OGC® DOCUMENT: 24-010

External identifier of this OGC® document: http://www.opengis.net/doc/ER/RBT-GPKG-Sprint/1.0

Open Geospatial Consortium

RELEASABLE BASEMAP TILES IN GEOPACKAGE ENGINEERING REPORT

ENGINEERING REPORT

PUBLISHED

Version:
Submission Date: 2024-05-30
Approval Date: 2024-06-21
Publication Date: 2025-MM-DD
Editor: Jérôme Jacovella-St-Louis

Note

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.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF RECOMMENDATIONS

I ABSTRACT

This Engineering Report describes an extension for storing Releasable Basemap Tiles (RBT) in GeoPackage developed during an OGC Code Sprint initiative sponsored by the US Army Geospatial Center (AGC) which took place in March 2024. During this initiative, participants Ecere Corporation, Compusult Limited and TechMaven Geospatial, assembled into this single extension document draft extensions developed in previous OGC initiatives for storing vector tiles, semantic annotations and portrayal styles in GeoPackages. Minor updates and clarifications to the requirements of these draft extensions were made based on additional experimentation performed during the sprint. Additional requirements and recommendations were also introduced for this extension, including requirements specific to RBT content, a recommendation to include fonts used by styles, requirements to support compressed vector tiles, as well as requirements further restricting how to store vector tiles in the context of this extension to improve interoperability.

II KEYWORDS

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

ogcdoc, OGC document, GeoPackage, Vector Tiles, RBT, Basemap

III PREFACE

This Engineering Report describes an extension for storing Releasable Basemap Tiles (RBT) in GeoPackage developed during an OGC Code Sprint initiative sponsored by the US Army Geospatial Center (AGC) which took place in March 2024. The participants in this initiative included Ecere Corporation, Compusult Limited and TechMaven Geospatial.

This extension was written with the intent to be adapted into a National System for Geospatial Intelligence (NSG) standard.

This RBT GeoPackage extension was assembled from multiple draft GeoPackage extensions developed in previous OGC initiatives including the Vector Tile Pilots (Phase 1+extension and Phase2), Testbed 15 and Testbed 16, which at the time of the initiative, had not yet been adopted as OGC standards. These extensions had been developed primarily by Image Matters LLC, in collaboration with the other participants of these initiatives, which performed several Technology Integration Experiments to validate interoperability across implementations from different technology vendors. In addition to a significant re-organization of the content, minor changes and clarifications were made to the technical details (such as some of the SQLite tables) sourced from these extensions as part of this initiative.

Since these different source draft specifications may still eventually evolve into multiple approved OGC standards and conformance classes, this specification maintains a separation into *conceptual* classes of requirements so as to facilitate maintaining the alignment of these individual extensions with this candidate NSG standard during further OGC standardization efforts.

For the purpose of improving interoperability, this extension defines additional requirements, beyond what would be specified in these more general and flexible OGC GeoPackage extensions, which further restrict the nature and encoding of content stored in an RBT GeoPackage.

From the perspective of this RBT extension, the entirety of this document describes a single GeoPackage extension and requirements class, consisting of mandatory requirements and optional recommendations.

An annex of this report includes details of the implementation by participants of both producers and consumers (viewers) for RBT content stored in GeoPackages using this extension. The annex also presents challenges and lessons learned during the initiative. Participants produced GeoPackages from RBT content and portrayal information provided by AGC as Mapbox Vector Tiles and TileJSON stored in MBTiles, as well as Mapbox / MapLibre styles. Participants also performed Technology Integration Experiments testing consumption of RBT GeoPackages produced by the other implementations. An additional annex defines an Abstract Test Suite for the extension, and another annex provides suggestions for future related work.

IV SUBMITTING ORGANIZATIONS

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

- US Army Geospatial Center (AGC)
- Ecere Corporation
- Compusult Limited
- TechMaven Geospatial

V CONTRIBUTORS

The following individuals contributed to this draft specification Engineering Report:

Name	Affiliation	OGC member
Jeff Harrison (sponsor)	U.S. Army Geospatial Center	Yes
Jérôme Jacovella-St-Louis (editor)	Ecere Corporation	Yes
Adam Parsons	Compusult Limited	Yes
Jordan Bess	Tech Maven Geospatial	Yes

In addition, the specification defined in this Engineering Report builds upon the work of previous previous initiatives such as the Vector Tile Pilots to which others contributed, in particular Jeff Yutzler, Image Matters LLC.

VI SECURITY CONSIDERATIONS

No security considerations have been made for this Standard.

1

SCOPE

1 SCOPE

This Engineering Report describes an extension for storing Releasable Basemap Tiles (RBT) in GeoPackage developed during an OGC Code Sprint initiative.

This extension regroups requirements specific to storing RBT data products in GeoPackages, requirements for storing vector tiles in GeoPackages, requirements for defining semantic annotations in GeoPackages, as well as requirements to include portrayal information in GeoPackages.

In addition to the extension itself, this Engineering Report includes details of three implementations developed during the code sprint, including Technology Integration Experiments demonstrating interoperability between them.

2

CONFORMANCE

2 CONFORMANCE

This extension defines of a single mandatory conformance class, but is broken down into conceptual requirement classes which may eventually correspond to individual standardized OGC GeoPackage extensions.

The requirements apply to either of these two standardization target types:

- data products, and
- software producing such data products.

Conformance with this extension 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 NSG Compliance Testing Policies and Procedures and the NSG Compliance Testing web site.

In order to conform to this candidate NSG data product standard, a software implementation producing a compliant data product shall implement the core requirement classes defined in this extension.

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

3

NORMATIVE REFERENCES

3 NORMATIVE REFERENCES

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

Jeff Yutzler: OGC 12-128r19, *OGC® GeoPackage Encoding Standard*. Open Geospatial Consortium (2024). http://www.opengis.net/doc/IS/geopackage/1.4.0.

Joan Masó, Jérôme Jacovella-St-Louis: OGC 17-083r4, *OGC Two Dimensional Tile Matrix Set and Tile Set Metadata*. Open Geospatial Consortium (2022). http://www.opengis.net/doc/IS/tms/2.0.0.

4

TERMS AND DEFINITIONS

4 TERMS AND DEFINITIONS

This document uses the terms defined in OGC Policy Directive 49, 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 document and OGC documents do not use the equivalent phrases in the ISO/IEC Directives, Part 2.

This document also uses terms defined in the OGC Standard for Modular specifications (OGC 08-131r3), also known as the 'ModSpec'. The definitions of terms such as standard, specification, requirement, and conformance test are provided in the ModSpec.

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

4.1. 2-Dimensional Tile Matrix Set (2DTMS)

tiling scheme consisting of a set of tile matrices defined at different scales covering approximately the same area and having a common coordinate reference system.

[**SOURCE:** OGC 17-083r4]

4.2. geodataclass

a URI identifying a class of geospatial data whose component data layers conform to a particular logical schema

A registry of geodataclasses would ideally resolve these URIs to metadata including these schemas. A geodataclass serves multiple purposes, including the ability to easily identify, using a simple identifier comparison, relevant datasets for a particular purpose, the compatibility of datasets as inputs for processes, the compatibility of the output of a process with the input of another process, as well as the compatibility of portrayal style information with a particular dataset (see also https://github.com/opengeospatial/styles-and-symbology/issues/12).

4.3. Mapbox vector tiles (MVT)

a specification developed by Mapbox for encoding tiled vector data using protocol buffers.

4.4. releasable basemap tiles (RBT)

tilesets of foundational geospatial data developed by the US Army Geospatial Center for public release, including vector tilesets of cultural and physical features, hillshaded elevation data and optional imagery

4.5. technology integration experiment (TIE)

experiments performed between different implementations of a specification or standard to test and demonstrate interoperability between them, and in the case of TIEs performed in the context of an initiative to validate a draft specification or standard, to validate the soundness and completeness of the requirements specified therein

5

OVERVIEW

5 OVERVIEW

This Engineering Report describes an extension for storing RBT in GeoPackage developed during the March 2024 OGC Code Sprint initiative sponsored by the US Army Geospatial Center (AGC).

The report is organized into four main sections:

- **Section 6 — Releasable Basemap Tiles GeoPackage Extension** describes the RBT extension as a whole, including requirements for declaring conformance to this extension, as well as requirements specific to storing RBT data products;
- **Section 7 — Vector Tiles** describes requirements related to storing vector tiles in GeoPackages, based on the draft extensions developed in previous initiatives such as the Vector Tile Pilots (Phase 1+extension and Phase2);
- **Section 8 — Semantic Annotations** describes requirements related to semantic annotations, also based on previously developed draft extensions, providing a mechanism to identify content and associate portrayal rules with content;
- **Section 9 — Styling** describes requirements related to including styles (portrayal information) in GeoPackages, also based on previously developed draft extensions;

and three annexes:

- **Annex A — Abstract Test Suite** defines an Abstract Test Suite for this extension, upon which an Executable Test Suite could be based;
- **Annex B — Implementations** describes three implementations of both producers and consumers (viewers) of GeoPackages using this extension developed during the code sprint by three participants: **Ecere Corporation**, **Compusult Limited** and **TechMaven Geospatial**, including Technology Integration Experiments demonstrating interoperability between these different implementations, as well as **Challenges and Lessons Learned** from these implementation efforts; and
- **Annex C — Future Work** provides recommendations for future work that could be undertaken to build upon the results from the sprint, improve interoperability and performance for RBT content, and move this extension along the standardization process.

