**Open Geospatial Consortium**

Submission Date: 2016-04-04

Approval Date:   2016-dd-mm

Publication Date:   2016-dd-mm

External identifier of this OGC® document: http://www.opengis.net/doc/IS/cdb-core-primer/1.0

Internal reference number of this OGC® document:    15-120r2

Category: OGC® Best Practice

Editor:   Carl Reed

**Volume 0:** Primer for the OGC CDB Standard:

Model and Physical Data Store Structure

**Copyright notice**

Copyright © 2016 Open Geospatial Consortium  
To obtain additional rights of use, visit <http://www.opengeospatial.org/legal/>.

**Warning**

This document is not an OGC Standard. This document is distributed for review and comment. This document is subject to change without notice and may not be referred to as an OGC Standard.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type:    OGC® Best Practice

Document subtype:    if applicable

Document stage:    Draft

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.

Contents

[1. What is a CDB Conformant Data Store? 8](#_Toc442446092)

[2. Conformance 10](#_Toc442446093)

[3. References 10](#_Toc442446094)

[4. Terms and Definitions 11](#_Toc442446095)

[4.1 4.1 Abbreviations 11](#_Toc442446096)

[5. Using OGC standards with a CDB structured data store 11](#_Toc442446097)

[6. Background and informative material 13](#_Toc442446098)

[6.1 Intended Audience 13](#_Toc442446099)

[6.2 Problem Definition 13](#_Toc442446100)

[6.3 Use of a CDB conformant data store as an Off-line Data store Repository 15](#_Toc442446101)

[6.4 Use of a CDB conformant data store as a Combined Off-line and Run-time Data store Repository 16](#_Toc442446102)

[6.5 Client-Device Independence 16](#_Toc442446103)

[6.6 Runtime Publisher 16](#_Toc442446104)

[6.7 Synthetic Environment Scalability & Adaptability 17](#_Toc442446105)

[6.8 Simulator Wide Unique Data Representation 21](#_Toc442446106)

[6.9 Key Benefits of the CDB Standard 22](#_Toc442446107)

[6.9.1 Improved Synthetic environment DB Generation Timeline 22](#_Toc442446108)

[6.9.2 Interoperable Simulation-Ready Synthetic environment DB 23](#_Toc442446109)

[6.9.3 Improved Client-device Robustness/Determinism 23](#_Toc442446110)

[6.9.4 Runtime-Adjustable Synthetic Environment DB Correlation and Fidelity 23](#_Toc442446111)

[6.9.5 Increased Synthetic Environment DB Longevity 24](#_Toc442446112)

[6.9.6 Reduced Synthetic Environment DB Storage Infrastructure Cost 24](#_Toc442446113)

[6.10 CDB Model Overview 25](#_Toc442446114)

[6.10.1 CDB Standard Data Representation and Organization 25](#_Toc442446115)

[6.10.2 CDB Standard Logical Structure 26](#_Toc442446116)

[6.10.3 Tile/Layer/Level-of-Detail Structure 27](#_Toc442446117)

[6.10.4 CDB Structure, Organization on Media and Conventions 30](#_Toc442446118)

[6.10.5 Use of CDB in Applications Requiring Dynamic Synthetic Environments 30](#_Toc442446119)

[6.10.6 Synthetic Environment Data store Correlation 31](#_Toc442446120)

1. **Abstract**

The CDB standard addresses the interoperability challenge of full plug-and-play interoperability and reuse of synthetic environment databases used for high fidelity simulation and mission rehearsal. The conception of synthetic environment as a three -dimensional digital representation of the natural environment is not inconsistent with many GIS applications, such as in City planning, where accurate simulations of the natural environment and man's impact on the environment is essential. Users are provided with a means of examining the spatial relationships of the digital objects that closely resemble their real world counterpart[[1]](#footnote-1).

The CDB standard defines a model and structure for a single, versionable, simulation-rich, virtual representation of the earth. A CDB structured database provides for a synthetic environment repository that is plug-and-play interoperable between data store authoring workstations. A CDB data store is entirely platform independent, i.e. it does NOT provide the implementation details of specific off-line database compilers or runtime publishers attached to specific client-devices. A critical aspect of such an environment is performance. A typical use case is a helicopter pilot using a training or mission simulator. The 3Dpresentation of the environment (natural and man-made) has to be visually displayed in real time at the highest fidelity possible. This critical requirement has driven many of the decisions, captured as requirements, in the CDB standard.

Moreover, a CDB structured data store can be used as a common online (or runtime) repository from which various simulator client-devices can simultaneously retrieve and modify, in real-time, relevant information to perform their respective runtime simulation tasks. In this case, a CDB compliant data store is plug-and-play interoperable between CDB-compliant simulators. A CDB structured data store can be readily used by existing simulation client-devices (legacy Image Generators, Radar simulator, Computer Generated Forces, etc.) through a data publishing process that is performed on-demand in real-time.

The application of the CDB standard to future simulator architectures will significantly reduce runtime-source level and algorithmic correlation errors, while reducing development, update and configuration management timelines. With the addition of the HLA/FOM[[2]](#footnote-2) and DIS[[3]](#footnote-3) protocols, the application of the CDB standard provides a Common Environment within which inter-connected simulators share a common view of the simulated environment.

The CDB standard specifies an open format for the storage, access and modification of a synthetic environment data store. The standard defines the data representation, organization and storage structure of a worldwide synthetic representation of the earth as well as the conventions necessary to support all of the subsystems of a full-mission simulator. The CDB standard makes use of several commercial and simulation data formats endorsed by leaders of the database tools industry.

The CDB synthetic environment is a representation of the natural environment including external features such as man-made structures and systems. A CDB structured data store can encompass terrain relief, terrain imagery, three-dimensional (3D) models of natural and man-made cultural features, 3D models of dynamic vehicles, the ocean surface, and the ocean bottom, including features (both natural and man-made) on the ocean floor. In addition, the synthetic environment includes the specific attributes of the synthetic environment data as well as their relationships.

A CDB data store contains datasets organized in layers, tiles and levels-of-detail. Together, these datasets represent the features of a synthetic environment for the purposes of distributed simulation applications. The organization of the synthetic environmental data in a CDB data store is specifically tailored for real-time applications.

1. **Keywords**

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

ogcdoc, OGC document, CDB, Common Data Base, simulation, synthetic environment, virtual, primer, data store

1. **Preface and Patent Call**

The initial CDB specification was developed under a competitive contract awarded to CAE to meet requirements of the United States Special Operations Command.

The CDB standard is widely implemented by multiple, independent industry contractors for end-user simulation and mission rehearsal customers in many different countries over a period of ten years.

The industry-maintained CDB model and data store structure has been discussed and demonstrated at OGC Technical Committee meetings beginning in September, 2013.

At the suggestion of several attendees at the first CDB ad-hoc meeting in September, 2014, the current version of the existing CDB standard was slightly reformatted for consideration as an OGC Best Practice. The OGC Best Practice is the basis of work for this OGC 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.*

1. **Submitting organizations**

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

CAE Inc.  
Carl Reed, OGC Individual Member  
Envitia, Ltd  
Glen Johnson, OGC Individual Member  
KaDSci, LLC  
Laval University  
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 Document 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).

1. **Submitters**

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

|  |  |
| --- | --- |
| Name | Affiliation |
| Carl Reed | Carl Reed & Associates |
| David Graham | CAE Inc. |

# What is a CDB Conformant Data Store?

A CDB[[4]](#footnote-4) conformant data store provides a representation of the whole earth optimized for very high speed access and visualization. The physical data store model divides the earth geographically into geodetic tiles. Each tile is defined by a latitude/longitude boundary. Each tile contains one or more specific sets of features. The CDB model uses the WGS-84 coordinate reference system. Details of the model along with implementation requirements are provided in Volume 1: OGC CDB Core [16-113]. Each of the simulator client-devices accesses CDB geospatial content using the WGS-84 coordinate reference system. The physical and logical tiling system specified in the CDB standard is an equal area implementation of a Discrete Global Grid System (DGGS)[[5]](#footnote-5). However, the implementation predates the OGC DGGS standards work and as such does not implement the OGC DGGS requirements.

