Volume 11

OGC CDB Core Standard Conceptual Model

Open Geospatial Consortium

Submission Date: 2020-01-21

Approval Date: 2020-08-24

Publication Date: 2020-xx-xx

External identifier of this OGC® document: http://www.opengis.net/doc/IS/CDB-core-model/1.2

Internal reference number of this OGC® document: 16-007r5

Version: 1.2

Category: OGC® Implementation Standard

Editor: Sara Saeedi

Volume 11: OGC CDB Core Standard Conceptual Model

Copyright notice

Copyright © 2020 Open Geospatial Consortium

To obtain additional rights of use, visit http://www.opengeospatial.org/legal/

Warning

This document is an OGC Member approved international standard. This document is available on a royalty free, non-discriminatory basis. Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type:	OGC® Standard
Document subtype:	
Document stage:	Approved
Document language:	English

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications. This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

Table of Contents

1. Overview of the OGC CDB Conceptual Model . . . 9
2. CDB General Data Organization . . . 14
   2.1. LoD and Geocell . . . 17
   2.2. CDB File System . . . 17
   2.3. Model Type . . . 20
      2.3.1. 3D Moving Model . . . 21
   2.4. Vector Data Model . . . 22
3. Metadata and Controlled Vocabulary Schema . . . 24
   3.1. Light Names Hierarchy . . . 24
   3.2. Client Specific Lights Definition . . . 24
   3.3. Model Components Definition . . . 25
   3.4. Base Materials Table . . . 25
   3.5. Composite Material Tables . . . 26
   3.6. Default Values Definition Table . . . 28
   3.7. Version . . . 28
   3.8. Configuration . . . 29
   3.9. CDB Vector Attributes . . . 31
   3.10. 3D Model Metadata . . . 32
4. 3D Model Extensions . . . 37
5. Feature Data Dictionary . . . 38
Annex A: Conformance Class Abstract Test Suite (Normative) . . . 40
   A.1. CDB Overall Conceptual Model . . . 40
Annex B: UML notations . . . 41
   B.1. Class Diagrams Notation . . . 41
Annex C: Revision History . . . 43

i. Abstract

This Open Geospatial Consortium (OGC) standard defines the conceptual model for the OGC CDB Standard. The objective of this document is to provide an core conceptual model for a CDB data store (repository). The model is represented using UML (Unified Modeling Language). The conceptual model is comprised of concepts, schema, classes and categories as well as their relationships, which are used to understand, and/or represent an OGC CDB data store. This enables a comparison and description of the CDB data store structure on a more detailed level. This document was created by reverse-engineering the UML diagrams and documentation from the original CDB submission OGC Common DataBase Volume 1 Best Practice, 2015 as a basis for supporting OGC interoperability. One of the important roles of this conceptual model is to provide a UML model that is consistent with the other OGC standards and to identify functional gaps between the current CDB data store and the OGC standards baseline. This document references sections of Volume 1: OGC CDB Core Standard: Model and Physical Database Structure [OGC 15-113r5].

NOTE	The simulation community uses the term “synthetic environment data” to mean all the digital data stored in some database or structured data store that is required for use by simulation clients. From the geospatial community perspective, these data are essentially the same as GIS data but with, in some cases, special attributes, such as radar reflectivity.

ii. Keywords

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, UML, conceptual model, raster, tiles, vector, CDB, Common Data Base, simulation, visualization, synthetic environment.

iii. Preface

The industry-maintained CDB model and data store structure has been discussed and demonstrated at OGC Technical Committee meetings since September 2013. This document, the first UML conceptual model for OGC CDB standard, is one of the 15 documents that comprise the OGC CDB modular standard. The UML conceptual model establishes a single set of consistent concepts that could be also implemented using other encoding mechanisms.

The CDB standard is originally based on the OGC CDB Best Practice documents, which were submitted to the OGC by CAE Inc. on behalf of the CDB implementation community and user group. CDB is currently widely implemented in the defense and aviation simulation communities. The intent is that this initial OGC version of the CDB standard be backwards compatible with existing implementations but that terminology and concepts be aligned as appropriate with the OGC technical baseline. Future work is planned to align the standard with other OGC standards and to provide Best Practices focused on how to use CDB with the existing OGC standards baseline, such as CityGML, Web Map Service (WMS), Web Feature Service (WFS), and Web Coverage Service (WCS). A GeoPackage capability was defined for version 1.2 of the OGC CDB standard.

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

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

iv. Submitting organizations

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

Organization name(s)

- CAE Inc.
- Carl Reed, OGC Individual Member
- Envitia, Ltd
- Glen Johnson, OGC Individual Member
- KaDSci, LLC
- Open Site Plan
- University of Calgary
- UK Met Office

The OGC CDB standard is based on and derived from an industry developed and maintained specification, which has been approved and published as OGC Best PracticeDocument 15-003: OGC Common DataBase Volume 1 Main Body. An extensive listing of contributors to the legacy industry-led CDB specification is at Chapter 11, pp 475-476 in that OGC Best Practices Document (https://portal.opengeospatial.org/files/?artifact_id=61935).

v. Submitters

All questions regarding this submission should be directed to the editor or the submitters:

Name	Affiliation
Sara Saeedi	University of Calgary
Steve Liang	University of Calgary
Carl Reed	Carl Reed & Associates
David Graham	CAE Inc.

vi. Future Work

The CDB community anticipates that additional standardization will be required to prescribe content appropriate to targeted simulation applications. In its current form, the CDB standard does not mandate synthetic environmental richness, quality and resolution. In Version 1.1, additional informative clauses were incorporated that provide guidance on how to include and encode global (data store wide) and local (data set specific) geospatial metadata.

The OGC CDB Standards Working Group (SWG) members understand there is a requirement for

eventual alignment of the CDB standard with the OGC/ISO standards baseline. In Version 1 of the CDB standard, effort was invested to begin aligning terminology and concepts, specifically in the coordinate reference system discussions and requirements.

The current version of the CDB standard is fully backwards compatible with version 1.0 of the CDB standard as defined and implemented by the current CDB implementer and user community. The requirements for a CDB data store are focused on the ability to store, manage, and access extremely large volumes of geographic content. In this version of the standard, initial harmonization with the OGC and ISO standards baseline has begun. For example, where appropriate, the CDB simulation community terms and definitions have been replaced with OGC/ISO terms and definitions. Further, the standards documents have been reorganized and structured to be consistent with the OGC Modular Specification Policy. However, the CDB SWG and community recognize the need to further harmonize and align this standard with the OGC baseline and other IT best practices. There has already been considerable discussion in this regard.

Based on such discussions and comments received during the public comment period, the following future work tasks are envisioned:

1. Describe explicitly how the CDB model may or may not align with the OGC DGGS standard;
2. Provide best practice details on how to use WMS, WFS, and WCS to access existing CDB data stores. This work may require Interoperability Experiments to better understand the implications of these decisions;
3. Extend the supported encodings and formats for a CDB data store to include the use of the OGC GeoPackage, CityGML, and InDoorGML standards as well as other broadly used community encoding standards, such as GeoTIFF. This work may require performing OGC Interoperability Experiments to better understand the implications of these decisions. Please note that in CDB version 1.2, GeoPackage containers can be used in a CDB data store.
4. Further align CDB terminology to be fully consistent with OGC/ISO terminology.

Making these enhancements will allow the use and implementation of a CDB structured data store for application areas other than aviation simulators.

vii. CDB Document Guide

This document contains a number of annexes related to the OGC CDB Core standard.

For the purposes of being able to cross reference this OGC Best Practice with the previous versions of the CDB standard, the following annex “crosswalk” is provided.

OGC CDB Best Practice and CDB 3.2	**OGC CDB Standard Version 1.0**
Formerly Annex A10 in Volume 2	Annex B Rationale: Sensor Simulation - Achieving Device-Independence
Main Body: Rationale for using JPEG	Annex C Reasons for Using JPEG
Formerly Annex B in Volume 2	Annex D: TIFF Implementation Requirements
Formerly Annex D in Volume 2	Annex E: ShapeFile dBASE III Guidance

Formerly Annex A.11 in Volume 2	Annex F: Annex F Rationale: Partitioning the Earth into Tiles
Formerly Annex A.12	Annex G Rationale: Importance of Level of Detail
Formerly Annex A.17 Volume 2	Annex H: JPEG Informative annex
Formerly Annex U, Volume 2	Annex I ZIP File Informative annex
Formerly Annex E, Volume 2	Annex J: Light Hierarchy
Formerly Annex M, Volume 2	Annex M: CDB Directory Naming and Structure
Formerly Annex O, Volume 2	Annex O: List of Texture Component Selectors
Formerly Annex Q, Volume 2	Annex Q: Table of Dataset Codes
Formerly Annex R, Volume 2	Annex R: Derived Datasets within the CDB
Formerly Annex S, Volume 2	Annex S: Default Read and Write values to be used by Simulator Client-Devices

For ease of editing and review, the standard has been separated into 16 Volumes, one being a schema repository.

- Volume 0: OGC CDB Companion Primer for the CDB standard (Best Practice).
- Volume 1: OGC CDB Core Standard: Model and Physical Data Store Structure. The main body (core) of the CDB standard (Normative).
- Volume 2: OGC CDB Core Model and Physical Structure Annexes (Best Practice).
- Volume 3: OGC CDB Terms and Definitions (Normative).
- Volume 4: OGC CDB Rules for Encoding CDB Vector Data using Shapefiles (Best Practice).
- Volume 5: OGC CDB Radar Cross Section (RCS) Models (Best Practice).
- Volume 6: OGC CDB Rules for Encoding CDB Models using OpenFlight (Best Practice).
- Volume 7: OGC CDB Data Model Guidance (Best Practice).
- Volume 8: OGC CDB Spatial Reference System Guidance (Best Practice).
- Volume 9: OGC CDB Schema Package: http://schemas.opengis.net/cdb/ provides the normative schemas for key features types required in the synthetic modeling environment. Essentially, these schemas are designed to enable semantic interoperability within the simulation context (Normative).
- Volume 10: OGC CDB Implementation Guidance (Best Practice).
- Volume 11: OGC CDB Core Standard Conceptual Model (Normative).
- Volume 12: OGC CDB Navaids Attribution and Navaids Attribution Enumeration Values (Best Practice).
- Volume 13: OGC CDB Rules for Encoding CDB Vector Data using GeoPackage (Normative, Optional Extension).
- Volume 14: OGC CDB Guidance on Conversion of CDB Shapefiles into CDB GeoPackages (Best Practice).
- Volume 15: OGC CDB Optional Multi-Spectral Imagery Extension (Normative).

viii. Terms, Definitions, and Abbreviations

Please refer to Volume 3: Terms and Definitions for terms used in this document (http://www.opengeospatial.org/standards/cdb). Abbreviations used in this CDB Volume are:

BMT Base Material Table

CMT Composite Material Table

DEM Digital Elevation Model

DIGEST Digital Geographic Exchange Standard

DGIWG Defence Geospatial Information Working Group

FDD Feature Data Dictionary

LOD Level of Detail

SEDRIS Synthetic *Environment* Data Representation and Interchange Specification

UHRB Ultra-High Resolution Building (data)

Chapter 1. Overview of the OGC CDB Conceptual Model

A conceptual model is a representation of a system, made of the composition of concepts which are used to help people know, understand, or simulate a subject the model represents. A documented conceptual model represents 'concepts' (entities), the relationships between them, and a vocabulary. This document details the conceptual model for a CDB data store.

