LOD nodes within the LODSet is not defined, but it is recommended to use an order increasing by the LODValue, from lower to higher levels of detail.

In case of multiple LODs, a special group node (e.g. X3D LOD node) may be defined containing all levels as child nodes. The group node acts as a switch and selects one of the child node depending on the distance to the current viewpoint. In order to function properly, this LOD node requires threshold values which are compared to the current distance. The behavior of the server to group all available LODs into an LOD node can be triggered by setting the GetScene parameter LODSelection to “combined”. The threshold values will be taken from the DefaultRange value in the LOD data structure (see Table 13). The DefaultRange value refers to the distance in meters at which the LOD will become visible.

7.3.4.9 TileSet

In some cases it is useful to configure and provide data as set of adjacent rectangular tiles. These tiles may be accessed using indices for tile level, row and column in a GetTile request. In order to enable a GetTile request on the layer, a TileSet definition must be provided which contains information on the spatial alignment of available tiles (see Table 14).

A TileSet consists of multiple levels, each having a different tile size. The tile sizes are strictly related by powers of two, meaning that the next higher level must have a tile size which is exactly half of the tile size of the next lower level. Tile sizes are defines as a list (element TileSizes) of double precision floating point numbers, separated by space. This list is ordered by decreasing tile size. The tile level index corresponds with the position in the TileSizes list, starting at zero.

Each tile level can be described as a grid with origin at the LowerCorner coordinate, which is the lower left or south east corner of the extent covering all data in the layer. The LowerCorner is defined only once in a TileSet. Each tile level therefore shares the same origin. For example, tile level n is defined by a grid originating at LowerCorner and grid size of value n in the TileSizes list. Tile extends are between the grid lines.

This data structure is comparable to an image pyramid or quad tree. Each tile is divided into four quarters in the next higher level representing a higher degree of accuracy. This concept is often used for surface data (terrain, geological strata) and enables continuous multiresolution surface representations.

Example:

```
<w3ds:TileSet>
    <ows:Identifier>dem_tileset</ows:Identifier>
    <w3ds:CRS>EPSG:4326</w3ds:CRS>
    <w3ds:TileSizes>180 90 45 22.5 11.25 5.625 2.8125 1.40625 0.703125
    0.3515625 0.17578125</w3ds:TileSizes>
    <w3ds:LowerCorner>-180.0 -90.0</w3ds:LowerCorner>
</w3ds:TileSet>
```


7.3.4.10 StartTime and EndTime

The StartTime and EndTime elements can be used to indicate the availability of historical content in a layer. Features may exist only within a defined historical time span. If the date of construction and/or date of demolition of each building is available, then a reconstruction of past cityscapes becomes possible or the development over time can be analyzed by submitting a series of GetScene requests with different times. Features may be also available in different representations showing temporal changes like extensions, renovations, decay etc. The data format of the Time parameters is defined in ISO 8601:2004(E) . If only one time parameter is included in the layer meta data, then the temporal availability is not limited to a specific time span. For instance, a building data set which is maintained by an authority and updated regularly may have a StartTime indicating the date at which the recording began, and no EndTime, since it must be possible to request the present situation.

Table 12 — Parts of LODSet data structure

Names [a]	Definition	Data type and values	Multiplicity and use
lod LOD	Set of discrete LODs available for this layer	LOD data structure	One or more (mandatory)
a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.			

Table 13 — Parts of LOD data structure

Names [a]	Definition	Data type and values	Multiplicity and use
title Title	Title of this LOD, normally used for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Recommended and usually included [b] Include one for each language represented
abstract Abstract	Brief narrative description of this LOD, normally available for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Include one for each language represented
keywords Keywords	Unordered list of one or more commonly used or formalised word(s) or phrase(s) used to describe this dataset	See MD_Keywords class in ISO 19115	Zero or more (optional) One for each keyword authority used
identifier Identifier	Unambiguous identifier or name of this LOD, unique for the containing layer	Character String type, not empty	One (mandatory)
metadata Metadata	Reference to more metadata about this dataset	ows:Metadata, see [OGC 06-121r3] Table 32	Zero or more (optional)
lodValue	Optional prefix plus value or	URI	One (mandatory)

Names [a]	Definition	Data type and values	Multiplicity and use
LODValue	magnitude of this LOD		
defaultRange DefaultRange	Default distance threshold value in meters at which the LOD becomes visible	Double Type	One (mandatory)
isDefault IsDefault	This LOD is used when no LOD is specified for this layer in the request	Boolean type	Zero or one (optional) Default is false

a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.

b Software may display the "Identifier" value when the "Title" is absent

Table 14 — Parts of TileSet data structure

Names [a]	Definition	Data type and values	Multiplicity and use
identifier Identifier	Unambiguous identifier or name of this TileSet, unique for this server	Character String type, not empty	One (mandatory)
crs CRS	CRS for which this TileSet is available and in which the data is natively stored on the server	URI	One (mandatory)
tileSizes TileSizes	List of tile sizes ordered descending. The unit of the values in defined by the CRS	Ordered sequence of double values	One (mandatory)
lowerCorner LowerCorner	Coordinates of the lower left or south east corner of the tile matrix	Ordered sequence of double values [b]	One (mandatory) Usually corresponds with the lowerCorner of the Layer's BoundingBox

a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.

b Only 2 axes shall be used representing east-west direction and south-north direction. The order of the axes and the increasing direction is defined by the CRS

Table 15 — Parts of Style data structure

Names [a]	Definition	Data type and values	Multiplicity and use
title Title	Title of this style, normally used for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Recommended and usually included [b] Include one for each language represented
abstract Abstract	Brief narrative description of this style, normally available for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Include one for each language represented
keywords	Unordered list of one or	See MD_Keywords	Zero or more

Names [a]	Definition	Data type and values	Multiplicity and use
Keywords	more commonly used or formalised word(s) or phrase(s) used to describe this dataset	class in ISO 19115	(optional) One for each keyword authority used
identifier Identifier	Unambiguous identifier or name of this style, unique for the containing layer	Character String type, not empty	One (mandatory)
metadata Metadata	Reference to more metadata about this dataset	ows:Metadata, see [OGC 06-121r3] Table 32	Zero or more (optional)
isDefault IsDefault	This style is used when no style is specified for this layer in the request	Boolean type	Zero or one (optional) Default is false

a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.

b Software may display the "Identifier" value when the "Title" is absent

Table 16 — Parts of Background data structure

Names [a]	Definition	Data type and values	Multiplicity and use
title Title	Title of this LOD, normally used for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Recommended and usually included [b] Include one for each language represented
abstract Abstract	Brief narrative description of this LOD, normally available for display to a human	LanguageString data structure, see [OGC 06-121r3] clause 10.7	Zero or more (optional) Include one for each language represented
keywords Keywords	Unordered list of one or more commonly used or formalised word(s) or phrase(s) used to describe this dataset	See MD_Keywords class in ISO 19115	Zero or more (optional) One for each keyword authority used
identifier Identifier	Unambiguous identifier or name of this LOD, unique for the containing layer	Character String type, not empty	One (mandatory)
metadata Metadata	Reference to more metadata about this dataset	ows:Metadata, see [OGC 06-121r3] Table 32	Zero or more (optional)

a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.

b Software may display the "Identifier" value when the "Title" is absent

The "Multiplicity and use" columns in Table 5, Table 6, **Error! Reference source not found.**and Table 10 through Table 16 specify the optionality of each listed parameter and data structure in the GetCapabilities operation response. All the "mandatory" parameters and data structures shall be implemented by all W3DS servers, using a specified value(s).

All other “optional” parameters and data structures, in the GetCapabilities operation response, should be implemented by all W3DS servers using specified values, whenever and wherever each is considered useful information for that server.

7.3.5 Capabilities document XML encoding

A XML schema fragment for a W3DS service metadata document extends ows:CapabilitiesBaseType in owsCommon.xsd of [OGC 06-121r3], and is:

```
<!-- ========================================================== -->
<element name="Capabilities" type="w3ds:CapabilitiesType"/>
<!-- ========================================================== -->
<complexType name="CapabilitiesType">
    <complexContent>
        <extension base="ows:CapabilitiesBaseType">
            <sequence>
                <element ref="w3ds:Contents" minOccurs="0"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<!-- ========================================================== -->
<element name="Contents" type="w3ds:ContentsType"/>
<!-- ========================================================== -->
<complexType name="ContentsType">
    <complexContent>
        <extension base="ows:ContentsBaseType">
            <sequence>
                <element name="Background" type="w3ds:BackgroundType"
minOccurs="0" maxOccurs="unbounded"/>
                <element name="Layer" type="w3ds:LayerType" minOccurs="0"
maxOccurs="unbounded"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

As indicated, this XML Schema Document uses the owsServiceIdentification.xsd, owsServiceProvider.xsd, and owsOperationsMetadata.xsd schemas specified in [OGC 06-121r3]. It also uses an XML Schema Document for the “Contents” section of the W3DS Capabilities XML document, which shall be as attached in the w3dsCapabilities.xsd file. All these XML Schema Documents contain documentation of the meaning of each element, attribute, and type, and this documentation shall be considered normative as specified in Subclause 11.6.3 of [OGC 06-121r3].

7.3.6 Capabilities document example

In response to GetCapabilities operation request, a W3DS server might generate a document that looks like:

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



```
<w3ds:Capabilities xmlns="http://www.opengis.net/w3ds/0.4.0"
xmlns:w3ds="http://www.opengis.net/w3ds/0.4.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink"
xsi:schemaLocation="http://www.opengis.net/w3ds/0.4.0/w3dsCapabilities.
xsd" version="0.4.0">
    <ows:ServiceIdentification>
        <ows:Title xml:lang="en">Web 3D Service Reference
Implementation</ows:Title>
        <ows:Abstract xml:lang="en">W3DS containing some sample data
for very detailed 3D city models and an terrain model</ows:Abstract>
        <ows:Keywords>
            <ows:Keyword>City Model</ows:Keyword>
            <ows:Keyword>3D</ows:Keyword>
            <ows:Keyword>Digital Elevation Model</ows:Keyword>
            <ows:Keyword>Scene Graph</ows:Keyword>
        </ows:Keywords>
        <ows:ServiceType>OGC W3DS</ows:ServiceType>
        <ows:ServiceTypeVersion>0.4.0</ows:ServiceTypeVersion>
        <ows:Fees>none</ows:Fees>
        <ows:AccessConstraints>none</ows:AccessConstraints>
    </ows:ServiceIdentification>
    <ows:ServiceProvider>
        <ows:ProviderName>University of Bonn, Department of Geography,
Cartography</ows:ProviderName>
        <ows:ProviderSite xlink:href="http://www.giub.uni-
bonn.de/karto"/>
        <ows:ServiceContact>
            <ows:IndividualName>Arne Schilling</ows:IndividualName>
            <ows:PositionName>Senior Engineer</ows:PositionName>
            <ows:ContactInfo>
                <ows:Phone>
                    <ows:Voice>+49 0228 73 2401</ows:Voice>
                    <ows:Facsimile>+49 0228 73 2401</ows:Facsimile>
                </ows:Phone>
                <ows:Address>
                    <ows:DeliveryPoint>University of
Bonn</ows:DeliveryPoint>
                    <ows:City>Bonn</ows:City>

<ows:AdministrativeArea>NRW</ows:AdministrativeArea>
                    <ows:PostalCode>53115</ows:PostalCode>
                    <ows:Country>Germany</ows:Country>

<ows:ElectronicMailAddress>schilling@geographie.uni-
bonn.de</ows:ElectronicMailAddress>
                </ows:Address>
            </ows:ContactInfo>
        </ows:ServiceContact>
    </ows:ServiceProvider>
    <ows:OperationsMetadata>
        <ows:Operation name="GetCapabilities">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                        <ows:Constraint name="GetEncoding">
                            <ows:AllowedValues>
```



```
                            <ows:Value>KVP</ows:Value>
                        </ows:AllowedValues>
                    </ows:Constraint>
                </ows:Get>
            </ows:HTTP>
        </ows:DCP>
    </ows:Operation>
    <ows:Operation name="GetScene">
        <ows:DCP>
            <ows:HTTP>
                <ows:Get xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                    <ows:Constraint name="GetEncoding">
                        <ows:AllowedValues>
                            <ows:Value>KVP</ows:Value>
                        </ows:AllowedValues>
                    </ows:Constraint>
                </ows:Get>
                <ows:Post xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                    <ows:Constraint name="PostEncoding">
                        <ows:AllowedValues>
                            <ows:Value>XML</ows:Value>
                        </ows:AllowedValues>
                    </ows:Constraint>
                </ows:Post>
            </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="ContentEncoding">
            <ows:AllowedValues>
                <ows:Value>plain</ows:Value>
                <ows:Value>gzip</ows:Value>
            </ows:AllowedValues>
        </ows:Parameter>
        <ows:Parameter name="ExceptionFormat">
            <ows:AllowedValues>
                <ows:Value>text/xml</ows:Value>
                <ows:Value>text/plain</ows:Value>
                <ows:Value>blank</ows:Value>
            </ows:AllowedValues>
        </ows:Parameter>
    </ows:Operation>
    <ows:Operation name="GetFeatureInfo">
        <ows:DCP>
            <ows:HTTP>
                <ows:Get xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                    <ows:Constraint name="GetEncoding">
                        <ows:AllowedValues>
                            <ows:Value>KVP</ows:Value>
                        </ows:AllowedValues>
                    </ows:Constraint>
                </ows:Get>
                <ows:Post xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                    <ows:Constraint name="PostEncoding">
                        <ows:AllowedValues>
                            <ows:Value>XML</ows:Value>
```



```
                            </ows:AllowedValues>
                        </ows:Constraint>
                    </ows:Post>
                </ows:HTTP>
            </ows:DCP>
        </ows:Operation>
        <ows:Operation name="GetLayerInfo">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                        <ows:Constraint name="GetEncoding">
                            <ows:AllowedValues>
                                <ows:Value>KVP</ows:Value>
                            </ows:AllowedValues>
                        </ows:Constraint>
                    </ows:Get>
                    <ows:Post xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                        <ows:Constraint name="PostEncoding">
                            <ows:AllowedValues>
                                <ows:Value>XML</ows:Value>
                            </ows:AllowedValues>
                        </ows:Constraint>
                    </ows:Post>
                </ows:HTTP>
            </ows:DCP>
        </ows:Operation>
        <ows:Operation name="GetTile">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                        <ows:Constraint name="GetEncoding">
                            <ows:AllowedValues>
                                <ows:Value>KVP</ows:Value>
                            </ows:AllowedValues>
                        </ows:Constraint>
                    </ows:Get>
                    <ows:Post xlink:href="http://gdi3d.giub.uni-
bonn.de/W3DS_AIRPORT/W3DS?">
                        <ows:Constraint name="PostEncoding">
                            <ows:AllowedValues>
                                <ows:Value>XML</ows:Value>
                            </ows:AllowedValues>
                        </ows:Constraint>
                    </ows:Post>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="ContentEncoding">
                <ows:AllowedValues>
                    <ows:Value>plain</ows:Value>
                    <ows:Value>gzip</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="ExceptionFormat">
                <ows:AllowedValues>
                    <ows:Value>text/xml</ows:Value>
```



```
                    <ows:Value>text/plain</ows:Value>
                    <ows:Value>blank</ows:Value>
                </ows:AllowedValues>
            </ows:Parameter>
        </ows:Operation>
    </ows:OperationsMetadata>
    <w3ds:Contents>
        <ows:DatasetDescriptionSummary>
            <ows:BoundingBox crs="EPSG:26916">
                <ows:LowerCorner>202759.03723318398
3310170.2612183597</ows:LowerCorner>
                <ows:UpperCorner>213200.2944077917
3320896.6813003295</ows:UpperCorner>
            </ows:BoundingBox>
            <ows:WGS84BoundingBox>
                <ows:LowerCorner>-90.07769903770375
29.888741108449068</ows:LowerCorner>
                <ows:UpperCorner>-89.97244224350582
29.983057950776132</ows:UpperCorner>
            </ows:WGS84BoundingBox>
        </ows:DatasetDescriptionSummary>
        <w3ds:Background>
            <ows:Title>Clouded Sky</ows:Title>
            <ows:Abstract>includes a background sphere showing an image
of a sky with clouds</ows:Abstract>
            <ows:Identifier>sky1</ows:Identifier>
        </w3ds:Background>
        <w3ds:Background>
            <ows:Title>Blue Color</ows:Title>
            <ows:Abstract>simple blue colored background</ows:Abstract>
            <ows:Identifier>blue</ows:Identifier>
        </w3ds:Background>
        <w3ds:Background>
            <ows:Title>Gradient Color</ows:Title>
            <ows:Abstract>includes a gradient color background node
from white at horizon to blue</ows:Abstract>
            <ows:Identifier>gradient</ows:Identifier>
        </w3ds:Background>
        <w3ds:Layer>
            <ows:Title>DigitalElevationModel</ows:Title>
            <ows:Abstract xml:lang="en">Digital Elevation Model of the
Airport and sorrounding areas.</ows:Abstract>
            <ows:Identifier>dem</ows:Identifier>
            <ows:BoundingBox crs="EPSG:26916">
                <ows:LowerCorner>202759.03723318398
3310170.2612183597</ows:LowerCorner>
                <ows:UpperCorner>213200.2944077917
3320896.6813003295</ows:UpperCorner>
            </ows:BoundingBox>
            <ows:OutputFormat>model/x3d</ows:OutputFormat>
            <ows:OutputFormat>model/kml</ows:OutputFormat>
            <ows:OutputFormat>model/kmz</ows:OutputFormat>
            <ows:AvailableCRS>EPSG:26916</ows:AvailableCRS>
            <w3ds:WGS84BoundingBox>
                <ows:LowerCorner>-90.07769903770375
29.888741108449068</ows:LowerCorner>
                <ows:UpperCorner>-89.97244224350582
29.983057950776132</ows:UpperCorner>
```