A CDB data store contains the features and modeled representation of the synthetic environment. A CDB data store can contain:

* terrain altimetry;
* planimetry,
* raster imagery,
* attribution,
* 3D features with their modeled geometry,
* texture and attribution.

This standard also makes provision for the representation of moving models. An example of a moving model is a tank moving across the terrain as viewed by the helicopter pilot. The representation of moving models is comprehensive and goes well beyond other visualization standards because it makes provisions for the standardized simulator naming conventions, material and feature attribution, radar/laser reflectivity, aircraft and airport lighting systems, armaments, special effects, collision points/volumes just to name a few.

The CDB standard governs all requirements of the CDB data store, as follows:

* Data content and representation of the synthetic environment
* Structure and organization of the synthetic environment
* File format of the synthetic environment as stored on media

The CDB standard describes a modeled environment representation for distributed simulation applications. Section [6.10.5](#_Use_of_CDB) of this document, Use of CDB in Applications Requiring Dynamic Synthetic Environments, discusses how a CDB-compliant simulator could be adapted to handle modifications of the synthetic environment in real-time.

Given that a CDB data store is a structured representation and attribution of terrain, objects and entities, it is possible to design a broad range of synthetic environment simulations that modify synthetic environments in real-time. Such simulations can modify the CDB data store and notify other systems in a simulation federate[[6]](#footnote-6) that share a CDB data store about the changes. This provides a synthetic environment (SE) that is persistent and fully correlated across all simulation federates. For example, terrain “trafficability” could be handled by a new SE simulation that dynamically calculates soil moisture content as a function of localized rain precipitation and soil types/composition. A second simulation would then derive the resulting soil physics and determine vehicle wheel slippage across the varying terrain conditions.

The modification/notification approach is well-suited for a broad gamut of SE simulations. However, some simulations are very data intensive and require excessive broadcasting bandwidths to other federates. In such cases, these dynamic simulations would have to be replicated in the other client-devices of the federation. Good examples of this are visual system special effects (e.g., rotor downwash effect, missile plumes, muzzle flashes, cast landing lights). Typically such simulations are proprietary and intrinsic to the client-devices. Other examples of this include the varying effects of weather[[7]](#footnote-7) (local winds, temperature, humidity, particulates, etc.) and oceans (currents, temperature, etc.).

For ease of editing and review, the standard has been separated into 12 Volumes, one of which is 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 CBD 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 Use of Shapefiles for Vector Data Storage (Best Practice).
* Volume 5: OGC CDB Radar Cross Section (RCS) Models (Best Practice).
* Volume 6: OGC CDB Rules for Encoding Data 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: provides the normative schemas for key features types required in the synthetic modelling 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)

# Conformance

Not applicable for this Best Practice document.

# Informative References

There are no normative references in this document.

The following informative references are applicable for this Primer.

HLA [IEEE Std 1516]: 1516-2010 - IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules

DIS [IEEE Std 1278]: 1278.1-2012 - IEEE Standard for Distributed Interactive Simulation--Application Protocols

# Terms and Definitions

A complete glossary of Terms and Definitions used in this standard are contained in OGC CDB Volume 3. The following description of a synthetic environment is provided as context for this Primer.

The synthetic environment is a representation of the natural environment at a specific geographical location including the external features of the man-made structures and systems. Therefore, the synthetic environment includes the terrain, the terrain features (both natural and manmade), three-dimensional (3D) models of vehicles, the ocean surface, and the ocean bottom, including features (both natural and man-made) on the ocean floor. In addition, the synthetic environment includes the specific attributes of the synthetic environment data as well as their relationships. The CDB Standard is more than just a means of creating visual (aka out-the-window) scenery. Unlike other simulation industry standards that only deal with data representational types of polygons, colors, and textures, CDB deals with all the data types needed in high-end virtual and constructive simulation applications.

