Table of Contents

Open Geospatial Consortium

Submission Date: <yyyy-mm-dd>

Approval Date: <yyyy-mm-dd>

Publication Date: <yyyy-mm-dd>

External identifier of this OGC® document: http://www.opengis.net/doc/{doc-type}/{standard}/{m.n}

Internal reference number of this OGC® document: 19-066

Version: n.n

Category: OGC® Best Practice

Editor: Michala Hill

OGC Best Practices for Conversion of CDB Vector Data to Geopackage

Warning

This document defines an OGC Best Practice on a particular technology or approach related to an OGC standard. This document is not an OGC Standard and may not be referred to as an OGC Standard. It is subject to change without notice. However, this document is an official position of the OGC membership on this particular technology topic.

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:	
Document stage:	Draft
Document language:	English

License Agreement

i. Abstract

This best practices document describes the conversion process for converting CDB structured Shapefile to CDB structued GeoPackages. This is the companion document to the optional OGC CDB extension document, which defines the requirements and provides CDB specific guidance on using GeoPackage containers in a CDB data store.

ii. Keywords

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

OGC, ogcdoc, OGC document, CDB, GeoPackage

iii. Preface

Background for this optional CDB Extension

The original requirement for the optional CDB extension was documented in OGC Change Request 545. This OGC change request was submitted based on work performed in OGC Testbed 13. The testbed activity and related change request captured a broad community requirement for being able to use GeoPackage containers in a CDB data store. At the same time, an additional requirement was identified to test and identify best practices for moving CDB vector files stored as Shapefiles into one or more GeoPackages.

In 2019, the CDB SWG executed the CDB Vector Data in GeoPackage Interoperability Experiment (IE). The participants in this IE tested transforming CDB Shapefile vector data into one or more GeoPackage(s) and storing the result in a CDB data store. GeoPackage Version 1.2 and CDB Version 1.1 and related Best Practices were the standards baseline used for this experiment. The IE built on the work described in the OGC CDB, Leveraging the GeoPackage Discussion Paper. A primary objective of the IE was to agree and document possible change requests and/or best practices for storing vector data in a CDB data using encodings and/or containers other than Shapefiles.

OGC Patent Declaration

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):

Cognitics, Inc.

v. Submitters

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

Name	Affiliation
Michala Hill *(editor)*	Cognitics, Inc.
Kevin Bentley	Cognitics, Inc.
Ryan Franz	FlightSafety Visual Systems
Bernard Leclerc	CAE, Inc.
Carl Reed	Carl Reed & Associates
David Graham	CAE Inc.

Chapter 1. Scope

NOTE

This Best Practice compliments the optional OGC CDB Extension that defines the behavior and requirements for encoding vector data in a GeoPackage container for use in a CDB data store. Both are grounded in the existing CDB Core requirements and the GeoPackage core requirements for vector data.

This document discusses the results and conclusions of the Interoperability Experiment conducted to determine the optimal use of GeoPackage in CDB, assuming relatively minor changes to the CDB current standard. It also offers general guidance on the use of GeoPackage in CDB, with details on the flattened attribute schema.

Before reading this standard, please remember that the idea is to restrict the encoding of a dataset to a single vector format per CDB Version. Since a “CDB” is made of one or more “Versions” (as specified by Configuration.xml or Version.xml), and that each CDB Version can have a different encoding for a given dataset, the result is that a “CDB” may pretty well exist with multiple encodings for the same dataset. This means that if you wish to use GeoPackage containers, you need to create a version that will just contain GeoPackages of the vector data and not include any Shapefiles.

Chapter 2. Conformance

This best practice defines [TBD] requirements / conformance classes.

The standardization targets of all conformance classes are "GeoPackages in CDB Data Store". The main requirements class is:

Core. [Add link to Annex A later]

The Core specifies requirements that all GeoPackages that are to be stored in a CDB store that shall be implemented.

Conformance with the standard and best practice shall be checked using all the relevant tests specified in Annex A (normative) of this document. The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing web site.

In order to conform to this OGC® interface best practice, a software implementation shall choose to implement the conformance levels specified in Annex A (normative)

All requirements-classes and conformance-classes described in this document are owned by the standard(s) identified.