6

RELEASABLE BASEMAP TILES GEOPACKAGE EXTENSION

6 RELEASABLE BASEMAP TILES GEOPACKAGE EXTENSION

6.1. Extension definition

6.1.1. Introduction

The Releasable Basemap Tiles GeoPackage extension defines the requirements for GeoPackage distribution of Releasable Basemap Tiles.

The extension defines how to efficiently store tilesets of foundational geospatial data in GeoPackages ("Releasable Basemap Tiles").

This foundational data to be contained in these GeoPackage distribution consists of two vector tilesets for physical and cultural features, and one map tileset of a pre-rendered hillshaded digital elevation model. The extension also defines two optional types of map tilesets that may be included for global imagery and/or for high-resolution imagery of select cities in unified combatant command (COCOM) areas of responsibility.

These tilesets are required to be using the WorldMercatorWGS84Quad 2D Tile Matrix Set, defined in the EPSG:3395 coordinate reference system.

This 2D Tile Matrix Set is **not to be confused** with the *WebMercatorQuad* 2D Tile Matrix Set based on the spherical Web Mercator projection (EPSG:3857), which is ubiquitous in Web mapping applications, but is not fully conformal. In Web Mercator, the angles between lines on the Earth surface are not preserved exactly on the map. Unlike Web Mercator, the world Mercator projection is a fully conformal map projection, preserving directions, angles, and shapes.

This extension, like all GeoPackage extensions, is intended to be transparent and to not interfere with GeoPackage-compliant software packages that do not support the extension. However, these software packages may not recognize the vector data encoded using the new vector tiles data type defined in this extension.

6.1.2. Extension Name

The name of this GeoPackage extension (to be used in the `extension_name` field of the `gpkg_extensions` table) is **`nsg_rbt`**.

Since this extension re-defines several requirements shared with other existing draft extensions which may eventually become OGC standards, conformance to and use of these other individual extensions could also be declared in the `gpkg_extensions` for the same SQLite tables, as long as the requirements remain aligned with those defined herein.

6.1.3. Extension Type

This extension provides new requirements dependent on GeoPackage Clause 2.2 (tiles).

6.1.4. Applicability

This extension defines specific classes of tiled raster and vector data to be encoded within GeoPackages (using Mapbox Vector Tiles), together with associated styles and symbology resources, for the purpose of serving as basemaps for a wide variety of operational systems.

6.1.5. Scope

read-write

6.1.6. Releasable Basemap Tiles

REQUIREMENTS CLASS 1	
TARGET TYPE	Data Product
PREREQUISITES	http://www.geopackage.org/spec140/#tiles http://www.geopackage.org/spec140/#extension_mechanism
LABEL	https://fgs-dps.gs.mil/#rbt/req

The extension consists of a single mandatory conformance class, but is broken down into conceptual classes of requirements organized in different sections, which may eventually correspond to individual standardized OGC GeoPackage extensions and/or requirement classes.

REQUIREMENT 1	
IDENTIFIER	`/req/rbt/extensions`
STATEMENT	For declaring extended GeoPackage tables
A	GeoPackages this extension SHALL include the entries listed in Table 1 in the `gpkg_extensions` table.

Table 1 — gpkg_extensions Table Rows

TABLE_NAME	COLUMN_NAME	EXTENSION_NAME	DEFINITION	SCOPE
tile pyramid user data table name	tile_data	nsg_rbt	OGC 24-010	read-write
gpkgext_content_types	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_semantic_annotations	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_sa_reference	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_vt_layers	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_vt_fields	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_styles	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_stylesheets	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_symbols	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_symbol_images	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_symbol_content	NULL	nsg_rbt	OGC 24-010	read-write
gpkgext_fonts	NULL	nsg_rbt	OGC 24-010	read-write

See Figure B.24 for an implementation example.

NOTE 1: Once this extension is published as an NSG standard, the values in the `definition` column SHALL refer to the permanent link to that standard. Until then, the `definition` SHOULD reference this engineering report using its OGC document number (`OGC 24-010`), or using the URL to the published Engineering Report.

NOTE 2: As described in the Future Work section of this Engineering Report, there is a possibility that the NSG standard be defined as a profile of official OGC GeoPackage extensions. In this case, the *extension_name* and *definition* columns would then refer to those defined in these extensions rather than *nsg_rbt*.

The following requirements pertains to the inclusion of specific content encoded in a specific manner in an RBT GeoPackage.

REQUIREMENT 2

IDENTIFIER	`/req/rbt/geodataclasses`
STATEMENT	For associating tilesets with a GeoDataClass
A	All required and optional tilesets in an RBT GeoPackage SHALL be identified with their respective GeoDataClass URI using a `GeoDataClass` semantic annotation (`/req/rbt/semantic-annotations`) associated with its corresponding `gpkg_contents` entry, as well as all of its corresponding `gpkgext_vt_layers` entries (for vector tilesets).

See Clause 8 for details on how to define semantic annotations.

See Figure B.22 and Figure B.23 for an implementation example of the `gpkgext_semantic_annotations` and `gpkgext_sa_reference` tables for defining RBT GeoDataClass semantic annotations.

REQUIREMENT 3

IDENTIFIER	`/req/rbt/world-mercator`
STATEMENT	For defining the 2D Tile Matrix Set of the tilesets
A	All required and optional tilesets in an RBT GeoPackage SHALL be tiled according to the `http://www.opengis.net/def/tilematrixset/OGC/1.0/WorldMercatorWGS84Quad` 2D Tile Matrix Set, which is based on the EPSG:3395 world Mercator coordinate reference system.

See Figure B.5 and Figure B.6 for an implementation example of the `gpkg_tile_matrix_set` and `gpkg_tile_matrix` tables for World Mercator.

PERMISSION 1

IDENTIFIER	`/per/rbt/polar-tilesets`
STATEMENT	For supporting polar regions
A	For every class of required and optional tileset, an RBT GeoPackage MAY include a corresponding tileset using the `http://www.opengis.net/def/tilematrixset/OGC/1.0/UPSArcticWGS84Quad` (EPSG:5041) and/or the `http://www.opengis.net/def/tilematrixset/OGC/1.0/UPSAntarcticWGS84Quad` (EPSG:5042) 2D Tile Matrix Set, for polar regions (beyond ±~85° of latitude) outside the bounds where WorldMercatorQuad tilesets are applicable. These tilesets would be identified using the same GeoDataClass as their corresponding world Mercator tilesets.

REQUIREMENT 4	
IDENTIFIER	/req/rbt/map-tiles
STATEMENT	For defining how map tiles are encoded
A	All required (hillshaded digital elevation model) and optional (imagery) map tilesets in an RBT Geo Package SHALL be encoded in separate tables using the tiles (/opt/tiles) data_type, without any additional content encoding (compression) applied.
B	All required (hillshaded DEM) and optional (imagery) map tilsets SHALL be identified as JPEG and/or PNG as applicable, using entries in the gpkgext_content_types table (/req/rbt/content-types) specifying the image/jpeg and/or image/png media type and a NULL encoding.
C	All optional imagery tilesets SHALL be encoded as JPEG and/or PNG.
D	The required hillshaded digital elevation model tilesets SHALL be encoded as PNG (required for translucent shadows).

REQUIREMENT 5	
IDENTIFIER	/req/rbt/physical-cultural-features
STATEMENT	For encoding physical and cultural features
A	An RBT GeoPackage SHALL include a multi-layer vector tileset representing physical features, identified using the GeoDataClass http://www.opengis.net/def/geodataclass/NSG/0/rbt-physical.
B	An RBT GeoPackage SHALL include a multi-layer vector tileset representing cultural features, identified using the GeoDataClass http://www.opengis.net/def/geodataclass/NSG/0/rbt-cultural.
C	The physical and cultural vector features in an RBT GeoPackage SHALL be encoded as gzip'ed Mapbox Vector Tiles (/req/rbt/mapbox-vector-tiles) in separate tables using the vector-tiles (/req/rbt/vector-tiles) data_type.
D	The content of the cultural and physical vector features tables SHALL be identified as gzip'ed Mapbox Vector Tiles using entries in the gpkgext_content_types table (/req/rbt/content-types) specifying the application/vnd.mapbox-vector-tile media type and gzip encoding.
E	The attributes (feature properties) for both the physical and cultural features SHALL be embedded within the Mapbox Vector Tiles.

See Clause 7 for details on how to include vector tilesets.