In this document, conceptual modeling is used as a structural methodology for describing how the components of a CDB data store may be implemented based on the requirements and guidance specified in the OGC CDB Core Standard. The following table defines the general CDB data store requirement and an overview of key elements implemented in a compliant CDB data store.

Requirement Class 1	
Req. ID	http://opengis.net/spec/CDB/1.0/core/OveralConceptualModel
Dependency	Requirements as specified in the OGC CDB core document
Req. Text	A minimum compliant CDB shall contain the version metadata. When any dataset is provided in a CDB, that dataset shall comply with the corresponding mandatory requirements of the OGC CDB Core Standard (Volume 1 http://www.opengeospatial.org/standards/cdb).

This section describes the conceptual model for an OGC CDB compliant data store. This model can used as the basis for the CDB standard in other application domains, along with its requirements, extension, file-based structure, data formats, access, and the discovery of services.

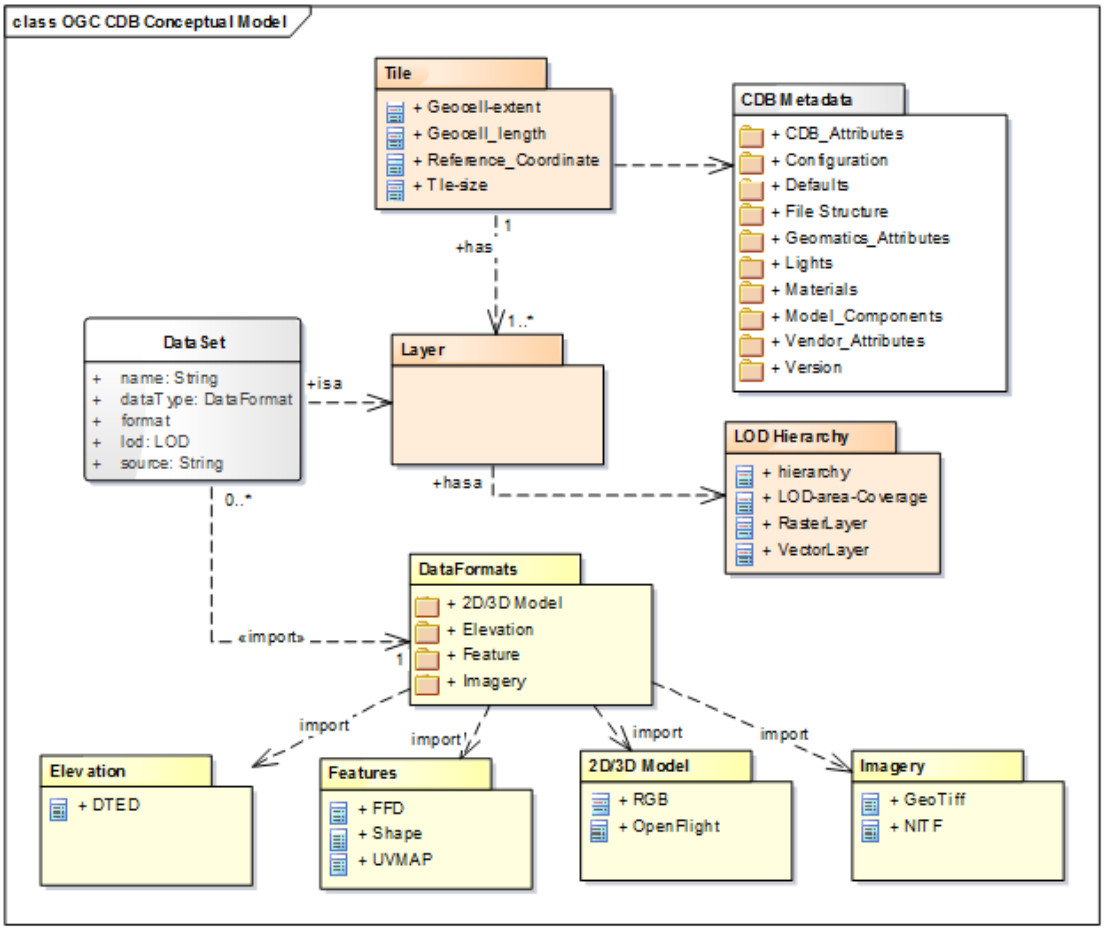


Figure 1. Package diagram of OGC CDB data store conceptual model

The CDB data store structure is designed to provide efficient access to any location enabled content accessible in the data store. The main properties of the CDB data store UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity

Tile	geometric shape with known properties that may or may not be the result of a tiling (tessellation) process. A tile consists of a single connected "piece" (topological disc) without "holes" or "lines". [OGC Tiling Abstract Specification] Note: In CDB, geographically divides the world into geodetic tiles (bounded by latitudes and longitudes), each containing at least a dataset	Dataset type.	One or more (mandatory)
LoD Hierarchy	Each dataset layer has a LoD hierarchy.	Hierarchy of raster, vector and models.	One (mandatory)
Dataset	Defines the basic storage unit used in a CDB data store.	Layers of data	One (mandatory)
Models	Includes 3D representations of cultural features and moving models such as buildings, pylons and posts, aircraft and other moving platforms. 3D models have various model components	Model data formats supported in CDB standard.	Zero or more (optional)
Imagery	There are various imagery types in a CDB data store such as representation of geo-referenced terrain, elevation, and texture.	Image data formats supported in the CDB standard such as GeoTIFF, JPEG 2000, etc.	Zero or more (optional)

Vector Features	This includes all the vector feature datasets in a CDB which are defined based on the Feature Data Dictionary.	Vectors data formats supported by the CDB such as shapefile and etc.	Zero or more (optional)
Elevation	Depicted by a grid of elevation data elements at regular geographic intervals, which include DEM, MinMaxElevation and MaxCulture [1].	Grid of terrain altimetry data	Zero or more (optional)
Metadata/Controlled Vocabularies	CDB XML files that include the default hierarchies, naming, and values to be used by client devices.	XML association	one or more (mandatory)

As it can be seen in Figure 2, the CDB standard relies on three important concepts to organize geospatial data: Tiles, Layers (or datasets) and Levels of Detail (LoD) which are described here.

- Tiles: Tiles organize the data into zones defined by location with respect to a WGS84 reference system [2]. The CDB storage structure allows efficient searching, retrieval and storage of any information contained within a CDB data store. The storage structure portion of the standard geographically divides the world into geodetic tiles (bound by latitudes and longitudes), each containing a specific set of features (such as terrain altimetry, vectors) and models (such as 3D and Radar Cross Sections models), which are in turn represented by their respective datasets. The datasets define the basic storage unit used in a CDB data store. The geographic granularity is at the tile level while in each tile, the information granularity is at the dataset level defined by layers.
- Layers: Layers organize different types of data in a tile. The CDB standard data store model is also logically organized as distinct layers of information. The layers are independent from each other (i.e., changes in one layer do not impose changes in other layers).
- Levels-of-Detail (LoDs): LoDs organize the data in each layer of each tile by its detail. The availability of LoD representations is critical to real-time performance. Most simulation client-devices can readily take advantage of an LoD structure because, in many cases, less detail/information is necessary at increasing distances from the viewpoint of a simulation rendering. The CDB standard requires that each geographic area be represented in an LoD hierarchy in accordance with the availability of source data.

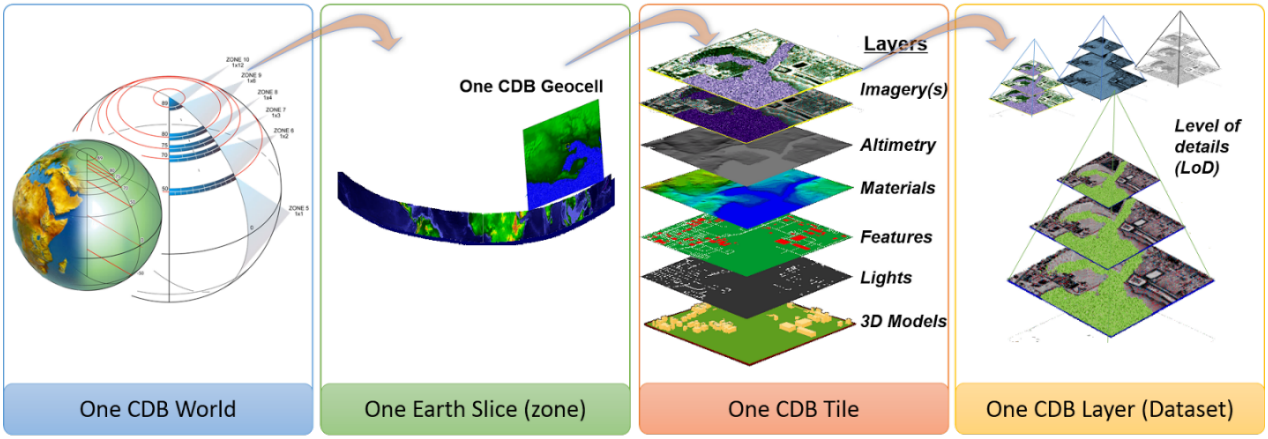


Figure 2. CDB data organization structure

[1] The values of this component are based on the heights of culture features with respect to the corresponding LoD of the culture, be it its bounding sphere, its bounding box or its modeled representation (if supplied).

[2] Please see CDB Spatial Reference System Guidance - Volume 8

Chapter 2. CDB General Data Organization

The CDB is composed of several datasets that share a common structure. The following sections present the general organization and structure of all CDB datasets. The CDB standard does not define or enforce an operating system or file system. Nonetheless, the implementation of a CDB storage sub-system must conform to absolute minimum file system requirements called for by the standard. A CDB data store uses existing common file formats for storing data in various formats such as TIFF/GeoTIFF (raster data), JPEG 2000 (imagery data), OpenFlight (3D models), GeoPackages (vector data and radar cross sections), ShapeFiles (vector data and radar cross sections), SGI image format aka RGB (textures), XML (Metadata) and ZIP (file collection). The current version of CDB uses a consolidation of data dictionaries from DIGEST, DGIWG, SEDRIS and UHRB (See Volume 3: CDB Terms and Definitions). In addition, it is possible to extend the CDB Feature Data Dictionary (FDD) by using the extension capabilities and adding a new FDD XML schema file to access additional feature data codes. The UML diagram in Figure 3 describes how the data is categorized in tiles, layers and LoDs. This is the basis for the CDB geospatial data categorization.