Chapter 3. References

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

ISO / TC 211: ISO 19115-1:2014 Geographic information — Metadata — Part 1: Fundamentals (2014)

OGC: OGC 15-113r5, Volume 1: OGC CDB Core Standard: Model and Physical Data Store Structure (2017)

OGC: OGC 16-070r3, Volume 4: OGC CDB Best Practice use of Shapefiles for Vector Data Storage (2017)

OGC: OGC 12-128r15, OGC GeoPackage Encoding Standard - with Corrigendum (2018)

OGC: OGC ???, OGC Vector GeoPackage Extension (TBD)

Chapter 4. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OGC 06-121r8], which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this Best Practice.

For the purposes of this document, the following additional terms and definitions apply.

4.1. coordinate reference system

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

4.2. coordinate system

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

4.3. dataset

collection of data, published or curated by a single agent, and available for access or download in one or more formats [DCAT]

Note: The use of 'collection' in the definition from [DCAT] is broader than the use of the term collection in this standard. See the definition of feature collection.

4.4. feature

abstraction of real world phenomena [ISO 19101-1:2014]

Note: If you are unfamiliar with the term 'feature', the explanations on Spatial Things, Features and Geometry in the W3C/OGC Spatial Data on the Web Best Practice document provide more detail.

4.5. feature collection; collection

a set of features from a dataset

Note: In this standard, 'collection' is used as a synonym for 'feature collection'. This is done to make this document easier to understand for those that are not geo-experts.

4.6. height

Distance of a point from a chosen reference surface measured upward along a line perpendicular to that surface. [ISO 19111]

Note 1 to entry: A height below the reference surface will have a negative value, which would embrace both gravity-related heights and ellipsoidal heights.

4.7. linestring

curve composed of straight-line segments (aka line) [ISO 19136:2007]

4.8. point

0-dimensional geometric primitive, representing a position [ISO 19136:2007]

Note to entry: The boundary of a point is the empty set.

4.9. polygon

planar surface defined by 1 exterior boundary and 0 or more interior boundaries [ISO 19136:2007]

4.10. GeoPackage file

a platform-independent SQLite database file that contains GeoPackage data and metadata tables with specified definitions, integrity assertions, format limitations and content constraints.

4.11. tile

a geometric shape with known properties that is the result of the tiling (tessellation) of a plane. A tile consists of a single connected "piece" without "holes" or "lines" (topological disc).

4.12. Valid GeoPackage

A GeoPackage that contains features per clause Features and/or tiles per clause Tiles and row(s) in the gpkg_contents table with data_type column values of "features" and/or "tiles" describing the user data tables.

4.13. vector and vector data

quantity having direction as well as magnitude

Note to entry: A directed line segment represents a vector if the length and direction of the line segment are equal to the magnitude and direction of the vector. The term vector data refers to data that represents the spatial configuration of features as a set of directed line segments. [ISO 19123:2005]

4.14. vector geometry

representation of geometry through the use of constructive geometric primitives [ISO 19107:2003]

4.15. Version

A collection of pure CDB Datasets and/or user-defined datasets. Please see section 3.2 of Volume 1: OGC CDB Core Standard: Model and Physical Data Store Structure for details on the CDB versioning strategy and structure.

Chapter 5. Conventions

This sections provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, use of XML schema, or special notes regarding how to read the document.

5.1. Identifiers

The normative provisions in this standard are denoted by the URI http://www.opengis.net/spec/cdb-vector-geopackage/1.2

All requirements and conformance tests that appear in this document are denoted by partial URIs which are relative to this base.

Chapter 6. Results from Interoperability Experiments

OGC CDB and GeoPackage are both standards increasingly used in M&S and GEOINT, but they both contain weaknesses and strengths when it comes to the combined needs of both industries. The combined technology of GeoPackage and OGC CDB is hoped to be a deterministic repository of easily read geospatial datasets suitable for storage, runtime access, and dissemination for live, virtual, constructive, gaming, and Mission Command (MC) systems. The use of CDB raised concerns in both industries because it utilizes Esri Shapefiles and Presagis OpenFlight, both of which are now open standards but were once proprietary. The primary issue is that Shapefiles leverage the dBase IV format to store feature attribution. To address the limitations of ESRI Shapefiles within CDB, potential designs were explored to integrate OGC GeoPackage with OGC CDB. GeoPackage technology was designed to ensure Mobile / Handheld devices could interoperate with systems within the larger Command Post and U.S. Army Enterprise. Current Geopackage adaptations can natively store 3D information, routing information, various types of data 'tiles', and other geographically related datasets. An implementation of OGC GeoPackage within CDB aligns CDB vector data storage with the U.S. Army architecture.