> **_WARNING_**
>
> _Since a registry of GeoDataClasses was not yet set up on the OGC definition server at the time of publishing this Engineering Report, the proposed RBT_

> *GeoDataClass URIs defined in this requirement class and used for the RBT GeoPackages produced by the participants are provisional and are not resolvable. See also the Styling section about the possibility to use these GeoDataClasses in the `url` field of MapboxGL style `sources`, which would rely on the ability of the definition server to return a Mapbox TileJSON representation of the schemas associated with a GeoDataClass at these GeoDataClass end-points.*

REQUIREMENT 6	
IDENTIFIER	`/req/rbt/hillshade`
STATEMENT	For encoding a hillshaded digital elevation model
A	An RBT GeoPackage SHALL include a pre-rendered map tileset, in a monochrome translucent hillshaded style, of a digital elevation model, identified using the GeoDataClass `http://www.opengis.net/def/geodataclass/NSG/0/rbt-hillshade`.

See Figure B.4 for an implementation example of the `gpkg_contents` table for RBT content.

PERMISSION 2	
IDENTIFIER	`/per/rbt/gridded-data`
STATEMENT	For encoding the source digital elevation model
A	The source digital elevation model used to generate the hillshaded tileset MAY be encoded using the OGC GeoPackage Extension for Tiled Gridded Coverage Data as a separate tileset using the `data_type: 2d-gridded-coverage`, identified using the geodataclass `http://www.opengis.net/def/geodataclass/NSG/0/rbt-dem`.

PERMISSION 3	
IDENTIFIER	`/per/rbt/imagery`
STATEMENT	For encoding imagery
A	An RBT GeoPackage MAY include an imagery tileset using the GeoDataClass `http://www.opengis.net/def/geodataclass/NSG/0/rbt-imagery`.
B	An RBT GeoPackage MAY include a tileset for high-resolution imagery of select cities in unified combatant command (COCOM) areas of responsibility identified using the GeoDataClass `http://www.opengis.net/def/geodataclass/NSG/0/rbt-cocom`.

REQUIREMENT 7	
IDENTIFIER	`/req/rbt/included-styles`
STATEMENT	For including portrayal information
A	An RBT GeoPackage SHALL include at least one style (`/req/rbt/styles`) including at minimum a MapboxGL style sheet (`/req/rbt/mapboxgl-style`).
B	Any style sheet included in an RBT GeoPackage SHALL at minimum define how to portray the physical and cultural vector tilesets.
C	All image resources used within the included styles SHALL be included within the GeoPackage at minimum as a sprite sheet, where multiple images of a particular style sheet are contained within a single `gpkgext_symbol_content` entry and the `sprite` property of the MapboxGL style reference that content entry by its `uri` (additional `gpkgext_symbol_images` entries for different resolutions and/or for individually cut images may also be included).

See Clause 9 for details on how to include styles.

> ***WARNING***
>
> *As mentioned for the physical and cultural tilesets in the above warning, the GeoDataClass URIs specified for the hillshade, elevation model and imagery are also provisionary.*

RECOMMENDATION 1	
IDENTIFIER	`/rec/rbt/included-fonts`
STATEMENT	For including fonts used in styles
A	An RBT GeoPackage SHOULD include all fonts (`/req/rbt/fonts`) used by the styles it contains in TrueType and/or OpenType formats, if their size is negligible compared to the size of the GeoPackage as a whole, or if the GeoPackage is intended for offline use and the target recipent does not already have these resources otherwise pre-installed on their system.
B	An RBT GeoPackage SHOULD include all fonts (`/req/rbt/fonts`) used by the styles it contains in zipped protobuf ranges of signed distanced field glyphs (as returned by the Mapbox Fonts API), if their size is negligible compared to the size of the GeoPackage as a whole, or if the GeoPackage is intended for offline use and the target recipent does not already have these resources otherwise pre-installed on their system.

See Requirement 20: `/req/rbt/fonts` for details on how to include fonts.

The subsequent sections define additional requirements, considered part of this same extension, pertaining to how vector tilesets are encoded in the GeoPackage using Mapbox Vector Tiles, how styles used to portray both raster and vector data are included in the GeoPackage, and how semantic annotations are defined, in order to identify the GeoDataClass of tilesets and associate styles to the tilesets.

7

VECTOR TILES

7 VECTOR TILES

7.1. Vector Tiles Requirements

7.1.1. Introduction

These requirements define how to encode tiled feature data (commonly known as *vector tiles*) in a GeoPackage data store.

For vector tilesets, all of the Tiles Option applies.

REQUIREMENT 8	
IDENTIFIER	`/req/rbt/vector-tiles`
STATEMENT	For defining how vector tiles are encoded
A	Vector tilesets SHALL be stored in a GeoPackage with individual tiles stored in the `tile_data` blob of a tile pyramid user data table.
B	The `data_type` of a vector tileset entry in the `gpkg_contents` table SHALL be `vector-tiles`.
C	The Coordinate Reference System (SRS) specified for a vector tileset in the `gpkg_spatial_ref_sys` (see clause 1.1.2 in the core GeoPackage standard) SHALL be compatible (see 2DTMS *6.2.1.1. TileMatrixSet CRS Compatibility*) with the Coordinate Reference System of the 2D Tile Matrix Set definition for that tileset.

```
CREATE TABLE tiles_physical(
   id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
   zoom_level INTEGER NOT NULL,
   tile_column INTEGER NOT NULL,
   tile_row INTEGER NOT NULL,
   tile_data BLOB NOT NULL,
   UNIQUE (zoom_level, tile_column, tile_row)
)
```

Listing 1 — Example SQL statement for creating a user-defined table storing vector tiles

See Figure B.8 for an implementation example of a user-defined vector tiles table.

There are two additional required metadata tables for vector tiles, `gpkgext_vt_layers` and `gpkgext_vt_fields`, that mirror the vector_layers key from the JSON object from the metadata from MBTiles. This allows client software to understand the feature schemas without having to open individual tiles. As with other GeoPackage tables, this extension takes no position on how either of these tables are to be used by a client.

REQUIREMENT 9

IDENTIFIER	`/req/rbt/vector-tiles-layers`
STATEMENT	For describing the layers contained in vector tilesets
A	GeoPackage containing vector tiles SHALL include a `gpkgext_vt_layers` table describing the layers in a vector tile set. The columns in this table are: • `id`, the primary key • `table_name`, which matches the entry in `gpkg_contents` • `name`, the layer name • `description`, an optional text description • `minzoom` and `maxzoom`, the optional integer minimum and maximum zoom levels • `attributes_table_name`, the optional name of an attributes table containing the attributes (when encoding attributes into a separate attributes tables rather than embedding them in the vector tiles — should be NULL for this RBT extension) • `geometry_dimension`, the dimension of the geometry found in the layer (NULL: unknown/mixed, 0: Point/MultiPoint, 1: LineString/MultiLineString, 2: Polygon/MultiPolygon)

```sql
CREATE TABLE gpkgext_vt_layers (
   id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
   table_name TEXT NOT NULL REFERENCES gpkg_contents (table_name),
   name TEXT NOT NULL,
   description TEXT,
   minzoom INTEGER,
   maxzoom INTEGER,
   attributes_table_name TEXT,
   geometry_dimension INTEGER
)
```

Listing 2 — Example SQL statement for creating the `gpkgext_vt_layers` table

See Figure B.11 for an implementation example of the `gpkgext_vt_layers` table.

REQUIREMENT 10

IDENTIFIER	`/req/rbt/vector-tiles-fields`
STATEMENT	For describing the fields contained in vector tilesets
A	GeoPackage containing vector tiles SHALL include a `gpkgext_vt_fields` table describing the fields (attributes) for a tiled feature data layer. The columns in this table are:

REQUIREMENT 10

- `id`, the primary key
- `layer_id`, a foreign key to id in `gpkgext_vt_layers`
- `name`, the field name
- `type`, either “String”, “Number”, or “Boolean”

```
CREATE TABLE gpkgext_vt_fields (
   id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
   layer_id INTEGER REFERENCES gpkgext_vt_layers,
   name TEXT NOT NULL,
   type TEXT
)
```

Listing 3 — Example SQL statement for creating the `gpkgext_vt_fields` table

See Figure B.12 for an implementation example of the `gpkgext_vt_fields` table.

REQUIREMENT 11

IDENTIFIER	`/req/rbt/content-types`
STATEMENT	For specifying the media types and content encoding of user data tables
A	The GeoPackage SHALL include a `gpkgext_content_types` table identifying the media type containing the following columns: • `content_id`, a foreign key to the corresponding entry in the `gpkg_contents` • `media_type`, text indicating a media type used in the `tile_data` blobs of the corresponding tile pyramid user data table (e.g., `image/png` or `application/vnd.mapbox-vector-tile`) • `encoding`, the content encoding (e.g., `deflate` or `gzip` compression), *NULL* if no additional encoding is used for the same `tile_data` blobs

```
CREATE TABLE gpkgext_content_types (
   content_id INTEGER REFERENCES gpkg_contents,
   media_type TEXT,
   encoding TEXT
)
```

Listing 4 — Example SQL statement for creating the `gpkgext_content_types` table

NOTE: Multiple entries for the same `content_id` indicate that multiple data media types can be found in the corresponding tiles table (e.g., JPEG and PNG).