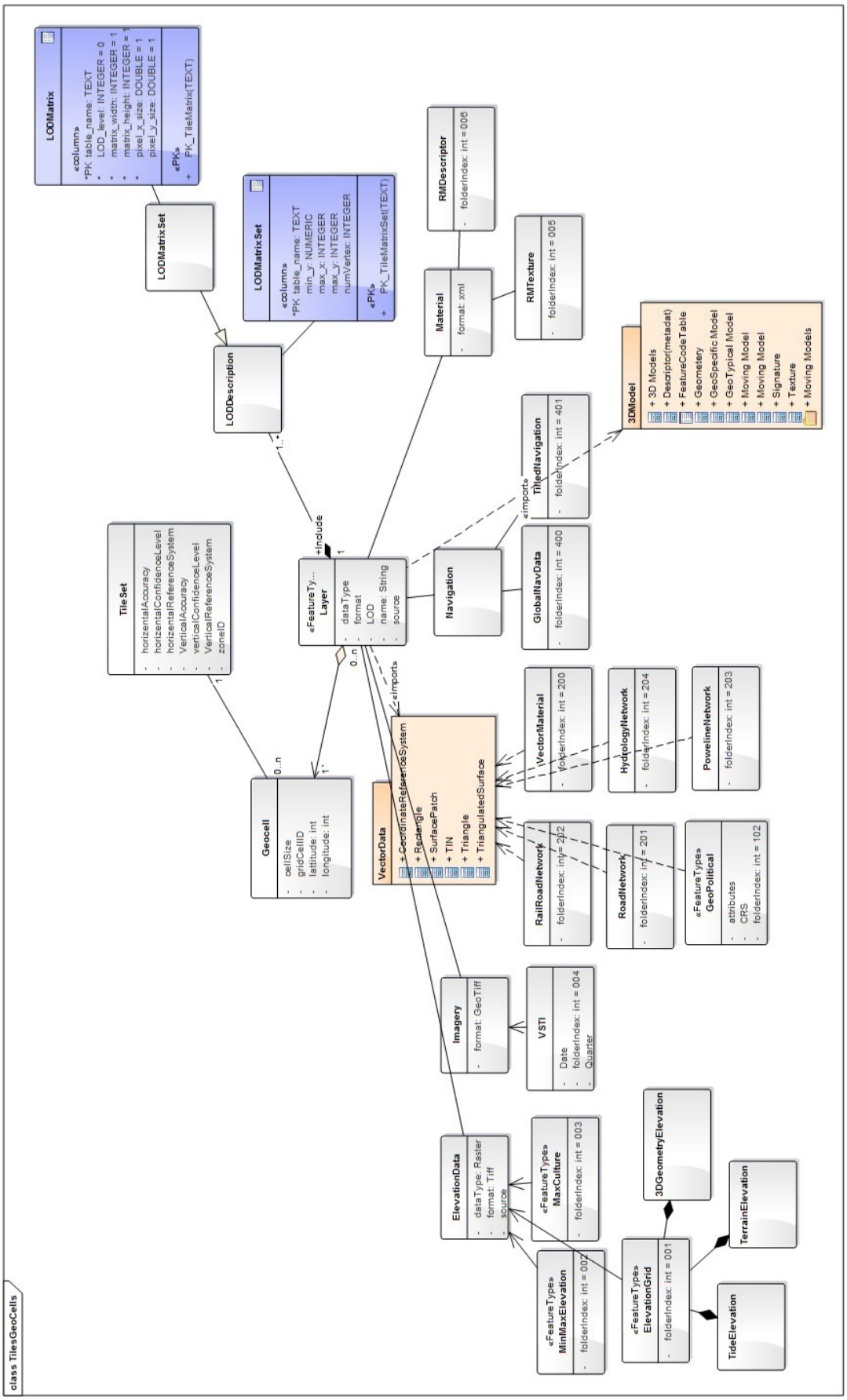


Figure 3. UML diagram of the CDB general data organization

This diagram is the general data organization for the CDB. The main properties of the CDB general data organization UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity
Raster Dataset	Data elements are organized into a regular grid evenly positioned. Raster Datasets always have a fixed number of elements corresponding to their LoD specification.	Raster data formats supported in CDB Core Standard for elevation, imagery, texture and grid data.	Zero or more (optional)
Vector Dataset	The point, the lineal, and the areal (polygon) features of the CDB are organized into several Vector Datasets and into LoDs. For each CDB LoD, the maximum number of points allowed per Tile-LoD and the resulting average Feature Density is defined.	RMDescriptor, GSFeature, GTFeature, GeoPolitical, VectorMateria, RoadNetwork, RailRoadNetwork, PowerLineNetwork, HydrographyNetwork, vectors(Shape)	Zero or more (optional)
Model Dataset	Includes 3D GTModel, GSModel, MModel & 2D Model or cultural feature such as air platforms, buildings and pylons and posts. 3D models have various model components.	GSModelGeometry, GSModelTexture, GSModelSignature, GSModelDescriptor, GSModelMaterial, GSModelInteriorGeometry, GSModelInteriorTexture, GSModelInteriorDescriptor, GSModelInteriorMaterial, GSModelCMT, T2DModelGeometry, OpenFlight models	Zero or more (optional)
Navigation	Navigation library is composed of a single dataset.	NavData	Zero or more (optional)

2.1. LoD and Geocell

This section shows the relationship between the tile structure, layers and LoDs. As can be seen in Figure 4, any TileGeoCell class may have any number of layers and each layer is associated with a LoD matrix set.

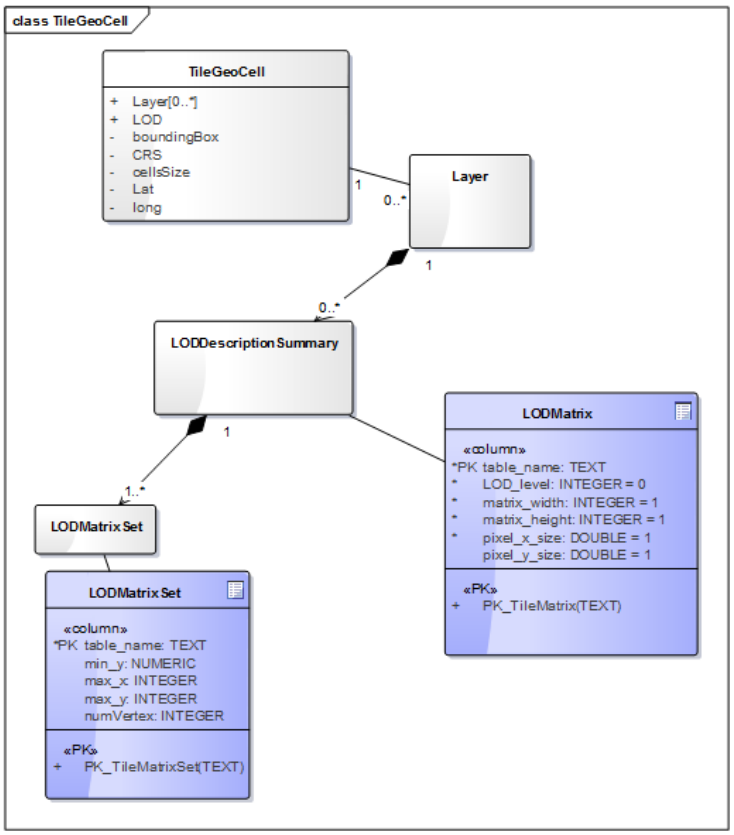


Figure 4. UML diagram of the Geocell, tile and LoD concept

2.2. CDB File System

This section describes how a current version of a CDB conformant data store uses the computer's native file system to store data in files and directories, what the CDB versioning structure is, and how the data is categorized. Further, this section defines the structure of a CDB conformant data store, i.e., the name of all directories forming the CDB hierarchy, as well as the name of all files found in the CDB hierarchy. An important feature of the CDB standard is that all CDB file names are unique and that the filename alone is sufficient to infer the path of the file.

The CDB data store is composed of several datasets that usually reside in their own directory. However, some datasets share a common structure. The top-level directory of the CDB data store follows the following structures.

- \CDB\: This is the root directory and does not need to be "\CDB\" and can be any valid path name on any disk device or volume under the target file system it is stored on.
- \CDB\Metadata\: This directory contains the specific XML metadata files which are global to the CDB.
- \CDB\GTModel\: This is the entry directory that contains the Geotypical [3] Models Datasets.
- \CDB\MModel\: This is the entry directory that contains the Moving Models Datasets.

- \CDB\Tiles\: This is the entry directory that contains all tiles within the CDB instance.
- \CDB\Navigation\: This is the entry directory that contains the global Navigation datasets.

Most of the CDB datasets are organized in a tiled structure and stored under \CDB\Tiles\ directory. The tiled structure facilitates access to the information in real-time by any runtime client-devices. However, for some datasets such as Moving Models or Geotypical Models that require minimal storage, there is no significant advantage to their being added into such a tile structure. Such datasets are referred to as global datasets. They consist of data elements that are global to the earth.

A CDB Version is a collection of CDB and/or user-defined datasets. A CDB Version contains data belonging to a single version of a CDB conformant data store. One CDB Version may refer to another one, which is the basis for the CDB File Replacement Mechanism. The concept of a CDB Version is illustrated using the following UML diagram (Figure 4).