Four approaches were testing in the Interoperability Experiment:

- Experiment 1: Conversion of Shapefiles in one or more CDB data stores into GeoPackages as required for Experiments 2, 3, and/or 4.
- Experiment 2: One-to-one conversion of a Shapefile into a GeoPackage.
- Experiment 3: Store each CDB Level of Detail (LOD) as a layer in GeoPackage.
- Experiment 4: Store each Geocell of Vector Data as a layer in GeoPackage.

Experiment 2 was further broken down into Options 1a – 1d. Each experiment offered its own set of potential advantages, disadvantages, and lessons learned.

Utilizing Option 1c:

- Fully flattened instance, class, and extended level attribute tables
- There is a 4:1 to 7:1 reduction in the number of files.
- Some duplication of data, resulting in larger files.
- There is one layer per GeoPackage.
- The Feature Class and Extended Attributes are populated for each feature.
- Full "Off the Shelf" GeoPackage software compatibility.
- Best Experiment 2 option in terms of file size and performance time.

Utilizing Option 1d:

- Flatten CDB standard instance and class attribute – maximum GIS tools compatibility
- "Off the Shelf" GeoPackage software compatibility for CDB standard attributes.
- Table for extended attributes using Related Tables GeoPackage extension

- This approach utilizes a standard normalized relational database design, utilizing foreign keys.
- Some duplication of data, resulting in larger files.
- There is one layer per GeoPackage.
- The Feature Class and Extended Attributes are populated for each feature.

The final method selected for conversion of a Shapefile CDB to a GeoPackage CDB is a hybrid of Options 1c and 1d. Because “Off the Shelf” GeoPackage software will not be aware of the class and extended level attributes, we needed to “flatten” the class level attributes with the instance level attributes. It was not possible to effectively flatten the extended level attributes at this time, and those will remain in a separate table with regular table relationships. The decision was made not to use the Related Tables GeoPackage extension.

Several observations were made during the experiments. The output created by the recommended method of conversion allows backward compatibility. However, the GeoPackage CDB results in larger size on disk, which could be problematic. The GeoPackage format (especially as outputted by GDAL) is very space inefficient when it comes to encoding tiles with small quantities of data. But it does reduce the total number of files on disk significantly and eliminates the occurrence of dBase files containing 0 KB.

For more details on the experiments and information above, please see the OGC CDB Vector Data in GeoPackage Interoperability Experiment Engineering Report.

Chapter 7. Guidance on the use of GeoPackages

An GeoPackage is an open, standards-based, platform-independent, portable, self-describing, compact format for transferring geospatial information. It is a platform-independent SQLite database file. The GeoPackage specification was developed by OGC. This document describes the encoding of Vector Data in GeoPackage only.

Per Requirement 4 of the CDB Vector GeoPackage Extension, the position of all points is expressed using WGS-84 geographic coordinates (latitude, longitude, altitude), as explained in Volume 8: CDB Spatial Reference Systems Guidance.

As detailed in the following table of GeoPackage Geometry Codes(Table 30 of Appendix G of the OGC GeoPackage Specification), the following geometry types are supported in GeoPackage:

Code	Name
0	GEOMETRY
1	POINT
2	LINESTRING
3	POLYGON
4	MULTIPOIPNT
5	MULTILINESTRING
6	MULTIPOLYGON
7	GEOMETRYCOLLECTION

Per Requirement 2 of the CDB Vector GeoPackage Extension, while the GeoPackage model supports encoding of 8 different types that can be stored in the same GeoPackage, the CDB standard requires a maximum of 3 feature geometry types per tile.

Chapter 8. Flattening the Schema

The CDB Shapefile standard provides three attribution schemas to represent attribution data. Those three have been flattened into two for use with GeoPackage.

- Instance-level attribution schema
- Extended-level attribution schema

As CNAM in the CDB Shapefile schema was only used as a relationship field, it has been removed from the flattened schema. Figure 1 shows the flattened instance and class-level attributes, and how the extended attributes link to a feature. I_CEAI, I_GEAI, and I_VEAI are all instance-level attributes and are linked to a specific feature. C_CEAI, C_GEAI, and C_VEAI are all class-level attributes and are linked to all features in a layer.