```
            </w3ds:WGS84BoundingBox>
            <w3ds:DefaultCRS>EPSG:26916</w3ds:DefaultCRS>
            <w3ds:Queriable>true</w3ds:Queriable>
            <w3ds:Tiled>true</w3ds:Tiled>
            <w3ds:TileSet>
                <ows:Identifier>dem_tileset</ows:Identifier>
                <w3ds:CRS>EPSG:26916</w3ds:CRS>
                <w3ds:TileSizes>4000.0 2000.0 1000.0
500.0</w3ds:TileSizes>
                <w3ds:LowerCorner>202759.03723318398
3310170.2612183597</w3ds:LowerCorner>
            </w3ds:TileSet>
            <w3ds:Style>
                <ows:Title>default</ows:Title>
                <ows:Identifier>default</ows:Identifier>
                <w3ds:IsDefault>true</w3ds:IsDefault>
            </w3ds:Style>
        </w3ds:Layer>
        <w3ds:Layer>
            <ows:Title>Buildings</ows:Title>
            <ows:Abstract xml:lang="en">Some 3D
Buildings.</ows:Abstract>
            <ows:Identifier>bldgs</ows:Identifier>
            <ows:BoundingBox crs="EPSG:26916">
                <ows:LowerCorner>202759.03723318398
3310170.2612183597</ows:LowerCorner>
                <ows:UpperCorner>213200.2944077917
3320896.6813003295</ows:UpperCorner>
            </ows:BoundingBox>
            <ows:OutputFormat>model/x3d</ows:OutputFormat>
            <ows:OutputFormat>model/kml</ows:OutputFormat>
            <ows:OutputFormat>model/kmz</ows:OutputFormat>
            <ows:AvailableCRS>EPSG:26916</ows:AvailableCRS>
            <w3ds:WGS84BoundingBox>
                <ows:LowerCorner>-90.07769903770375
29.888741108449068</ows:LowerCorner>
                <ows:UpperCorner>-89.97244224350582
29.983057950776132</ows:UpperCorner>
            </w3ds:WGS84BoundingBox>
            <w3ds:DefaultCRS>EPSG:26916</w3ds:DefaultCRS>
            <w3ds:Queriable>true</w3ds:Queriable>
            <w3ds:LODSet>
                <w3ds:LOD>
                    <ows:Title>LOD1</ows:Title>
                    <ows:Abstract>prismatic building
shells</ows:Abstract>
                    <ows:Identifier>bldgs_lod1</ows:Identifier>
                    <w3ds:LODValue>CityGML:1</w3ds:LODValue>
                    <w3ds:DefaultRange>3000.0</w3ds:DefaultRange>
                </w3ds:LOD>
                <w3ds:LOD>
                    <ows:Title>LOD2</ows:Title>
                    <ows:Abstract>buildings with roof
structures</ows:Abstract>
                    <ows:Identifier>bldgs_lod2</ows:Identifier>
                    <w3ds:LODValue>CityGML:2</w3ds:LODValue>
                    <w3ds:DefaultRange>1500.0</w3ds:DefaultRange>
                </w3ds:LOD>
```



```
                <w3ds:LOD>
                    <ows:Title>LOD3</ows:Title>
                    <ows:Abstract>fully textured complex building
models</ows:Abstract>
                    <ows:Identifier>bldgs_lod3</ows:Identifier>
                    <w3ds:LODValue>CityGML:3</w3ds:LODValue>
                    <w3ds:DefaultRange>750.0</w3ds:DefaultRange>
                </w3ds:LOD>
            </w3ds:LODSet>
            <w3ds:Style>
                <ows:Title>default</ows:Title>
                <ows:Abstract>dont style anything</ows:Abstract>
                <ows:Identifier>default</ows:Identifier>
                <w3ds:IsDefault>true</w3ds:IsDefault>
            </w3ds:Style>
            <w3ds:Style>
                <ows:Title>Cartographic</ows:Title>
                <ows:Abstract>Shows the buildings in different colors
according to their usage</ows:Abstract>
                <ows:Identifier>carto</ows:Identifier>
            </w3ds:Style>
        </w3ds:Layer>
    </w3ds:Contents>
</w3ds:Capabilities>
```

7.3.7 Exceptions

When a W3DS server encounters an error while performing a GetCapabilities operation, it shall return an exception report message as specified in Clause 8 of [OGC 06-121r3]. The allowed exception codes shall include those listed in Table 5 of Subclause 7.4.1 of [OGC 06-121r3], if the updateSequence parameter is implemented by the server.

8 GetScene operation (mandatory)

8.1 Introduction

The GetScene operation returns a 3D scene representing a subset of the natural or manmade structures on the earth surface. Upon receiving a GetScene request, a W3DS shall either satisfy the request or issue a service exception.

The required parameters for retrieving a 3D scene from a W3DS comprise spatial, thematic, and other constraints. The minimum set of parameters include, in addition to the mandatory parameters service, request and version, which are part of every W3DS operation, the spatial extent of the scene given as bounding box, the CRS in which the scene shall be provided, the format, and the list of layers. The bounding box defines a rectangular region with edges perpendicular to the selected CRS.

The BoundingBox parameter for selecting the area on the earth surface is only two-dimensional. This parameter is mandatory since it cannot be derived from the camera definition. For limiting the vertical extent of the scene, the two parameters MinHeight

and MaxHeight can be used. Either both parameters together for defining a cube, or only one of them for providing an upper or lower boundary.

8.1.1 Camera definition

Optionally, the position and viewing direction of a virtual camera can be specified. The virtual camera will be integrated as so called Viewpoint into the 3D scene, allowing the user to quickly move the scene into the viewport. In 3D browsers the available viewpoints are usually presented as drop down list. The viewing direction is indirectly described as combination of a camera position and a focus point. The line of sight is determined by subtracting both tuples. All coordinates are provided in world space, i.e. in the CRS provided in the GetScene request. The camera parameters are:

a) Point of Interest (POI): the exact location in x,y,z space of the viewer's focus point which will in the center of screen.

b) Point of Camera (POC): the exact location in x,y,z space of the viewer.

c) Up Vector (UP): a vector perpendicular to the line POI-POC pointing upwards from the viewer's perspective. This can be used to roll the camera around the line of sight.

d) Field of view (FOV): The angular extent of the visible scene, meaning the angle between the left and right border of the viewport.

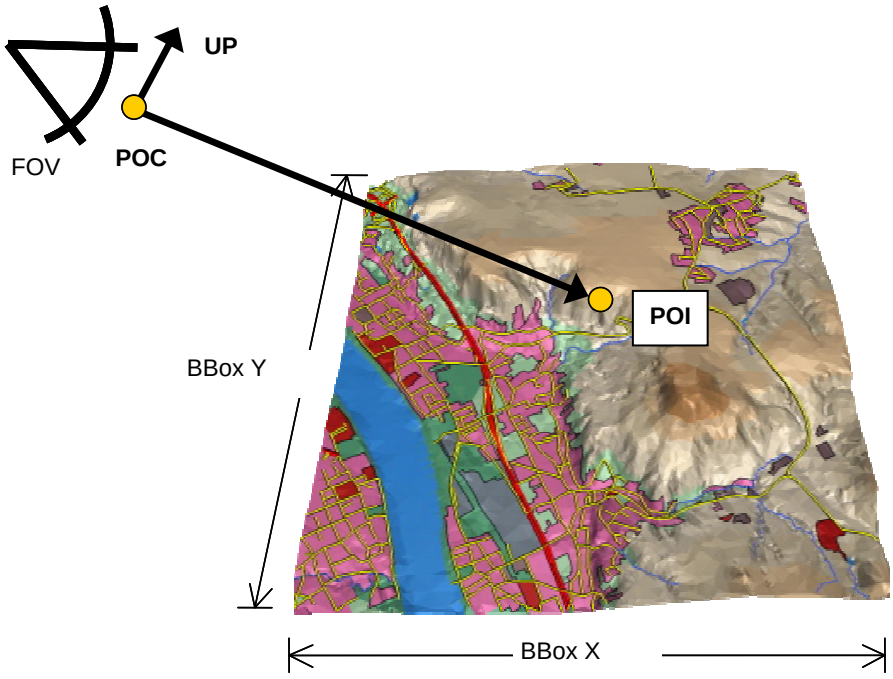

Figure 5 — GetScene parameters defining a virtual viewpoint

8.1.2 World coordinates versus computer graphics coordinates

For the internal representation of the scene graph which is passed to the rendering pipeline of the client, usually shortened coordinates are used. Coordinates of map projections (for instance UTM, Gauss Kruger) or polar coordinate systems (WGS84) require a very high number of digits for representing smaller objects accurately. Modern graphics hardware is working with single precision floating point numbers with only 32 bits of precision, which is not sufficient to represent city models. The GetScene parameter Translate can be used to shift all coordinates closer to the origin.

Additionally the W3DS will usually do an axes transformation from real world to computer graphics coordinate systems, depending on the output format. Real world coordinates (like UTM, Gauss Kruger or geographical coordinates) are typically using left-hand Cartesian coordinate system with z indicating the height above sea level. Computer graphics coordinate systems used in X3D are based on a right-hand Cartesian coordinate system. Therefore the W3DS automatically switches the axes of the real world CRSs as shown in Figure 6.

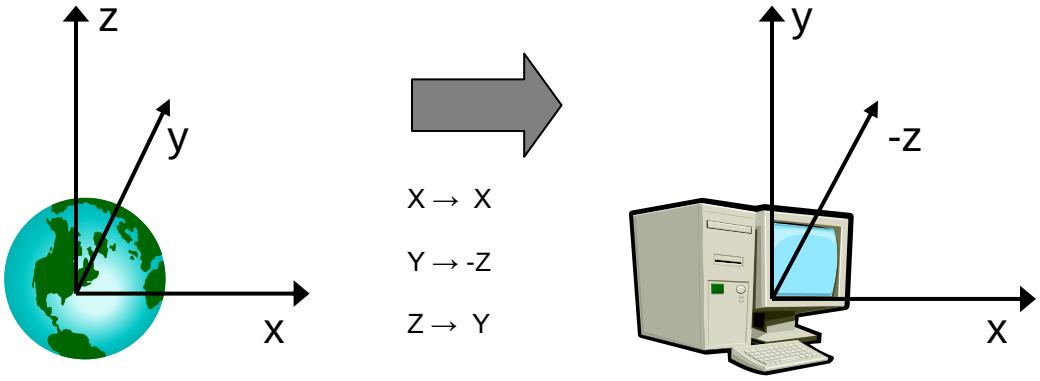

Figure 6 — Axes transformation from real world coordinates (left) to 3D computer graphics coordinate system (right)

8.2 GetScene operation request

8.2.1 GetScene request parameters

A request to perform the GetScene operation shall include the use the data structure specified in Figure 7 and Table 17. This table also specifies the UML model data type, source of values, and multiplicity of each listed parameter, plus the meaning to servers when each optional parameter is not included in the operation request. Although some values listed in the “Names” column appear to contain spaces, they shall not contain spaces.

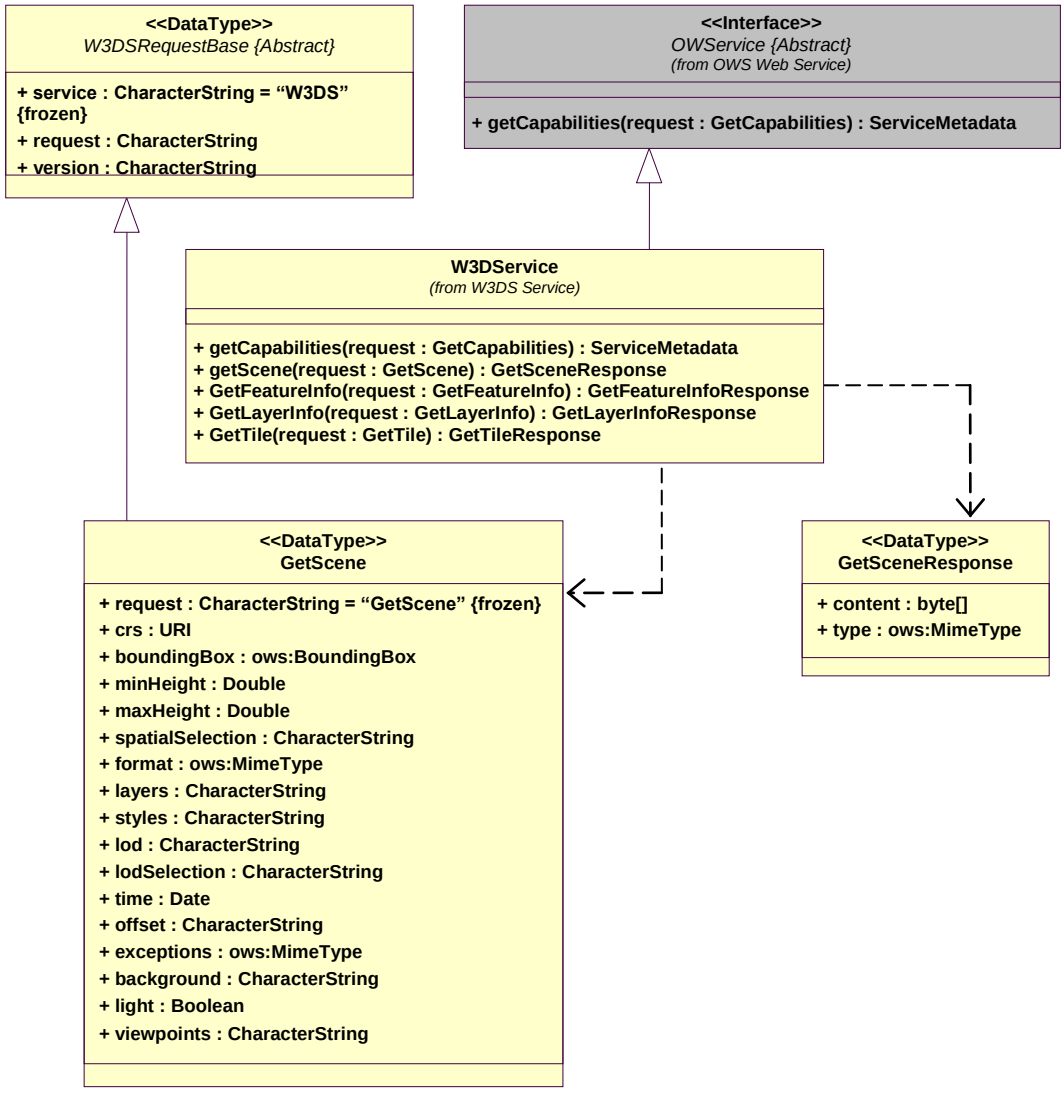

Figure 7 — GetScene operation request UML class diagram

NOTE 1 To reduce the need for readers to refer to other documents, the first three parameters listed below are largely copied from Table 21 in Subclause 9.2.1 of [OGC 06-121r3].

Table 17 — Parameters in GetScene operation request

Names [a]	Definition	Data type and values	Multiplicity and use
service service	Service type identifier	Character String type, not empty Value shall be "W3DS"	One (mandatory)
request request	Operation name	Character String type, not empty Value shall be "GetScene"	One (mandatory)
version version	Standard version for operation	Character String type, not empty Value is specified by each Implementation Standard and Schemas version	One (mandatory)
crs CRS	CRS of the returned scene	URI [b]	One (mandatory)
boundingBox BoundingBox	Bounding rectangle surrounding selected dataset, in available CRS.	BoundingBox data structure, see [OGC 06-121r3] clause 10.2	One (mandatory)
minHeight MinHeight	Vertical lower limit for boundingBox selection criteria	Double type	Zero or one (optional)
maxHeight MaxHeight	Vertical upper limit for boundingBox selection criteria	Double type	Zero or one (optional)
spatialSelection SpatialSelection	Indicates method of selecting objects with BoundingBox	Character String type, not empty. Value is one of "contains_center", "overlaps", "cut". Default is "overlaps".	Zero or one (optional)

format Format	Format encoding of the scene	ows:MimeType, see [OGC 06-121r3] clause 10.5	One (mandatory)
layers Layers	List of layer to retrieve the data from	Character String type, comma separated list of layer Identifiers	One (mandatory)
styles Styles	List of server styles to be applied to the layers	Character String type, comma separated list of style Identifiers. List must have the same length as in parameter Layers	Zero or one (optional)
lods LODs	List of LODs requested for the layer	Character String type, comma separated list of URIs[e]. List must have the same length as in parameter Layers	Zero or one (optional)
lodSelection LODSelection	Indicates method for selecting LODs	Character String type, either "equals" or "equals_or_smaller" or "combined"	Zero or one (optional)
time Time	Date and time	Date type, not empty. Data type defined in ISO 8601:2004(E). Default is "current"	Zero or one (optional)
offset Offset	Offset vector which shall be applied to the scene, i.e. subtracted from the scene	Character String type, list of coordinate value of length 3, separated by comma. Coordinate order in CRS space [c]	Zero or one (optional)
exceptions Exceptions	Format of exceptions	ows:MimeType, see [OGC 06-121r3] clause 10.5	Zero or one (optional)
background Background	Identifier of the background to be used	Character String type, not empty	Zero or one (optional)
light Light	Add light source	Boolean type. Default is false	Zero or one (optional)
viewpoints Viewpoints	Add Viewpoints to choose from	Character String type, list of tuple value, separated by comma.	Zero or one (optional)

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 06-121r3].

b The CRS shall be specified using either the European Petroleum Survey Group form "'EPSG:<POSC Code>" or the URL format defined in OWS Common. Examples: "EPSG:31467", "urn:ogc:def:crs:EPSG:6.3:31467 ", see [OGC 06-121r3] clause 10.3 and [OGC 04-046r3].

c This means that the coordinates are described as in the CRS, usually east, north, up.

d e.g. 0x00FF00 for green

e The LOD shall be specified using a combination of a prefix and a number, e.g. "CityGML:4" for indoor models

NOTE 2 The data type of many parameters is specified as "Character String type, not empty". In the XML Schema Documents specified herein, these parameters are encoded with the xsd:string type, which does NOT require that these strings not be empty.

NOTE 3 The UML class diagram contained in Subclause C.6 provides a useful graphical view of the contents of the GetScene operation request listed in Tables Table 17 - Table 18.

The "Multiplicity and use" columns in Table 17 and Table 18 specify the optionality of each listed parameter and data structure in the GetScene operation request. All the

“mandatory” parameters and data structures shall be implemented by all W3DS clients, using a specified value(s). Similarly, all the “mandatory” parameters and data structures shall be implemented by all W3DS servers, checking that each request parameter or data structure is received with any specified value(s).

All the “optional” parameters and data structures, in the GetScene operation request, should be implemented by all W3DS clients using specified values, for each implemented W3DS to which that parameter or data structure applies. Similarly, all the “optional” parameters and data structures shall be implemented by all W3DS servers, for each implemented W3DS to which that parameter or data structure applies.

8.2.1.1 CRS

The parameter value for the coordinate reference system (CRS) is defined in [OGC 06-121r3] clause 10.3 and [OGC 04-046r3]. When using a 2D CRS, then the height is measured in meters above sea level.

8.2.1.2 BoundingBox

The mandatory BoundingBox parameter allows a Client to request a particular spatial subset. The value of the BoundingBox parameter is a list of comma-separated real numbers in the form "minx,miny,maxx,maxy". The BoundingBox data structure is defined in [OGC 06-121r3] clause 10.2. The units, ordering and direction of increment of the X and Y axes are as defined by the GetScene CRS parameter.

If a request contains an invalid BoundingBox (e.g., one whose minimum X is greater than or equal to the maximum X, or whose minimum Y is greater than or equal to the maximum Y) the server shall throw a service exception. If the Bounding Box values are not defined for the given CRS (e.g., latitudes greater than 90 degrees in CRS:84), the server should return empty content for areas outside the valid range of the CRS.

The default selection method is that any features that are partly or entirely contained in the Bounding Box shall be returned.

8.2.1.3 MinHeight

By indicating a minimum height the selection of the displayed objects can be further limited. Only objects and/or partial objects are displayed, whose elevation coordinate values are larger or equal to the indicated elevation value. If MinHeight is omitted, there is no restriction of the elevation coordinate values downward. MinHeight refers to the original elevation coordinates of the real world objects before any translation or other transformation takes place.

8.2.1.4 MaxHeight

The MaxHeight parameter is analogous to MinHeight but refers to objects whose elevation coordinate values are lower or equal to the indicated elevation value.

8.2.1.5 SpatialSelection

If a spatial selection method other than intersecting the BoundingBox with the feature's spatial extent is desired, the SpatialSelection parameter can be used. The default behavior of selecting a feature is to check the spatial relation between the BoundingBox and the feature's 2D footprint. If the 2D footprint is contained in or intersects with the BoundingBox, then the feature is selected. The according default value of SpatialSelection is "overlaps".

The second possible value of SpatialSelection is "contains_center". This spatial selection method shall check whether the feature's center point is contained or intersects with the BoundingBox. How the center point is computed by the server is not defined, but it shall be inside the convex hull of the feature.

The third possible value of SpatialSelection is "cut". This spatial selection method shall not return features or part of features that lie outside of the BoundingBox. Features that are completely contained in the BoundingBox shall be returned unmodified. Features that intersect with the borders of the BoundingBox shall be split und parts that lie outside cut away. The parts that lie inside of the BoundingBox shall be included in the GetScene response. Multiple GetScene requests with adjacent BoundingBoxes shall generate feature geometries that fit seamlessly together without gaps or cracks.

8.2.1.6 Format

The mandatory Format parameter specifies the target encoding of the returned scene provided as ows:MimeType, see [OGC 06-121r3] clause 10.5. Available formats are described in the server's meta data.

8.2.1.7 Layers

The Layers parameter specifies a comma separated list of layer identifiers to be displayed. The concept of the layer is a metaphor to the traditional (two-dimensional) cartography, with which geo objects of different classes were drawn on different foils resulting in a map with an overall view of this foils. The order in which the layers are listed in the Layers parameter does not influence the visual appearance of the generated scene. However, the order of the lists in the (optional) Styles and LOD parameters shall correlate with the Layers list. Each entry in the Layers comma separated list shall refer to a layer identifier as described in the server's meta data.

8.2.1.8 Styles

The Styles parameter specifies for each layer, which style shall be applied by the server when generating the display elements. The Styles parameter contains a comma separated list of style identifiers. The length of this list shall be equal to the length of the list in the Layers parameter. The order of this list shall correlate with the list in the Layers parameter, meaning that style n shall be applied to layer n. Each entry in this list shall refer to a style identifier as described in the server's meta data. Blanks may be used to indicate the default style.

Example:

```
Layers=dtm,buildings,vegetation&Styles=orthophoto,,simple
```

8.2.1.9 LODs

The parameter LODs specifies for each layer which Level of Detail to choose from when accessing the server's data repository. The parameter value is a comma separated list of URIs referring to the available Levels of Detail as specified in the server's meta data layer section. The length of this list shall be equal to the length of the list in the Layers parameter . The order of this list shall correlate with the list in the Layers parameter, meaning that LOD n shall be selected from layer n.

Each entry in this list shall be a URI consisting of a prefix and the actual numeric value, separated by a colon, e.g. "CityGML:1". The prefix indicates the spectrum of possible values and how these values must be interpreted. The LOD definitions of CityGML can be found in [08-007r1] clause 6.2.

8.2.1.10 LODSelection

In conjunction with the LOD parameter, the LODSelection parameter may be used for telling the server how to interpret the LOD value for each layer. Three selection methods are available: "equals", "equals_or_smaller", and "combined".

The method "equals" is the default selection method. For each feature, the available LODs are compared with the LOD value provided in the GetScene operation. If the specified LOD is available for this feature, then the according model shall be selected and included in the scene. If the specified LOD is not available for this feature, then the feature shall be omitted in the scene altogether.

The method "equals_or_smaller" causes the server to compile a scene from multiple available LODs. Only one LOD for each feature shall be selected. If the specified LOD is available for this feature, then the according model shall be selected and included in the scene. If the specified LOD is not available for this feature, then the server shall select the next lower LOD available for this feature. If no LOD equal or lower than the specified LOD is available for this feature, then this feature shall be omitted in the scene altogether. Cumulative LOD models, e.g. building blocks representing multiple buildings as a single geometry, shall be handled so that no overlaps with higher LODs occur. If a higher LOD for one building included in the block is available, then the block shall be omitted.

The method "combined" causes the server to include multiple LODs for each feature in the scene, if available. Multiple LODs are combined in a "LOD" node which switches between multiple child models depending on the current distance to the viewpoint. The availability of this method depends on the requested format, the "LOD" node is supported by X3D. The server shall include for each feature all LODs equal or smaller than the specified LOD value in the GetScene operation. Cumulative LOD models shall be handled so that overlaps with higher LODs occur.

8.2.1.11 Time

The Time parameter can be used to include a temporal criterion to the feature selection method. The server shall select only features that exist at the given point in time if the layer meta data contains StartTime and/or EndTime elements. In this case the Time parameter includes a date which may be the present system time or a historical date.

The specification of valid parameter values is in accordance with ISO 8601:2004(E). The special keyword "Time=current" may be used to indicate the most current data available at the current timestamp. This is also the default value.

Examples:

```
Example 1: Time=2003-10-25
Specifies the 25th of October 2003

Example 2: Time=2003-10-25T14:28:43Z
Specifies the 25th of October 2003 at 14:28 hours and 43 seconds. Z
means Zulu time which is equivalent to UTC.

Example 3: Time=-1279
Means year 1279 B.C. Ramses II becomes pharaoh of Egypt
```

8.2.1.12 Offset

Depending on the used CRS, coordinate values of 3D objects may become very big. For example UTM coordinates require a mantissa of 9 digits for achieving accuracy in centimeters. Clients using single precision floating point numbers (32 bit) are not able to handle large coordinates very well. The Offset parameter can be used in order to define a point in 3D which is subtracted from all coordinate values, thus reducing the number of digits. This is useful if no format with geo extension is used.

8.2.1.13 Exceptions

The optional Exceptions parameter specifies the behavior of the server upon detecting an error, e.g. invalid request or internal server error. The default value is "text/xml". The value shall be one of the MIME types offered in the server's meta data parameter value, see 7.3.2.2.

If the Exceptions parameter is set to "blank", then the server shall, upon detecting an error, return a document of the MIME type specified in the format parameter whose content is uniformly "off", i.e. a document with correct headers according to the format and zero content. This "silent" mode is useful if the scene graph in the client is composed of a large number W3DS GetScene requests and is not prepared to process server exceptions.

8.2.1.14 Background

The optional parameter Background can be used to include a predefined background advertised by the server in its meta data. The value of this parameter refers to an existing background identifier of the server. The background node will be included in the returned

scene. No specific node type is specified, it is up to the server how to implement the background, it may consist of a simple color, a color gradient, a cube or sphere with a projected image, videos, or any scene graph node that is supported by the requested format.

8.2.1.15 Light

The optional parameter Light can be used to include a light source in the scene. Most 3D viewers add a headlight if they find no light source in the scene. In order to enable proper lighting a global light source must be defined. If this parameter is set to true, then the server shall add direct and indirect light components simulating the real solar and atmospheric conditions.

8.2.1.16 Viewpoints

The optional parameter Viewpoints can be used to include a list of viewpoints in the scene the user can choose from. Viewpoints are presented in most viewers as list which can be used to move quickly to another pre-defined position and view direction. If multiple viewpoints are defined, then the first in the list will be used as default viewpoint when loading the scene.

The parameter value contains a comma separated list of tuple values for defining for each viewpoint a) a name, b) the Point of Interest (POI), c) the Point of Camera (POC), d) the up vector (UP), and e) the Field of View (FOV). Thus 11 Values are used to define one viewpoint. In case of multiple viewpoints, all values are appended to the list consecutively. The list of Viewpoints parameter values shall be divisible by 11.

Template:

Viewpoints=NAME,POI_x,POI_y,POI_z,POC_x,POC_y,POC_z,UP_x,UP_y,UP_z,FOV

All tuple values refer to the spatial reference system specified by the CRS parameter. The FOV is given as angle value.

8.2.2 GetScene request KVP encoding (optional)

Servers may implement HTTP GET transfer of the GetScene operation request, using KVP encoding. The KVP encoding of the GetScene operation request shall use the parameters specified in Table 18. The parameters listed in Table 18 shall be as specified in Table 17 above.

Table 18 — GetScene operation request URL parameters

Name and example [a]	Optionality and use	Definition and format
service=W3DS	Mandatory	Service type identifier
request= GetScene	Mandatory	Operation name

version=0.4.0	Mandatory	Standard and schema version for this operation
crs=EPSG:26916	Mandatory	Coordinate Reference System of the returned scene
boundingbox=202759.0,3310170.0,213200.0,3320896.0	Mandatory	Bounding rectangle surrounding selected dataset, in available CRS.
minheight=0.0	Optional	Vertical lower limit for boundingBox selection criteria
maxheight=1000.0	Optional	Vertical upper limit for boundingBox selection criteria
spatialselection=contains_center	Optional	Indicates method of selecting objects with BoundingBox
format=model/x3d	Mandatory	Format encoding of the scene
layers=dem,bldgs	Mandatory	List of layer to retrieve the data from
styles=default,Cartographic	Optional	List of server styles to be applied to the layers
lods=,CityGML:1	Optional	List of LODs requested for the layer
lodselection=equals_or_smaller	Optional	Indicates method for selecting LODs
time=2009-07-13-07:00	Optional	Date and time
offset=202000,3310000,0	Optional	Offset vector which shall be applied to the scene, i.e. subtracted from the scene
exceptions=text/xml	Optional	Format of exceptions
background=sky1	Optional	Identifier of the background to be used
light=true	Optional	Add light source
Viewpoints=vpoint1,202000,3310000,200, 202000,3305000,200,0,1,0,60	Optional	Add Viewpoints to choose from

a All parameter names are here listed using mostly lower case letters. However, any parameter name capitalization shall be allowed in KVP encoding, see Subclause 11.5.2 of [OGC 06-121r3].

EXAMPLE An example GetScene operation request KVP encoded for HTTP GET is:

```
http://hostname:port/path?SERVICE=W3DS&REQUEST=GetScene&VERSION=0.4.0&CRS=EPSG:26916&FORMAT=model/x3d&BoundingBox=202759.0,3310170.0,213200.0,3320896.0&LAYERS=Terrain&STYLES=aerial_view_RGB
```

8.2.3 GetScene request XML encoding (mandatory)

All W3DS servers shall implement HTTP POST transfer of the GetScene operation request, using XML encoding only. The following schema fragment specifies the contents and structure of a GetScene operation request encoded in XML:

```
    <!-- ======================================================== -->
<element name="GetScene" type="w3ds:GetSceneType"/>
<!-- ======================================================== -->
<complexType name="GetSceneType">
    <annotation>
```



```
            <documentation>XML encoded W3DS GetScene operation request.
</documentation>
        </annotation>
        <complexContent>
            <extension base="w3ds:W3DSRequestBaseType">
                <sequence>
                    <element name="CRS" type="anyURI"/>
                    <element ref="ows:BoundingBox"/>
                    <element name="MinHeight" type="double" minOccurs="0"/>
                    <element name="MaxHeight" type="double" minOccurs="0"/>
                    <element name="SpatialSelection" type="w3ds:SpatialSelectionType"
minOccurs="0"/>
                    <element name="Format" type="ows:MimeType" minOccurs="0"/>
                    <element name="Layers" type="w3ds:IdentifierListType"/>
                    <element name="Styles" type="w3ds:IdentifierListType" minOccurs="0"/>
                    <element name="LODs" type="w3ds:IdentifierListType" minOccurs="0"/>
                    <element name="LODSelection" type="w3ds:LODSelectionType"
minOccurs="0"/>
                    <element name="Time" type="dateTime" minOccurs="0"/>
                    <element name="Offset" type="w3ds:PositionType3D" minOccurs="0"/>
                    <element name="Exceptions" type="ows:MimeType" minOccurs="0"/>
                    <element name="Background" type="ows:CodeType" minOccurs="0"/>
                    <element name="Light" type="boolean" minOccurs="0"/>
                    <element name="Viewpoints" type="w3ds:ViewpointListType"
minOccurs="0"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
```

EXAMPLE An example GetScene operation request XML encoded for HTTP POST is:

```
    <?xml version="1.0" encoding="UTF-8"?>
    <w3ds:GetScene service="W3DS" request="GetScene" version="0.4.0">
        <w3ds:CRS>EPSG:26916</w3ds:CRS>
        <ows:BoundingBox crs="EPSG:26916">
            <ows:LowerCorner>202759.03723318398
3310170.2612183597</ows:LowerCorner>
            <ows:UpperCorner>213200.2944077917
3320896.6813003295</ows:UpperCorner>
        </ows:BoundingBox>
        <w3ds:Format>model/x3d</w3ds:Format>
        <w3ds:Layers>
            <ows:Identifier>dem</ows:Identifier>
            <ows:Identifier>bldgs</ows:Identifier>
        </w3ds:Layers>
        <w3ds:Styles>
            <ows:Identifier>default</ows:Identifier>
            <ows:Identifier>Cartographic</ows:Identifier>
        </w3ds:Styles>
    </w3ds:GetScene>
```

8.3 GetScene operation response

The response to a valid GetScene request is a document of the MIME type as specified in the Format parameter. The document contains a 3D scene assembled from the features of

the selected layers within the specified BoundingBox, converted into the specified Format and CRS. The coordinate axes will be usually shifted to computer graphics coordinate axes as shown in Figure 5Figure 2.

If the GetScene request contains a list of server style identifiers or user defined styles, then the features from the individual layers are portrayed according to the respective style. In case of server defined styles, the method of how to modify the appearance and other properties of the features may not be transparent to the client. Therefore it is recommended to include a detailed style description in the server's meta data. In case of user styles included in the GetScene request, the client has full control over the styling.

Some output formats other than the default X3D format may not support elements which are referred to in the parameters Environment, BGColor, and the parameters for defining a virtual camera/viewpoint. For instance a light node may not be available in formats focusing on the geometry description. In this case these parameters may be ignored, and no exception shall be thrown. If the output format allows a camera definition, then a default camera is included in the scene, whose location is derived from the provided Viewpoint parameters.

If the resulting file contains references to additional files which must be loaded by the client, for instance URLs to textures, then those files must be accessible to the client, either stored on the same server as the W3DS or on another server accessible over the Internet. Relative paths in the URL without protocol and server domain may be used for referencing to files in the directory of the W3DS or in subdirectories. Contained URLs may also point to other service end points producing the required MIME type, for instance a WCS may be used as source for terrain textures. Scene graph nodes within the GetScene response might also point to other W3DS servers, using include statements. The configuration and design of how to access the contents of the scene, is up to the system administrator.

In case of an incorrect request or an occurring error while generating the scene, the response must be supplied in the requested format for exceptional cases (parameters EXCEPTION) by the server. With a request via HTTP the MIME type of the returned document must be set by the server according to contents of the resulting file.

8.3.1 GetScene exceptions

When a W3DS server encounters an error while performing a GetScene operation, it shall return an exception report message as specified in Subclause 7.4 of [OGC 06-121r3]. The allowed standard exception codes shall include those listed in Table 19. For each listed exceptionCode, the contents of the "locator" parameter value shall be as specified in the right column of Table 19.

NOTE To reduce the need for readers to refer to other documents, the first three values listed below are copied from Table 20 in Subclause 8.3 of [OGC 06-121r3].

Table 19 — Exception codes for GetScene operation

exceptionCode value	Meaning of code	"locator" value

MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value
OptionNotSupported	Request is for an option that is not supported by this server	Identifier of option not supported
CRSNotSupported	Operation request contains a value in the CRS parameter which is not supported by the server	Name of unsupported CRS
InvalidCameraDefinition	Operation request contains an invalid camera definition	None, omit "locator" parameter
InvalidListLength	Operation request contains Style or LOD parameter whose value does not contain a list of the same length as in parameter Layers	Name of parameter with invalid value
UnknownLayer	Operation request contains an identifier in the Layers parameter which is unknown to the server	Identifier of invalid layer
UnknownStyle	Operation request contains an identifier in the Style parameter which is unknown to the server	Identifier of invalid style
LODNotSupported	Operation request contains a value in the LOD parameter which is not available for the according layer	Name of unsupported LOD
FormatNotSupported	Operation request contains a MIME type in the Format parameter which is not supported by the server	Name of unsupported format
ExceptionNotSupported	Operation request contains a value in the Exception parameter which is not supported by the server	Name of unsupported exception format
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter

9 GetFeatureInfo operation (optional)

9.1 Introduction

The GetFeatureInfo operation is designed to provide clients with additional attribute information about features within a scene that is currently displayed. The canonical use case for GetFeatureInfo is that a user explores the response of a GetScene request and points at an object within the scene for which to obtain more information. The concept of this operation is that the client determines a location in 3D space by clicking on an object and calculates either the intersection point of the object geometry with the picking ray or the center point of the object and submits this location together with additional parameters to the server. The location can be also determined by other 3D input devices or by any other means. Since the W3DS protocol is stateless, also the current CRS needs to be submitted so that the W3DS is able to reconstruct the location within the CRS of its data store. Also a list of layers shall be provided with the GetFeatureInfo request so that the search for attribute information can be restricted to selected data sets.

The actual method of how the W3DS server decides which features should be selected based on the location, what kind of information is returned, and how this information is structured is left up to the W3DS provider. However, the format of the response must be provided as MIME type, e.g. text/xml, enabling the client to parse the result properly.

The GetFeatureInfo operation is only supported for those Layers for which the attribute queryable="1" (true) has been defined or inherited. A client shall not issue a GetFeatureInfo request for other layers. A W3DS shall respond with a properly formatted service exception response (code = OperationNotSupported) if it receives a GetFeatureInfo request but does not support it.

9.2 GetFeatureInfo operation request

9.2.1 GetFeatureInfo request parameters

A request to perform the GetFeatureInfo operation shall include the data structure specified in Figure 7 and Table 20. This table also specifies the UML model data type, source of values, and multiplicity of each listed parameter, plus the meaning to servers when each optional parameter is not included in the operation request. Although some values listed in the “Names” column appear to contain spaces, they shall not contain spaces.

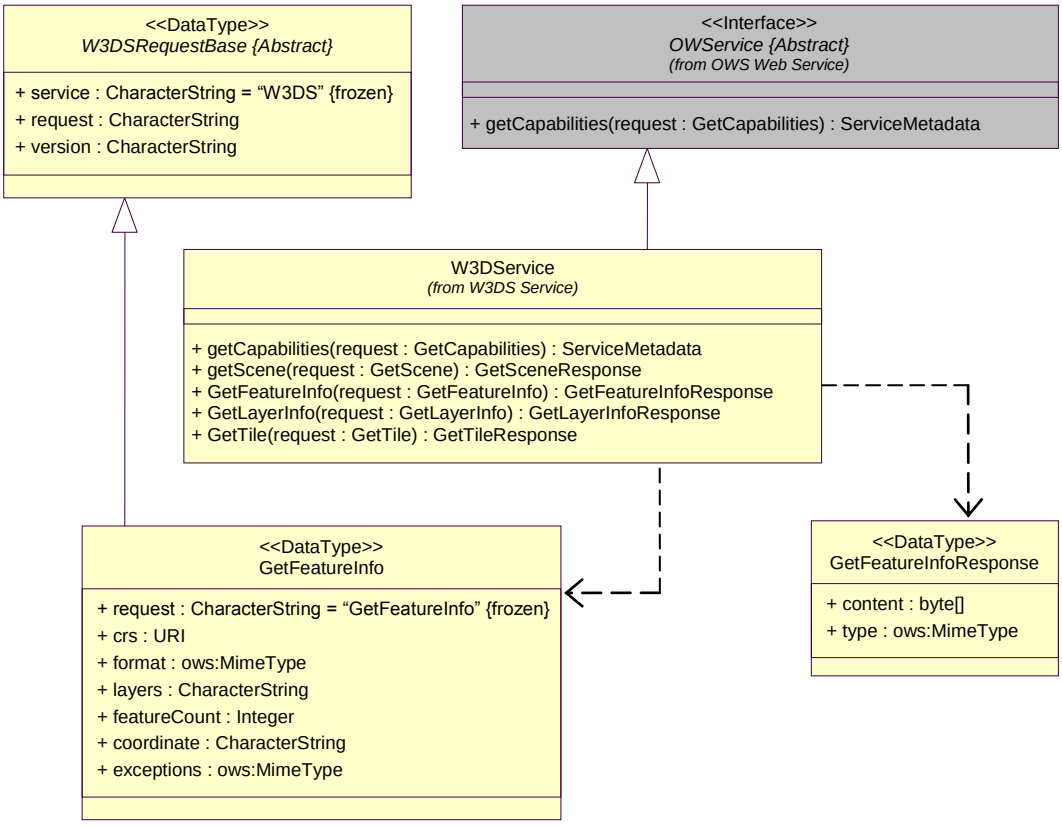

Figure 8 — GetFeatureInfo operation request UML class diagram

NOTE 1 To reduce the need for readers to refer to other documents, the first three parameters listed below are largely copied from Table 21 in Subclause 9.2.1 of [OGC 06-121r3].

Table 20 — Parameters in GetFeatureInfo operation request

Names [a]	Definition	Data type and values	Multiplicity and use
service service	Service type identifier	Character String type, not empty Value shall be "W3DS"	One (mandatory)
request request	Operation name	Character String type, not empty Value shall be "GetScene"	One (mandatory)
version version	Standard version for operation	Character String type, not empty Value is specified by each Implementation Standard and Schemas version	One (mandatory)
crs CRS	CRS of the returned scene	URI [b]	One (mandatory)
format Format	Format encoding of the result	ows:MimeType, see [OGC 06-121r3] clause 10.5	One (mandatory)
layers Layers	List of layer to retrieve the data from	Character String type, comma separated list of layer Identifiers	One (mandatory)
featurecount FeatureCount	Number of features to return information from.	Integer type, default=1	Zero or one (optional)
coordinate Coordinate	Position used to search for features to return information from	Character String type, list of coordinate value of length 3, divided by comma. Coordinate order in CRS space [c]	One (mandatory)
exceptions Exceptions	Format of exceptions	ows:MimeType, see [OGC 06-121r3] clause 10.5	Zero or one (optional)

a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.

b The CRS shall be specified using either the European Petroleum Survey Group form "'EPSG:<POSC Code>" or the URL format defined in OWS Common. Examples: "EPSG:31467", "urn:ogc:def:crs:EPSG:6.3:31467 ", see [OGC 06-121r3] clause 10.3 and [OGC 04-046r3].

c This means that the coordinates are described as in the CRS, usually east, north, up.

NOTE 2 The data type of many parameters is specified as "Character String type, not empty". In the XML Schema Documents specified herein, these parameters are encoded with the xsd:string type, which does NOT require that these strings not be empty.

NOTE 3 The UML class diagram contained in Subclause C.7 provides a useful graphical view of the contents of the GetFeatureInfo operation request listed in Tables Table 20 - Table 21.

The "Multiplicity and use" columns in Table 20 and Table 21 specify the optionality of each listed parameter and data structure in the GetFeatureInfo operation request. All the "mandatory" parameters and data structures shall be implemented by all W3DS clients,

using a specified value(s). Similarly, all the "mandatory" parameters and data structures shall be implemented by all W3DS servers, checking that each request parameter or data structure is received with any specified value(s).

All the "optional" parameters and data structures, in the GetFeatureInfo operation request, should be implemented by all W3DS clients using specified values, for each implemented W3DS to which that parameter or data structure applies. Similarly, all the "optional" parameters and data structures shall be implemented by all W3DS servers, for each implemented W3DS to which that parameter or data structure applies.

9.2.1.1 CRS

The parameter value for the coordinate reference system (CRS) is defined in [OGC 06-121r3] clause 10.3 and [OGC 04-046r3]. When using a 2D CRS, then the height is measured in meters above sea level.

9.2.1.2 Layers

The Layers parameter specifies a comma separated list of layer identifiers from which the server spatially selects features based on the coordinate, and collects attribute information from. The order in which the layers are listed in the Layers parameter determines the order in which the feature information will be displayed in the GetFeaturInfo response. Each entry in the Layers comma separated list shall refer to a layer identifier as described in the server's meta data.

9.2.1.3 Format

The mandatory Format parameter specifies the target encoding of the returned attribute information provided as ows:MimeType, see [OGC 06-121r3] clause 10.5. Available formats are described in the server's meta data.

9.2.1.4 FeatureCount

The optional FeatureCount parameter specifies the maximum number of features per layer for which feature information shall be returned. The parameter value shall be a positive integer. The default value is 1 if this parameter is omitted or is other than a positive integer.

9.2.1.5 Coordinate

The mandatory Coordinate parameter provides a location in 3D space within the scene from which feature information will be generated. The parameter value is a list of comma separated floating point numbers describing the x,y, and z values of the coordinate. Note that x, y, z values are not in computer graphics coordinate system, but they shall have the same axis orientation and direction as defined in the CRS parameter. This location should be within or at the border of a feature geometry, but it does not have to be. The W3DS server shall detect the feature(s) which geometry is containing the location or lying nearest to it.

9.2.1.6 Exceptions

The optional Exceptions parameter specifies the behavior of the server upon detecting an error, e.g. invalid request or internal server error. The default value is “text/xml”. The value shall be one of the MIME types offered in the server’s meta data parameter value, see 7.3.2.2.

9.2.2 GetFeatureInfo request KVP encoding (optional)

Servers may implement HTTP GET transfer of the GetFeatureInfo operation request, using KVP encoding. The KVP encoding of the GetFeatureInfo operation request shall use the parameters specified in Table 21. The parameters listed in Table 21 shall be as specified in Table 20 above.

Table 21 — GetFeatureInfo operation request URL parameters

Name and example [a]	Optionality and use	Definition and format
service=W3DS	Mandatory	Service type identifier
request=GetFeatureInfo	Mandatory	Operation name
version=0.4.0	Mandatory	Standard and schema version for this operation
crs=EPSG:26916	Mandatory	Coordinate Reference System of the returned scene
format=text/html	Mandatory	Format encoding of the result
layers=bldgs	Mandatory	List of layer to retrieve the data from
featurecount=5	Optional	Number of features to return information from (default=1)
coordinate=202042.233,3310094.983,334.0	Mandatory	Position used to search for features to return information from
exceptions=text/xml	Optional	Format of exceptions

a All parameter names are here listed using mostly lower case letters. However, any parameter name capitalization shall be allowed in KVP encoding, see Subclause 11.5.2 of [OGC 06-121r3].

EXAMPLE An example GetFeatureInfo operation request KVP encoded for HTTP GET is:

```
http://hostname:port/path?SERVICE=W3DS&REQUEST=GetFeatureInfo&VERSION=0
.4.0&CRS=EPSG:26916&FORMAT=text/xml&LAYERS=bldgs&FEATURECOUNT=5&COORDIN
ATE=202042.233,3310094.983,334.0&EXCEPTIONS=text/html
```

9.2.3 GetFeatureInfo request XML encoding (mandatory)

All W3DS servers shall implement HTTP POST transfer of the GetFeatureInfo operation request, using XML encoding only. The following schema fragment specifies the contents and structure of a GetFeatureInfo operation request encoded in XML:

```
<!-- ========================================================== -->
<element name="GetFeatureInfo" type="w3ds:GetFeatureInfoType"/>
<!-- ========================================================== -->
<complexType name="GetFeatureInfoType">
    <annotation>
```



```
            <documentation>XML encoded W3DS GetFeatureInfo operation request.
</documentation>
        </annotation>
        <complexContent>
            <extension base="w3ds:W3DSRequestBaseType">
                <sequence>
                    <element name="CRS" type="anyURI"/>
                    <element ref="ows:BoundingBox"/>
                    <element name="Format" type="ows:MimeType"/>
                    <element name="Layers" type="w3ds:IdentifierListType"/>
                    <element name="FeatureCount" type="integer"  minOccurs="0"/>
                    <element name="Coordinate" type="w3ds:PositionType3D"/>
                    <element name="Exceptions" type="ows:MimeType" minOccurs="0"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
```

EXAMPLE An example GetFeatureInfo operation request XML encoded for HTTP POST is:

```
<?xml version="1.0" encoding="UTF-8"?>
<w3ds:GetFeatureInfo service="W3DS" request="GetFeatureInfo"
version="0.4.0">
    <w3ds:CRS>EPSG:26916</w3ds:CRS>
    <w3ds:Format>text/html</w3ds:Format>
    <w3ds:Layers>
        <ows:Identifier>bldgs</ows:Identifier>
    </w3ds:Layers>
    <w3ds:FeatureCount>5</w3ds:FeatureCount>
    <w3ds:Coordinate>202042.233 3310094.983 334.0</w3ds:Coordinate>
    <w3ds:Exceptions>text/html</w3ds:Exceptions>
</w3ds:GetFeatureInfo>
```

9.3 GetFeatureInfo operation response

The normal response to a valid GetFeatureInfo operation request shall be a table or equivalent data structure in which the attribute information of each found feature is listed. Each row in this table describes one feature. The response is a document encoded in the MIME type as specified in the Format parameter.

9.3.1 GetFeatureInfo response example

A GetFeatureInfo operation response for W3DS can look like this encoded in text/html:

```
<html>
  <head></head>
  <body>
    <table width="95%" >
      <tr>
        <th bgcolor='#ffffff'><b>Buildings</b></td>
      </tr>
      <tr>
```