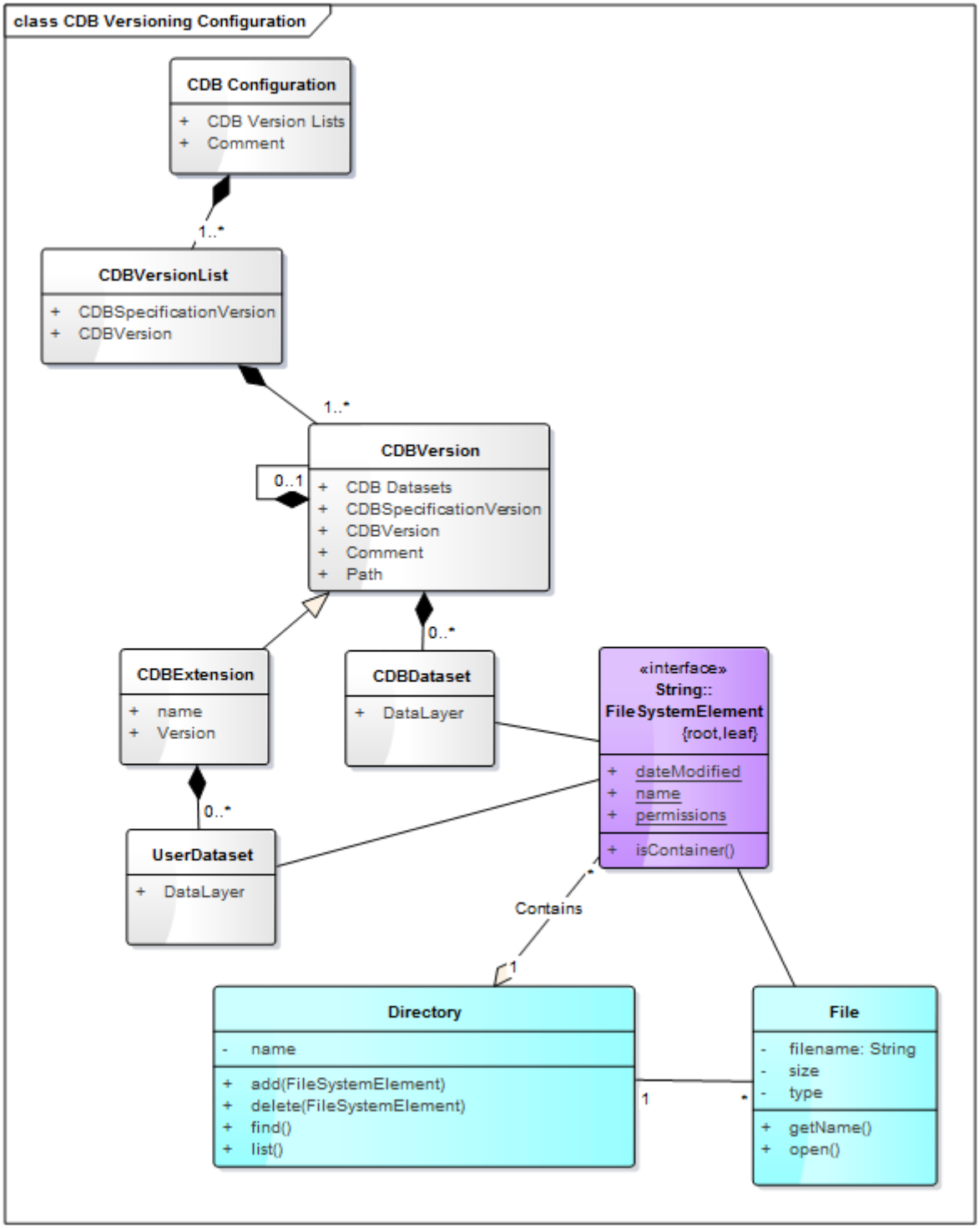


Figure 5. UML diagram of CDB version concept

The diagram shows that a CDB Version contains CDB Datasets. In addition, it states which CDB Version Number has been used to build the CDB content. Finally, the CDB Version has a reference to another CDB Version. This reference allows the creation of a chain of CDB Versions. By chaining two CDB Versions together, the user can replace files in a previous CDB Version with new ones in a newer CDB Version data store. The diagram shows that a CDB Extension inherits all the attributes of a CDB Version and adds its own attributes, a name and a version number (of the extension). The client application checks the name attribute to recognize and process known CDB Extensions and unrecognized CDB Extensions are skipped.

2.3. Model Type

The term Model refers to all of the modeled representations of a cultural feature. The model type features of a CDB can be represented using the following UML diagram. 3DModel, referred to as a GSModel, is unique. In the case where the 3DModel is instanced, it is referred to as a GTModel. A 3DModel that is capable of movement is called a MModel. In the case where a MModel is positioned by the modeler, it is called a statically-positioned MModel.

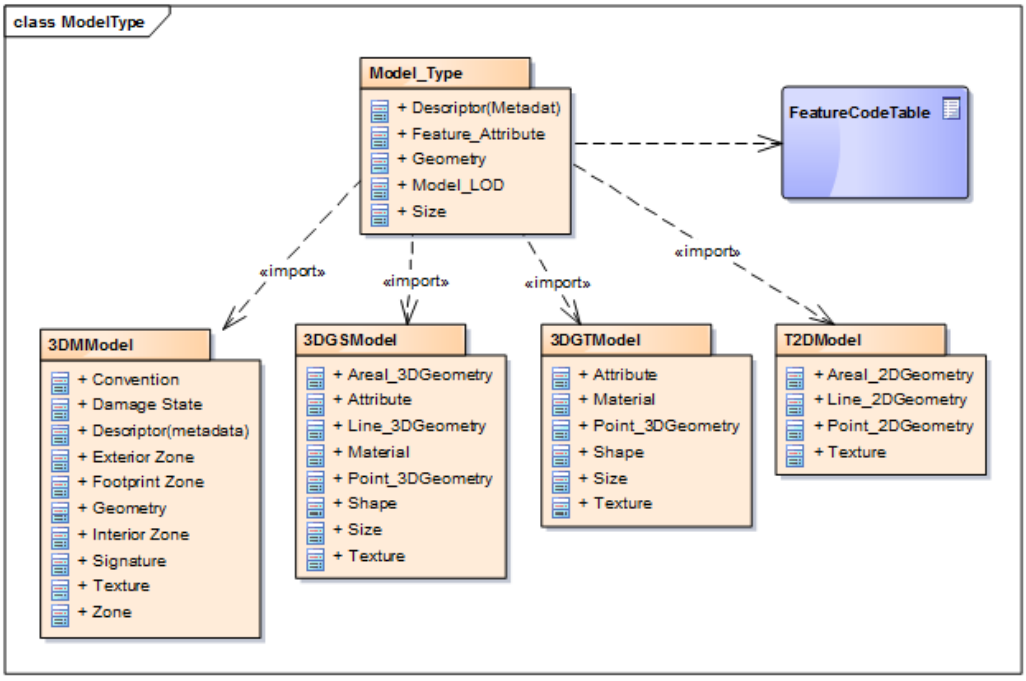


Figure 6. UML package diagram of the model type

The term Model-LoD refers to a specific level of detail of a Model. The main properties of the CDB 2D/3D model type UML diagram are listed below.

Name	Definition	Data type & Value	Multiplicity
Model_Type	The modeled representation of a feature primarily consists of its geometry and textures and encompasses its exterior and interior.	3D model formats supported by the CDB such as OpenFlight	Zero or more (optional)
3DGTModel	Geotypical 3D Model is a geotypical representation of a point-feature that is anchored to the ground.	3D model formats supported by the CDB such as OpenFlight	Zero or more (optional)

3DGSModel	Geospecific 3D Model is geospecific representation of a point, lineal- or areal feature that is anchored to the ground.	3D model formats supported by the CDB such as OpenFlight	Zero or more (optional)
T2DModel	Tiled 2D Model is geospecific or geotypical representations of lineal and areal (polygon) features that are anchored to the ground.	2D model formats supported by the CDB such as shapefiles	Zero or more (optional)
3DMModel	3D modeled representations of point-features that are not anchored to the ground.	3D model formats supported by the CDB such as OpenFlight	Zero or more (optional)

2.3.1. 3D Moving Model

A moving model is typically characterized as if the feature can move (on its own) or be moved. More specifically within the context of this standard, the model is not required to be attached to a cultural point feature (geographic location).

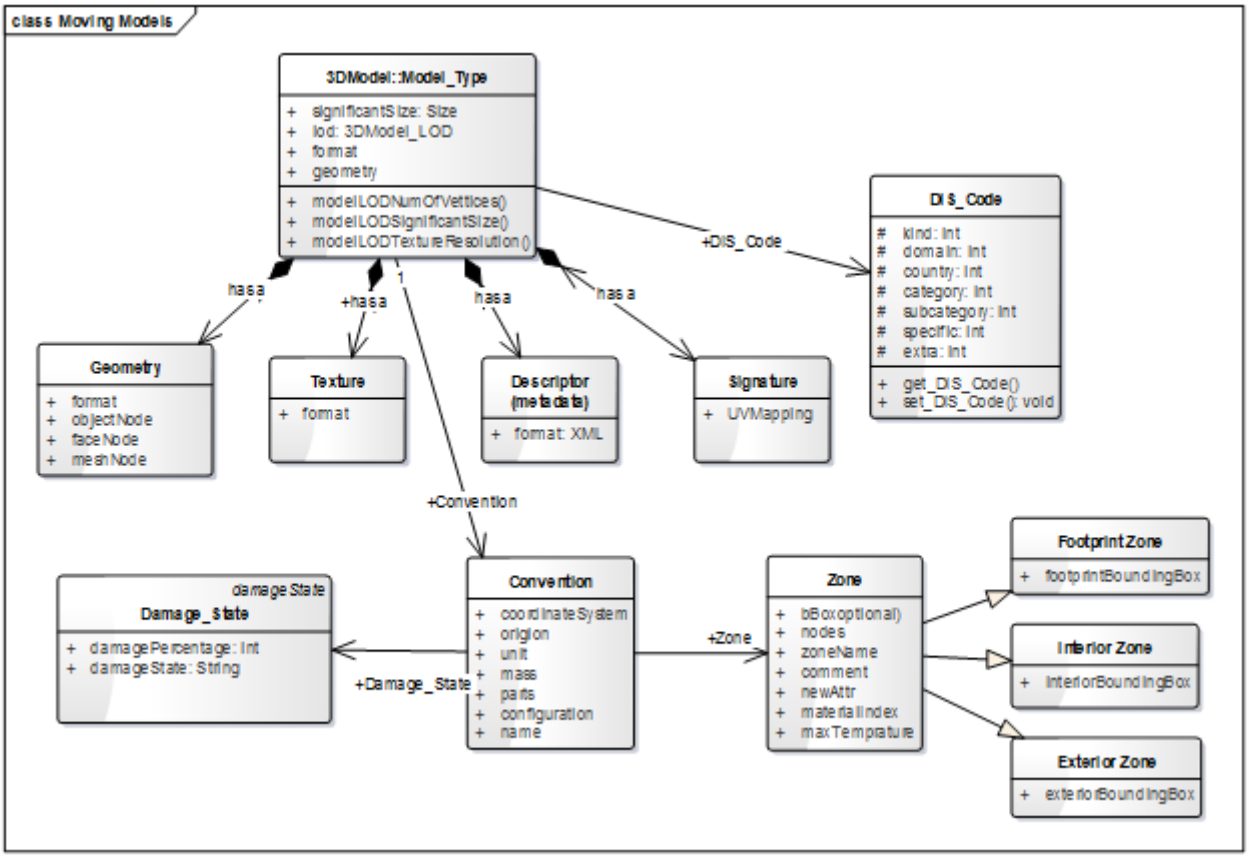


Figure 7. UML diagram of the 3D moving model

During the course of a multi-player simulation, each client-device is typically solicited to provide a modeled representation of each player. The activation of such players requires the client-device to access the appropriate modeled representation of each player. There are a large number of simulations where the player types are characterized by their Distributed Interactive Simulation [4] (DIS) code. To this end, the CDB data store provides a moving model library whose structure provides a convenient categorization of models by their DIS code as shown in the following diagram.