Combined Instance and Class-Level Attributes - Linear Features - Highway																	
ID	GEOM	AO1	LNEL	LPN	RTAI	SCALx	SCALy	SCALz	I_CEAI	I_GEAI	I_VEAI	BSR	FACC	FSC	C_CEAI	C_GEAI	C_VEAI
1	Linestring	-	82 500m	1	71%	-	-	-	4	-	-	-	AP030	002	-	-	9
2	Linestring	-	33 565m	1	99%	-	-	-	-	-	-	-	AP030	002	-	-	9
3	Linestring	-	53 565m	1	99%	-	-	-	-	-	-	-	AP030	002	-	-	9

Extended-Level Attributes - Linear				
ID	LNK	GRP	EAC	EAV
4	5	CDB	5	CYUL
5	8	CDB	54	06L
8	0	CDB	25	B23
9	10	ABC Inc	1	1234567890ABCDEF
10	0	ABC Inc	1	GHIJ

Combined Instance and Class-Level Attributes - Linear Features - Railroad																	
ID	GEOM	AO1	LNEL	LPN	RTAI	SCALx	SCALy	SCALz	I_CEAI	I_GEAI	I_VEAI	BSR	FACC	FSC	C_CEAI	C_GEAI	C_VEAI
1	Linestring	-	154 000m	1	85%	-	-	-	-	-	-	-	AN010	000	-	1	-
2	Linestring	-	206 000m	1	85%	-	-	-	-	-	-	-	AN010	000	-		-

Extended-Level Attributes - Linear				
ID	LNK	GRP	EAC	EAV
1	2	DIGEST	1	1
2	3	DIGEST	2	6.45
3	0	DIGEST	3	XYZ...

Combined Instance and Class-Level Attributes - Polygon Features - House																	
ID	GEOM	AO1	LNEL	LPN	RTAI	SCALx	SCALy	SCALz	I_CEAI	I_GEAI	I_VEAI	BSR	FACC	FSC	C_CEAI	C_GEAI	C_VEAI
1	Polygon	5.2"	-	-	89%	1	1	1	-	-	-	-	AL015	016	-	-	-
2	Polygon	35.0"	-	-	89%	1	1	1	-	-	-	-	AL015	016	6	-	-

Extended-Level Attributes - Polygon				
ID	LNK	GRP	EAC	EAV
6	7	CDB	29	1
7	0	CDB	60	84

Figure1. CDB GeoPackage Flattened Schema

Example: Record ID 4 in the extended-level attributes table is related to record ID 1 in the linear features table and is an instance-level attribute. Record IDs 5 and 8 are also linked to record ID 4, and all apply to the same linear feature. Note that a 0 in the LNK column denotes the end of the applicable record list. Record 9 in the extended-level attributes table is a class-level attribute related to all features that are present in the highway layer.

All of the information that is needed to instance features is organized in accordance to the CDB tile structure. All the tiled GeoPackage dataset files are located in the same directory. The dataset's second component selector (CS2) is used to differentiate between files with the same extension or with the same Vector features. Table 1 presents the list of codes that are allocated. Note that Vector datasets do not necessarily use all of the Dataset Component Selector 2 reserved codes. Users of the CDB standard should refer to the appropriate section for an enumeration of the supported File Component Selector 2 codes as well as the ones specific to the Dataset.

The Vector dataset concept and the feature code concepts overlap somewhat; some of the Vector datasets are generalizations or specializations of feature codes. Note that the same feature should not have two representations.

Table 1: Component Selector 2 for Vector Dataset File Names

CS2	File Extension	Dataset Component Name	Supported Shape Type
001	*.gpkg	Point features	Point, PointZ, PointM, MultiPoint, MultiPointZ, MultiPointM
003	*.gpkg	Lineal features	PolyLine, PolyLineZ, PolyLineM
005	*.gpkg	Polygon features	Polygon, PolygonZ, PolygonM, Multipatch
007	*.gpkg	Lineal figure point features	Point, PointZ, PointM, MultiPoint, MultiPointZ, MultiPointM
009	*.gpkg	Polygon figure point features	Point, PointZ, PointM, MultiPoint, MultiPointZ, MultiPointM
011	*.gpkg	2D relationship tile connections	N/A
015	*.gpkg	2D relationship dataset connections	N/A