7.2. Mapbox Vector Tiles Encoding Requirements

These requirements define how to encode vector tiles in a GeoPackage data store using the Mapbox Vector Tiles (MVT) specification version 2.1, based on Google Protocol Buffers for encoding the data of each tile.

REQUIREMENT 12	
IDENTIFIER	`/req/rbt/mapbox-vector-tiles`
STATEMENT	For describing how tiles are encoded using Mapbox Vector Tiles
A	Tilesets encoded using Mapbox Vector Tiles SHALL be stored as blobs in the `tile_data` fields of user-defined tiles tables.
B	The `tile_data` blobs SHALL be encoded Google Protocol Buffers (PBF) using the schema defined for MVT.
C	If an encoding such as `gzip` or `deflate` is specified in the `gpkgext_content_types` table, the PBF blob SHALL have this additional encoding applied.

RECOMMENDATION 2	
IDENTIFIER	`/rec/rbt/mvt-id`
STATEMENT	For specifying a unique feature ID
A	Features contained in Mapbox Vector Tiles contained in a GeoPackage vector tileset SHOULD make use of the MVT `id` field to store a persistent numeric identifier uniquely identifying a particular feature of the parent layer. This enables client to recognize features across tile boundaries (for example, to reconstruct the original features) without resorting to attribute comparison, which might not be accurate.

8

SEMANTIC ANNOTATIONS

8 SEMANTIC ANNOTATIONS

8.1. Semantic Annotations Requirements

A semantic annotation is a semantically grounded term that can be applied to another concept. These requirements define how semantic annotations can be applied to any business object in the current GeoPackage (layers, features, tiles, styles, etc.).

REQUIREMENT 13	
IDENTIFIER	`/req/rbt/semantic-annotations`
STATEMENT	For defining semantic annotations
A	A semantic annotation SHALL be defined as an entry in a `gpkgext_semantic_annotations` table with the following columns: • `id`, the a primary key • `type`, a semantically grounded type (category) for the annotation • `title`, a human-readable title for the annotation • `description`, an optional human-readable text description for the annotation • `uri`, the resolvable URI for the semantic concept

```sql
CREATE TABLE gpkgext_semantic_annotations (
   id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
   type TEXT NOT NULL, title TEXT NOT NULL,
   description TEXT,
   uri TEXT
)
```

Listing 5 — Example SQL statement for creating the `gpkgext_semantic_annotations` table

See Figure B.22 for an implementation example of the `gpkgext_semantic_annotations` table.

NOTE 1: This RBT extension relies on semantic annotations for the association of both tilesets and styles to a `GeoDataClass` `type` of semantic annotation.

REQUIREMENT 14	
IDENTIFIER	/req/rbt/sa-reference
STATEMENT	For associating semantic annotations
A	Associating a particular business object with a semantic annotation SHALL be done by adding an entry to a gpkgext_sa_reference table with the following columns: • table_name, the name of the table containing the business object • key_column_name, the name of the integer column in the specified table that acts as a key; if no such column exists, rowid can be used • key_value, the value of the key column that uniquely identifies the row • sa_id, a foreign key to gpkgext_semantic_annotations

```
CREATE TABLE gpkgext_sa_reference (
   table_name TEXT NOT NULL,
   key_column_name TEXT NOT NULL,
   key_value INTEGER NOT NULL,
   sa_id INTEGER NOT NULL REFERENCES gpkgext_semantic_annotations,
   UNIQUE(table_name, key_column_name, key_value, sa_id)
)
```

Listing 6 — Example SQL statement for creating the `gpkgext_sa_reference` table

See Figure B.23 for an implementation example of the gpkgext_sa_reference table.

NOTE 2: There can be a many-to-many mapping between business object rows and semantic annotations.

9

STYLING

9 STYLING

9.1. Styling Requirements

These requirements provide a mechanism to include the style sheets, symbols and fonts needed to portray the data stored in a GeoPackage in one or more style, as intended by the producer.

REQUIREMENT 15	
IDENTIFIER	`/req/rbt/styles`
STATEMENT	For defining styles
A	Styles to portray the data in a GeoPackage SHALL be defined as entries in the `gpkgext_styles` table containing the following columns: • `id`, a primary key • `style`, text naming a specific style • `description`, an optional text description • `uri`, a resolvable URI

```
CREATE TABLE gpkgext_styles (
   id INTEGER PRIMARY KEY,
   style TEXT NOT NULL,
   description TEXT,
   uri TEXT,
   UNIQUE(uri)
)
```

Listing 7 — Example SQL statement for creating the `gpkgext_styles` table

See Figure B.13 for an implementation example of the `gpkgext_styles` table.

RECOMMENDATION 3	
IDENTIFIER	`/rec/rbt/styles-uris`
STATEMENT	Guidance on style URIs
A	Where possible, style URIs SHOULD resolve to a network-accessible resource.

RECOMMENDATION 3	
B	When that is not possible, style URIs SHOULD be unique.
C	If no existing URI scheme is available, a style URI SHOULD take the form of: `gpkgstyle::[geodataclass]::[org]::[style]`

REQUIREMENT 16	
IDENTIFIER	`/req/rbt/style-sheets`
STATEMENT	For encoding style sheets
A	Style sheets, a list of styling rules encoded in a particular styling language, corresponding to a particular style SHALL be defined as entries in the `gpkgext_stylesheets` table containing the following columns: • `id`, a primary key • `style_id`, a foreign key to `gpkgext_styles` • `format`, the format of the stylesheet (e.g., `mbstyle` or `sld`) • `stylesheet`, the actual stylesheet BLOB (since some stylesheets are binary)

```
CREATE TABLE gpkgext_stylesheets (
   id INTEGER PRIMARY KEY,
   style_id INTEGER REFERENCES gpkgext_styles,
   format TEXT NOT NULL,
   stylesheet BLOB NOT NULL,
   UNIQUE(style_id, format)
)
```

Listing 8 — Example SQL statement for creating the `gpkgext_stylesheets` table

See Figure B.14 for an implementation example of the `gpkgext_stylesheets` table.

REQUIREMENT 17	
IDENTIFIER	`/req/rbt/symbols`
STATEMENT	For defining symbols
A	Symbols SHALL be defined in a `gpkgext_symbols` table contains containing the following columns: • `id`, a primary key • `symbol`, text naming a specific symbol • `description`, an optional text description

REQUIREMENT 17	
	• uri, a resolvable URI

```
CREATE TABLE gpkgext_symbols (
   id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
   uri TEXT,
   symbol TEXT NOT NULL,
   title TEXT NOT NULL,
   description TEXT
)
```

Listing 9 — Example SQL statement for creating the `gpkgext_symbols` table

See Figure B.18 for an implementation example of the `gpkgext_symbols` table.

RECOMMENDATION 4	
IDENTIFIER	`/rec/rbt/symbol-uris`
STATEMENT	Guidance on symbol URIs
A	Where possible, symbol URIs SHOULD resolve to a network-accessible resource.
B	When that is not possible, symbol URIs SHOULD be unique.
C	If no existing URI scheme is available, a symbol URI SHOULD take the form of: `gpkgsym::[geodataclass]::[org]::[style]::[symbol]` (`[style]` could be replaced by e.g., `multiple` if the symbol applies to multiple styles)

REQUIREMENT 18	
IDENTIFIER	`/req/rbt/symbol-images`
STATEMENT	For defining symbols
A	Images representing symbols SHALL be defined in a `gpkgext_symbol_images` table contains containing the following columns: • `id`, a primary key • `symbol_id`, a foreign key to `gpkgext_symbols` • `content_id`, a foreign key to `gpkgext_symbol_content` • `width`, `height`, optional parameters that are required for sprites or for when there are multiple versions of the same image with different sizes • `offset_x`, `offset_y`, `pixel_ratio`, optional parameters for sprite information (NULL if the entire image is used)

```sql
CREATE TABLE gpkgext_symbol_images (
   id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
   symbol_id INTEGER NOT NULL REFERENCES gpkgext_symbols,
   content_id INTEGER NOT NULL REFERENCES gpkgext_symbol_content,
   width INTEGER,
   height INTEGER,
   offset_x INTEGER,
   offset_y INTEGER,
   pixel_ratio INTEGER
)
```

Listing 10 — Example SQL statement for creating the `gpkgext_symbol_images` table

See Figure B.19 for an implementation example of the `gpkgext_symbol_images` table.

NOTE 1: If a symbol sprite sheet is included as a single image containing multiple symbols, multiple entries of this table will reference the same sprite sheet symbol content entry.

NOTE 2: Multiple images may be defined for the same symbol so as to offer different resolutions, different styles and/or for both an individual and sprite sheet version of the same symbol.