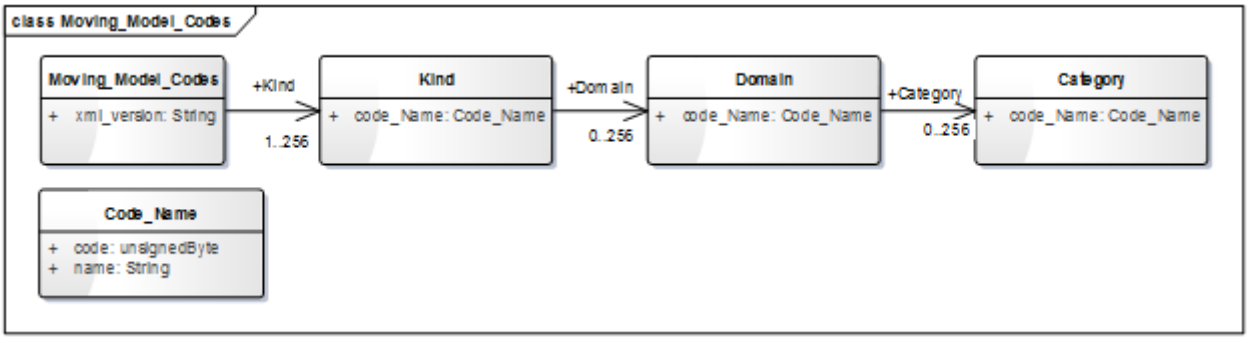


Figure 8. UML diagram of moving model codes

The “xml_version” attribute of a moving model code is used to indicate the version of the XML file containing the list of codes. It is independent from the version of the Standard and also the version of the Schema.

2.4. Vector Data Model

Tiled vector data differs from their raster counterpart in three important ways. First of all, the tiled

vector data internal structure permits a non-uniform distribution of elements within the tile: i.e., the position of each element within the tile is explicit. Secondly, the tiled vector data's internal structure permits a variable number of elements within a tile's boundary. Finally, the distribution of the element types from a single list can be controlled.

Conceptually, the LoD for tiled vector data implicitly provides the average density of elements within the tile. The run-time LoD behavior that controls the rendered number of data elements depends on various parameters and on the off-line filtering process.

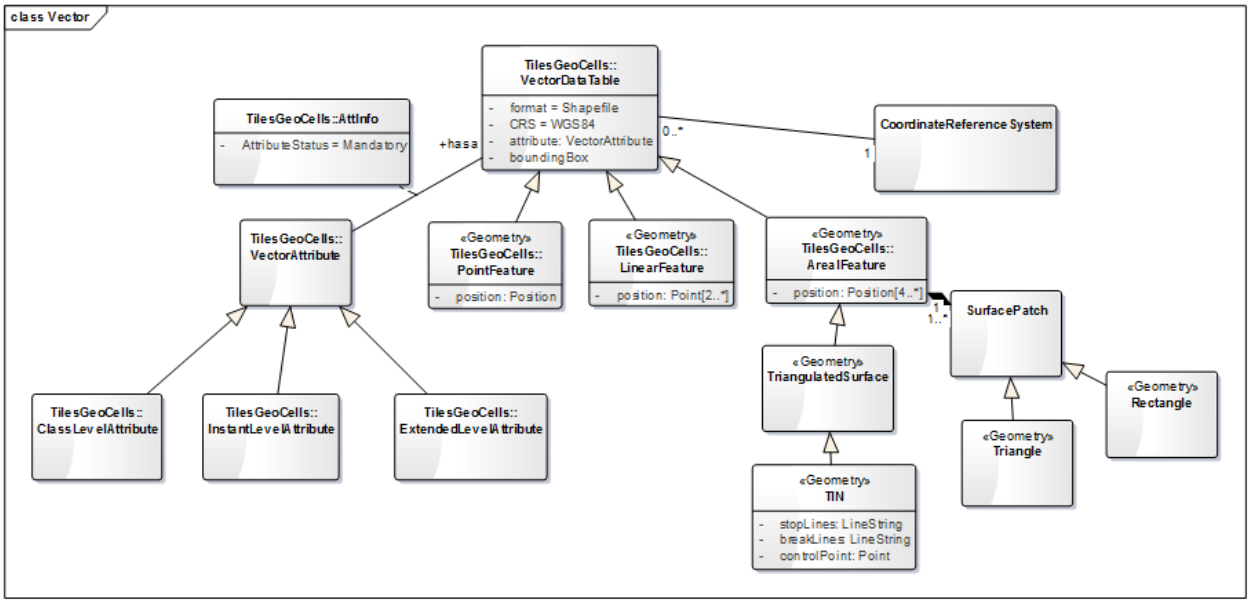


Figure 9. UML diagram of vector data model

[3] A model is said to be geotypical if it instanced multiple times within a CDB data store. Geotypical models correspond to representative (in shape, size, texture, materials and attribution) models of real-world manmade or natural 3D cultural features.

[4] IEEE 1278 series Distributed Interactive Simulation.

Chapter 3. Metadata and Controlled Vocabulary Schema

Metadata and controlled vocabulary datasets contain information, global to a CDB implementation that defines its structure, naming hierarchies, default values, allowable values, and status. All metadata files are formatted using XML files, and their XSD schemas can be found in the \CDB\Metadata\Schema\ folder delivered with the CDB Standard.

3.1. Light Names Hierarchy

The light name hierarchy for a CDB compliant data store is described in detail within the table found in Volume 2 Annex J of the OGC CDB Core Standard [5]. This Annex provides a description of the entire naming hierarchy, including the hierarchical relationship of the levels with respect to each other and the position of each light type within this hierarchy. To this end, the lights hierarchy definition controlled vocabulary is stored in an XML file in the metadata CDB directory. “Lights.xml” file contains the name of each light type and a unique code with each light type. Light codes have a one-to-one association with light types; consequently, the light codes are unique among all light types. For run-time access of this data, clients must be able to retrieve such information. The below diagram (figure 9) shows the UML diagram of light data hierarchy to define and validate the content of the CDB light names hierarchy found in /CDB/Metadata/Lights.xml.

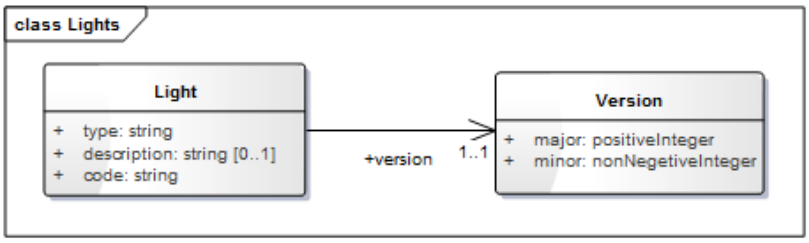


Figure 10. UML diagram of light names hierarchy

This attribute “Version” represents the version number of the Light.xml file.

3.2. Client Specific Lights Definition

Client-devices use the light type code as an index to lookup the client-specific properties and characteristics of each light type using a Lights_xxx.xml. The CDB standard offers a complementary approach to modifying the appearance of lights. This approach provides basic control over light intensity, color, lobe width and aspect, frequency and duty cycle for potential use by simulation implementations. This approach also permits a modeller to add new light types to the CDB light hierarchy. The below UML diagram (figure) presents the schema for the fields of the Lights_xxx.xml which is generated from the schema file located at \CDB\Schema\Lights_Tuning.xsd.

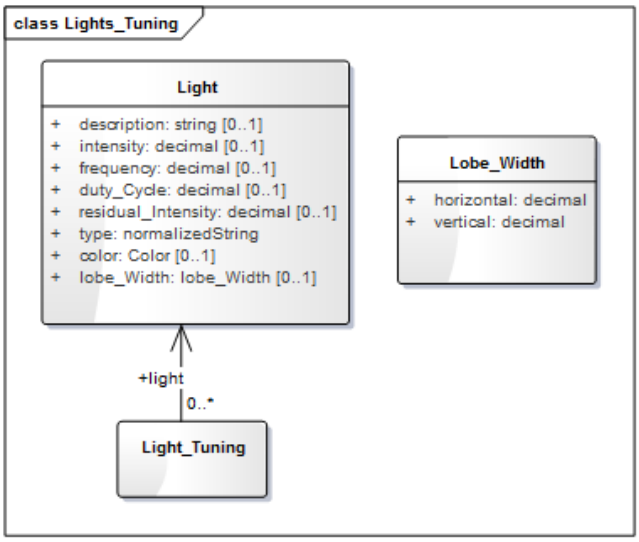


Figure 11. UML diagram of client specific lights definition

3.3. Model Components Definition

The CDB Standard provides the means to unambiguously tag any portions of a 3D model (moving model or cultural feature with a modeled representation) with a descriptive name. Component model names are stored in the model components' definition file, at \CDB\Metadata\Model_Components.xml which is delivered with the standard distribution package.

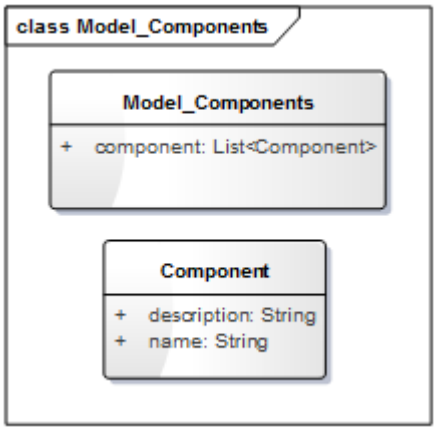


Figure 12. UML diagram of the model components definition

CDB Model Components is a list of components which are made up of the component names along with their descriptions.

3.4. Base Materials Table

A Base Material represents a basic material such as water, vegetation, concrete, glass, or steel. Each Base Material used in a CDB data store has a unique name. The components of a Base Material are listed in Volume 1 Section 2.5.1 of the CDB Core standard in Table 2-6: Components of a Base Material. A Base Material Table (BMT) is provided for run-time access by client applications. More details on the file format can be found in Volume 1 Core Standard section 5.1.3, Base Material Table. CDB Base Materials are listed and stored in an XML file named \CDB\Metadata\Materials.xml. The format of the file is defined by the following UML diagram generated from an XML schema that is delivered with the CDB standard in the file named \CDB\Metadata\Schema\Base_Material_Table.xsd.

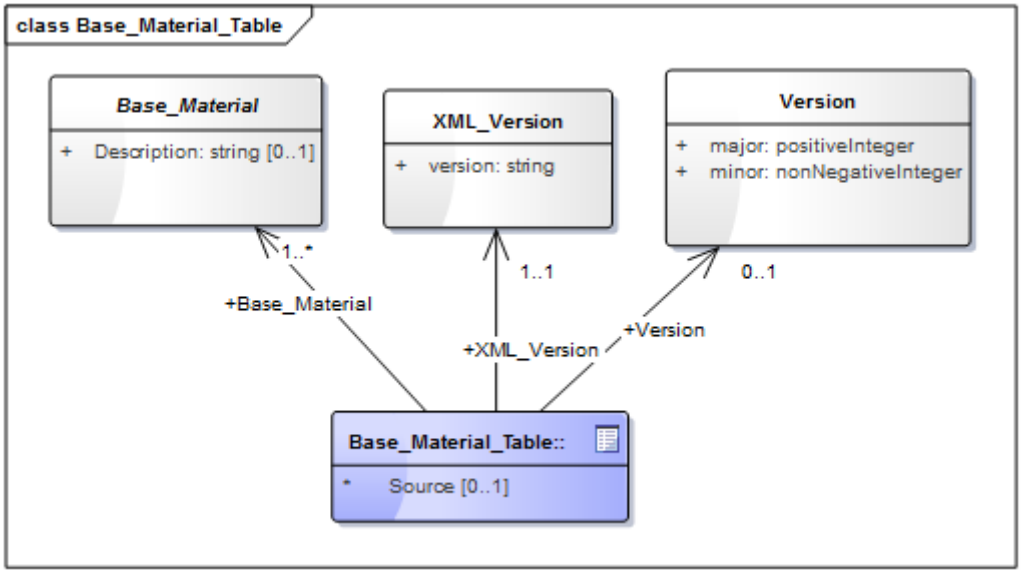


Figure 13. UML diagram of the base materials table

The main properties of the base materials table UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity
Base_Material	This element defines one CDB Base Material by giving it a unique name. It is recommended to provide a description.	XML	One or more (mandatory)
XML_Version	This element indicates the version of the XML file containing the list of CDB Default Values. It is independent from the version of the Standard.	string	One (mandatory)
Version	This attribute represents the version number of this file.	Version	One (mandatory)

3.5. Composite Material Tables

Composite Material Tables provide a structured arrangement by which Composite Materials can be defined. There are several Composite Material Tables spread across the CDB hierarchy. A CMT is a list of one or more composite materials. Note that all Composite Material Tables follow the following UML diagram.