```
        <th bgcolor='#fafafa'></td>
        <th bgcolor='#dddddd'>id </td>
        <th bgcolor='#dddddd'>objkey </td>
        <th bgcolor='#dddddd'>strkey </td>
        <th bgcolor='#dddddd'>hsno </td>
        <th bgcolor='#dddddd'>description </td>
        <th bgcolor='#dddddd'>sockelhoehe </td>
        <th bgcolor='#dddddd'>traufhoehe </td>
        <th bgcolor='#dddddd'>height   </td>
      </tr>
      <tr>
        <td bgcolor='#fafafa'>1</td>
        <td bgcolor='#fafafa'>34891270</td>
        <td bgcolor='#fafafa'>1123 Hochschulgebaeude</td>
        <td bgcolor='#fafafa'>2390</td>
        <td bgcolor='#fafafa'>1</td>
        <td bgcolor='#fafafa'>Alte Universitaet</td>
        <td bgcolor='#fafafa'>2.500000</td>
        <td bgcolor='#fafafa'>15.000000</td>
        <td bgcolor='#fafafa'>115.347270</td>
      </tr>
      <tr>
        <td bgcolor='#fafafa'></td>
      </tr>
    </table>
  </body>
</html>
```

9.3.2 GetFeatureInfo exceptions

When a W3DS server encounters an error while performing a GetFeatureInfo operation, it shall return an exception report message as specified in Subclause 7.4 of [OGC 06-121r3]. The allowed standard exception codes shall include those listed in Table 22. For each listed exceptionCode, the contents of the "locator" parameter value shall be as specified in the right column of Table 22.

NOTE To reduce the need for readers to refer to other documents, the first four values listed below are copied from Table 20 in Subclause 8.3 of [OGC 06-121r3].

Table 22 — Exception codes for GetFeatureInfo operation

exceptionCode value	Meaning of code	"locator" value
OperationNotSupported	Request is for an operation that is not supported by this server	Name of operation not supported
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value
OptionNotSupported	Request is for an option that is not supported by this server	Identifier of option not supported
CRSNotSupported	Operation request contains a value in the CRS parameter which is not supported by the server	Name of unsupported CRS

UnknownLayer	Operation request contains an identifier in the Layers parameter which is unknown to the server	Identifier of invalid layer
FormatNotSupported	Operation request contains a MIME type in the Format parameter which is not supported by the server	Name of unsupported format
ExceptionNotSupported	Operation request contains a value in the Exception parameter which is not supported by the server	Name of unsupported exception format
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter

10 GetLayerInfo operation (optional)

10.1 Introduction

The purpose of the GetLayerInfo request is to collect information on the available attribute names and the values in the attribute table of a specific layer. The attribute table is managed by the W3DS server in a database table, as dbf file, or otherwise. The entries in the attribute table can be linked to the geometries that can be retrieved using the GetScene request.

The GetLayerInfo operation is optional. A W3DS server shall respond with a properly formatted service exception (XML) response (code = OperationNotSupported) if it receives a GetLayerInfo request but does not support it.

The GetLayerInfo request contains a mandatory Layer parameter for identifying the data set from which attribute information shall be received and an optional ColumnName parameter. If only the Layer parameter is used, then the response of the request shall contain only a list of all available attribute or column names in the format specified by the Format parameter. The received attribute names can then be used to receive additional information on the available values in the attribute table. If additionally to the Layer parameter also a ColumnName parameter is present, then the attribute table shall be queried for all available values that the features in the layer may have. The response to such a request contains a full list of unique values. In this list no duplicate values must occur. Examples of XML formatted responses are given in 10.3.3.

10.2 GetLayerInfo operation request

10.2.1 GetLayerInfo request parameters

A request to perform the GetLayerInfo operation shall include the use the data structure specified in Figure 8 and Table 23. This table also specifies the UML model data type, source of values, and multiplicity of each listed parameter, plus the meaning to servers when each optional parameter is not included in the operation request. Although some values listed in the "Names" column appear to contain spaces, they shall not contain spaces.

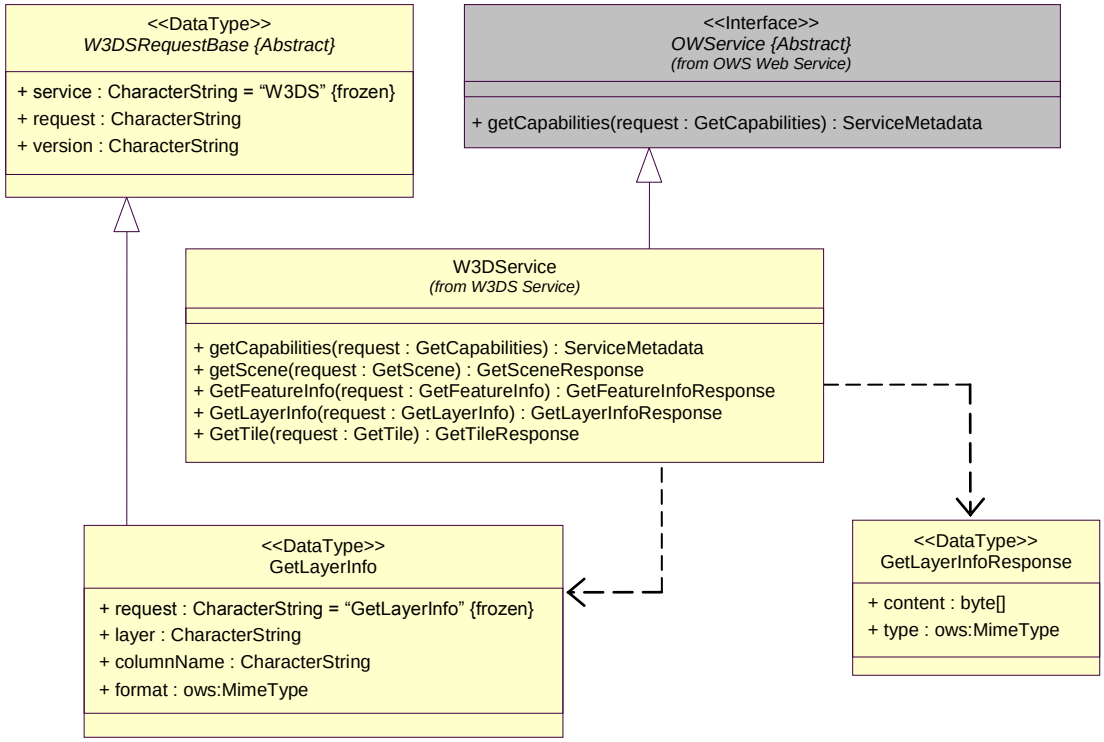

Figure 9 — GetLayerInfo operation request UML class diagram

NOTE 1 To reduce the need for readers to refer to other documents, the first three parameters listed below are largely copied from Table 21 in Subclause 9.2.1 of [OGC 06-121r3].

Table 23 — Parameters in GetLayerInfo operation request

Names [a]	Definition	Data type and values	Multiplicity and use
service service	Service type identifier	Character String type, not empty Value shall be "W3DS"	One (mandatory)
request request	Operation name	Character String type, not empty Value shall be "GetScene"	One (mandatory)
version version	Standard version for operation	Character String type, not empty Value is specified by each Implementation Standard and Schemas version	One (mandatory)
layer Layer	The layer to get information from	Character String type, not empty	One (mandatory)
columnnames ColumnNames	A list of column names to get information from	Character String type, comma separated list of column names	Zero or one (optional)

format Format	Format encoding of the result	ows:MimeType, see [OGC 06-121r3] clause 10.5	One (mandatory)
exceptions Exceptions	Format of exceptions	ows:MimeType, see [OGC 06-121r3] clause 10.5	Zero or one (optional)
a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.			

NOTE 2 The data type of many parameters is specified as "Character String type, not empty". In the XML Schema Documents specified herein, these parameters are encoded with the xsd:string type, which does NOT require that these strings not be empty.

NOTE 3 The UML class diagram contained in Subclause C.8 provides a useful graphical view of the contents of the GetLayerInfo operation request listed in Tables Table 23 - Table 24.

The "Multiplicity and use" columns in Table 23 through Table 29 specify the optionality of each listed parameter and data structure in the GetLayerInfo operation request. All the "mandatory" parameters and data structures shall be implemented by all W3DS clients, using a specified value(s). Similarly, all the "mandatory" parameters and data structures shall be implemented by all W3DS servers, checking that each request parameter or data structure is received with any specified value(s).

All the "optional" parameters and data structures, in the GetLayerInfo operation request, should be implemented by all W3DS clients using specified values, for each implemented W3DS to which that parameter or data structure applies. Similarly, all the "optional" parameters and data structures shall be implemented by all W3DS servers, for each implemented W3DS to which that parameter or data structure applies.

10.2.1.1 Layer

The mandatory Layer parameter specifies the layer from which information shall be retrieved. The parameter value shall refer to a layer identifier as described in the server's meta data. If information from multiple layers needs to be collected, then multiple requests must be sent to the W3DS server. If the layer identifier is not defined in the server's meta data, then server shall issue a service exception (code = UnknownLayer).

10.2.1.2 ColumnNames

The optional ColumnNames parameter specifies one or several table column(s) or attribute name(s) of the selected layer in the Layer parameter from which all available unique values shall be retrieved. The value is a comma-separated list of one or more attribute names. If all available attributes of a layer shall be queried, then the value of the ColumnNames parameter can be set to "ALLINFO". If any layer in the ColumnNames parameter is not defined in the service metadata of the server, then the server shall issue a service exception (code = ColumnNameNotDefined).

10.2.1.3 Format

The mandatory Format parameter specifies the target encoding of the returned column and attribute information provided as ows:MimeType, see [OGC 06-121r3] clause 10.5. Available formats are described in the server's meta data.

10.2.2 GetLayerInfo request KVP encoding (optional)

Servers may implement HTTP GET transfer of the GetLayerInfo operation request, using KVP encoding. The KVP encoding of the GetLayerInfo operation request shall use the parameters specified in Table 24. The parameters listed in Table 24 shall be as specified in Table 23 above.

Table 24 — GetLayerInfo operation request URL parameters

Name and example [a]	Optionality and use	Definition and format
service=W3DS	Mandatory	Service type identifier
request= GetLayerInfo	Mandatory	Operation name
version=0.4.0	Mandatory	Standard and schema version for this operation
layer=Buildings	Mandatory	The layer to get information from
columnnames=id,category	Optional	A list of column names to get information from
format=text/xml	Mandatory	Format encoding of the result
exceptions=text/xml	Optional	Format of exceptions

a All parameter names are here listed using mostly lower case letters. However, any parameter name capitalization shall be allowed in KVP encoding, see Subclause 11.5.2 of [OGC 06-121r3].

EXAMPLE An example GetLayerInfo operation request KVP encoded for HTTP GET is:

```
http://hostname:port/path?SERVICE=W3DS&REQUEST=GetLayerInfo&VERSION=0.4
.0&LAYER=Buildings&COLUMNNAMES=id,category&FORMAT=text/xml
```

10.2.3 GetLayerInfo request XML encoding (mandatory)

All W3DS servers shall implement HTTP POST transfer of the GetLayerInfo operation request, using XML encoding only. The following schema fragment specifies the contents and structure of a GetLayerInfo operation request encoded in XML:

```
<!-- ========================================================== -->
<element name="GetLayerInfo" type="w3ds:GetLayerInfoType"/>
<!-- ========================================================== -->
<complexType name="GetLayerInfoType">
    <annotation>
        <documentation>XML encoded W3DS GetLayerInfo operation request.
</documentation>
    </annotation>
    <complexContent>
        <extension base="w3ds:W3DSRequestBaseType">
            <sequence>
                <element name="Layer" type="ows:CodeType"/>
                <element name="ColumnNames" type="w3ds:IdentifierListType"/>
                <element name="Format" type="ows:MimeType"/>
                <element name="Exceptions" type="ows:MimeType" minOccurs="0"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```


EXAMPLE An example GetLayerInfo operation request XML encoded for HTTP POST is:

```
<?xml version="1.0" encoding="UTF-8"?>
<w3ds:GetLayerInfo service="W3DS" request="GetLayerInfo"
version="0.4.0">
    <w3ds:Layer>bldgs</w3ds:Layer>
    <w3ds:ColumnNames>
        <ows:Identifier>id</ows:Identifier>
        <ows:Identifier>category</ows:Identifier>
    </w3ds:ColumnNames>
    <w3ds:Format>text/xml</w3ds:Format>
    <w3ds:Exceptions>text/xml</w3ds:Exceptions>
</w3ds:GetLayerInfo>
```

10.3 GetLayerInfo operation response

10.3.1 Normal response parameters

The normal response to a valid GetLayerInfo operation request shall be document encoded in MIME type text/xml containing attribute names and optionally attribute values of the selected layer. More precisely, a response from the GetLayerInfo operation shall include the parts listed in Table 25. This table also specifies the UML model data type plus the multiplicity and use of each listed part.

Table 25 — Parts of GetLayerInfo operation response

Names [a]	Definition	Data type and values	Multiplicity and use
layerinfo LayerInfo	Base xml element	LayerInfo data type	One (mandatory)
a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.			

Table 26 — Parts of LayerInfo data structure

Names	Definition	Data type and values	Multiplicity and use
li_layer LI_Layer	Layer from which attribute data was collected	LI_Layer data type	One (mandatory)

Table 27 — Parts of LI_Layer data type

Names	Definition	Data type and values	Multiplicity and use
identifier Identifier	Identifier of this layer	Character String type, not empty	One (mandatory)
attribute Attribute	Attribute or column found in the server's data repository for this layer	Attribute data type	Zero or more (optional)

Table 28 — Parts of Attribute data structure

Names	Definition	Data type and values	Multiplicity and use
name Name	Name of this attribute	Character String type, not empty	One (mandatory)
type Type	Data type of this attribute	Character String type, not empty	One (mandatory)
uniqueCount UniqueCount	Number of unique values for this attribute	Long type	Zero or one (optional)
values Values	List of unique values stored in the server's data repository for the selected layer and attribute/column	Values data type	Zero or one (optional)

Table 29 — Parts of Values data structure

Names	Definition	Data type and values	Multiplicity and use
value Value	Unique attribute value	Character String type, may be empty	Zero or more (optional)

NOTE The UML class diagram contained in Subclause C.8 provides a graphical view of the contents of the GetLayerInfo operation response listed in Table 25 - Table 29.

10.3.2 Normal response XML encoding

The following schema fragment specifies the contents and structure of a GetLayerInfo operation response, always encoded in XML:

```
    <!-- ========================================================== -->
    <element name="LayerInfo" type="w3ds:LayerInfoType"/>
    <!-- ========================================================== -->
    <complexType name="LayerInfoType">
        <sequence>
            <element name="LI_Layer" type="w3ds:LI_Layer" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
    </complexType>
    <!-- ========================================================== -->
    <complexType name="LI_Layer">
        <sequence>
            <element ref="ows:Identifier" />
            <element name="Attribute" type="w3ds:Attribute" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
    </complexType>
    <!-- ========================================================== -->
    <complexType name="Attribute">
        <sequence>
            <element name="Name" type="ows:CodeType"/>
```



```
            <element name="Type" type="string"/>
            <element name="Values" type="w3ds:Values" minOccurs="0" maxOccurs="1"/>
            <element name="UniqueCount" type="long" minOccurs="0" maxOccurs="1"/>
        </sequence>
    </complexType>
    <!-- ========================================================== -->
    <complexType name="Values">
        <sequence>
            <element name="Value" type="string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
```

10.3.3 GetLayerInfo response example

A GetLayerInfo operation response for W3DS can look like this encoded in XML:

EXAMPLE An example response to a GetLayerInfo request, including a Layer parameter only, XML encoded for HTTP POST is:

```
<?xml version="1.0" encoding="UTF-8"?>
<w3ds:LayerInfo>
    <w3ds:LI_Layer>
        <ows:Identifier>dem</ows:Identifier>
        <w3ds:Attribute>
            <w3ds:Name>id</w3ds:Name>
            <w3ds:Type>Long</w3ds:Type>
            <w3ds:UniqueCount>129764</w3ds:UniqueCount>
        </w3ds:Attribute>
        <w3ds:Attribute>
            <w3ds:Name>landuse</w3ds:Name>
            <w3ds:Type>String</w3ds:Type>
            <w3ds:UniqueCount>5</w3ds:UniqueCount>
        </w3ds:Attribute>
    </w3ds:LI_Layer>
</w3ds:LayerInfo>
```

EXAMPLE An example response to a GetLayerInfo request, including a Layer and a ColumnNames parameter, XML encoded for HTTP POST is:

```
<?xml version="1.0" encoding="UTF-8"?>
<w3ds:LayerInfo>
    <w3ds:LI_Layer>
        <ows:Identifier>dem</ows:Identifier>
        <w3ds:Attribute>
            <w3ds:Name>landuse</w3ds:Name>
            <w3ds:Type>String</w3ds:Type>
            <w3ds:Values>
                <w3ds:Value>train</w3ds:Value>
                <w3ds:Value>block</w3ds:Value>
                <w3ds:Value>green</w3ds:Value>
                <w3ds:Value>street</w3ds:Value>
                <w3ds:Value>forest</w3ds:Value>
            </w3ds:Values>
            <w3ds:UniqueCount>129764</w3ds:UniqueCount>
        </w3ds:Attribute>
    </w3ds:LI_Layer>
</w3ds:LayerInfo>
```


10.3.4 GetLayerInfo exceptions

When a W3DS server encounters an error while performing a GetLayerInfo operation, it shall return an exception report message as specified in Subclause 7.4 of [OGC 06-121r3]. The allowed standard exception codes shall include those listed in Table 30. For each listed exceptionCode, the contents of the "locator" parameter value shall be as specified in the right column of Table 30.

NOTE To reduce the need for readers to refer to other documents, the first four values listed below are copied from Table 20 in Subclause 8.3 of [OGC 06-121r3].

Table 30 — Exception codes for GetLayerInfo operation

exceptionCode value	Meaning of code	"locator" value
OperationNotSupported	Request is for an operation that is not supported by this server	Name of operation not supported
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value
OptionNotSupported	Request is for an option that is not supported by this server	Identifier of option not supported
UnknownLayer	Operation request contains an identifier in the Layers parameter which is unknown to the server	Identifier of invalid layer
ColumnNameNotDefined	Operation request contains a column name which is not defined for the selected layer.	Name of invalid column
FormatNotSupported	Operation request contains a MIME type in the Format parameter which is not supported by the server	Name of unsupported format
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter

11 GetTile operation (optional)

11.1 Introduction

Data in layers as described in the server's meta data may be spatially partitioned into rectangular tiles. The GetTile operation is an alternative entry point for accessing tiled layers using tile level, row, and column indices. The information on how to compute these indices is provided by a TileSet definition in the layer element.

The GetTile operation is only supported for those Layers for which the attribute tiled=true has been defined or inherited. A client shall not issue a GetTile request for

other layers. A W3DS shall respond with a properly formatted service exception response (code = OperationNotSupported) if it receives a GetTile request but does not support it.

11.2 GetTile operation request

11.2.1 GetTile request parameters

A request to perform the GetTile operation shall include the data structure specified in Figure 10 and Table 31. This table also specifies the UML model data type, source of values, and multiplicity of each listed parameter, plus the meaning to servers when each optional parameter is not included in the operation request. Although some values listed in the "Names" column appear to contain spaces, they shall not contain spaces.

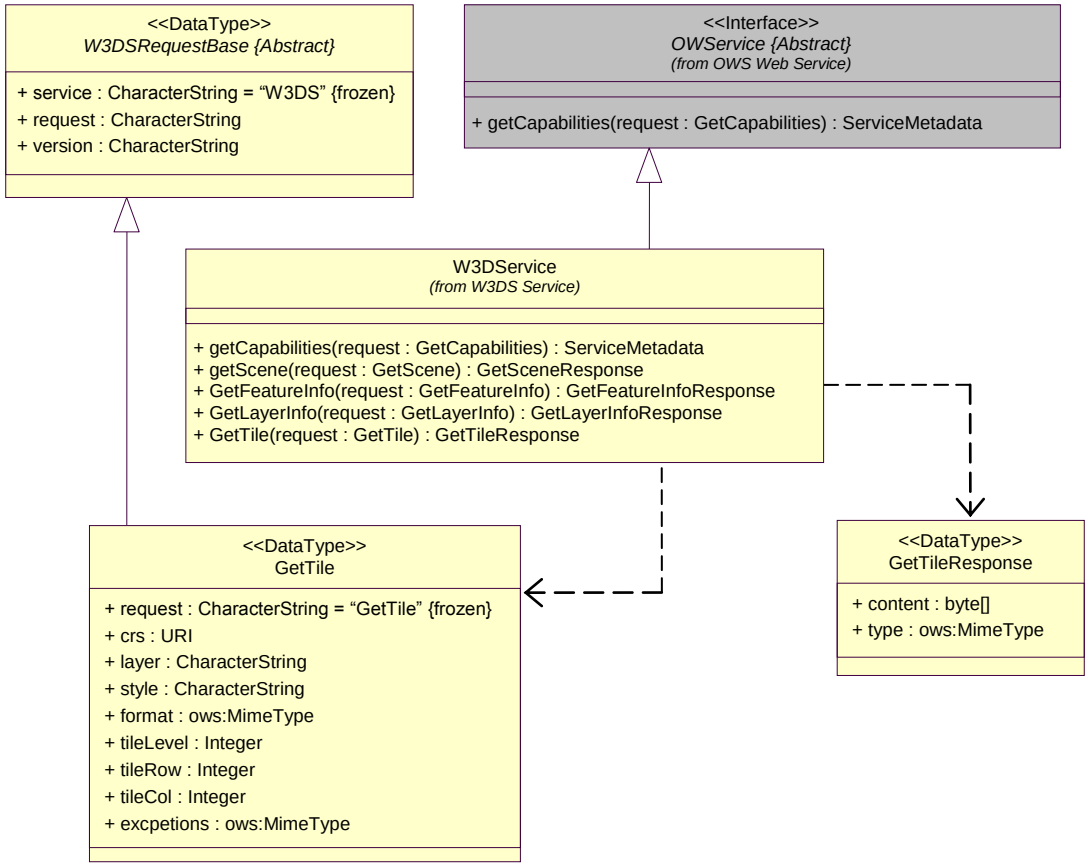

Figure 10 — GetTile operation request UML class diagram

NOTE 1 To reduce the need for readers to refer to other documents, the first three parameters listed below are largely copied from Table 21 in Subclause 9.2.1 of [OGC 06-121r3].

Table 31 — Parameters in GetTile operation request

Names [a]	Definition	Data type and values	Multiplicity and use
service service	Service type identifier	Character String type, not empty Value shall be "W3DS"	One (mandatory)
request request	Operation name	Character String type, not empty Value shall be "GetTile"	One (mandatory)
version version	Standard version for operation	Character String type, not empty Value is specified by each Implementation Standard and Schemas version	One (mandatory)
crs CRS	Coordinate Reference System of the returned tile	URI [b]	One (mandatory)
layer Layer	Identifier of layer to access data from	Character String type	One (mandatory)
style Style	Identifier of server style to be applied to the layer	Character String type	Zero or one (optional)
format Format	Format encoding of the tile	ows:MimeType, see [OGC 06-121r3] clause 10.5	One (mandatory)
tileLevel TileLevel	Level of requested tile	Non negative Integer type	One (mandatory)
tileRow TileRow	Row index of requested tile	Non negative Integer type	One (mandatory)
tileCol TileCol	Column index of requested tile	Non negative Integer type	One (mandatory)
exceptions Exceptions	Format of exceptions	ows:MimeType, see [OGC 06-121r3] clause 10.5	Zero or one (optional)

a Although some values listed in the "Name" column appear to contain spaces, they shall not contain spaces.

b The CRS shall be specified using either the European Petroleum Survey Group form "EPSG:<POSC Code>" or the URL format defined in OWS Common. Examples: "EPSG:31467", "urn:ogc:def:crs:EPSG:6.3:31467 ", see [OGC 06-121r3] clause 10.3 and [OGC 04-046r3].