REQUIREMENT 19	
IDENTIFIER	`/req/rbt/symbol-content`
STATEMENT	For defining symbols
A	The actual data for an image SHALL be encoded in a `gpkgext_symbol_content` table containing the following columns: • `id`, a primary key • `format`, the media type (formerly MIME type, e.g., `image/svg+xml` or `image/png`) of the symbol • `content`, the actual symbol BLOB • `uri`, a resolvable name to uniquely reference a specific content entry e.g., for use in Mapbox GL styles `sprite` property to reference a particular sprite sheet

```sql
CREATE TABLE gpkgext_symbol_content (
   id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
   format TEXT NOT NULL,
   content BLOB NOT NULL,
   uri TEXT NOT NULL
)
```

Listing 11 — Example SQL statement for creating the `gpkgext_symbol_content` table

See Figure B.16 for an implementation example of the `gpkgext_symbol_content` table.

NOTE 3: Multiple versions of the same image may be included (e.g., both SVG and PNG).

9.2. Fonts Requirements

These requirements define how fonts required by styles can be included in the GeoPackage.

REQUIREMENT 20	
IDENTIFIER	`/req/rbt/fonts`
STATEMENT	For optionally including fonts
A	If included, fonts required by styles SHALL be encoded in a `gpkgext_fonts` table containing the following columns: • `id`, a primary key • `name`, the name of the font • `font`, the TrueType or OpenType font as a BLOB, • `glyphs`, a BLOB consisting of a zipped protobuf ranges of signed distanced field glyphs (as returned by the Mapbox Fonts API) with `{rangeLo-rangeHi}.pbf` filenames within the zip
B	At least `font` or `glyphs` SHALL be included in every entry

```
CREATE TABLE gpkgext_fonts(
   id INTEGER PRIMARY KEY,
   name TEXT UNIQUE,
   font BLOB,
   glyphs BLOB
)
```

Listing 12 — Example SQL statement for creating the `gpkgext_fonts` table

See Figure B.20 for an implementation example of the `gpkgext_fonts` table.

9.3. Mapbox GL Styling Specification Requirements

These requirements define how to include style sheets specified in the Mapbox GL Styling Specification within a GeoPackage.

REQUIREMENT 21	
IDENTIFIER	`/req/rbt/mapboxgl-style`
STATEMENT	For including MapboxGL style sheets
A	Styling rules encode according to the MapboxGL style sheets SHALL be included in the `gpkgext_stylesheets` table.
B	The MapboxGL style sheet SHALL refer to a particular tileset by setting the `url` field of a `source` to the GeoDataClass of that tileset, as associated using a `GeoDataClass` type of semantic annotations (`/req/rbt/semantic-annotations`).

NOTE 1: By supporting a TileJSON representation of the schemas associated with a GeoDataClass, it would be possible for the `url` field to both function as a representative data source for the style, while also corresponding to a GeoDataClass. However, it does not however limit the use of this style strictly with that representative data source. The style should be suitable to portray any RBT data source conforming to the same schemas associated with the GeoDataClass.

NOTE 2: An alternative was considered to specify the GeoDataClasses for the style in a separate `geoDataClass` property of the source, instead the `url` field. A final decision on this approach should consider the possibility of the GeoDataClass registry on the OGC definition server to return a TileJSON representation of the schemas, and would benefit from additional implementation testing, as detailed in the Future Work section of this Engineering Report.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] David Graham, Carl Reed: OGC 20-092, *CDB X Conceptual Model with Prototyping Examples and Recommendations*. Open Geospatial Consortium (2022). http://www.opengis.net/doc/DP/CDB-X-Study.

[2] Jeff Yutzler: OGC 18-101, *Vector Tiles Pilot Extension Engineering Report*. Open Geospatial Consortium (2019). http://www.opengis.net/doc/PER/VTPExt.

[3] Martin Klopfer: OGC 19-018, *OGC Testbed-15: Open Portrayal Framework Engineering Report*. Open Geospatial Consortium (2020). http://www.opengis.net/doc/PER/t15-D015.

[4] Jeff Yutzler: OGC 20-019r1, *OGC Testbed-16: GeoPackage Engineering Report*. Open Geospatial Consortium (2021). http://www.opengis.net/doc/PER/t16-D010.

[5] Gobe Hobona, Terry Idol: OGC 19-088r2, *OGC Vector Tiles Pilot 2: Summary Engineering Report*. Open Geospatial Consortium (2020). http://www.opengis.net/doc/PER/VTP2-summary.

[6] Sergio Taleisnik: OGC 19-082r1, *OGC Vector Tiles Pilot 2: Tile Set Metadata Engineering Report*. Open Geospatial Consortium (2020). http://www.opengis.net/doc/PER/vtp2-D001.

[7] Jeff Yutzler: OGC 18-074, *OGC Vector Tiles Pilot: GeoPackage 1.2 Vector Tiles Extensions Engineering Report*. Open Geospatial Consortium (2019). http://www.opengis.net/doc/PER/vtp-VTPD005.

[8] Sam Meek: OGC 18-086r1, *OGC Vector Tiles Pilot: Summary Engineering Report*. Open Geospatial Consortium (2019). http://www.opengis.net/doc/PER/vtp-summary.

[9] Jens Ingensand, Kalimar Maia: OGC 18-076, *OGC Vector Tiles Pilot: Tiled Feature Data Conceptual Model Engineering Report*. Open Geospatial Consortium (2019). http://www.opengis.net/doc/PER/vtp-conceptualModel.

[10] CDB 2.0 2023 Summer Workshop GitHub Repository, https://github.com/opengeospatial/CDBV2-2023-Summer-Workshop

[11] MBTiles Specification version 1.3, https://github.com/mapbox/mbtiles-spec/blob/master/1.3/spec.md

[12] Mapbox Vector Tiles Specification version 2.1, https://github.com/mapbox/vector-tile-spec/tree/master/2.1

[13] MapLibre Style Specification, https://maplibre.org/maplibre-style-spec/

[14] Mapbox GL Style Specification, https://docs.mapbox.com/style-spec/

[15] TileJSON Specification version 3.0, https://github.com/mapbox/tilejson-spec/tree/master/3.0.0

A

ANNEX A (NORMATIVE) CONFORMANCE CLASS ABSTRACT TEST SUITE

A ANNEX A (NORMATIVE) CONFORMANCE CLASS ABSTRACT TEST SUITE

A.1. Conformance Class Releasable Base Map Tiles GeoPackage Extension

CONFORMANCE CLASS A.1	
IDENTIFIER	`https://fgs-dps.gs.mil/#rbt/conf`
REQUIREMENTS CLASS	https://fgs-dps.gs.mil/#rbt/req
TARGET TYPE	Data Product
CONFORMANCE TESTS	Abstract test A.1: `/conf/rbt/extensions` Abstract test A.2: `/conf/rbt/geodataclasses` Abstract test A.3: `/conf/rbt/world-mercator` Abstract test A.4: `/conf/rbt/map-tiles` Abstract test A.5: `/conf/rbt/physical-cultural-features` Abstract test A.6: `/conf/rbt/hillshade` Abstract test A.7: `/conf/rbt/included-styles` Abstract test A.1-8: `/conf/rbt/vector-tiles` Abstract test A.1-9: `/conf/rbt/vector-tiles-layers` Abstract test A.1-10: `/conf/rbt/vector-tiles-fields` Abstract test A.1-11: `/conf/rbt/content-types` Abstract test A.1-12: `/conf/rbt/mapbox-vector-tiles` Abstract test A.1-13: `/conf/rbt/semantic-annotations` Abstract test A.1-14: `/conf/rbt/sa-reference` Abstract test A.1-15: `/conf/rbt/styles` Abstract test A.1-16: `/conf/rbt/style-sheets` Abstract test A.1-17: `/conf/rbt/symbol-images` Abstract test A.1-18: `/conf/rbt/symbol-content` Abstract test A.1-19: `/conf/rbt/fonts`

CONFORMANCE CLASS A.1
Abstract test A.1-20: `/conf/rbt/mapboxgl-style`

A.1.1. Abstract Test for Requirement RBT Extensions

ABSTRACT TEST A.1	
IDENTIFIER	`/conf/rbt/extensions`
REQUIREMENT	Requirement 1: `/req/rbt/extensions`
TEST PURPOSE	Verify that the RBT GeoPackage properly declare extended tables
TEST METHOD	**Given:** a GeoPackage conforming to the core GeoPackage standard **When:** querying the content of the `gpkg_extensions` table **Then:** - assert that all entries listed in Table 1 are present, including an entry for every user data table making use of these extensions.