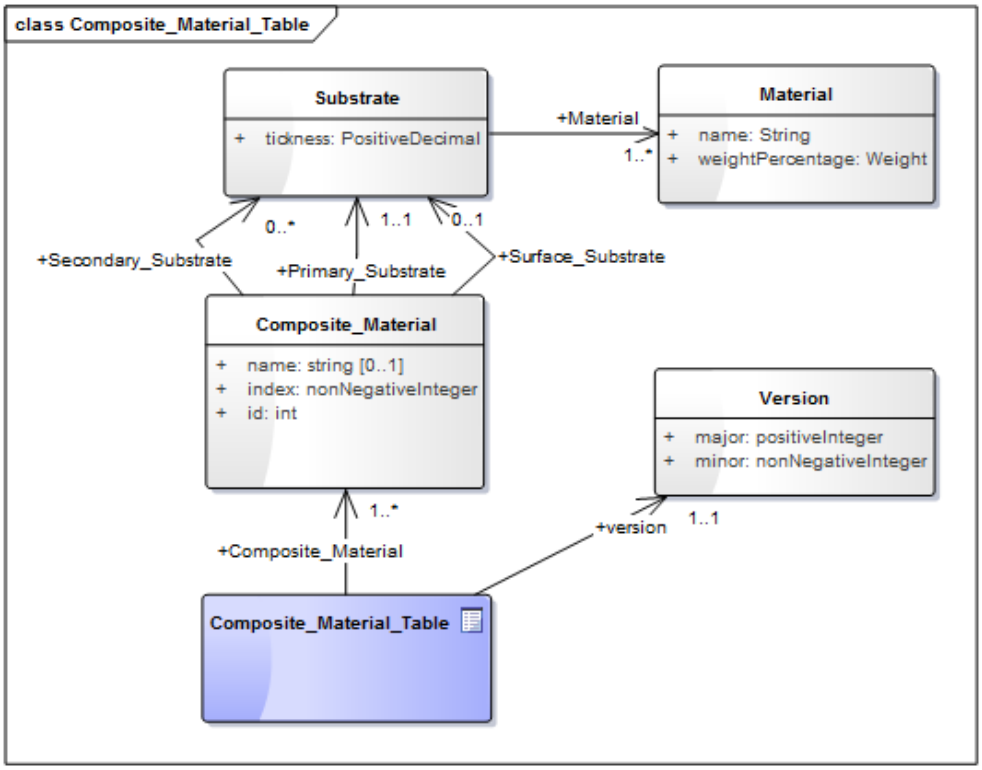


Figure 14. UML diagram of composite material tables

The main properties of the composite material tables' UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity
Composit_Material	Each composite material has a unique identification number, a name, and one or more substrates.	XML	One or more (mandatory)
Material	Each material is identified by the name of its base material and by its proportion in the substrate. This class has a weightPercentage which is an integer in the range [1,100].	Array of strings	One or more (mandatory)
Substrate	A substrate has a certain thickness and is composed of one or more base materials.	positive decimal	One or more (mandatory)
Version	This attribute represents the version number of this file.	Version	One (mandatory)

3.6. Default Values Definition Table

Default values for all datasets can be stored in the default values' metadata file "\CDB\Metadata\Defaults.xml". Default values, defined throughout the CDB standard, are listed in Volume 2 Annexes for the Core CDB Standard (normative) - Annex S and the below UML diagram indicates the schema provided in \CDB\Metadata\Schema\Defaults.xsd to define and validate the content of Defaults.xml. There are two types of default values: read and write default values ('R' or 'W'.) Generally, read default values are values to be used when optional information is not available. Write default values are default values to be used by CDB creation tools to fill mandatory content when information is either missing or not available. The default value name is a unique name identifying a default value for a given dataset. Valid default value names are listed in Annex S. Each default value has a type. Valid default value data types are "float", "integer" and "string".

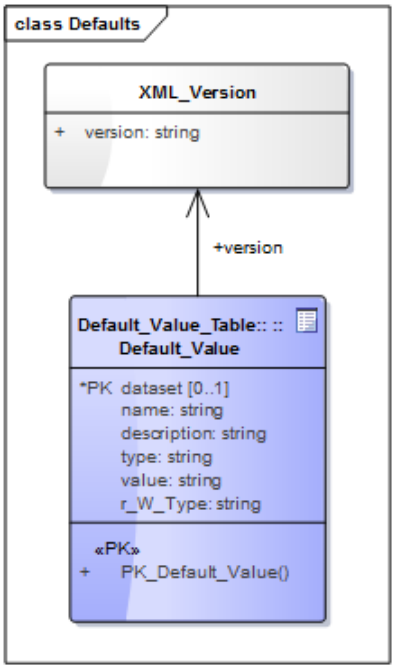


Figure 15. UML diagram of the default values definition table

The "XML_Version" attribute is used to indicate the version of the XML file containing the list of CDB Default Values. It is independent from the version of the standard.

3.7. Version

Each CDB version has a version control file that is called Version.xml. Its contents should be defined and validated by the following UML diagram which is generated from the content of Defaults.xsd in the schema folder of the CDB.

The optional <PreviousIncrementalRootDirectory> element is used to refer to another CDB Version. This is the mechanism used to chain together two CDB versions. The mandatory <Specification> element indicates the CDB standard that is used to produce the content of the CDB Version. Note that version numbers of the standard are limited to the version numbers from the legacy industry-maintained CDB specification, specifically 3.2, 3.1, and 3.0. For the OGC standard, allowed versions are 1.0, 1.1, and 1.2. All the OGC versions are backwards compatible. Other values are not permitted. Finally, the optional <Extension> element indicates that this CDB Version is in fact a CDB Extension. A version control file that does not have a CDB Extension indicates that the CDB Version

holds content that strictly follows the CDB standard.

A CDB Extension corresponds to user defined information, which is not described or supported by the CDB standard, stored within the CDB Version. As an example, such additional information could be client or vendor-specific information used to increase system performance. Any user defined information shall not replace or be used in place of existing CDB information. A CDB Extension only contains vendor or device specific information.

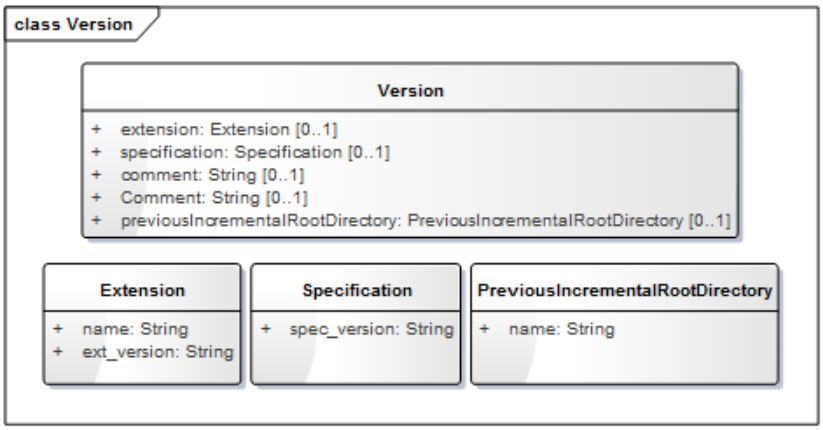


Figure 16. UML diagram of the version

3.8. Configuration

The CDB Configuration and CDB Version mechanisms allow users to manage the CDB by offering the following capabilities:

- The CDB can have multiple simultaneous independent CDB Configurations.
- Each CDB Configuration is defined by an ordered list of CDB Versions.
- A CDB Version is either a collection of CDB Datasets or a collection of user-defined datasets called a CDB Extension

The Configuration metadata file provides the means of defining CDB Configurations. The complete XML schema is provided in /CDB/Metadata/Schema/Configuration.xsd delivered with the standard and displayed below.

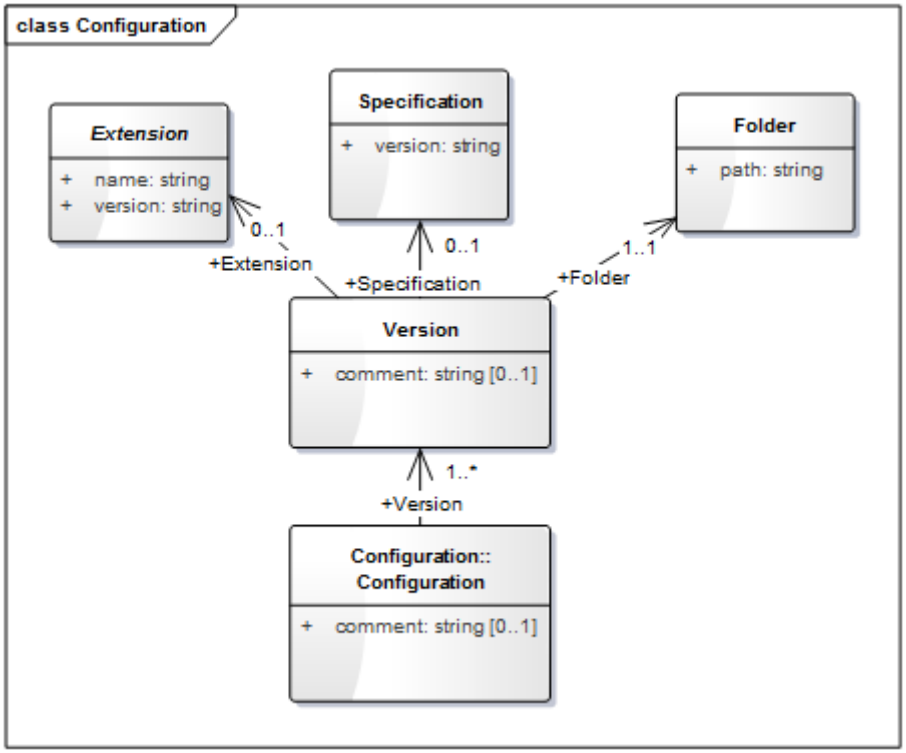


Figure 17. UML diagram of configuration metadata

A single XML file, named Configuration.xml, completely defines the configuration of a CDB. This way, the client application does not have to traverse the linked list of CDB Versions through the 'PreviousIncrementalRootDirectory' element found in Version.xml. The main properties of the configuration metadata UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity
Configuration	The CDB Configuration is a simple list of one or more CDB Versions.	XML	One or more (mandatory)
Extension	Indicates that the CDB Version contains extensions to the CDB Specification. The CDB Extension is identified by a name and a version number.	Array of strings	Zero or more (optional)
Folder	Provides a non-empty path to a folder. A relative path is preferred although an absolute path is supported.	string	One (mandatory)

Specification	Specifies the version of the CDB Specification/Standard used to generate the current CDB Version. If 'Specification' is omitted, the version number is deemed to be 3.0. For the OGC version of the standard, the Version number is 1.0, 1.1, and so forth.	string	Zero or more (optional)
Version	A CDB Version points to the folder where the data for that version resides. An optional comment can be used to describe the version. It is possible to indicate to which version of the CDB Specification/Standard the CDB Version complies. Finally, the CDB Version can indicate if it contains extensions to the standard.	String	One (mandatory)

3.9. CDB Vector Attributes

The CDB attributes are listed and described in Volume 1: CDB Core section 5.7.1.3 CDB Attributes. The controlled vocabulary for these attributes is stored in \CDB\Metadata\CDB_Attributes.xml and the following diagram indicates the schema file as provided in the CDB schema folder, Vector_Attributes.xsd. In essence, the file is the transposition of CDB Attributes into a format more appropriate for a computer program.