NOTE 2 The data type of many parameters is specified as "Character String type, not empty". In the XML Schema Documents specified herein, these parameters are encoded with the xsd:string type, which does NOT require that these strings not be empty.

NOTE 3 The UML class diagram contained in Subclause C.9 provides a useful graphical view of the contents of the GetTile operation request listed in Tables Table 31 - Table 32.

The "Multiplicity and use" columns in Table 20 and Table 21 specify the optionality of each listed parameter and data structure in the GetTile operation request. All the "mandatory" parameters and data structures shall be implemented by all W3DS clients, using a specified value(s). Similarly, all the "mandatory" parameters and data structures shall be implemented by all W3DS servers, checking that each request parameter or data structure is received with any specified value(s).

All the "optional" parameters and data structures, in the GetTile operation request, should be implemented by all W3DS clients using specified values, for each implemented W3DS to which that parameter or data structure applies. Similarly, all the "optional" parameters and data structures shall be implemented by all W3DS servers, for each implemented W3DS to which that parameter or data structure applies.

11.2.1.1 CRS

The parameter value for the coordinate reference system (CRS) is defined in [OGC 06-121r3] clause 10.3 and [OGC 04-046r3]. When using a 2D CRS, then the height is measured in meters above sea level.

11.2.1.2 Layer

Identifier of the layer from which to access data. The layer must contain the attribute tiled=true in its meta data, otherwise, a OperationNotSupported exception is thrown.

11.2.1.3 Style

The Styles parameter specifies which server style shall be applied by the server when generating the display elements. The parameter value shall refer to a style identifier as described in the server's meta data. If this parameter is missing, then the default style is applied.

11.2.1.4 Format

The mandatory Format parameter specifies the target encoding of the returned tile provided as ows:MimeType, see [OGC 06-121r3] clause 10.5. Available formats are described in the server's meta data.

11.2.1.5 Exceptions

The optional Exceptions parameter specifies the behavior of the server upon detecting an error, e.g. invalid request or internal server error. The default value is "text/xml". The value shall be one of the MIME types offered in the server's meta data parameter value, see 7.3.2.2.

11.2.2 GetTile request KVP encoding (optional)

Servers may implement HTTP GET transfer of the GetTile operation request, using KVP encoding. The KVP encoding of the GetTile operation request shall use the parameters specified in Table 32. The parameters listed in Table 32 shall be as specified in Table 31 above.

Table 32 — GetTile operation request URL parameters

Name and example [a]	Optionality and use	Definition and format
service=W3DS	Mandatory	Service type identifier
request= GetTile	Mandatory	Operation name

version=0.4.0	Mandatory	Standard and schema version for this operation
crs=EPSG:26916	Mandatory	Coordinate Reference System of the returned tile
layer=dem	Mandatory	Identifier of layer to access data from
style=default	Optional	Identifier of server style to be applied to the layer
format= model/x3d	Mandatory	Format encoding of the result
tilelevel=4	Mandatory	Level of requested tile
tilerow=45	Mandatory	Row index of requested tile
tilecol=123	Mandatory	Column index of requested tile
exceptions=text/xml	Optional	Format of exceptions

a All parameter names are here listed using mostly lower case letters. However, any parameter name capitalization shall be allowed in KVP encoding, see Subclause 11.5.2 of [OGC 06-121r3].

EXAMPLE An example GetTile operation request KVP encoded for HTTP GET is:

```
http://hostname:port/path?SERVICE=W3DS&REQUEST=GetTile&VERSION=0.4.0&CRS=EPSG:26916&FORMAT=model/x3d&LAYER=dem&TILELEVEL=4&TILEROW=45&TILECOL=123&EXCEPTIONS=text/xml
```

11.2.3 GetTile request XML encoding (mandatory)

All W3DS servers shall implement HTTP POST transfer of the GetTile operation request, using XML encoding only. The following schema fragment specifies the contents and structure of a GetTile operation request encoded in XML:

```
<!-- ========================================================== -->
<element name="GetTile" type="w3ds:GetTileType"/>
<!-- ========================================================== -->
<complexType name="GetTileType">
    <annotation>
        <documentation>XML encoded W3DS GetTile operation request.
</documentation>
    </annotation>
    <complexContent>
        <extension base="w3ds:W3DSRequestBaseType">
            <sequence>
                <element name="CRS" type="anyURI"/>
                <element name="Layer" type="ows:CodeType"/>
                <element name="Style" type="ows:CodeType" minOccurs="0"/>
                <element name="Format" type="ows:MimeType"/>
                <element name="TileLevel" type="integer"/>
                <element name="TileRow" type="integer"/>
                <element name="TileCol" type="integer"/>
                <element name="Exceptions" type="ows:MimeType" minOccurs="0"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```


EXAMPLE An example GetTile operation request XML encoded for HTTP POST is:

```
<?xml version="1.0" encoding="UTF-8"?>
<w3ds:GetTile service="W3DS" request="GetTile" version="0.4.0">
    <w3ds:CRS>EPSG:26916</w3ds:CRS>
    <w3ds:Layer>dem</w3ds:Layer>
    <w3ds:Style>default</w3ds:Style>
    <w3ds:Format>model/x3d</w3ds:Format>
    <w3ds:TileLevel>4</w3ds:TileLevel>
    <w3ds:TileRow>45</w3ds:TileRow>
    <w3ds:TileCol>123</w3ds:TileCol>
    <w3ds:Exceptions>text/xml</w3ds:Exceptions>
</w3ds:GetTile>
```

11.3 GetTile operation response

The response to a valid GetTile request is a document of the MIME type as specified in the Format parameter. The document contains 3D display elements of the selected layer within the spatial extent of the requested tile, specified by the TileLevel, TileRow and TileCol values. The result is converted into the specified CRS. The coordinate axes will be usually shifted to computer graphics coordinate axes as shown in Figure 2.

If the GetTile request contains a server style identifier, then the display elements of the layer are portrayed according to the respective style.

If the resulting document contains references to additional files which must be loaded by the client, for instance URLs to textures, then those files must be accessible to the client, either stored on the same server as the W3DS or on another server accessible over the Internet. Relative paths in the URL without protocol and server domain may be used for referencing to files in the directory of the W3DS or in subdirectories. Contained URLs may also point to other service end points producing the required MIME type, for instance a WCS may be used as source for terrain textures.

If the GetTile request contains a TileLevel, TileRow or TileCol index which is not within the range of the layer as described in the server's meta data, then the W3DS server shall not respond with an exception report message. The response to a GetTile request shall be always a document of the MIME type as specified in the Format parameter. If any of the indices is out of range, then this document shall contain no content, but all headers that are necessary in order to parse the document correctly.

In case of an incorrect request or an occurring error while accessing the tile, the response must be supplied in the requested format for exceptional cases (parameters EXCEPTION) by the server. With a request via HTTP the MIME type of the returned document must be set by the server according to contents of the resulting file.

11.3.1 GetTile exceptions

When a W3DS server encounters an error while performing a GetTile operation, it shall return an exception report message as specified in Subclause 7.4 of [OGC 06-121r3]. The allowed standard exception codes shall include those listed in Table 33. For each listed

exceptionCode, the contents of the "locator" parameter value shall be as specified in the right column of Table 33.

NOTE To reduce the need for readers to refer to other documents, the first four values listed below are copied from Table 20 in Subclause 8.3 of [OGC 06-121r3].

Table 33 — Exception codes for GetTile operation

exceptionCode value	Meaning of code	"locator" value
OperationNotSupported	Request is for an operation that is not supported by this server	Name of operation not supported
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value
OptionNotSupported	Request is for an option that is not supported by this server	Identifier of option not supported
LayerNotTiled	No tiled data is available for the requested layer. The attribute Tiled in the server's meta data is set to false	Identifier of layer
TileSetNotDefined	No TileSet has been defined for the requested layer in the server's meta data. This indicates a false configuration of the server	Identifier of layer
CRSNotSupported	Operation request contains a value in the CRS parameter which is not supported by the server	Name of unsupported CRS
UnknownLayer	Operation request contains an identifier in the Layer parameter which is unknown to the server	Identifier of invalid layer
FormatNotSupported	Operation request contains a MIME type in the Format parameter which is not supported by the server	Name of unsupported format
ExceptionNotSupported	Operation request contains a value in the Exception parameter which is not supported by the server	Name of unsupported exception format
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter

Annex A
(normative)

Abstract test suite

In each Implementation Standard document, Annex A shall specify the Abstract Test Suite, as specified in Clause 9 and Annex A of ISO 19105. That Clause and Annex specify the ISO/TC 211 requirements for Abstract Test Suites. Examples of Abstract Test Suites are available in an annex of most ISO 191XX documents, one of the more useful is in ISO 191TBD. Note that this guidance may be more abstract than needed in an OpenGIS® Implementation Standard.

Inclusion of the Abstract Test Suite is expected in version 1.0.0 of each OGC Implementation Standard. In earlier versions, the following paragraph can be used:

An abstract test suite is not provided in this version of this Implementation Standard, but will be provided in version 1.0.0.

A.1 General

A paragraph.

Annex B
(normative)

XML Schema Documents

In addition to this document, this standard includes several normative XML Schema Documents. These XML Schema Documents are bundled in a zip file with the present document. After OGC acceptance of a Version 1.0.0 of this standard, these XML Schema Documents will also be posted online at the URL:

http://schemas.opengis.net/W3DS/1.0.0.

In the event of a discrepancy between the bundled and online versions of the XML Schema Documents, the online files shall be considered authoritative.

The W3DS abilities now specified in this document use 8 specified XML Schema Documents included in the zip file with this document. These XML Schema Documents combine the XML schema fragments listed in various subclauses of this document, eliminating duplications. These XML Schema Documents roughly match the UML packages described in Annex C, and are named:

w3dsCommon.xsd

w3dsGetCapabilities.xsd

w3dsCapabilities.xsd

w3dsGetScene.xsd

w3dsGetFeatureInfo.xsd

w3dsGetLayerInfo.xsd

w3dsLayerInfo.xsd

w3dsGetTile.xsd

These XML Schema Documents use and build on the OWS common XML Schema Documents specified [OGC 06-121r3], named:

ows19115subset.xsd

owsCommon.xsd

owsDataIdentification.xsd

owsExceptionReport.xsd

owsGetCapabilities.xsd

owsOperationsMetadata.xsd

owsServiceIdentification.xsd

owsServiceProvider.xsd

All these XML Schema Documents contain documentation of the meaning of each element and attribute, and this documentation shall be considered normative as specified in Subclause 11.6.3 of [OGC 06-121r3].

Annex C
(informative)

UML model

C.1 Introduction

This annex provides a UML model of the W3DS interface, using the OGC/ISO profile of UML summarized in Subclause 5.3 of [06-121r3].

Figure C.1 is a simple UML diagram summarizing the W3DS interface. This class diagram shows that the W3DService class inherits the getCapabilities operation from the OGCWebService interface class, and adds the GetScene, GetFeatureInfo, GetLayerInfo, and GetTile operations. (The capitalization of names uses the OGC/ISO profile of UML.)

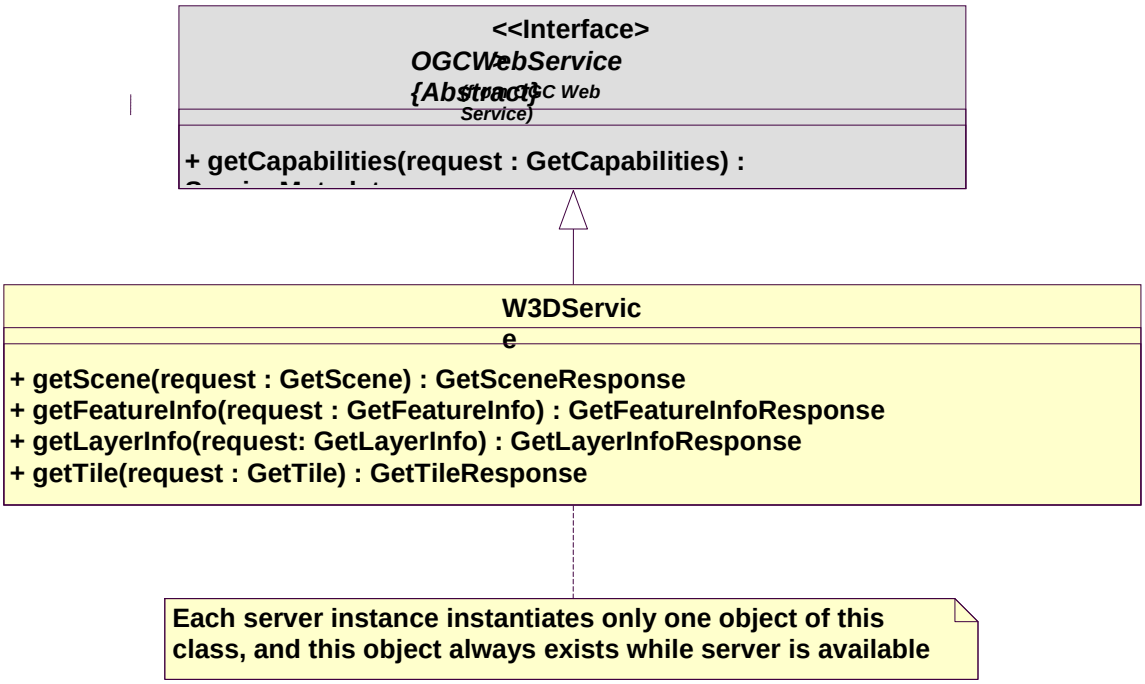

Figure C.1 — W3DS interface UML diagram

Each of the W3DS operations uses a request and a response data type, each of which is defined by one or more additional UML classes. The following subclauses provide a more complete UML model of the W3DS interface, adding UML classes defining the operation request and response data types.

C.2 UML packages

The W3DS interface UML model is organized in packages, as shown in the package diagram in Figure C.2. These W3DS-specific packages make use of five non-W3DS-specific packages, named OWS Web Service, OWS Operations Metadata, OWS Service Identification, OWS Service Provider, and ISO 19115 Subset. This package diagram shows the dependencies among the various packages shown.

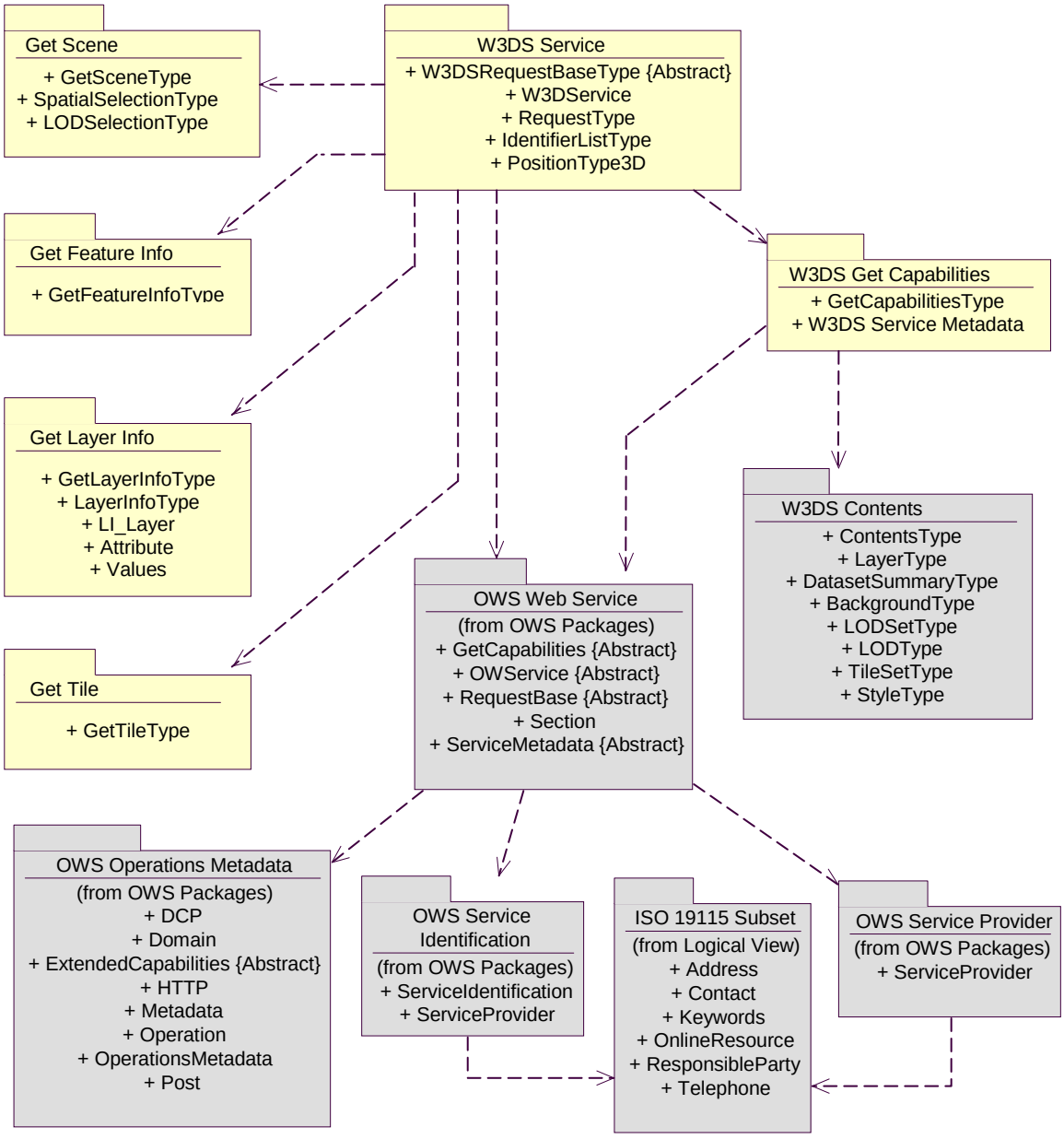

Figure C.2 — W3DS interface package diagram

Each of the six W3DS-specific packages shown in Figure C.2 is described in the following subclauses. The OWS Web Service, OWS Operations Metadata, OWS Service

Identification, OWS Service Provider, and ISO 19115 Subset packages are described in Annex B of [OGC 06-121r3].

C.3 W3DS Service package

The W3DS Service package is shown in the class diagram in Figure C.3. This diagram does not show the classes used by the W3DS operation requests and responses, which are shown (with part of this package) in the GetCapabilites, GetScene, GetFeatureInfo, GetLayerInfo, and GetTile packages. This diagram also shows one used classes from the OWS Web Service package, which is common to all OGC Web Services, plus one used class from the W3DS package.

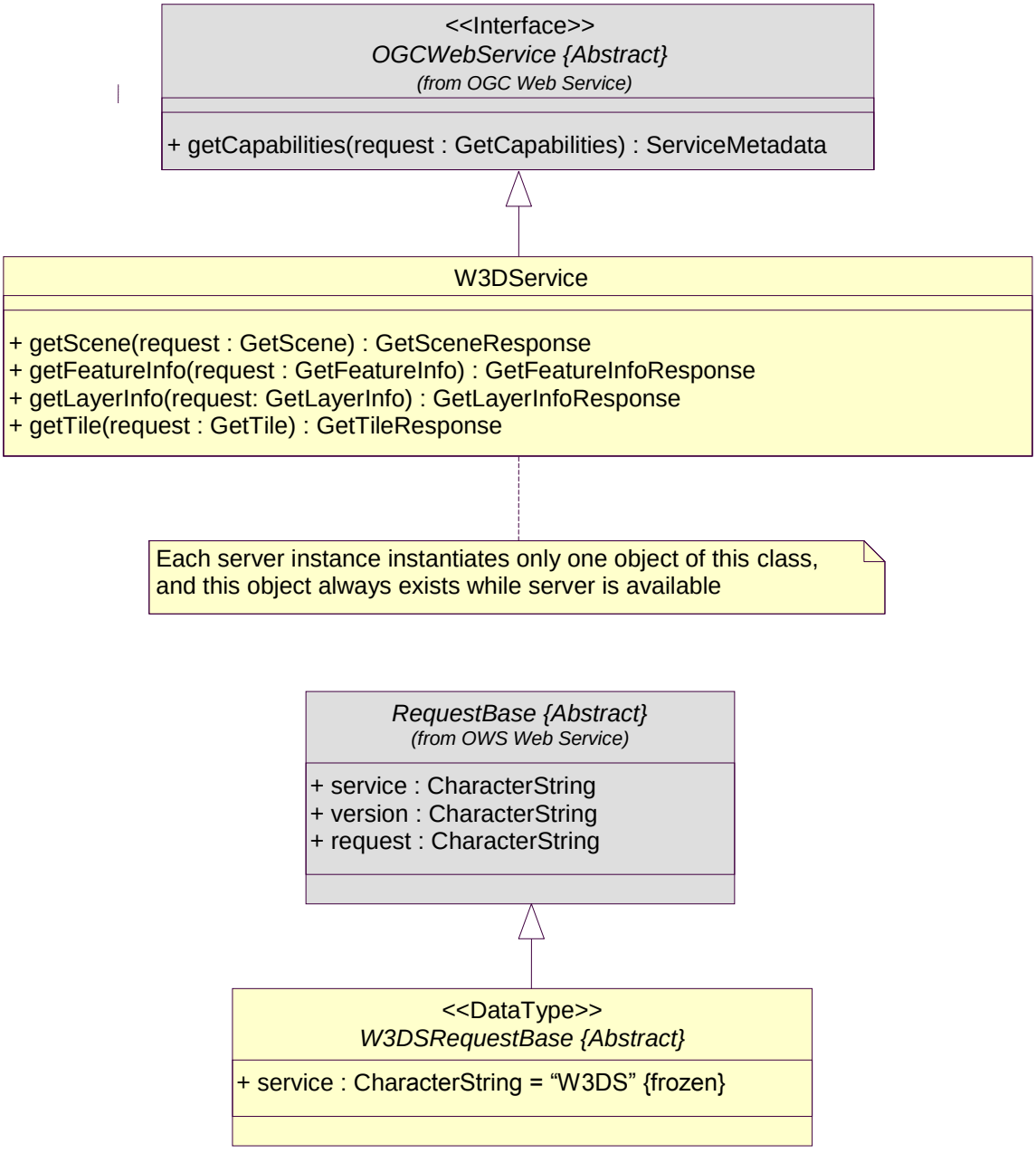

Figure C.3 — W3DS Service package class diagram

C.4 W3DS Get Capabilities package

The W3DS Get Capabilities package is shown in the class diagram in Figure C.4. This diagram also shows several classes from the OWS Get Capabilities package. The W3DSGetCapability class introduced by this package is further defined by Table 3 in this document.

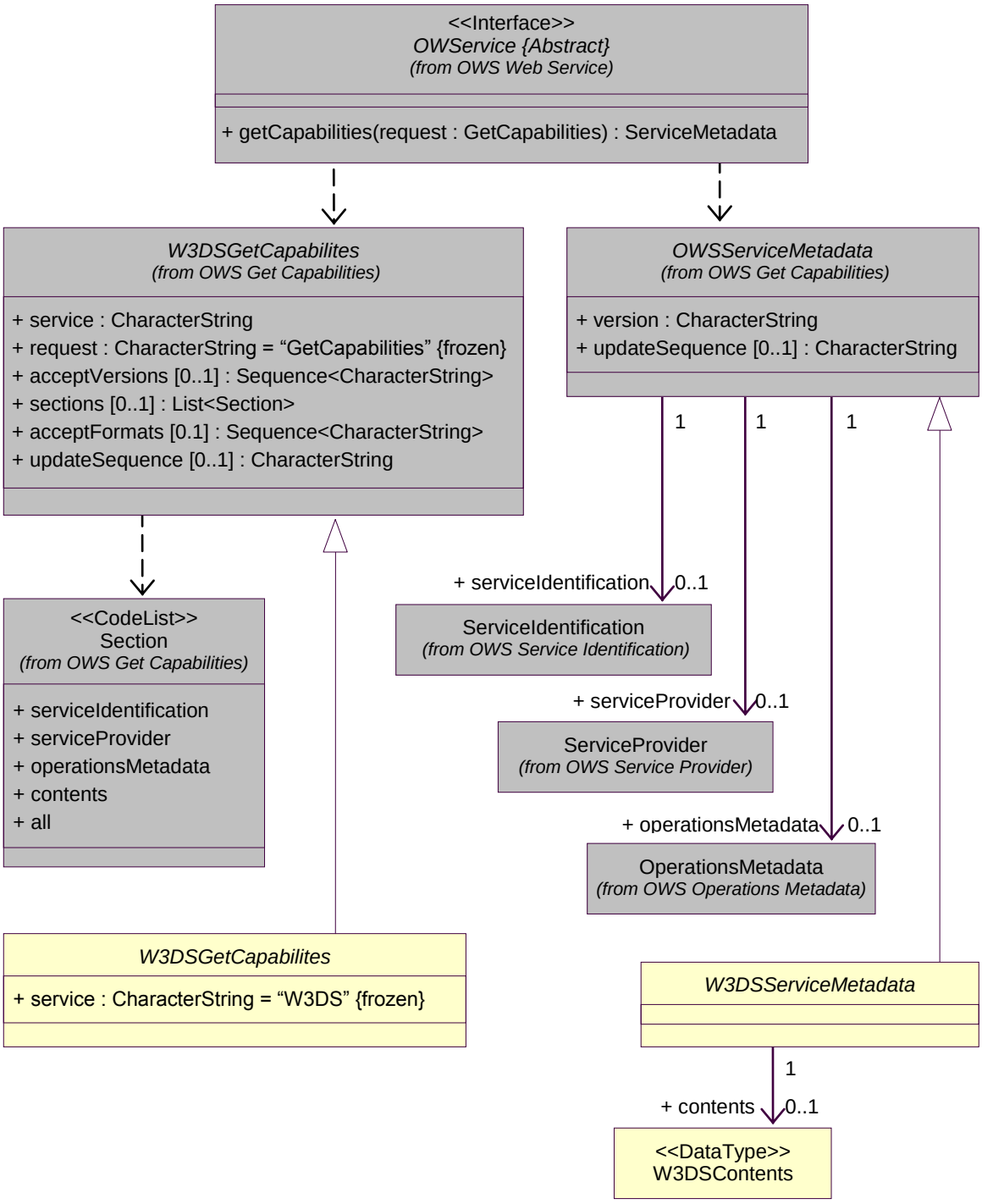

Figure C.4 — W3DS Get Capabilities package class diagram

C.5 W3DS Contents package

The W3DS Contents package is shown in the class diagram in Figure C.5. This diagram also shows classes from the OWS Common, and OWS Data Identification package. The classes introduced by this package is further defined by Table 10 through Table 16 in this document.

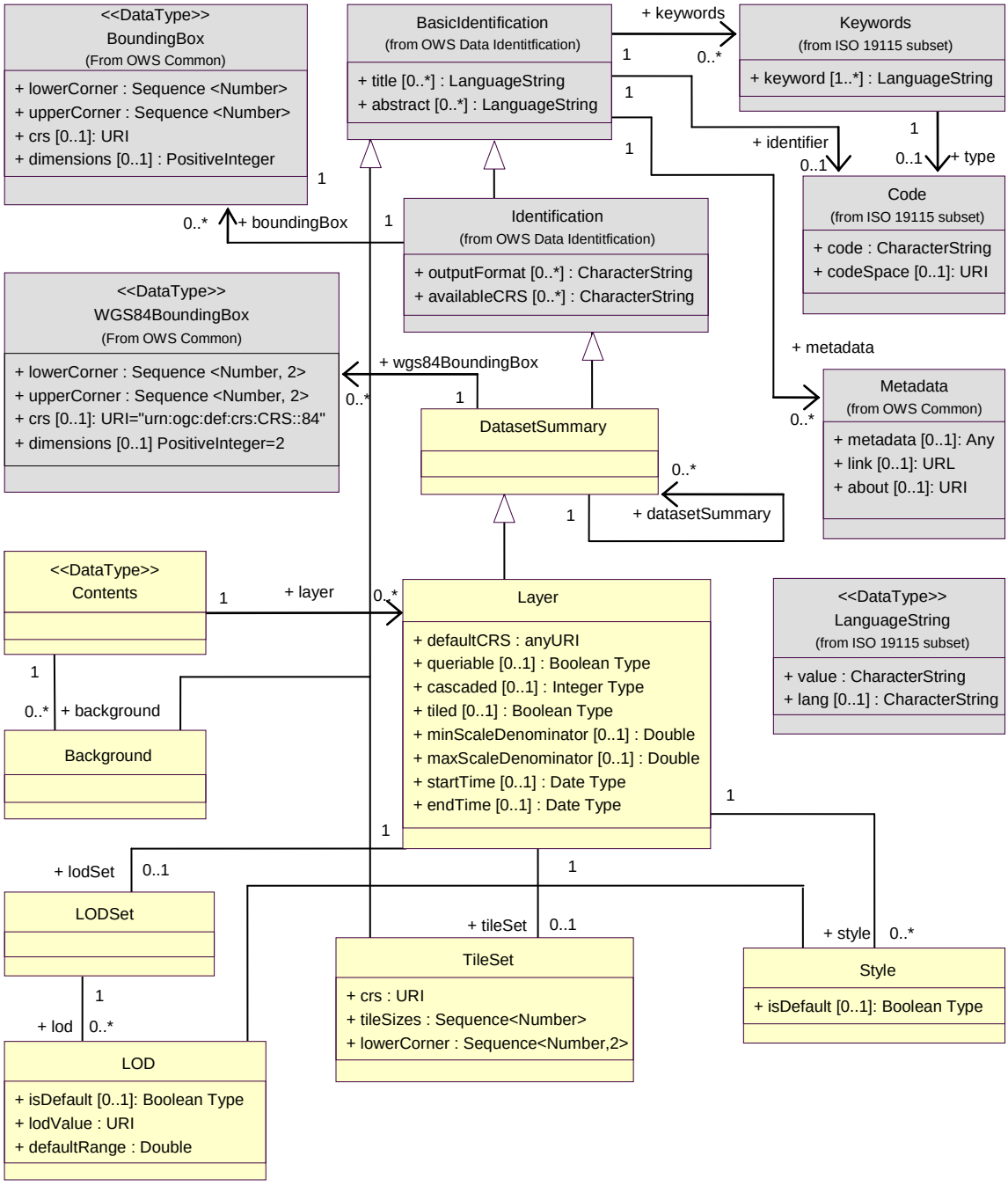

Figure C.5 — W3DS Contents package class diagram

C.6 W3DS Get Scene package

The W3DS Get Scene package is shown in the class diagram in Figure C.6. The GetScene operation is specified in subclause 8. This diagram also shows 2 classes from the OWS Common and OWS Get Capability package. The GetScene class introduced by this package is further defined by Table 17 in this document.

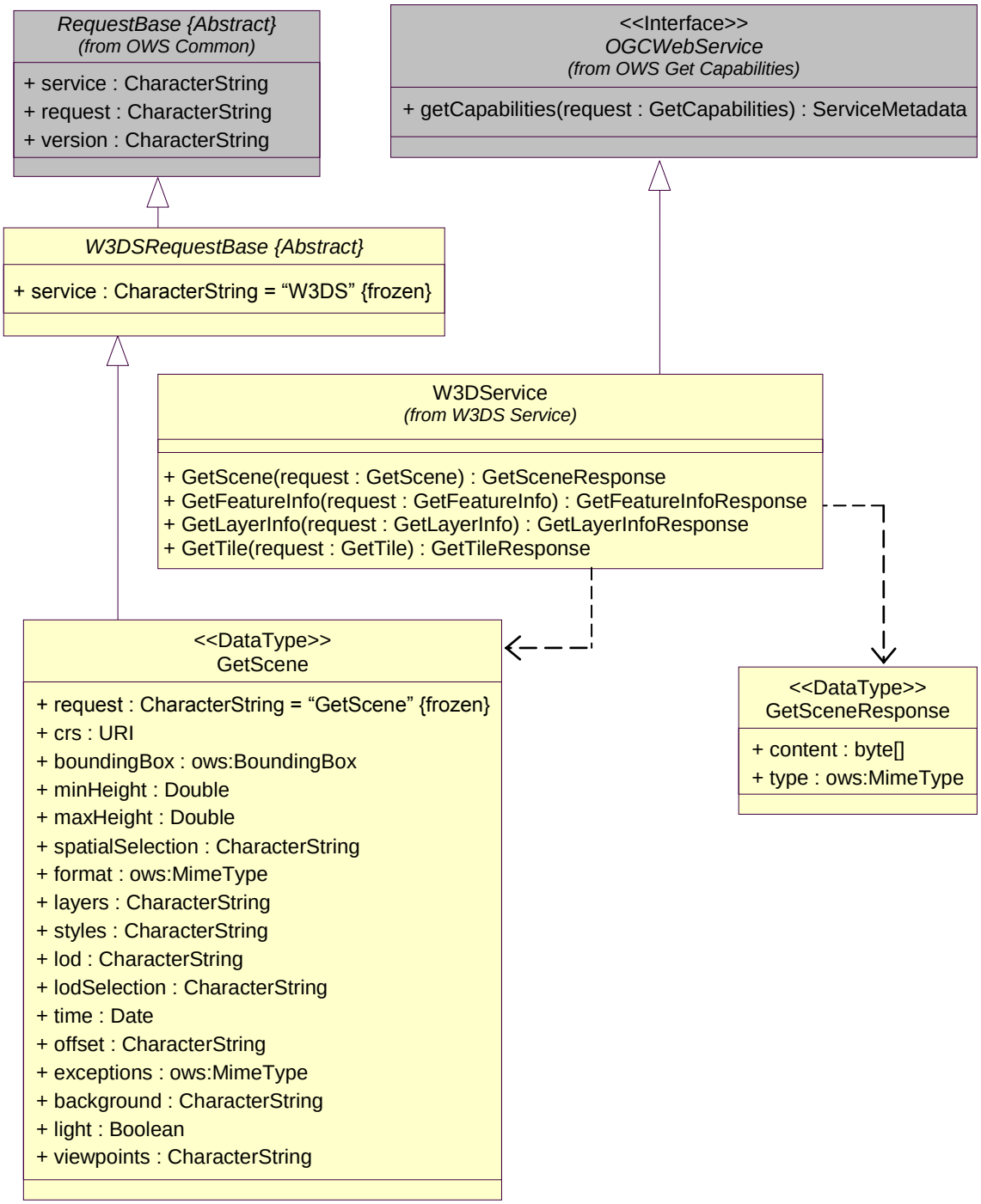

Figure C.6 — W3DS Get Scene package class diagram

C.7 W3DS Get Feature Info package

The W3DS Get Feature Info package is shown in the class diagram in Figure C.7. The GetFeatureInfo operation is specified in subclause 9. This diagram also shows 2 classes from the OWS Common and OWS Get Capability package. The GetFeatureInfo class introduced by this package is further defined by Table 20 in this document.

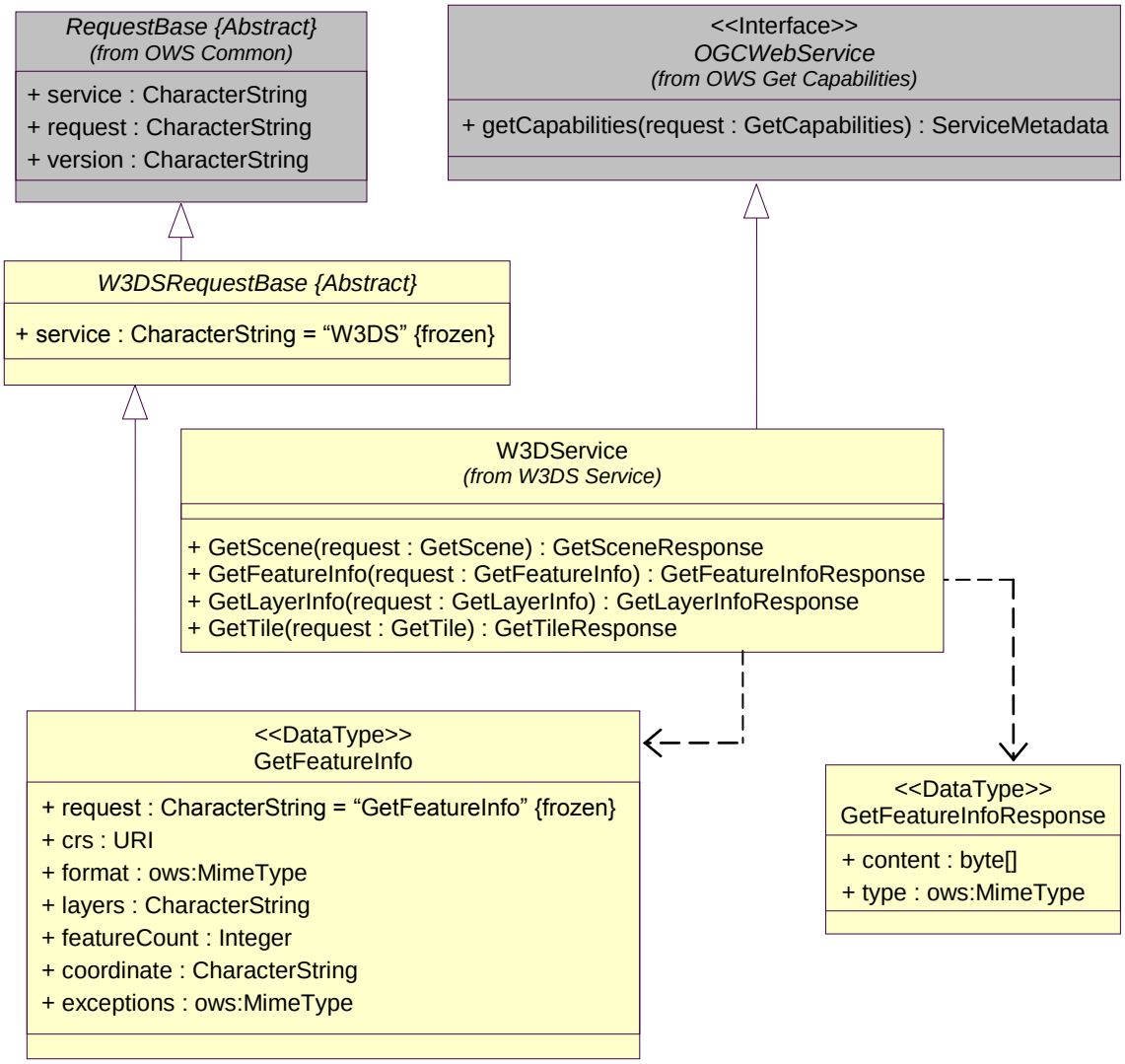

Figure C.7 — W3DS Get FeatureInfo package class diagram

C.8 W3DS Get Layer Info package

The W3DS Get Layer Info package is shown in the class diagram in Figure C.8. The GetLayerInfo operation is specified in subclause 10. This diagram also shows 2 classes from the OWS Common and OWS Get Capability package. The GetLayerInfo class introduced by this package is further defined by Table 23 in this document.

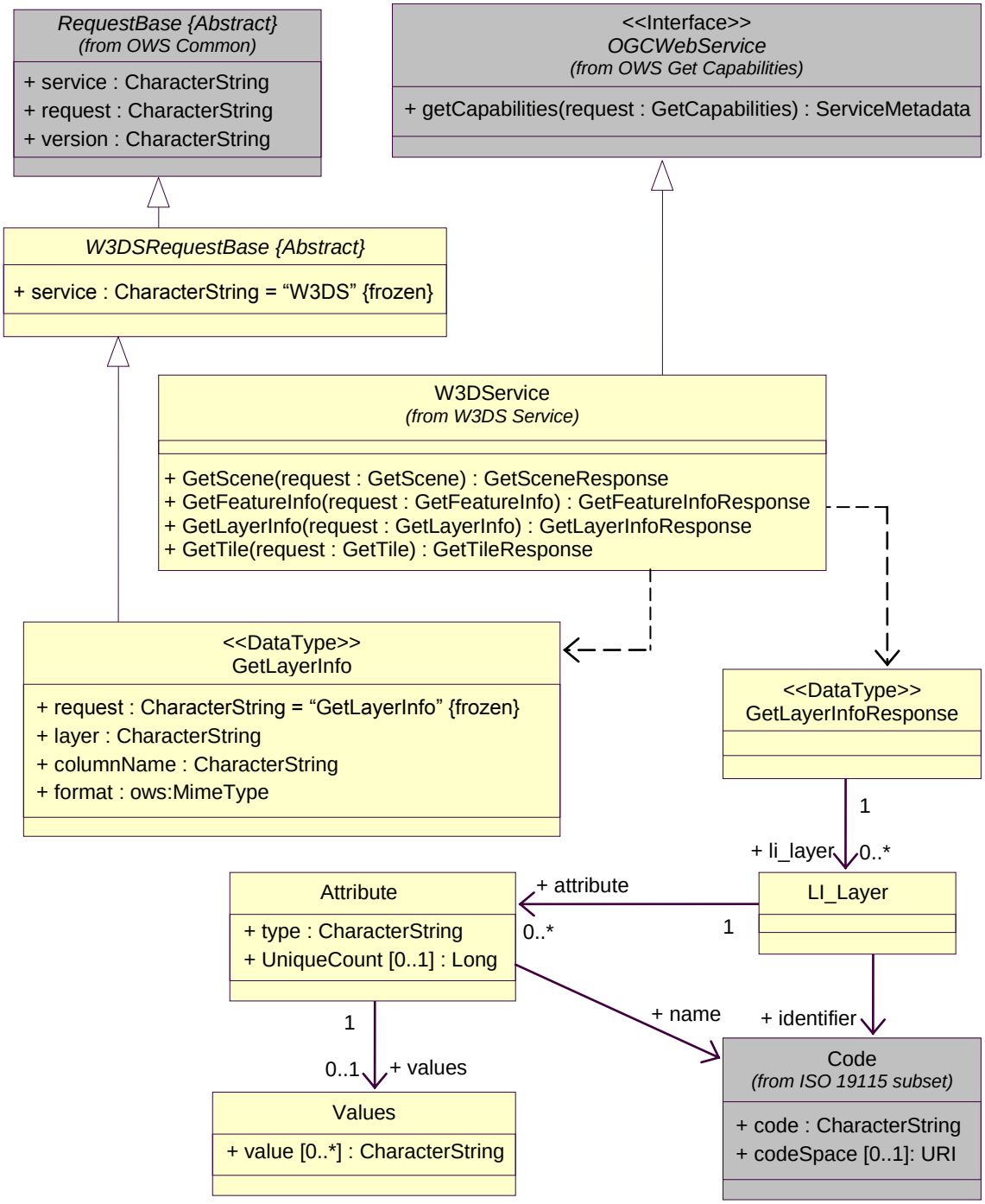

Figure C.8 — W3DS Get LayerInfo package class diagram

C.9 W3DS Get Tile package

The W3DS Get Tile package is shown in the class diagram in Figure C.9. The GetTile operation is specified in subclause 11. This diagram also shows 2 classes from the OWS Common and OWS Get Capability package. The GetLayerInfo class introduced by this package is further defined by Table 31 in this document.

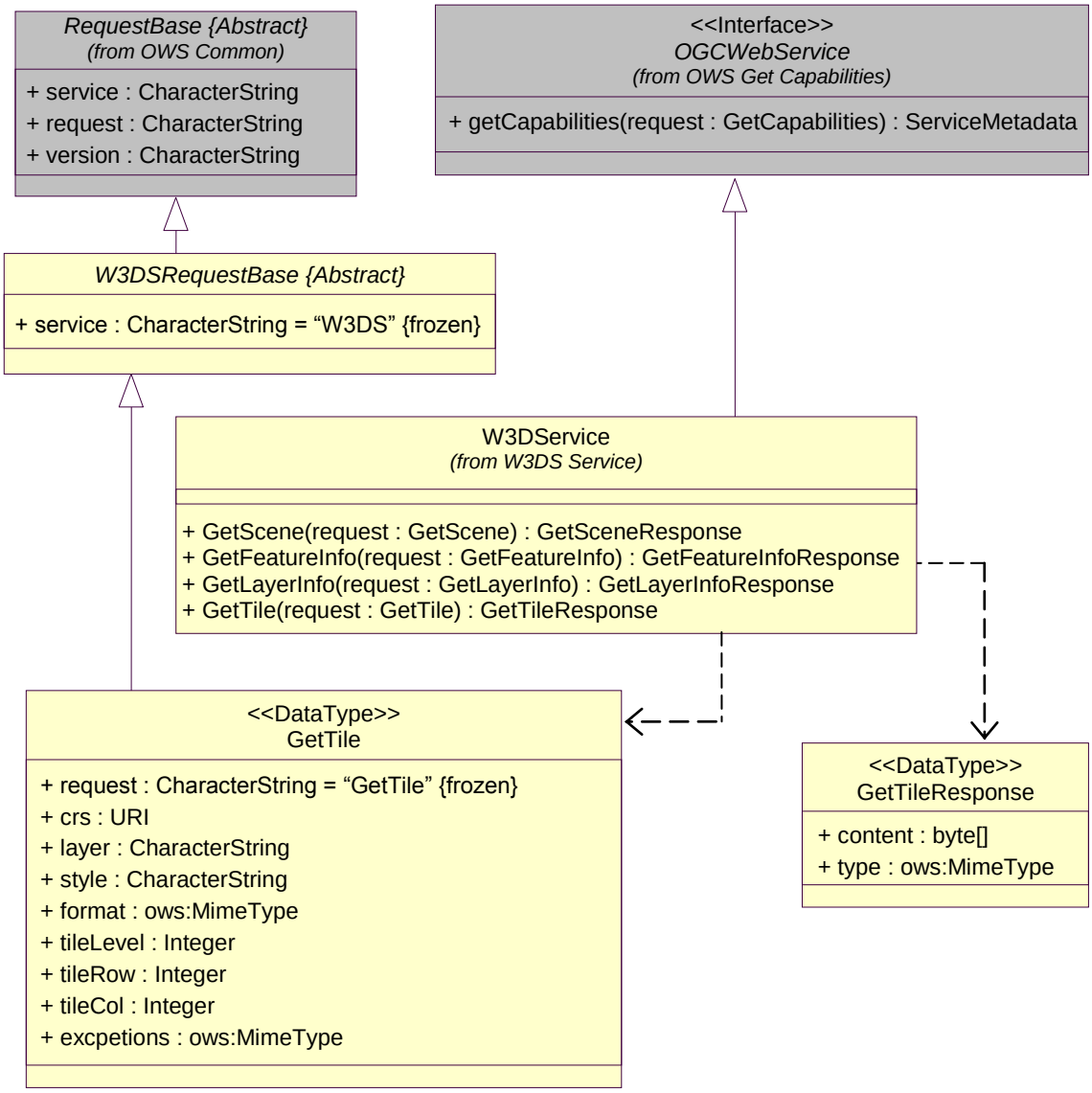

Figure C.9 — W3DS Get Tile package class diagram

Annex D
(informative)

Example XML documents

D.1 Introduction

This annex provides more example XML documents than given in the body of this document.

D.2 Level of Detail Selection modes

The following example shows how the LODSelection parameter in the GetScene request controls how multiple Levels of Detail (LODs) of a data set (if available) are handled by the server. The LODSelection parameter specifies whether the scene shall consist of either a single LOD, or a combination of several LODs which can be merged based on the relations between the features.