A.1.2. Abstract Test for Requirement GeoDataClasses

ABSTRACT TEST A.2	
IDENTIFIER	`/conf/rbt/geodataclasses`
REQUIREMENT	Requirement 2: `/req/rbt/geodataclasses`
TEST PURPOSE	Verify that the RBT GeoPackage properly identify RBT tilesets using GeoDataClass semantic annotations
TEST METHOD	**Given:** a GeoPackage conforming to the core GeoPackage standard passing the `/conf/rbt/semantic-annotations` and `/conf/rbt/sa-reference` tests **When:** querying the semantic annotations **Then:** - assert that corresponding entries exist annotating the `gpkg_contents` table for the mandatory *cultural* and *physical* tilesets, as well as for the optional imagery, COCOM and digital elevation model tilesets (if present) - assert that corresponding entries exist annotating the `gpkgext_vt_layers` table for the mandatory *cultural* and *physical* tilesets

A.1.3. Abstract Test for Requirement World Mercator 2DTMS

ABSTRACT TEST A.3	
IDENTIFIER	`/conf/rbt/world-mercator`
REQUIREMENT	Requirement 3: `/req/rbt/world-mercator`
TEST PURPOSE	Verify that the RBT GeoPackage tile sets use the World Mercator 2D Tile Matrix Set
TEST METHOD	**Given:** a GeoPackage conforming to the core GeoPackage standard passing the `/conf/rbt/geodataclasses` test **When:** inspecting the gpkg_tile_matrix and gpkg_tile_matrix_sets tables associated with the identified cultural, physical, imagery COCOM, and elevation model RBT tilesets **Then:** - assert that the entries correspond to those expected for the `http://www.opengis.net/def/tilematrixset/OGC/1.0/WorldMercatorWGS84Quad` 2D Tile Matrix Set using the EPSG:3395 world Mercator coordinate reference system.

A.1.4. Abstract Test for Requirement Map Tiles

ABSTRACT TEST A.4	
IDENTIFIER	`/conf/rbt/map-tiles`
REQUIREMENT	Requirement 4: `/req/rbt/map-tiles`
TEST PURPOSE	Verify that the RBT GeoPackage tile sets includes the mandatory hillshade tileset, and encodes all mandatory and optional imagery tilesets as expected
TEST METHOD	**Given:** a GeoPackage conforming to the core GeoPackage standard passing the `/conf/rbt/geodataclasses` and `/conf/rbt/content-types` tests **When:** inspecting the available user defined tiles tables and `gpkg_contents` table **Then:** - assert that the `data_type` of the `gpkg_contents` table for these tilesets is `tiles` - assert that the `tile_data` of the the user defined tiles table contains PNG for the *hillshade* tileset, and PNG and/or JPEG for the optional imagery tilesets, without any additional encoding applied - assert that the `gpkgext_content_types` table declares these tilesets as using JPEG (except for *hillshade*) or PNG using the `image/jpeg` and/or `image/png` media type and a NULL encoding

A.1.5. Abstract Test for Requirement Physical and Cultural Features

ABSTRACT TEST A.5	
IDENTIFIER	`/conf/rbt/physical-cultural-features`
REQUIREMENT	Requirement 5: `/req/rbt/physical-cultural-features`
TEST PURPOSE	Verify that the RBT GeoPackage tile sets includes and encodes the mandatory physical and cultural vector features tile sets as expected
TEST METHOD	**Given:** a GeoPackage conforming to the core GeoPackage standard passing the `/conf/rbt/geodataclasses`, `/conf/rbt/vector-tiles`, `/conf/rbt/vector-tiles-layers` and `/conf/rbt/content-types` tests **When:** inspecting the available user defined tiles tables, `gpkgext_vt_layers`, `gpkg_contents` and their associated semantic annotations tables **Then:** - assert that a physical features tileset with the GeoDataClass http://www.opengis.net/def/geodataclass/NSG/0/rbt-physical is included - assert that a cultural features tileset with the GeoDataClass http://www.opengis.net/def/geodataclass/NSG/0/rbt-cultural is included - assert that the `data_type` of the `gpkg_contents` table for these tilesets is `vector-tiles` and that the `tile_data` of the the user defined tiles table contains gzip'ed Mapbox Vector Tiles - assert that the `gpkgext_content_types` table declares these tilesets as using Mapbox Vector Tile using the `application/vnd.mapbox-vector-tile` and the `gzip` encoding - assert that the Mapbox Vector Tiles for these tilesets contain embedded attributes

A.1.6. Abstract Test for Requirement Hillshade

ABSTRACT TEST A.6	
IDENTIFIER	`/conf/rbt/hillshade`
REQUIREMENT	Requirement 6: `/req/rbt/hillshade`
TEST PURPOSE	Verify that the RBT GeoPackage tile sets includes and encodes the mandatory hillshaded digital elevation model as expected
TEST METHOD	**Given:** a GeoPackage conforming to the core GeoPackage standard passing the `/conf/rbt/map-tiles` test **When:** inspecting the content of the user defined tiles table for the hillshaded tileset **Then:**

ABSTRACT TEST A.6	
	- assert that a hillshade tileset with the GeoDataClass http://www.opengis.net/def/geodataclass/NSG/0/rbt-hillshade is included - assert that the content of the hillshade tileset is pre-rendered map tileset, in a monochrome translucent hillshaded style encoded as PNG images

A.1.7. Abstract Test for Requirement Included Styles

ABSTRACT TEST A.7	
IDENTIFIER	`/conf/rbt/included-styles`
REQUIREMENT	Requirement 7: `/req/rbt/included-styles`
TEST PURPOSE	Verify that the RBT GeoPackage tile sets includes at least one style in a MapboxGL representation
TEST METHOD	**Given:** a GeoPackage conforming to the core GeoPackage standard passing the `/conf/rbt/geodataclasses`, `/conf/rbt/styles`, `/conf/rbt/style-sheets`, `/conf/rbt/symbol-images`, `/conf/rbt/symbol-content` and `/conf/rbt/mapboxgl-style` tests **When:** inspecting the content of the `gpkgext_styles` table for styles associated to the *physical*, *cultural* and *hillshade* RBT tilesets using the `GeoDataClass` annotation **Then:** - assert that at least one style in a MapboxGL representation is available for styling the physical and cultural and hillshade RBT tilesets - assert that all style sheets applicable to any of the RBT tilesets include styling rules for at least the physical and cutural tilesets - assert that all symbols referenced by MapboxGL style sheets are at least available as a sprite sheet where the `sprite` property of the MapboxGL style corresponds to a `uri` of an entry in the `gpkgext_symbol_content` table whose `content` blob contains all symbols - assert that individual entries in the `gpkgext_symbol_images` table, with offsets and dimensions of individual symbols, exist for all symbols referencing the sprite sheets in the `gpkgext_symbol_content` table for each Mapbox GL style

NOTE: The Abstract Test Suites corresponding to the requirements in the Vector Tiles, Semantic Annotations and Styling sections will be elaborated in the corresponding OGC GeoPackage extensions to be developed as separate documents, as documented in the Future Work section.

B

ANNEX B (INFORMATIVE) IMPLEMENTATIONS

B

ANNEX B (INFORMATIVE) IMPLEMENTATIONS

B.1. Compusult

B.1.1. About

Compusult Limited is an established, innovative technology company that focuses on developing products and services, primarily in the geospatial industry. Established in 1985, Compusult supplies computer software, hardware and consulting services directly to governments and corporations – military and non-military – around the world. Compusult products and services are proven to be cost-effective for mass, customized and secure markets.

Compusult has a wide range of products that are suitable for different situations, yet work in combination with each other for an integrated IT solution. Compusult products include: Web Enterprise Suite (WES), SensorHub, GO Mobile, Meta Manager, FasseTrack, assistive technology solutions, and robotics (UGVs) solutions.

Compusult has been involved with many past testbeds, pilots and SWGs and has aided in developing many OGC standards, specifically the OGC GeoPackage standard and extensions. Compusult wanted to build off its initial work in Vector Tiles Pilot(s), and testbeds related to OGC GeoPackage symbology and portrayal to help finalize the existing GeoPackage extensions and to produce an RBT extension for storing RBT EPSG:3395 Vector Tiles in a GeoPackage.

B.1.2. RBT GeoPackage Producer

The WES GeoPackager is a web-based application for WES that provides users with the ability to select different raster/vector sources for generating OGC compliant GeoPackages. Prior to the RBT sprint the GeoPackager had some Vector Tile support including:

- Vector Tile generation (GeoJSON, Mapbox) from well known vector sources (i.e., Shapefile, GDB, SQLite, etc.)
- Embedded styling (Mapbox Style, SLD) using Semantic Annotations extension
- Feature simplifications and optimizations for tiling

- Embedded or external feature attributes using the `Related Tables` extension

To support the RBT GeoPackage Extension the WES GeoPackager was updated to:

- Ingest the MBTiles vector source including vector tiles and metadata
- Ingest TrueType fonts and glyphs to be stored in `gpkext_fonts`
- Produce `GeoDataClass` semantic annotations based on source definitions
- Associate styles and layers with the `GeoDataClass` annotations

The Compusult GeoPackager module was updated to support the ingestion of MBTiles as well as their associated Mapbox style documents and referenced fonts as a single zipped data package. Each MBTiles file results in a unique vector tiles layer, with all relevent metadata being used to populate the RBT GeoPackage data-model.