## Abbreviations

**CGF** - Computer Generated Forced

**DIS** - Distributed Interactive Simulation [IEEE std 1278]

**DBGF** - Database Generation Facility [IEEE std 1516]

**IG** – Image Generator

**LOS** Line of Sight

**NVG** Night Vision Googles

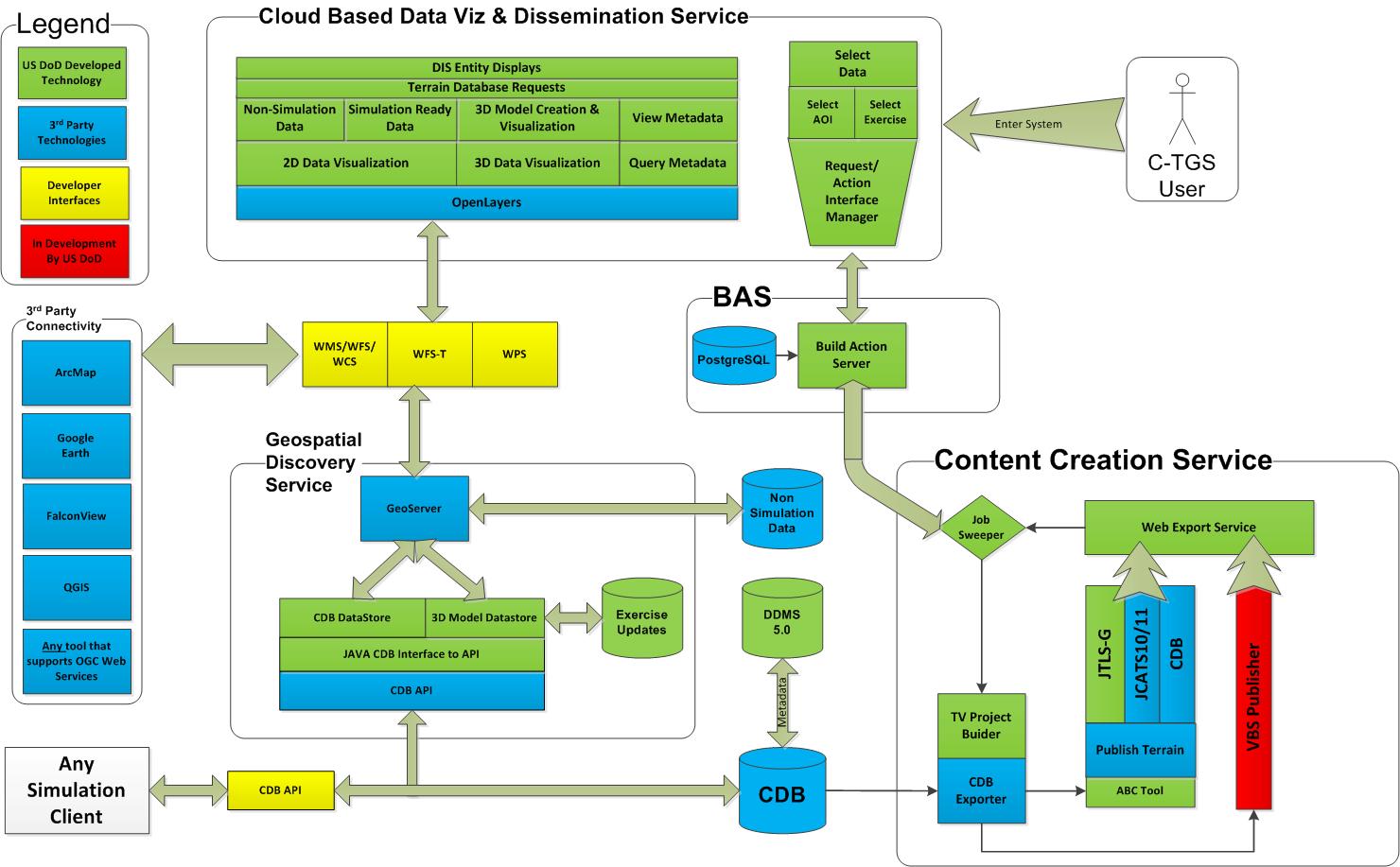
**OTW** Out the Window

**RTP** - Runtime Publisher

# Using OGC standards with a CDB structured data store[[8]](#footnote-8)

The U.S. Department of Defense (DoD) developed the Joint Training Data Services (JTDS) to support rapid scenario generation for the U.S. DoD M&S training community. The recently developed (2014) and deployed JTDS Common Terrain Generation Service (C-TGS) is a cloud-based service for the war fighter. C-TGS provides a robust capability to search, discover and retrieve Modeling and Simulation (M&S) terrain data. JTDS based the CB-TGS standard data structure on OGC-approved formats served by OGC Web Services. The goal for Terrain Generation Service (TGS): Eliminate the current paradigm where every simulation uses a different proprietary format. Current data efforts are fraught with simulation issues and errors, and data mining, correlation and processing to support multiple simulation formats is manpower intensive.

The TGS Solution: Have a repository and user interface that provides access to a single source of data through a persistent web service AND set and enforce a standard data structure based on industry standard Open Geospatial Consortium (OGC) approved formats, to normalize terrain data across Force Development simulations.



**Figure: DoD C-TGS Architecture**

**(Chambers and Sherman, 2014)**

The above figure represents the implementation architecture for CB-TGS. Implementation software represents a mix of commercial/proprietary and open source. At the heart of the C-TGS are two data stores: A CDB structured data store and a 3d model data store. All applications at the Visualization and Dissemination level interface the data stores via OGC service interfaces.

Connecting OGC Web Services to an open, simulation-optimized terrain data store, such as a CDB structured data store, results in high-end performance for serving and rendering geospatial content, the ability to dynamically modify terrain and support for all levels of simulation. a high-end simulation system using CDB can dynamically update the terrain (such as a bomb crater) and have that dynamically updated terrain instantly served by the Geospatial Discovery Service to any OGC compliant system, such as Esri® ArcMap, QGIS, or a mobile web page. The underlying architecture and framework of the C-TGS provide a mechanism to move high-end simulation to the point where data sharing, crowd-sourcing, and dynamic terrain are in-sync with GIS tools and simulation system edits of the terrain repository.

# Background and informative material

## Intended Audience

The primary audience for the CDB standard includes distributed simulation system developers and synthetic environment database tool developers whose applications are intended to read and/or write synthetic environment database files. To this end, this document discusses concepts incorporated in the standard and contains a detailed description of the physical layout of the files as represented on disk.

This document assumes the reader has some familiarity with:

1. Synthetic environment creation and generation processes.
2. Image Generation and radar simulation principles.
3. Sensor subsystems used in aircraft and other vehicles (Radar, Night Vision Image Intensifiers, Infrared (IR) sensors, Laser Range Finders, etc.) requiring the use of synthetic environment databases.
4. Simulator client-devices (such as Computer Generated Forces, Air/Ground Traffic Control, Weather, etc.) requiring the use of synthetic environment databases.

## Problem Definition

Complex mission simulators include a wide range of subsystems designed to simulate on-board equipment and to provide a rich gaming environment complete with weather, computer generated forces, ordnance, air traffic, networked players, etc. Each of these subsystems traditionally utilizes a proprietary runtime database (and format) for its synthetic representation of the gaming area. These application-specific formats have been generated off-line at a database generation facility using a variety of tools and processes. This approach has several inherent disadvantages, including length of time needed for synthetic environment production for multiple simulation applications, loss of correlation due to compilation differences, complexity in configuration management, and an inefficient update process. The abundance of distinct database formats creates several challenges for configuration management, resulting in mismatched correlations, and in the increased timeline needed to generate these databases. The CDB Standard specifies an approach that establishes a set of industry-standard formats and conventions for all simulator client subsystems (aka simulator client-devices).

The confluence of digital multispectral high-resolution satellite imagery, real time sensor feeds, and highly capable visual systems has created dramatic new Mission Planning and Mission Rehearsal capabilities. As a result, recent environment databases built to take advantage of these new capabilities require orders of magnitude more storage than equivalent databases just a few years ago.

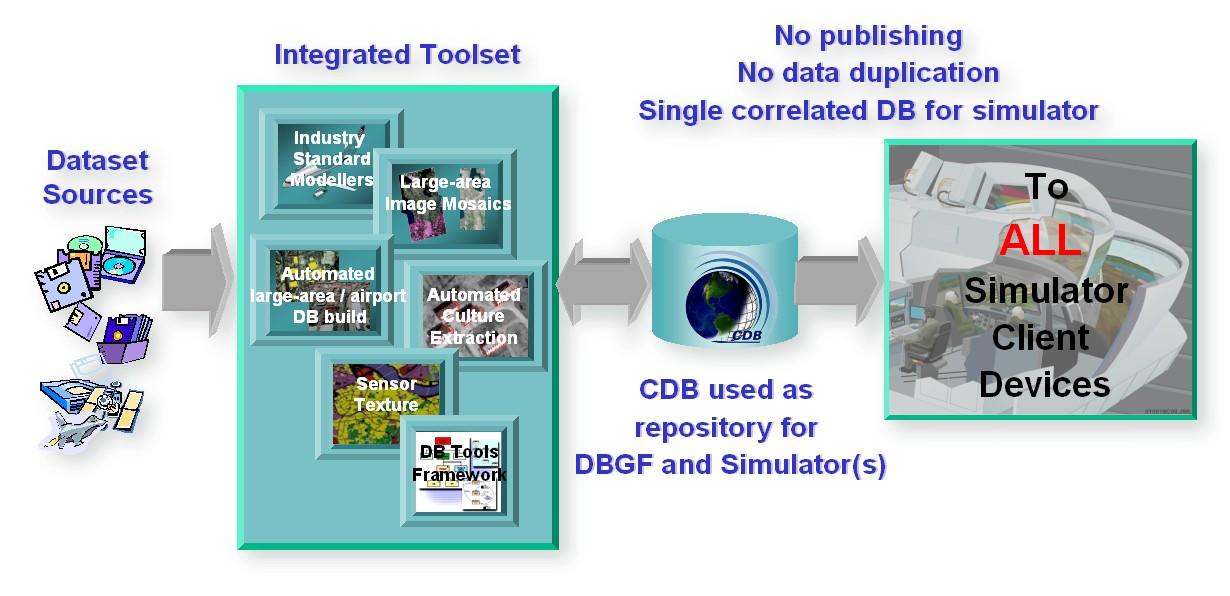
The CDB standard addresses these requirements through a common database model. CDB is intended as a simulation standard for use in producing a unified synthetic representation of the world. A database built to the CDB Standard is referred to as a CDB structured data store or CDB conformant data store. A CDB structured data store is a single-copy data repository from which various simulator client-devices are able to simultaneously retrieve, in real-time, relevant information to perform their respective runtime simulation tasks.

The CDB standard enhances unity and correlation between various simulator level client-devices (e.g., Visual, Radar), while improving maintainability. As a result, one of the main benefits of the CDB standard is the elimination of several types of correlation errors attributable to alternate, sometimes conflicting data representations required by each the simulator client-devices. The CDB standard achieves this by allowing all simulator clients-devices to share, in runtime, a single repository of the synthetic environment information. In addition, a CDB structured data store can also be used as a master repository for the authoring tools. As a result CDB content can be interchanged between DB workstations. Finally, in the case where one or more of the client-devices are not compliant to the CDB standard, it is possible to revert to the conventional compilation paradigm, (i.e., compile the CDB into one or more client-device loadable (usually proprietary) representations).