Table 2: Component Selector 2 for Vector Dataset Table Names

CS2	Dataset Component Name	Supported Shape Type
001	Point features	Point, PointZ, PointM, MultiPoint, MultiPointZ, MultiPointM
002	Point feature class-level attributes	N/A
003	Lineal features	PolyLine, PolyLineZ, PolyLineM
004	Lineal feature class-level attributes	N/A

CS2	Dataset Component Name	Supported Shape Type
005	Polygon features	Polygon, PolygonZ, PolygonM, Multipatch
006	Polygon feature class-level attributes	N/A
007	Lineal figure point features	Point, PointZ, PointM, MultiPoint, MultiPointZ, MultiPointM
008	Lineal figure point feature class-level attributes	N/A
009	Polygon figure point features	Point, PointZ, PointM, MultiPoint, MultiPointZ, MultiPointM
010	Polygon figure point feature class-level attributes	N/A
011	2D relationship tile connections	N/A
012	Deprecated	N/A
013	Deprecated	N/A
014	Deprecated	N/A
015	2D relationship dataset connections	N/A
016	Point feature extended-level attributes	N/A
017	Lineal feature extended-level attributes	N/A
018	Polygon feature extended-level attributes	N/A
019	Lineal Figure Point extended-level attributes	N/A
020	Polygon Figure Point extended-level attributes	N/A

Chapter 9. Layer Names within a GeoPackage

Each GeoPackage file shall contain one and only one feature table. The feature table shall include all attributes (fields) required by the CDB standard (Volume 1, 5.7.1.2. CDB Attribution). The single feature table shall be named identically to the GeoPackage filename (not including the ".gpkg" filename extension). A GeoPackage file may include an optional extended level attribute table. The extended level attribute table shall not contain geometry. The name of the extended level attribute table shall use the CS2 selector shown in Table 2. Table 3 shows the file naming convention.

Table 3: Tiled Dataset File Naming Convention 1

Field	Description
Lat	Geocell Latitude – Identical to the name of the directory defined in section 3.6.2.1, Directory Level 1 (Latitude Directory).
Lon	Geocell Longitude – Identical to the name of the directory defined in section 3.6.2.2, Directory Level 2 (Longitude Directory).
Dnnn	Character D followed by the 3-digit code assigned to the dataset.
Snnn	Character S followed by the 3-digit value of Component Selector 1 (CS1).
Tnnn	Character T followed by the 3-digit value of Component Selector 2 (CS2).
LOD	Level of Detail – As defined in section 3.3.8.5, Level of Detail.
Un	UREF – Identical to the name of the directory as defined in section 3.6.2.5, Directory Level 5 (UREF Directory).
Rn	RREF – A reference to the Right Index of a tile. Character R (Right direction) followed by the column number as described in this section.
xxx	File extension as per file type.

The example below and Table 4 show the filename structure for a road feature. GeoPackage Filename: N13E045_D201_S002_T003_L00_U0_R0.gpkg
Feature Table Name: N13E045_D201_S002_T003_L00_U0_R0
Extended Attribute Table Name: N13E045_D201_S002_T017_L00_U0_R0

Table 4: Basic Filename Structure

N13	E045	D201	S002	T003	L00	U0	R0
Latitude	Longitude	Dataset Code	CS1	CS2	LOD	UREF	RREF

Figure 2 shows the flattened schema with filenames as they might appear for a road dataset. The feature and extended attribute tables are shown contained within the GeoPackage file.

N13E045_D201_S002_T003_L00_U0_R0.gpkg

N13E045_D201_S002_T003_L00_U0_R0																	
ID	GEOM	AO1	LNEL	LPN	RTAI	SCALx	SCALy	SCALz	I_CEAI	I_GEAI	I_VEAI	BSR	FACC	FSC	C_CEAI	C_GEAI	C_VEAI
1	Linestring	-	82 500m	1	71%	-	-	-	4	-	-	-	AP030	002	-	-	9
2	Linestring	-	33 565m	1	99%	-	-	-	-	-	-	-	AP030	002	-	-	9
4	Linestring	-	53 565m	1	99%	-	-	-	-	-	-	-	AP030	002	-	-	9

N13E045_D201_S002_T017_L00_U0_R0				
ID	LNK	GRP	EAC	EAV
4	5	CDB	5	CYUL
5	8	CDB	54	06L
8	0	CDB	25	B23
9	10	ABC Inc	1	1234567890ABCDEF
10	0	ABC Inc	1	GHIJ

Figure2. CDB GeoPackage Flattened Schema Road Dataset Example

See Annex A: Sample Conversion Tools for information on the use of sample conversion tools found in the Cognitics GitHub repo.