B.1.2.1. Input

To produce a RBT GeoPackage the user must upload a ZIP file containing the relevent RBT raw content required to consume and view the vector tiles. The file upload module auto-matically detects the precense of MBTiles and gives the user the option to produce a GeoPackage.

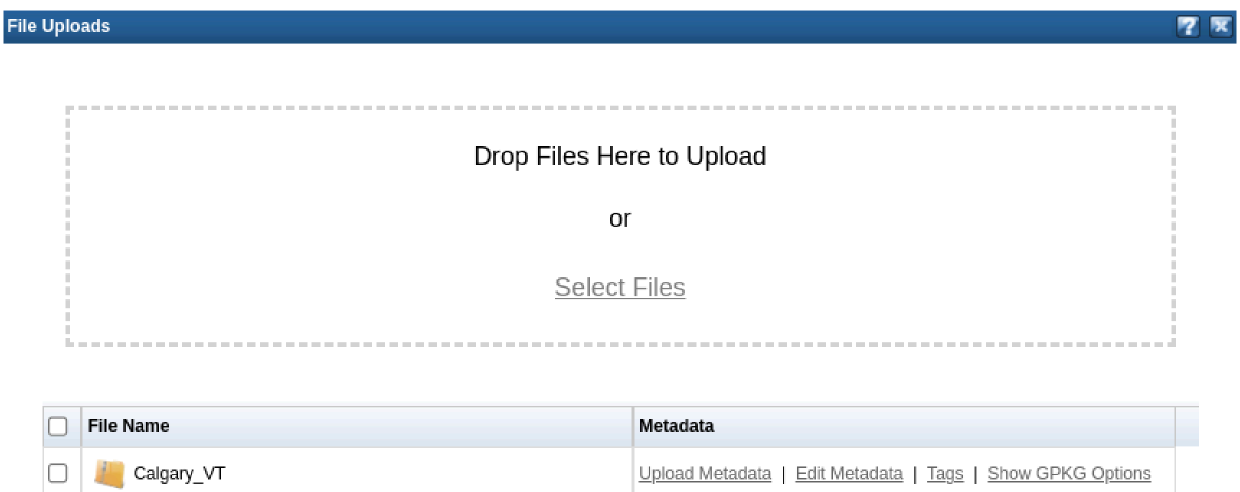


Figure B.1 — WES File Upload

The zipped raw content includes the following:

- `mbtiles` 1 or more mbtiles files containing the vector tiles and associated metadata
- `json` 0 or more Mapbox stylesheets to style the vector tiles
- `ttf/zip` 0 or more TrueType fonts or gzipped glyphs

NOTE: The names of the fonts much match the unique names specified in the stylesheet rules. If no fonts are specified, system fonts are used when visualizing the content.

Name	Size	Type
RBM-TLM-OVERLAY.json	212.1 kB	JSON docum...
RBT-JOG-3395.json	146.9 kB	JSON docum...
RBT-JOG-SATELLITE-3395.json	168.8 kB	JSON docum...
RBT-TLM-3DBLDG-3395.json	194.7 kB	JSON docum...
RBT-TLM-3395.json	168.7 kB	JSON docum...
RBT-TLM-DARK-3395.json	194.0 kB	JSON docum...
RBT-TLM-SATELLITE-3395.json	193.9 kB	JSON docum...
RBT-TPC-3395.json	150.2 kB	JSON docum...
NotoSansBold.ttf	150.9 kB	TrueType font
NotoSansItalic.ttf	150.9 kB	TrueType font
NotoSansRegular.ttf	151.0 kB	TrueType font
cultural.mbtiles	23.0 MB	unknown
hillshade3395.mbtiles	30.5 MB	unknown
imagery.mbtiles	100.3 MB	unknown
physical.mbtiles	19.5 MB	unknown

Figure B.2 — Calgary RBTs Input Package

B.1.2.2. Output

The GeoPackage producer uses the supplied input to extract the information it needs to produce and populate the RBT GeoPackage data-model. All required information is contained inside of the input, and requires no further human interaction.

The `gpkg_contents` table is populated using the information stored in the MBTiles `metadata` table as follows:

- `table_name` — to ensure a unique table name the `id` metadata field is used
- `data_type` — the `format` field [pbf=vector-tiles, png=tiles]
- `identifier` — the `name` field
- `description` — the `description` field
- `min_x,min_y,max_x,max_y` — the `bounds` field
- `srs_id` — EPSG:3395

metadata

	name	value
1	basename	cultural.mbtiles
2	id	cultural
3	filesize	45939150848
4	name	cultural
5	description	A tileset created for Army Geosp
6	version	2
7	minzoom	0
8	maxzoom	13
9	center	55.17334,25.065696,13
10	bounds	-180,-85.051129,180,85.05112
11	antimeridian_adjusted_bounds	-180.000000,-85.020708,180.0
12	type	overlay
13	attribution	<a href='https://www.agc.army.
14	format	pbf
15	generator	tile-join v2.38.0
16	json	{"vector_layers":[{"id":"adm0_la
17	scheme	tms

Figure B.3 — MBTiles metadata table

gpkg_contents

	table_name	data_type	identifier	description	last_change	min_x	min_y	max_x	max_y	srs_id
1	physical	vector-tiles	physical	water	2024-03-26T17:26:48.723Z	-20,037,508.342789244	-19,994,875.533921562	20,037,508.342789244	19,994,875.53392155	3,395
2	hillshade3395	tiles	hillshade	Hillshade generated with SRTM data courtesy of	2024-03-26T17:26:48.730Z	-20,035,356.5370322	-7,688,556.87605075	20,034,462.430622894	11,672,080.668577183	3,395
3	cultural	vector-tiles	cultural	A tileset created for Army Geospatial Center's (A	2024-03-26T17:26:48.744Z	-20,037,508.342789244	-19,994,875.533921562	20,037,508.342789244	19,994,875.53392155	3,395
4	imagery	tiles	imagery	Global Imagery Basemap	2024-03-26T17:26:48.767Z	-20,037,508.342789244	-15,454,428.346425207	20,037,508.342789244	18,722,101.52050451	3,395

Figure B.4 — RBT GeoPackage gpkg_contents table

The `gpkg_tile_matrix_set` table is populated by making a couple of assumptions:

- RBT Vector Tiles are stored in EPSG:3395 projection
- RBT uses the tile matrix set definition https://www.opengis.net/def/wkss/OGC/1.0/WorldMercatorWGS84

gpkg_tile_matrix_set

	table_name	srs_id	min_x	min_y	max_x	max_y
1	physical	3,395	-20,037,508.342789244	-20,037,508.342789244	20,037,508.342789244	20,037,508.342789244
2	hillshade3395	3,395	-20,037,508.342789244	-20,037,508.342789244	20,037,508.342789244	20,037,508.342789244
3	cultural	3,395	-20,037,508.342789244	-20,037,508.342789244	20,037,508.342789244	20,037,508.342789244
4	imagery	3,395	-20,037,508.342789244	-20,037,508.342789244	20,037,508.342789244	20,037,508.342789244

Figure B.5 — RBT GeoPackage gpkg_tile_matrix_set table

The `gpkg_tile_matrix` table is populated with entries for each MBTiles in the input making a couple of assumptions:

- `zoom_level` range is derived from the `minzoom` and `maxzoom` values in the MBTiles metadata table

- RBT uses the tile matrix definition https://www.opengis.net/def/wkss/OGC/1.0/WorldMercatorWGS84

gpkg_tile_matrix table_name ='cultural'

	table_name	zoom_level	matrix_width	matrix_height	tile_width	tile_height	pixel_x_size	pixel_y_size
1	cultural	0	1	1	256	256	156,543.033928041	156,543.033928041
2	cultural	1	2	2	256	256	78,271.5169640205	78,271.5169640205
3	cultural	2	4	4	256	256	39,135.7584820102	39,135.7584820102
4	cultural	3	8	8	256	256	19,567.8792410051	19,567.8792410051
5	cultural	4	16	16	256	256	9,783.9396205026	9,783.9396205026
6	cultural	5	32	32	256	256	4,891.9698102513	4,891.9698102513
7	cultural	6	64	64	256	256	2,445.9849051256	2,445.9849051256
8	cultural	7	128	128	256	256	1,222.9924525628	1,222.9924525628
9	cultural	8	256	256	256	256	611.4962262814	611.4962262814
10	cultural	9	512	512	256	256	305.7481131407	305.7481131407
11	cultural	10	1,024	1,024	256	256	152.8740565704	152.8740565704
12	cultural	11	2,048	2,048	256	256	76.4370282852	76.4370282852
13	cultural	12	4,096	4,096	256	256	38.2185141426	38.2185141426
14	cultural	13	8,192	8,192	256	256	19.1092570713	19.1092570713