The CDB standard data representation model is based on the native formats used by industry-standard toolsets. This eliminates the time-consuming off-line database compilation process for each of the clients. The CDB standard redefines a new balance between offline and online compilation processes because modern computer platforms can now accomplish most of the compilation process in real-time[[9]](#footnote-9).

The CDB standard addresses the issues that have characterized the simulation industry for past decades (see Figure 1: CDB Approach to Synthetic Environment Generation), as follows:

1. Size of Synthetic Environment Storage: The CDB standard specifies how to consolidate the synthetic environment into a single data repository that provides a static representation of the earth. This includes all the relevant information so that all the simulator client-devices can perform their respective simulation tasks in order to meet the training and mission rehearsal requirements. This approach avoids any data content duplication. Storage intensive datasets can be optionally compressed using modern third-party algorithms. The CDB standard provides a fine-grained versioning scheme that avoids the replication of the entire DB when effecting small-area updates.
2. Scalability of Synthetic Environment Database: A CDB structured database can be built to a size or a density that far exceeds the capabilities of some or all of its client-devices. The data structure defined by the CDB standard makes it possible to implement runtime filtering schemes to adjust the loaded content density to the specific capabilities of the client-devices. As a result, a CDB structured data store can be scaled to take advantage of future simulator technological improvements.
3. Environment DB Correlation: Since CDB structured content is unique (without data duplication), runtime source level correlation errors among clients are eliminated, thereby ensuring inter-subsystem coherence and simulator interoperability. In addition, it is possible to improve correlation by adjusting the runtime publishing process associated with each client-device.
4. Database Availability Timeline: The CDB data store generation process allows for small incremental updates, thereby shortening generation and build process times. Furthermore, the translation step into a CDB structured data store is rather straightforward since the industry-standard formats and encodings are used.
5. Configuration Management: Configuration management effort is reduced, because a single CDB structured data store corresponds to the synthetic environment of all the client-devices in the simulator. Furthermore, while a CDB data store instance is conceptually a single, yet layered entity, the standard specifies how incremental updates are performed resulting in efficient storage and handling of CDB data store versions.



**Figure 1: CDB Approach to Synthetic Environment Generation**

## Use of a CDB conformant data store as an Off-line Data store Repository

CDB can be used as a structured offline repository. This capability is described in Volume 10 Implementation guidance in Section 6.2.

## Use of a CDB conformant data store as a Combined Off-line and Run-time Data store Repository

A data store conforming to this CDB standard can be both used an offline repository for data store authoring tools or as an on-line (or runtime) repository for simulators. This approach is described in Volume 10 Implementation Guidance, section 6.4.

## Client-Device Independence

The CDB standard is client-device independent. The standard is based on a requirement that all client-devices likely encountered on a tactical mission simulator need to be supported. The standard is completely independent of any vendor-specific devices.

## Runtime Publisher

The runtime publishers provide the necessary level of abstraction. The level of abstraction provided by the publishers is purely a function of the selected implementation of the client-device and its associated publishers. A client-device can “understand” and be completely “aware” of a CDB compliant data store as defined by the CDB standard. In this case, such a device would not require a runtime publisher, because it is already CDB data store native.

The runtime publishers bridge the gap between the information content requested by the attached client-device and the information content and structure available to them in a CDB conformant data store. As a result, the runtime publishers must each be theoretically “aware” of the following CDB concepts:

1. Tile: Ability to understand the concept of earth surface areas hierarchically subdivided in accordance to the CDB standard;
2. Level of detail: Ability to understand the concept of resolution or level-of-detail, for terrain, cultural vector data, raster imagery, and model geometry;
3. Terrain surface representation: Ability to understand concepts pertinent to earth surface geometry and attribution;
4. Cultural vector data (point, linear, areal): Ability to understand the concept of point, linear and areal cultural data and related attribution, fixed or conformed to earth surface;
5. Grid-organized data and meshes: Ability to understand the concept of grid-organized single-value datasets (e.g., elevation grid) and multi-value datasets (e.g., color triplets);
6. Imagery: Ability to understand the concept of color raster imagery mapped onto terrain surfaces or models;
7. Model geometry (incl. light points): Ability to understand the concept of general surface geometry;
8. Model positioning capability: Ability to differentiate between statically and dynamically positioned models;
9. Descriptive attribution: Ability to understand the attribution concepts pertinent to the client-device.

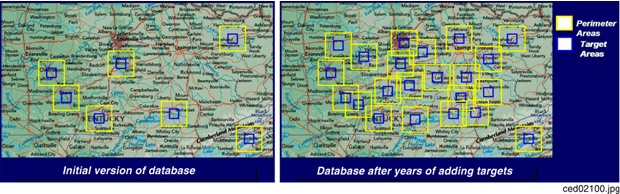
## Synthetic Environment Scalability & Adaptability

The concept of scalability when applied to synthetic environments not only applies to the geographic extent of the data store but more importantly it also reflects the ability to scale the environment in resolution and fidelity. These concepts are fully supported by the CDB Standard and are explained below:

1. Geographic extent: Correspond to the range of geographic extent of the earth surface that can be modeled.
2. Resolution: Correspond to the range of information density (for instance, the number of elevation values per km2) of the modeled datasets.
3. Fidelity: Correspond to the amount and type of synthetic environment data that can be modeled to support higher-fidelity simulator client-devices[[10]](#footnote-10). Consider for instance a simulator capable of supporting a single-surface earth skin representation versus one capable of representing tunnels, bathymetric data, location-dependent tide heights, location-dependent wave heights, etc. Clearly, the amount of information required by the higher-fidelity simulator is greater.
4. Precision: Correspond to the range of numerical precision (i.e., number of bits allocated to represent a quantity) of the modeled datasets.

Modelers face difficult challenges if they want a synthetic environment that is both scalable and adaptive. The solution to this difficult issue extends beyond the “mechanics” of achieving backward/forward compatibility. The solution also requires a complete “dissociation” of the data store from any constraints imposed by client-devices. Current practice today is for modelers to repeatedly adjust the content, resolution and density of synthetic environment databases to closely match the capabilities and performance of the targeted client-devices. When content is added to the database, previously modeled content is usually removed or simplified. Under such constraints it is difficult for a modeler to build a synthetic environment database capable of meeting anything but its immediate requirements.

Figure 2: *Typical Evolution of a Database* shows the evolution of a modeled region for use in a mission rehearsal. The initial version of the region may have only a few high-resolution/high-fidelity areas. However, over a given time period modelers will be asked to model additional target areas. As a result, the size, complexity, resolution and fidelity of the synthetic environment database gradually increase. Without built-in provisions for scalability, significant rework of the database is required each time a target area is added.



**Figure 2: Typical Evolution of a Database**

The CDB standard allows the “dissociation” of the synthetic environment database’s extent, resolution, fidelity, precision, structure and format from client-devices. This concept is not limited to aspects of formatting, numerical representation, internal structure, file structure, file systems, etc. More importantly the concept also covers aspects relating to synthetic environment database fidelity and resolution. Historically, the fidelity and resolution of synthetic environment databases has been intimately linked to the capabilities of the targeted simulator client-devices. More often than not, the source data was manipulated and adapted to constraints imposed by the client-devices. As a result, the content, resolution and density of synthetic environment databases were repeatedly adjusted to closely match the capabilities and performance of the targeted client-devices. The amount of time devoted to repeatedly adding/editing/removing content, and then repeatedly re-publishing would largely exceed the effort of creating and building the original synthetic environment database. The CDB standard offers the means to break this inter-dependence.

When assembling a CDB conformant data store, the synthetic environment database modeler is permitted (within their time-cost budget) to include content that significantly exceeds the capabilities of their simulator(s). The job of “adjusting” content to client-devices is relegated to the runtime publishers; the modeler is freed from this labor-intensive, repetitive task.