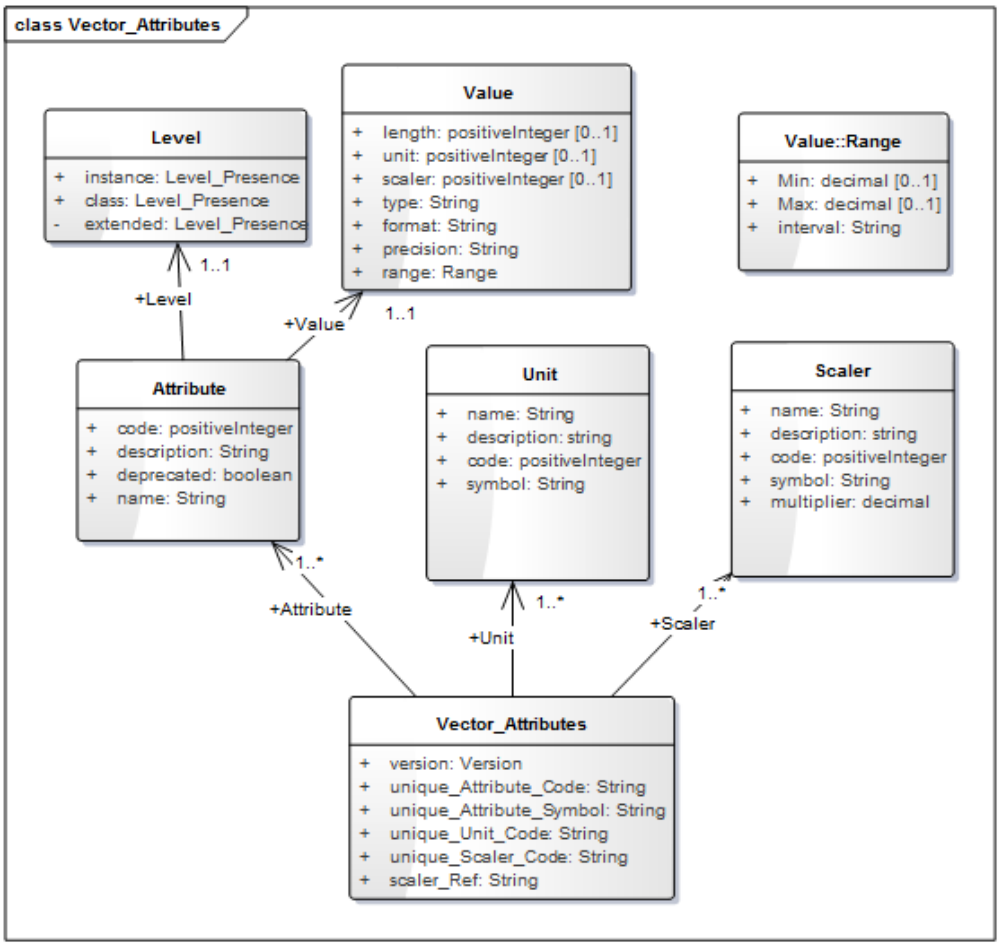


Figure 18. UML diagram of the CDB vector attributes

The UML diagram is composed of three majors sections (i.e., attributes, units and scalers), the first one being the most important. The file has a list of attributes, followed by two lists of units and scalers that are referenced by an individual attribute. The main properties of the vector attributes' UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity
Vector_Attributes	Attributes are defined through 3 lists: 1) the attributes themselves, 2) their units, and 3) their scalers.	Vector_Attributes	One (mandatory)
Version	This represents the version number of the file which has two components: major and minor.	String	One (mandatory)

3.10. 3D Model Metadata

This following UML presents an XML schema file in the CDB schema folder which defines the metadata associated with 3D models. These metadata are in accordance with the legacy industry-maintained versions of the CDB specification and includes name, feature data dictionary, mass,

part, texture and materials.

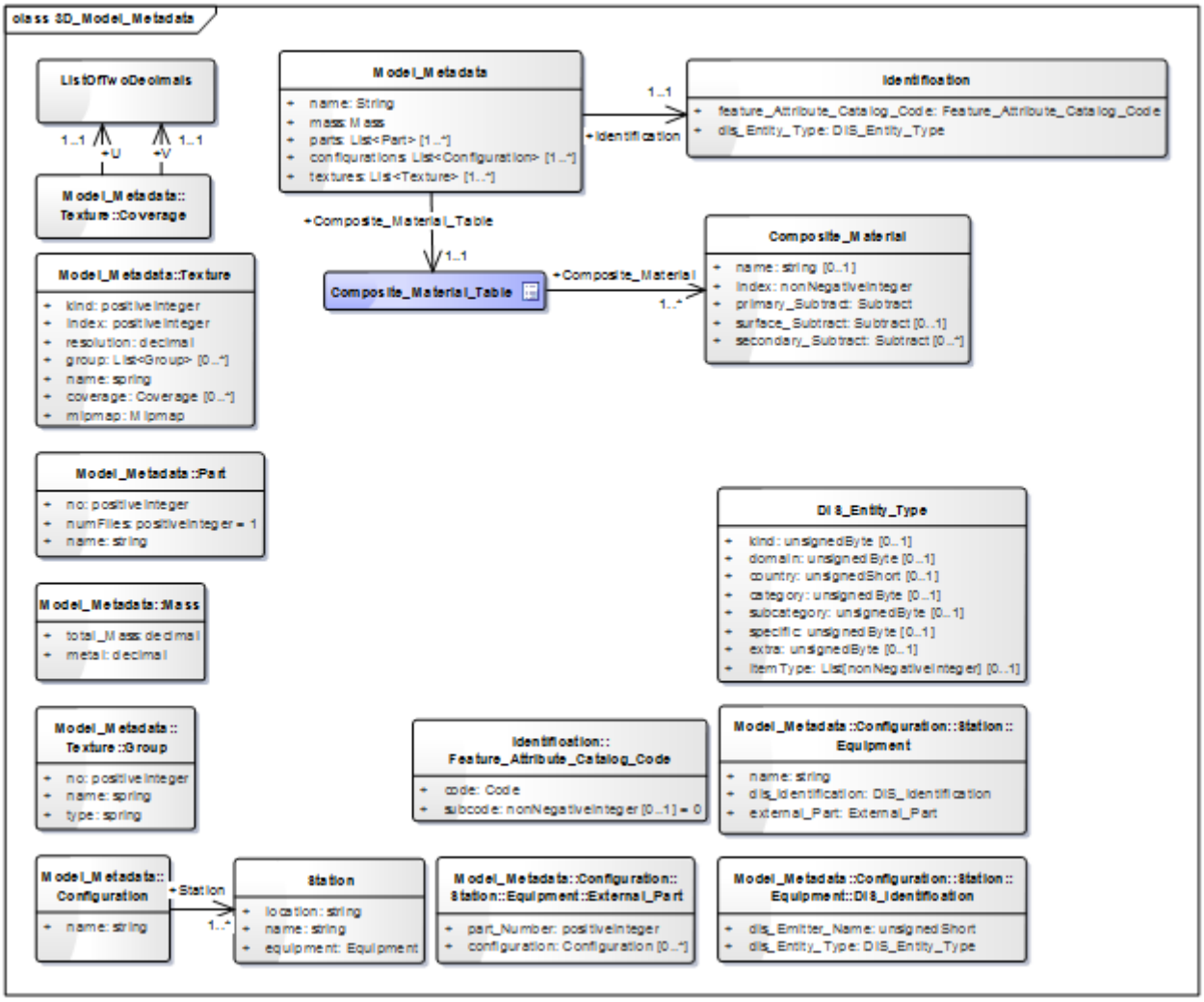


Figure 19. UML diagram of 3D model metadata

The main properties of the 3D model metadata UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity
3D_Model_Metadata	The metadata associated with a model is made of up to seven elements.	XML	One (mandatory)
Identification	A 3D model is either a moving model with a DIS Entity Type, or a cultural feature with a feature code (FC).	DIS or FC	One (mandatory)

DIS_Entity_Type	This type has two formats: 1) a simple list of up to 7 integers; or 2) a sequence of up to 7 elements providing the name of the fields whose values are being provided.	DIS	One (mandatory)
Feature_Attribute_Catalog_Code	This code is composed of two elements: a code and a subcode. The code is a string of 2 letters and 3 digits. The subcode is optional and defaults to 0.	FeatureCode	One (mandatory)
Mass	This is defined by two elements: total mass, and its metallic portion. By default, the metallic portion is assumed to be 0.	Mass	One (mandatory)
Part	When the list of parts is supplied, it contains at least one entry. If the list is absent, a single part stored in a single file is assumed. A part has a name and is made of a part number, and the number of files associated with the part.	Part	One (mandatory)
Textures	When the list of textures is supplied, it contains at least one entry. If the list is absent, the model does not have textures. Optionally, groups of textures may be defined and listed. An individual texture may optionally belong to texture groups.	Texture	One (mandatory)

Texture group	It is identified by its group number and its group name. Later, individual texture will refer to group numbers.	Texture group	One or more (mandatory)
Texture	A texture is defined by a sequence of 5 mandatory elements and 2 optional elements. The first 4 elements (Kind, Index, Mipmap, and Name) are used to compose the file name where the texture is stored. The Resolution can be used to select which mipmap to load. The optional Coverage provides the maximum extent of U and V mapping. The optional Group refers to the Texture_Group to which the texture belongs.	Texture	One or more (mandatory)
Configurations	the list of one or more configurations is supplied,	configuration	One (mandatory)
configuration	A configuration is a named list of one or more stations.	configuration	One or more (mandatory)
Station	A Station has a name and defines exactly one equipment in one location.	Station	One (mandatory)
Equipment	Equipment is defined by either a DIS key or an external part - and possibly both. An external part is identified by its part number. Optionally the part may have its own configuration.	Equipment	One (mandatory)

DIS identification	A DIS identification is either a DIS entity type or a DIS emitter name. A DIS emitter name is a 16-bit unsigned integer.	integer	One (mandatory)
Composite material table	A composite material table is a list of one or more composite materials. Each one has a unique identification number, a name, and one or more substrates. A substrate has a certain thickness and is composed of one or more base materials. Each material in a substrate is identified by the name of its base material and by its proportion in the substrate. A percentage is an integer in the range [1,100].	Composite	One (mandatory)

[5] http://www.opengeospatial.org/standards/cdb

Chapter 4. 3D Model Extensions

All of the statically positioned cultural features and the moving models are represented in the OpenFlight format. As such, OpenFlight plays a significant role, since. To add attributes to OpenFlight models, OpenFlight_Model_Extensions.xsd schema file can be used to verify the changes. The below diagram is generated from this schema file which is located in the CDB schema folder (/CDB/Metadata/Schema/).