```
   <?xml version="1.0" encoding="UTF-8"?>
   <w3ds:GetScene service="W3DS" request="GetScene" version="0.4.0">
       <w3ds:CRS>EPSG:26916</w3ds:CRS>
       <ows:BoundingBox crs="EPSG:26916">
           <ows:LowerCorner>200759.03723318398
3300170.2612183597</ows:LowerCorner>
           <ows:UpperCorner>223200.2944077917
3330896.6813003295</ows:UpperCorner>
       </ows:BoundingBox>
       <w3ds:Format>model/x3d</w3ds:Format>
       <w3ds:Layers>
           <ows:Identifier>dem</ows:Identifier>
           <ows:Identifier>bldgs</ows:Identifier>
       </w3ds:Layers>
       <w3ds:LODs>
           <ows:Identifier/>
           <ows:Identifier>CityGML:2</ows:Identifier>
       </w3ds:LODs>
       <w3ds:LODSelection>equals_or_smaller</w3ds:LODSelection>
       <w3ds:Offset>200000.0 3310000.0 0.0</w3ds:Offset>
       <w3ds:Exceptions>text/xml</w3ds:Exceptions>
       <w3ds:Light>true</w3ds:Light>
       <w3ds:Viewpoints>202000.0 3310000.0 1500.0 202500.0 3305000.0
200.0 0.0 1.0 0.0 60.0</w3ds:Viewpoints>
   </w3ds:GetScene>
```

The LODSelection parameter can have 3 values:

a) equals: exactly the LOD specified in the LODs parameter will be returned
b) equals_or_smaller: A combination of the LOD specified in the LODs parameter and additional lower Levels of Detail will be returned as static scene

c) combined: A combination of the LOD specified in the LODs parameter and additional lower Levels of Detail will be returned as dynamic scene containing special LOD nodes

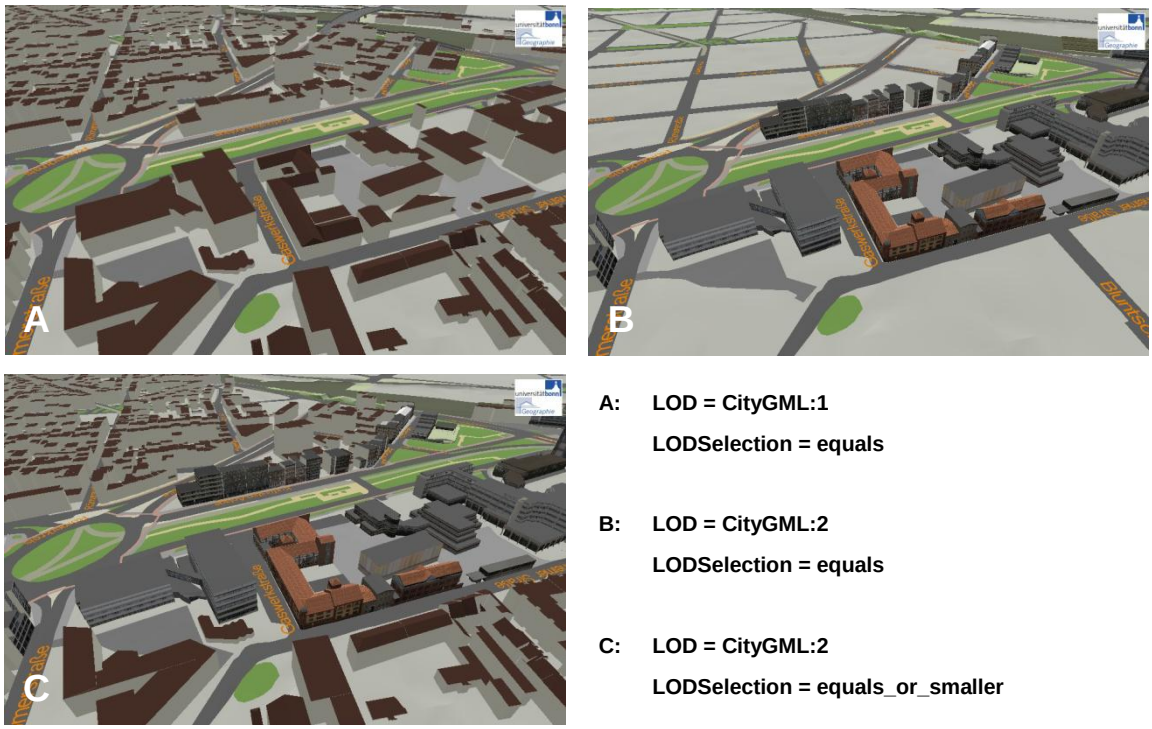

Figure D.1 — W3DS GetScene operation results. Variations of LOD and LODSelection parameters

D.2 TileSet Definition

The following example shows a TileSet definition as described in the server's meta data. It covers the entire earth. The TileSizes are described in decimal degrees, since the specified CRS is WGS84 (EPSG:4326). The Origin is at the south pole, -180 degrees. Thus the earth surface is divided into 2 base tiles, one for the western and one for the eastern hemisphere.

```
<w3ds:TileSet>
    <ows:Identifier>dem_tileset</ows:Identifier>
    <w3ds:CRS>EPSG:4326</w3ds:CRS>
    <w3ds:TileSizes>180 90 45 22.5 11.25 5.625 2.8125 1.40625 0.703125
    0.3515625 0.17578125</w3ds:TileSizes>
    <w3ds:LowerCorner>-180.0 -90.0</w3ds:LowerCorner>
</w3ds:TileSet>
```

Figure D.2 shows a scene which has been composed using multiple W3DS GetTile requests. Each Tile represents a quadratic parcel of the earth surface and is modeled as

Triangular Irregular Network (TIN). The coordinates have been transformed into UTM using the CRS parameter.

Figure D.2 — Scene composed of multiple W3DS GetTile requests, rendered in wireframe mode

Bibliography

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

[2] OGC 09-042, 3D-Symbology Encoding Discussion Draft.

[3] OGC 08-140, 3D Portrayal Services - Use Cases, August 2008.

[4] OGC 09-075r1, OWS-6 3D Flythrough (W3DS) Engineering Report, May 2009.

[5] Open Source Geospatial Foundation (OSGeo): WMS Tiling Client Recommendation, http://wiki.osgeo.org/wiki/WMS_Tiling_Client_Recommendation

[6] D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner: Level of Detail for 3D Graphics, Morgan Kaufmann, 2002, ISBN 1-55860-838-9.