In a typical CDB data store implementation, it is anticipated that client-devices will independently control their respective runtime publishers so that the runtime published synthetic environment corresponds to their inherent capabilities (resolution, fidelity, etc.).

Nonetheless, the runtime publishing paradigm offers interesting new possibilities. For instance, it would be possible to individually adjust the fidelity and resolution of the synthetic environment for each client-device; this adjustment could be done at any time. As a result, it is possible to control and adjust the overall simulator fidelity and correlation to a level that was previously unachievable.

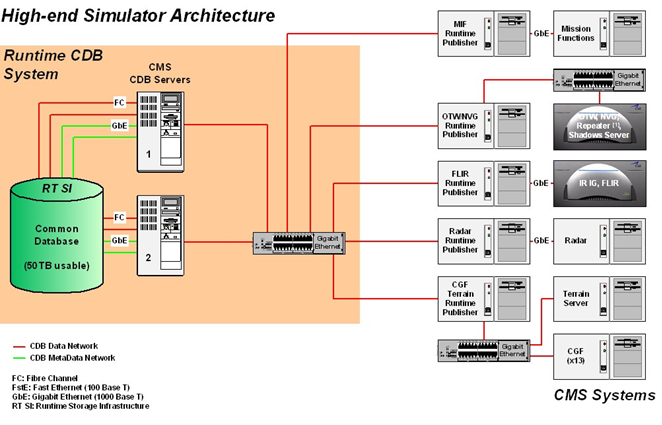
The CDB standard does not enforce a particular computer hardware infrastructure. The selected infrastructure allocated to the handling of a CDB data store is largely determined by the overall simulation requirements. Any leeway the simulator architect may have at their disposal when trading-off various simulator performance parameters against each other, are as follows:

1. Geographic extent
2. SE/Simulator Resolution
3. SE/Simulator Fidelity
4. SE/Simulator Precision
5. Desired level of client-devices correlation
6. Client-level SE load management
7. Simulated aircraft speed
8. CDB sharing

An experienced simulation engineer can typically undertake this analysis. However, the process requires a good understanding of the content available in the targeted CDB structured databases and of the content each client-device needs in order to meet the stated level of performance and fidelity.

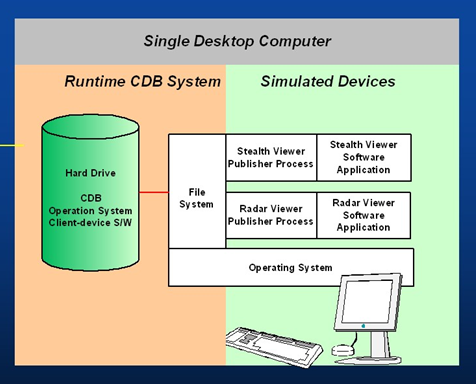
Alternately, it is possible to implement a simulator with client-devices (or attached publishers) that are capable of automatically adjusting resolution and fidelity in order to overcome performance limitations attributable to the CDB storage infrastructure and/or runtime publishers.

Figure 3: *Typical Implementation of the CDB standard on High-end Simulator provides* a system block diagram of a typical implementation of the CDB standard on a high-end tactical flight simulator. The runtime CDB system serves the combined needs of a mission functions computer, an OTW/NVG IG, a Forward Looking Infrared (FLIR) IG, Radar and a Computer Generated Forces (CGF) subsystem equipped with its own terrain server. The runtime CDB data store system is scaled to reflect the collective capabilities of the attached client-devices; as a result, the storage system is configured to supply the necessary bandwidth. Similarly, the IO bandwidth of the CDB data store servers and processing performance of the runtime publishers are scaled to satisfy the demands of their respective client-devices.



**Figure 3: Typical Implementation of CDB Specification on High-end Simulator**

Figure 4: *Typical Implementation of CDB Standard on Desktop Computer* shows a modest application of the CDB standard. This approach consists of a single desktop computer equipped with both stealth viewer and radar simulation application software. Each application is front-ended by a runtime publisher that in turn interfaces to the CDB data store via a standard file system. Given the more modest platform resources, some trade-offs in either resolution or fidelity might be required. This can be implemented in the runtime publisher or in the client-device application software.



**Figure 4: Typical Implementation of CDB Standard on Desktop Computer**

## Simulator Wide Unique Data Representation

A CDB data store is a single repository for the entire simulator’s synthetic environment DB. The internal structure ensures that all datasets are uniquely represented yet available (through a publishing process) to each of the simulator client-devices at runtime. This paradigm eliminates the extensive duplication of datasets that are shared by two or more client-devices.

The CDB standard is technically a normalized data model in the sense that it best meets the logical data design objectives of correctness, consistency, simplicity, non-redundancy and stability. Ignoring any tailoring or tuning for particular application requirements or performance, a normalized design provides the following advantages:

1. Minimize amount of space required to store data: Normalization precludes storing data items in multiple places.
2. Minimize risk of data inconsistencies within the data store: Since datasets are stored in only one place, there is no risk of datasets becoming inconsistent (uncorrelated).
3. Minimize possible update and delete anomalies: A normalized CDB data store eliminates the concerns related to update or delete operations.
4. Maximize the stability of the data structure: The process of normalization helps in associating attributes with entities based on the inherent properties of the data rather than on particular application requirements. Thus, new application requirements are less likely to force changes to the CDB model/data store design.

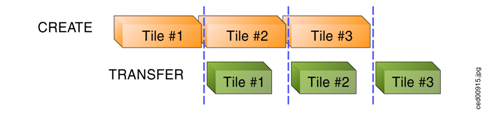
## Key Benefits of the CDB Standard

The following outline the key benefits of using the CDB standard.

### Improved Synthetic environment DB Generation Timeline

Important reductions in both the DB generation and DB update timeline are expected from an implementation of the CDB standard because:

1. There is no need to compile distinct synthetic environment runtime data stores for each of the simulator client-devices.
2. All of the datasets common to two or more client-devices need not be duplicated. All associated client-specific structures are also eliminated.
3. Tiles and layers are technically independent from each other; as a result, there is no need to reprocess neighboring tiles and coincident layers. However, one exception to this relates to the level-of-detail generation.
4. The tile structure permits users to “pipeline” or overlap the DB creation/update process, see Figure 5: Pipelined DB Update Process, with DB transfer process[[11]](#footnote-11).



**Figure 5: Pipelined DB Update Process**

1. The tile structure lends itself naturally to the concurrent or “parallel” production of the CDB data store.
2. The internal versioning mechanism lends itself well to CDB structured data store updates because only the affected tiles or layers need be re-transmitted to the simulator. This represents a significant timesaving, especially in cases where small updates are constantly applied to a comparatively large CDB structured data store.
3. The conversion of the synthetic environment from its tool-native representation into a form directly entered by each of the simulator client-devices is broken down into several publications accomplished in real-time on behalf of each of client-device. This approach permits the CDB standard representation of the synthetic environment to be “dissociated” from the resolution, fidelity, precision, structure and format imposed by each client-devices.

### Interoperable Simulation-Ready Synthetic environment DB

A CDB compliant data store is a fully interoperable, simulator-ready synthetic environment DB, (i.e., it can be used “as-is” on any CDB-compliant simulator). Because the data store is free of any simulator dependencies, there is no need to undertake a time-consuming and expensive rework of the runtime data store(s) in order to adapt it (them) to the format, structure, content, and precision constraints imposed by the simulator.

### Improved Client-device Robustness/Determinism

The CDB standard tile organization provides the means to implement deterministic loading and/or paging of the data store because each tile corresponds to the same amount of data (i.e., a “quanta” of information called a LOD-tile), regardless of its position on earth and regardless of its assigned LOD. This is a key characteristic of the CDB standard and is necessary to ensure deterministic operation of client-devices, even when the data resolution varies considerably from region to region or when the data is modeled at an extremely high-resolution. It is quite straightforward for a client-device to determine the amount of memory it must locally allocate when loading (or paging-in) a geographical area of interest. If the geographical area of interest exceeds the client-device’s capabilities, it can easily revert to a coarser LOD, hence gracefully degrading, as opposed to aborting (due to an internal failure in allocating sufficient memory) or “skipping” over the offending area.