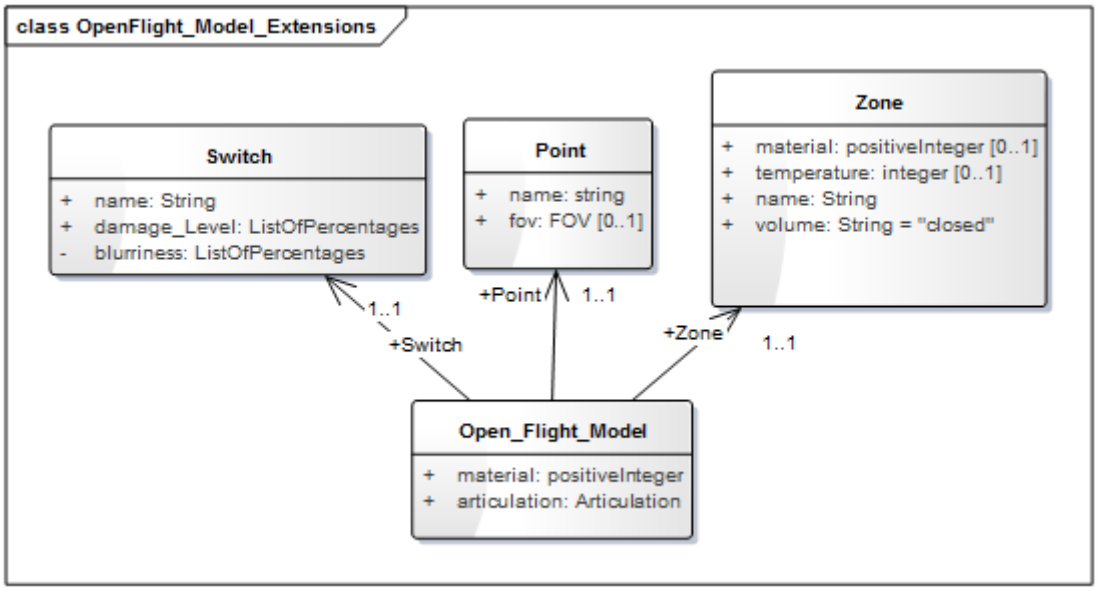


Figure 20. UML diagram of OpenFlight model extensions

Chapter 5. Feature Data Dictionary

The CDB Feature Data Dictionary (FDD) is provided with the CDB standard in the form of an XML file including the complete list of the supported feature codes. The following UML diagram is generated using the XML stylesheet which is provided to format and display the dictionary inside a standard Web browser. Furthermore, the schema can also be found in the schema subdirectory of the CDB Schema Distribution Package.

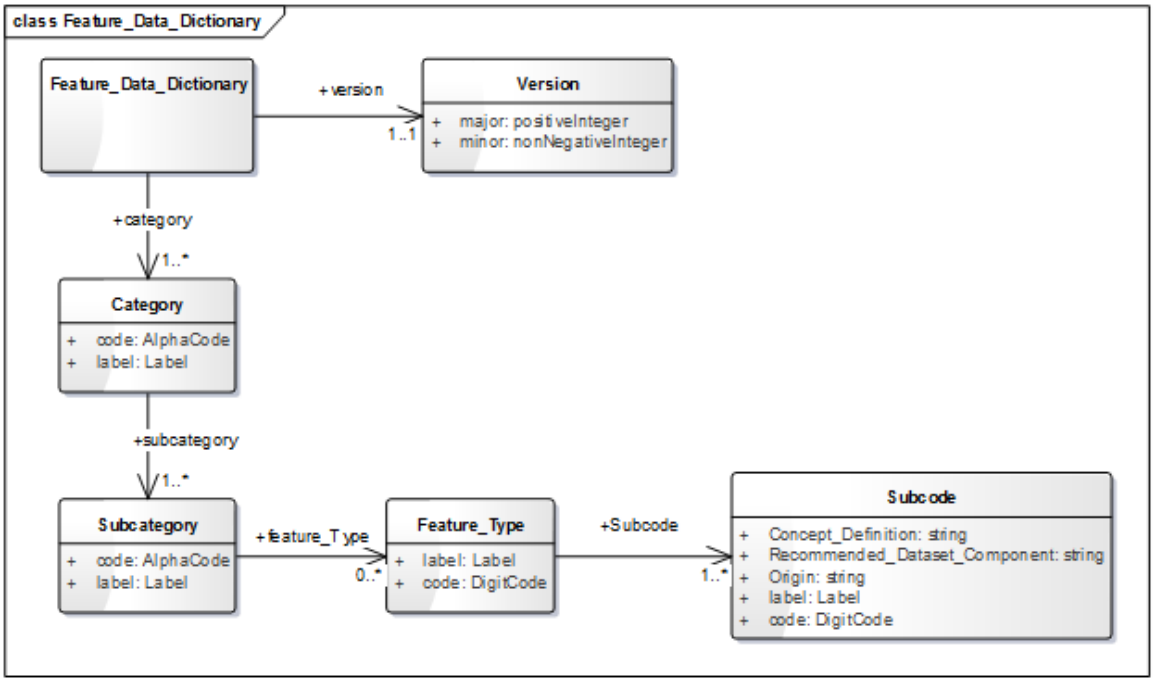


Figure 21. UML diagram of feature data dictionary

The main properties of the feature data dictionary UML diagram are documented below.

Name	Definition	Data type & Value	Multiplicity
AlphaCode	The code of the category or subcategory.	string	One (mandatory)
Feature_Data_Dictionary	This element represents the CDB Feature Data Dictionary root element. It has a version number and the list of all categories.	XML	One (mandatory)
Feature_Type	This element has a code attribute, a label and a list of subcodes.	Feature_Type	Zero or more (optional)
Label	A meaningful name to the code attribute.	string	One (mandatory)

Subcategory	Has a code attribute, a label and a list of feature types.	Feature_Type	One or more (mandatory)
Subcode	Has a code attribute, a label, a concept definition, a recommended dataset component and an origin.	Subcode	Zero or more (optional)
Version	This attribute represents the version number of FDD.	Array of strings	One (mandatory)

Annex A: Conformance Class Abstract Test Suite (Normative)

This section describes conformance test for the OGC CDB Conceptual Model Standard. A CDB dataset shall satisfy the following criteria to be conformant with the OGC CDB Conceptual Model Standard.

A.1. CDB Overall Conceptual Model

The following conformance class is designed to determine if any dataset claiming conformance to the CDB Conceptual model is described based on the comprehensive set of requirements.

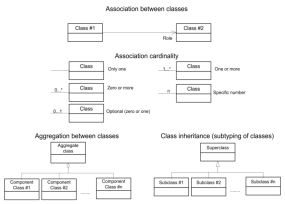
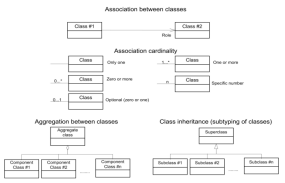
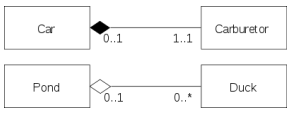
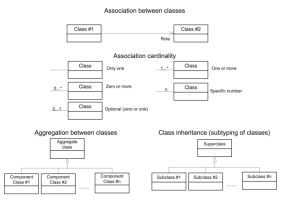
Conformance Class	http://opengis.net/spec/CDB/1.0/conf/core/OveralConceptualModel	
Requirements	http://opengis.net/spec/CDB/1.0/core/OveralConceptualModel	
Dependency	All of the requirements of the OGC CDB core document	
Test Class 1	Test purpose	Verify to test requirement class 1
	Test method	Review that the CDB dataset complies with the corresponding mandatory requirements of the OGC CDB core standard.
	Test type	Conformance

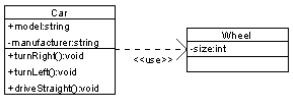
Annex B: UML notations

B.1. Class Diagrams Notation

A class diagram shows a collection of declarative (static) model elements, such as classes, types, and their contents and relationships. Classes and relationships represent real-world concepts to describe the structure of a system. The CDB Core Standard is presented in this document in diagrams using the Unified Modeling Language (UML) class diagrams. The UML notations used in this standard are described here (see ISO TS 19103, Geographic information - Conceptual schema language for the details).

A relationship is a general term covering the specific types of logical connections found on class and objects diagrams. UML shows the following relationships:

Relationships	**Definition**	**Diagram**
Association	It is semantic relationship between two or more classes that specifies links among their instances. In association, an attribute of the dependent class is an instance of the independent class.	
Aggregation	Form of association that specifies a whole-part relationship between the aggregate (whole) and a component part.	
Composition	It is an aggregation and stronger variant of the "has a" association relationship; if the container is destroyed, normally every instance that it contains is destroyed as well.	
Generalization	Taxonomic relationship between a more general element and a more specific element. It is also known as the inheritance or "is a" relationship. The superclass (base class) is also known as the "parent".	

Realizations	It is shown on classes, interfaces, components, and packages that connects a client element with a supplier element and shows that the class realizes the operations offered by the interface.	

The UML representation of an association is a line with an optional arrowhead indicating the role of the object(s) in the relationship, and an optional notation at each end indicating the multiplicity (the number of objects that participate in the association) of instances of that entity as listed in the below table:

0..1	No instances, or one instance
1, 1..1	Exactly one instance
0..*	Zero or more instances
n	Specific number
1..*	One or more instances

In the document, UML diagrams typically identifies a stereotype with a bracketed comment for each object identifying whether it is a class, interface, etc. Additionally, the UMLs are color-coded as follow to show different stereotypes:

Elements (e.g. Class & Object)	Interface	Description
Table	Data Type	Profile

Annex C: Revision History

Date	Release	Editor	Primary clauses modified	Description
2019-10-21	1.2	C. Reed	Various	Revision
2018-03-20	1.1	S. Saeedi	all	Revision
2016-04-04	1.0	S. Saeedi	all	Initial version