Annex A: Conformance Class Abstract Test Suite (Normative)

NOTE Ensure that there is a conformance class for each requirements class and a test for each requirement (identified by requirement name and number)

A.1. Conformance Class A

A.1.1. Requirement 1

Test id:	/conf/conf-class-a/req-name-1
Requirement:	/req/req-class-a/req-name-1
Test purpose:	Verify that…
Test method:	Inspect…

A.1.2. Requirement 2

Annex B: Sample Conversion Tools (Informative)

Sample conversion tools may be found in the Cognitics GitHub repo located at https://github.com/Cognitics/cdb-shp-geopackage-convert.

Warning: Use the sample conversion code at your own risk. It is intended to provide an example on how to convert Shapefiles to GeoPackage in a CDB, and is not an OGC tested application. Make sure you test on a backup copy of your CDB first.

The sample converter is written in Python, and uses GDAL (version 2.2.3 or greater) with the Python bindings. Binaries of GDAL can be found at https://trac.osgeo.org/gdal/wiki/DownloadingGdalBinaries.

To use the sample converter tool, use the following Syntax:

```
Usage: Convert.py <Input Root CDB Directory> <Output Directory for GeoPackage Files>
```

The converter starts by calling the function:

```
generateMetaFiles.generateMetaFiles(cDBRoot)
```

This function scans the CDB to find a list of Shapefiles. It stores a list of all Shapefiles in a file called shapeindex.py. Another file named shapemeta.py is generated with a list of shapefiles and the extents of each file.

The shapemeta.py file is not necessary for the conversion, but can be useful for analysis. That block of code can be removed for a faster conversion if it isn't needed.

Next, the index file is scanned, and each ShapeFile is processed through:

```
copyFeaturesFromShapeToGeoPackage(shpFilename,outputGeoPackageFile)
```

If the shapefile passed through this function is a feature file, and not a feature class or extended attribute file, the appropriate feature class and extended attributes DBF file (based on Selector2 in the filename) is ready by calling readDBF. All records in each file are read and stored in a dictionary.

Next, GDAL is used to create the appropriate GeoPackage file and layers. Then GDAL is used to open the ShapeFile and iterate through each feature. To flatten the feature class records, as each feature is read, any matching feature class records (based on the CNAM attribute) are added to the feature record.

After copyFeaturesFromShapeToGeoPackage has completed, the extended attribute DBF file that matches the feature file is read. The appropriate layer is created in the GeoPackage. The name of that layer is identical to the filename of the extended attribute DBF file, without the .dbf extension. Each record in the extended attribute DBF file is read and added to the new layer in the GeoPackage. Each record from the DFB file is copied into the GeoPackage layer.

Note that no additional relationships or foreign keys are necessary. The feature attributes that are used to link the features to the Extended Attributes in the GeoPackage are the same as the attributes from the ShapeFile (CEAI,GEAI, and VEAI) and have been renamed to indicate either the Instance or Class level (see Figure 1 in the Flattening the Schema of the Normative document).

After each feature ShapeFile has been processed, the conversion is complete. Note that none of the associated ShapeFile or DBF files are removed from the CDB.

Annex C: Revision History

Date	Release	Editor	Primary clauses modified	Description
2016-04-28	0.1	G. Editor	all	initial version

Annex D: Bibliography

NOTE

Example Bibliography (Delete this note).

The TC has approved Springer LNCS as the official document citation type.

Springer LNCS is widely used in technical and computer science journals and other publications

- For citations in the text please use square brackets and consecutive numbers: [1], [2], [3]

– Actual References:

[n] Journal: Author Surname, A.: Title. Publication Title. Volume number, Issue number, Pages Used (Year Published)

[n] Web: Author Surname, A.: Title, http://Website-Url

[1] OGC: OGC Testbed 12 Annex B: Architecture. (2015).