The tile organization lends itself to robust, deterministic management of memory within client-devices because memory can be allocated/de-allocated in fixed sized quantities. As a result, client-devices need not concern themselves with complex and non-deterministic memory de-fragmentation schemes that do not lend themselves to real-time applications.

### Runtime-Adjustable Synthetic Environment DB Correlation and Fidelity

The CDB standard plays a critical role in improving the internal correlation of a synthetic environment because it eliminates the replication of runtime data stores for each of the client-devices. The prior art in simulation mandated replicated copies of the synthetic environment data store. Each copy was constrained by the content, formats, and structures specific to each client-device. As a result, the potential for correlation errors abounded. The CDB standard resolves this issue by defining a single, usable in real-time, open, synthetic environment data store representation.

The “runtime publishing” paradigm now permits the simulator vendor the means to not only control client-device load but to globally re-examine and control the level of correlation within a simulator (or across simulators). While the CDB standard does not provide explicit jurisdiction over the implementation of this mechanism in the client-devices and/or publishers, it is nonetheless possible to improve parametric correlation, at runtime, via control of the client-devices/publishers.

This topic is discussed in more detail in the OGC CDB Best Practice, Volume 1 Section 1.6.6 , Synthetic Environment Database Correlation.

### Increased Synthetic Environment DB Longevity

The longevity of synthetic environment databases for use in simulation has traditionally been a source of considerable aggravation within the user-community.

Traditionally a considerable level of effort (both human and machine) was required to adapt source-level data to a form that is directly usable by each of the simulator client-devices, also known as the runtime-level vendor-specific database format; this is referred to as the “compilation” process. More often than not, the source data is manipulated and adapted to constraints imposed by one or more simulator client-devices. In most cases, the content, resolution and density of synthetic environment databases are repeatedly adjusted to closely match the capabilities and performance of the targeted client-devices. While it is true that the native tool format database remains independent of the targeted client-devices, it is clear that content of the source-level database roughly corresponds to the capabilities of the then-current client-devices. As a result, source-level databases become quickly outdated and do require a complete rebuild to take advantage of new simulator capabilities.

The CDB standard avoids these pitfalls because the model does not need to be adapted to the constraints imposed by simulator client-devices; that role is relegated to the runtime publishers. Hence, the synthetic environment database can be built, right from the start, to a level of fidelity commensurate with the anticipated useful life of the targeted simulator(s).

### Reduced Synthetic Environment DB Storage Infrastructure Cost

For equivalent synthetic environment DB content, the CDB standard offers a significant storage space savings and significant reductions in the required interconnect infrastructure to supply the synthetic environment DB to the simulator(s).

The reduction in equipment and labor can be attributed to the following features:

1. Simulator-wide unique data representation: eliminate duplication of datasets across client-devices.
2. Compression of storage-intensive datasets: provides effective compression of key datasets.
3. Fine-grain versioning: A CDB structured data store is internally versioned. It is possible to revert to prior representations of the SE without restoring stored back-ups of the data store. Because the underlying mechanism is fine-grained, only in affected geographic areas or datasets of the data store need to be versioned.

## CDB Model Overview

The following sections provide a general description and focus of the CDB standard.

### CDB Standard Data Representation and Organization

The CDB standard provides the means of describing all feature sets relevant to simulation (such as terrain and 3D objects). These feature sets are logical re-groupings of datasets that are used directly by the simulator client-devices.

The standard currently supports the following representations:

1. Terrain: A representation of the terrain shape/elevation, raster imagery, surface attribution and other surface characteristics relevant to distributed simulations.
2. Point feature: A representation of a single location in space or on the earth’s surface. A Point feature consists of a single <latitude, longitude> coordinate with or without an elevation. When a point feature does not have an elevation, it is deemed to be on the surface of the earth. The information includes point-feature type identification, location, orientation, connectivity, attribution and other characteristics relevant to simulation.
3. Linear feature (aka LineString): A representation of predominantly man-made multi-segmented line-oriented features conformed relative to the terrain (such as runways, roads, transmission lines, fences). The information includes linear feature type identification, location, orientation, lineal geometry, connectivity, attribution and other surface characteristics relevant to simulation.
4. Areal feature (aka Polygon): A representation of closed area features conformed relative to the terrain such as forested areas and fields. The information includes areal feature type identification, location, orientation, 2D geometry, connectivity, attribution and other surface characteristics relevant to simulation.
5. 3D cultural model (aka Built Environment): Is a model that is statically positioned on the terrain or bathymetry skin. Cultural models are often a 3D representation of a man-made or a natural object positioned and conformed relative to the terrain. The information includes its geometry, articulations, raster imagery (texture, normal map, light map, etc.), lighting systems, and other characteristics relevant to simulation.
6. Moving model: A model that is not fixed at one location in the synthetic environment data store. The simulation host can update the position and orientation of a moving model at every simulation iteration cycle. A moving model is a 3D representation of man-made objects free to move within the data store. The information includes feature type identification, (vehicle class, type, model, etc.), geometry, articulations, raster imagery (texture, normal map, light map, etc.), lighting systems, connectivity to special effects, attribution and other characteristics relevant to simulation
7. Materials: A symbolic representation of the surface materials for all of the elements contained within the data store. Client-devices are required to simulate the synthetic environment over different portions of the electromagnetic spectrum: IR (e.g., FLIR, NVG), microwaves (e.g., Radar), audio (e.g., Sonar), etc. The fidelity of the sensor synthetic environment simulation model that runs in each of the client-devices is highly dependent on the richness and completeness of properties that characterize the synthetic environment data store in the electromagnetic spectrum of interest[[12]](#footnote-12).
8. Navigational data: A representation of ARINC-424 and DAFIF data in the form of NAVAIDs, Airport/Heliport, Runway/Helipad, Waypoints, Routes, Holding Patterns, Airways, Airspace, etc.

In order to represent the above feature sets, the CDB standard logically organizes its data in mutually exclusive datasets. Furthermore, the CDB standard is platform and client-device independent. As a result, the internal data representation is free of client-device specifics and more closely aligned to DB structures/formats supported by prominent industry standard tools.

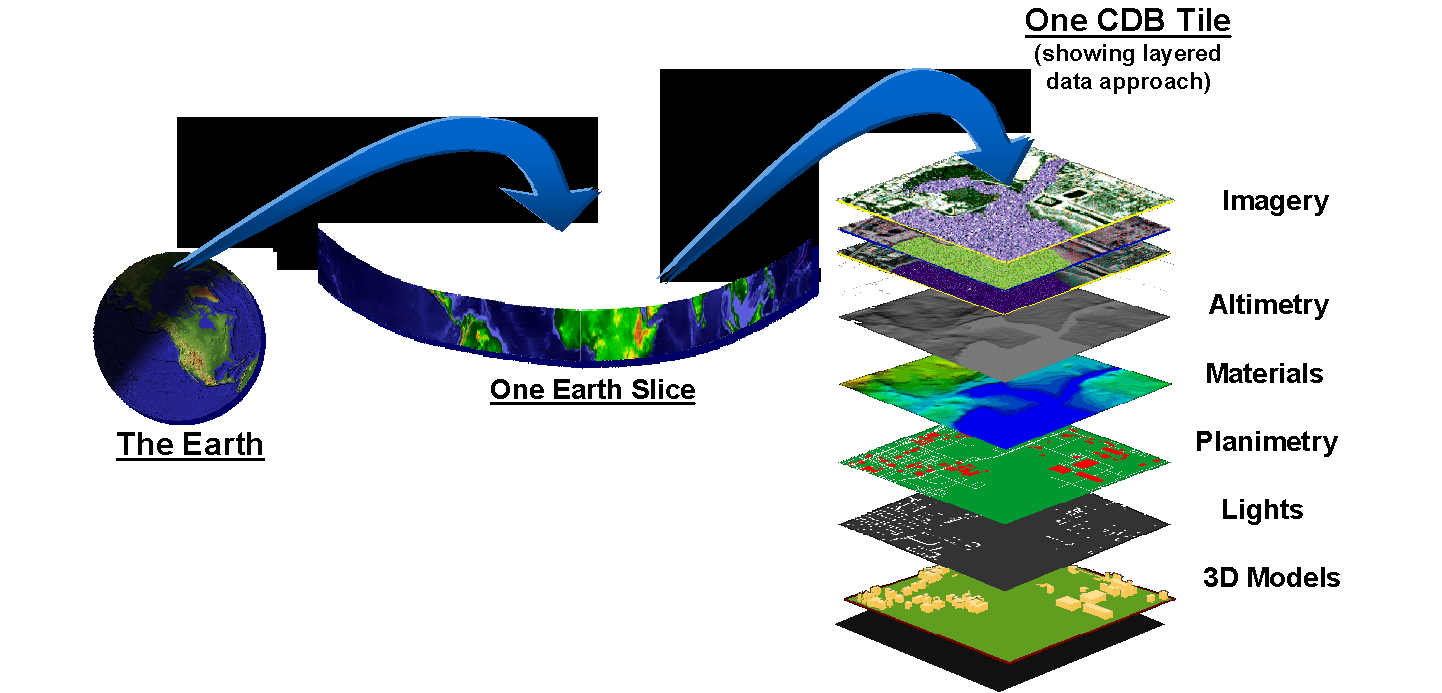
### CDB Standard Logical Structure

The CDB standard is a structure for simulation and as such its storage structure is optimized for simulator client-devices runtime performance. Therefore, the internal storage structure is designed with these specific considerations in mind:

1. Promotes efficient real-time data access by the simulator client-devices without degrading their performance.
2. Allows simultaneous accesses by all of the various simulator client-devices.
3. Promotes efficient data store updates and deployment in order to reduce the deployment of a CDB structured data store onto one or more simulators.

To address these objectives, the storage structure geographically divides the world into tiles (bounded by latitudes and longitudes), each containing a specific set of features (such as terrain altimetry, vectors), models (such as OpenFlight models and RCS models), which are in turn are represented by the datasets (see Figure 6: *CDB Standard Tile/Layer Structure*). The datasets define the basic storage unit used in a CDB structured data store. The geographic granularity is at the tile level while the information granularity is at the dataset level. As a result, the storage structure permits flexible and efficient updates due to the different levels of granularity with which the information can be stored or retrieved. At the data store generation facility, it is possible to effect small updates, either at the tile level; the feature set level or the dataset level. Similarly, the storage structure fully supports real-time retrieval of content at the tile level and the dataset level. Finally, the incremental versioning mechanism allows users to efficiently deploy updates to a data store; only the files whose content differs from a prior version need be generated and transferred from the data store generation facility to the simulation facility.

Another benefit of the CDB standard storage structure is its ability to support concurrent generation and deployment.



**Figure 6: CDB Standard Tile/Layer Structure**

### Tile/Layer/Level-of-Detail Structure

The CDB standard offers a structure that is well suited for the efficient access of its contents. To this end, it relies on three important means to organize the synthetic environment data:

* Tiles
* Layers
* Level-of-Detail (LOD)

#### Tiles

The CDB standard relies on a top-level tile structure designed to organize the data for efficient access in real-time. The tile structure provides an effective means for simulator client-devices to access the datasets of a geographical area at a selected LOD. Since the spatial dimension of tiles varies inversely with LOD (i.e., resolution), client-devices can readily predict the amount of data contained within the tile; as a result, the management of memory within client-devices is simplified and much more deterministic.

The CDB standard Data Representation Model (DRM) is logically organized as a strip of geo-unit aligned tiles along each earth slice. Each earth slice is divided into a decreasing number of tiles to account for the fact that the length of an earth slice decreases with increasing latitude.

#### Layers

The CDB standard DRM is also logically organized as distinct layers of information. The layers are theoretically independent from each other, (i.e., changes in one layer do not impose changes in other layers).

Layers are additive in fidelity, (i.e., the achievable level of the simulation fidelity increases with the number of layers).

Secondly, unavailable layers are automatically defaulted. A database modeler need not fully populate a CDB structured data store. There is no mandatory “coverage completeness requirement” imposed by the CDB standard. This feature permits the generation of a CDB conformant data store even when database modelers are confronted with limited data availability. The usefulness/faithfulness of the synthetic environment increases with the number of available layers.

The layering mechanism improves the efficiency of the client-devices since they need only access (and be aware) of the datasets that are relevant to them, at their prescribed level of fidelity. For instance, a simulator CGF client-device will not likely have a reason to use ground raster imagery. However, it is quite likely interested in accessing ground surface properties when determining traffic-ability.

#### Levels of Detail (LOD)

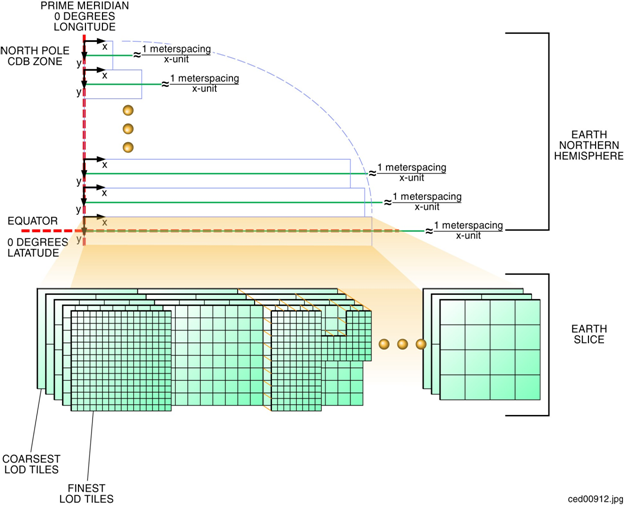
Level of detail (LoD) is a concept available in various disciplines from computer graphics and cartography to electrical circuit design. For GIS practitioners, the discipline where level of detail is most relevant and well known is 3D city modelling. In the CDB, LOD is a mechanism for allowing increasing levels of fidelity without impacting system performance. In CDB data stores, LODs are also an integral part of the physical database structure. The LOD requirement for simulation systems is similar with regard to requirements for increased fidelity in 3d City Models and standards such as CityGML.

The availability of LOD representations is critical to real-time performance, especially when dealing with grid-organized data. Most simulation client-devices can readily take advantage of a LOD structure because, in many cases, less detail/information is necessary at increasing distances from the simulated own-ship. As a result, many client-devices can reduce (by 100-fold or more) the required bandwidth to access synthetic environment data in real-time.

Additionally, the availability of LODs can be readily exploited by simulator client-devices to improve algorithmic efficiency. Devices such as Image Generators (IGs) can readily take advantage of an LOD structure because image perspective naturally reduces image detail with distance as a result of the perspective computation inherent to the IG. As a result, much less geometric detail and texture detail need be processed or considered at far range.

The spatial sampling of each successive LOD in a CDB data store follows power of two progressions. This is a prerequisite for the efficient and deterministic management of memory by all client-devices of a typical simulator.

The terrain LOD can be controlled to the tile level. Since the CDB standard specifies a variable LOD hierarchy spanning up to 34 levels, it is possible to control the allocation of LODs by area. The variable LOD hierarchy provides for efficient use of storage because the LOD hierarchy needs to be deepened only in the areas where higher resolution is desired. Figure 7: *Variable Allocation of LOD*, illustrates an earth strip made up of a series of tiles; some portions of the strip have been modeled with 3, 4, or 5 LODs according to the application requirements.



### CDB Structure, Organization on Media and Conventions

This CDB standard describes the data content, representation, as well as the logical organization of the data stored in the data store:

1. The Data store structure comprises all of the data structures used to access data store content (such as file paths, Level-of-Detail (LOD), lists) to speed up access, information layering, versioning and configuration management.
2. Naming Conventions: Used to describe a naming hierarchy for cultural models, moving models, lights and model components.

#### Other Applications of the CDB Standard

While the CDB standard was initially targeted primarily for use in flight simulator applications, it is entirely suited to a broad range of other applications that make use of the same datasets; these include (and are not limited to):

1. Ground warfare simulation
2. Anti-submarine warfare
3. Visualization
4. Modeling and Simulation
5. Urban Planning
6. Natural Resource Management
7. Emergency Management

The implementation of the CDB standard on legacy simulator client-devices usually mandates the use of the runtime publishers. These costs are largely offset by the consolidation (and substantial reduction) of storage and associated infrastructure. In the future, it is anticipated that the runtime publishing computer infrastructure will be largely “absorbed” by higher performance client-devices that will natively process the data store without an intermediate conversion into a legacy internal format.

In applications that are less demanding than flight simulation, the CDB standard can be implemented without resorting to high-performance computer platforms. For instance, the simulator repository mass storage system (shown in Figure 8: *Typical CDB Implementation on a Suite of Simulators*) can be a single disk storage device. In demanding applications, the simulator repository can be implemented as a large-scale Storage Area Network (SAN). Similarly, the Update Manager IUM), the CDB server and each of the client-device runtime publishers can be implemented as software tasks within a single computer platform, or can even be migrated to software tasks running internally within the client-devices. Figure 1-8, *Typical Implementation of CDB Standard on Desktop Computer*, illustrates a desktop trainer consisting of a stealth viewe[[13]](#footnote-13)r and radar simulations, implemented as software applications running on a single computer platform; both applications pull their synthetic environment data from a disk-resident CDB.

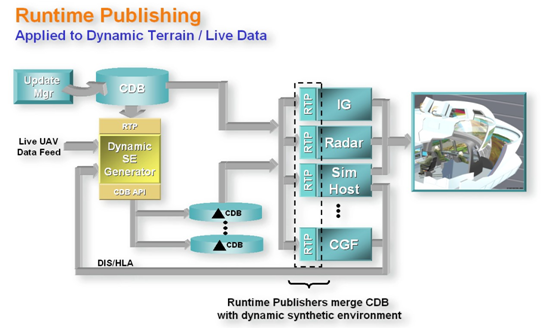
### Use of CDB in Applications Requiring Dynamic Synthetic Environments

A CDB-compliant simulator already incorporates most of the architectural features required to support the handling of Dynamic Synthetic Environments (aka DSE). Figure 1-14, *Versioning Paradigm Applied to Dynamic SE*, illustrates how CDB-compliant simulator architecture can be extended to permit the runtime modification of the CDB. This can be accomplished by leveraging simulator architectures that natively adhere to the CDB data schema and to the CDB versioning capabilities.

This capability requires that the simulator be equipped with a Dynamic Synthetic Environment Generator whose interfaces conform to the CDB standard; all other elements of the simulator architecture are identical to those of a CDB-compliant simulator.

Two applications of this principle could be envisaged:

1. ***DB update from DIS:*** This is the so-called “Dynamic Terrain/Synthetic Environment”, where a group of confederates interoperate over a DIS network by receiving update commands that trigger runtime modification of the CDB.
2. ***DB creation/update from live data feed:*** Terrain areas could be created or modified on-the-fly, based on live data (such as terrain imagery or LIDAR data transmitted by a UAV) and provided instantly to the confederates.



**Figure 8: Versioning Paradigm Applied to Dynamic SE**

### Synthetic Environment Data store Correlation

Please read section 1.6.6 of the OGC CDB Best Practice for additional details on correlation. https://portal.opengeospatial.org/files/?artifact\_id=61935

Annex A Revision history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Release | Author | Paragraph modified | Description |
| 10/20/15 |  | C. Reed | Many | First major “scrub” |
| 11/20/15 |  | C. Reed | Many | Prepare for first publication |
| 2/20/2016 |  | D Graham and C Reed | Many | Prepare for OAB Review |
| 4/3/2016 |  | D Graham |  | Prepare for RFC |

1. From [3D Synthetic Environment Reconstruction](https://books.google.com/books?id=AurSBwAAQBAJ&pg=PA12&lpg=PA12&dq=synthetic+environment+databases&source=bl&ots=BisRqpWBJD&sig=5i49U0wfRJNV7K6YE_0_ky0dM1k&hl=en&sa=X&ved=0ahUKEwjy29zHvJ3JAhUCnIgKHVtyBpwQ6AEIXDAI) [↑](#footnote-ref-1)
2. High Level Architecture/Federation Object Model (IEEE Std 1516™ Series) [↑](#footnote-ref-2)
3. Distributed Interactive Simulation (IEEE Std 1278™ Series) [↑](#footnote-ref-3)
4. Formerly known as Common Database the OGC Members determined that going forward that this standard shall be known as “CDB”. [↑](#footnote-ref-4)
5. Please go to <http://www.opengeospatial.org/projects/groups/dggsswg> for information on an OGC candidate DGGS standard. [↑](#footnote-ref-5)
6. This usage of federate and federation comes from IEEE Std 1516 in the context of Federation Object Models (FOM) and Simulation Object Models (SOM). Common HLA terminology: Federate - an HLA compliant simulation entity. Federation - multiple simulation entities connected via the RTI using a common OMT. [↑](#footnote-ref-6)
7. Time-varying weather simulation effects could be simulated by a “weather server” simulation subsystem which in turn can rely on the terrain elevation and terrain material datasets to perform its simulation of weather in real-time. [↑](#footnote-ref-7)
8. This section is based on presentations and a paper on the JTDS CB-TGS activity. Specifically, “Cloud Terrain Generation and Visualization Using Open Geospatial Standards, Chambers and Freeman, 2014. From proceedings of the Interservice/Industry Training, Simulation, and Education Conference. [↑](#footnote-ref-8)
9. The CDB standard does require however, that most of its dataset be generated in a level-of-detail hierarchy. [↑](#footnote-ref-9)
10. The CDB standard supports this concept through two mechanisms:

    Data defaulting: The CDB standard is tolerant to missing data, because all attributes, layers and datasets have Specification default behaviors. As more of the data is provided, the fidelity of the environmental database increases.

    Additive layering: The CDB standard offers a layered environment database organization. The layer organization is “fidelity-ordered”, (i.e., basic layers appear first in the hierarchy while the layers needed for a higher-fidelity simulation appear later in the hierarchy). [↑](#footnote-ref-10)
11. Assuming the DB toolset (used by the modelers at the DB Generation Facility) allow its users to manipulate Environmental DB content on a small-area basis. [↑](#footnote-ref-11)
12. One of the primary objectives of this standard is to provide and integrate all of the data required by all sensor devices, not just IGs producing the Out-The-Window (OTW) scenes. In addition, the mandate of this standard is to accomplish this objective in a device-independent fashion. The CDB standard provides a means for client-devices to retain their internal Sensor Synthetic Environment Model (SEM), and yet do so without introducing device dependencies within the CDB synthetic environment. To accomplish this objective, client-device vendors must provide the appropriate SEM properties for the prescribed CDB Base Materials (see Appendix L), since none of the (vendor/device-specific) material properties are stored within the CDB [↑](#footnote-ref-12)
13. A ‘stealth viewer’ is analogous to a camera viewpoint that is not ‘attached’ to any physical entity in a simulation [↑](#footnote-ref-13)