Figure B.6 — RBT GeoPackage gpkg_tile_matrix table

Each `table_name` in `gpkg_contents` is populated with the associated vector/raster tiles found in the `tiles` table in the MBTiles file. Tile population is accomplished by running 2 SQL commands on the GeoPackage:

```
attach database /path/to/mbtiles as {mbtiles_alias};
```

Listing B.1 — Attach Database

```
insert into {table_name}
   (zoom_level, tile_column, tile_row, tile_data)
   select m.zoom_level, m.tile_column,
      power(2, m.zoom_level) - 1 - m.tile_row
   as tile_row, m.tile_data
   from {mbtiles_alias}.tiles m;
```

Listing B.2 — Copy Tiles

NOTE: The tiles stored in MBTiles has the Y axis reversed from the commonly used XYZ coordinate reference system tiles | *Enter a SQL expression to filter results (use Ctrl+Space)*

	123 zoom_level	123 tile_column	123 tile_row	tile_data
1	0	0	0	½ Gµ0ÞõèGōô¼zggw }Ì... [35496]
2	1	0	1	{ |TÅ½ÿ 9¯=çìæ±y × ð... [12328]
3	2	0	2	l Å Çwvwvwvo}¾=Ûw>Û... [3023]
4	3	1	5	|½ \Õyçë]ÝzµÞÝz4 AYy ... [58561]
5	4	2	10	Ýw\ Çû p» ¨ cì'Ö¨ z ... [342721]
6	5	5	21	ìÝw\S÷ 8zN !)KDÔ { ... [173418]
7	6	11	42	ì¼ TTWôð{ïô¹S > X£"½... [123805]
8	7	23	84	t] \ ÉÓ ā ë'v' Xèéy... [52224]
9	7	23	85	|Ý \ [û8p ±A Q ± FEÊ... [111912]
10	8	46	169	ì½g\ ½ó/¼»ô¥#*¢â öÞE)... [46478]
11	8	46	170	ì¼wXTI° zÎä9ä D ... [216530]
12	8	46	171	ì½w\TÇ zìÝzï.»ËÂ .m];... [26791]
13	8	47	170	ì= _ÍYôßïÛßëÕāYÊ+z } -... [42412]

Figure B.7 — MBTiles tiles table

cultural | *Enter a SQL expression to filter results (use Ctrl+Space)*

	123 id	123 zoom_level	123 tile_column	123 tile_row	tile_data
1	1	0	0	0	½ Gµ0ÞõèGōô¼zggw }Ì... [35496]
2	2	1	0	0	{ |TÅ½ÿ 9¯=çìæ±y × ð... [12328]
3	3	2	0	1	l Å Çwvwvwvo}¾=Ûw>Û... [3023]
4	4	3	1	2	|½ \Õyçë]ÝzµÞÝz4 AYy ... [58561]
5	5	4	2	5	Ýw\ Çû p» ¨ cì'Ö¨ z ... [342721]
6	6	5	5	10	ìÝw\S÷ 8zN !)KDÔ { ... [173418]
7	7	6	11	21	ì¼ TTWôð{ïô¹S > X£"½... [123805]
8	8	7	23	43	t] \ ÉÓ ā ë'v' Xèéy... [52224]
9	9	7	23	42	|Ý \ [û8p ±A Q ± FEÊ... [111912]
10	10	8	46	86	ì½g\ ½ó/¼»ô¥#*¢â öÞE)... [46478]
11	11	8	46	85	ì¼wXTI° zÎä9ä D ... [216530]
12	12	8	46	84	ì½w\TÇ zìÝzï.»ËÂ .m];... [26791]
13	13	8	47	85	ì= _ÍYôßïÛßëÕāYÊ+z } -... [42412]
14	14	8	47	84	ì} [Ô¹óÿöe ¾ Û]ÏÞÏ³·... [50788]

Figure B.8 — RBT GeoPackage tiles table

The `gpkgext_vt_layers` and `gpkgext_vt_fields` tables are populated based on the TileJSON definition that can be found in the `json` entry in the `metadata` table for MBTiles.

The TileJSON contains a `vector_layers` entry for each layer associated with the MBTiles. This entry specifies the associated fields of the layer, as well as the `min` and `max zoom levels`. Each layer also has an entry in the `layers` array which defines the `geometry` type of the layer.

```
"vector_layers": [
  {
    "id": "adm0_labels",
    "description": "",
    "minzoom": 0,
    "maxzoom": 13,
    "fields": {
      "adm0_name": "String",
      "adm0_name1": "String",
      "area": "Number",
      "gns_deig_cd": "String",
      "gns_full_name": "String",
      "gns_name_rank": "Number",
      "ne_abbrev": "String",
      "ne_formal_en": "String",
      "ne_name": "String",
      "ne_name_long": "String",
      "status_cd": "Number",
      "status_nm": "String"
    }
  },
```

Figure B.9 — TileJSON Vector Layers

```
"layers": [
  {
    "layer": "adm0_labels",
    "count": 4707,
    "geometry": "Point",
    "attributeCount": 12,
    "attributes": [ … ] // 12 items
  },
```

Figure B.10 — TileJSON Layers

Using this information the `gpkgext_vt_layers` and `gpkgext_vt_fields` can be populated.

gpkgext_vt_layers | table_name = 'cultural'

	id	table_name	name	description	attributes_table_name	minzoom	maxzoom	geometry_dimension
1	15	cultural	adm0_labels		[NULL]	0	13	0

Figure B.11 — RBT GeoPackage gpkgext_vt_layers table gpkgext_vt_fields | layer_id = 15

	id	layer_id	name	type
1	26	15	area	Number
2	27	15	ne_name_long	String
3	28	15	adm0_name	String
4	29	15	status_cd	Number
5	30	15	ne_formal_en	String
6	31	15	ne_name	String
7	32	15	status_nm	String
8	33	15	gns_name_rank	Number
9	34	15	gns_full_name	String
10	35	15	ne_abbrev	String
11	36	15	gns_deig_cd	String
12	37	15	adm0_name1	String

Figure B.12 — RBT GeoPackage gpkgext_vt_fields table

The `gpkgext_stylesheets` and `gpkgext_styles` tables are populated depending on the stylesheets uploaded. Each Mapbox stylesheet represents a unique style that can be associated with a `GeoDataClass`

gpkgext_styles | *Enter a SQL expression to filter results (use Ctrl+Space)*

	id	style	description	uri
1	1	RBT-TLM-OVERLAY-3395	RBT-TLM-OVERLAY-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TLM-OVERLAY-3395/style.json
2	2	RBT-TLM-3DBLDG-3395	RBT-TLM-3DBLDG-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TLM-3DBLDG-3395/style.json
3	3	RBT-TLM-DARK-3395	RBT-TLM-DARK-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TLM-DARK-3395/style.json
4	4	RBT-TLM-SATELLITE-3395	RBT-TLM-SATELLITE-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TLM-SATELLITE-3395/style.json
5	5	RBT-JOG-SATELLITE-3395	RBT-JOG-SATELLITE-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-JOG-SATELLITE-3395/style.json
6	6	RBT-TLM-3395	RBT-TLM-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TLM-3395/style.json
7	7	RBT-TPC-3395	RBT-TPC-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TPC-3395/style.json
8	8	RBT-JOG-3395	RBT-JOG-3395 mbstyle style document	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-JOG-3395/style.json

Figure B.13 — RBT GeoPackage gpkgext_styles table

gpkgext_stylesheets | *Enter a SQL expression to filter results (use Ctrl+Space)*

	id	style_id	format	stylesheet
1	1	1	mbstyle	{"version":8,"name":"RBT-TLM-OVE... [212126]
2	2	2	mbstyle	{"version":8,"name":"RBT-TLM-3DB... [194721]
3	3	3	mbstyle	{"version":8,"name":"RBT-TLM-DAR... [193959]
4	4	4	mbstyle	{"version":8,"name":"RBT-TLM-SAT... [193872]
5	5	5	mbstyle	{"version":8,"name":"RBT-JOG-SAT... [168770]
6	6	6	mbstyle	{"version":8,"name":"RBT-TLM-339... [168698]
7	7	7	mbstyle	{"version":8,"name":"RBT-TPC-339... [150219]
8	8	8	mbstyle	{"version":8,"name":"RBT-JOG-339... [146855]

Figure B.14 — RBT GeoPackage gpkgext_stylesheets table

If any stylesheets are provided the `gpkgext_symbol_content` tables are populated using the avalable `sprite` source(s).

```
{
    "version": 8,
    "name": "RBT-TPC-3395",
    "owner": { … }, // 2 items
    "metadata": { … }, // 2 items
    "sources": { … }, // 3 items
    "sprite": "https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TPC-3395/sprite",
    "glyphs": "https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/fonts/{fontstack}/{range}.pbf",
    "layers": [ … ], // 146 items
    "id": "RBT-TPC-3395"
}
```


Figure B.15 — MBStyle Sprites

gpkgext_symbol_content | Enter a SQL expression to filter results (use Ctrl+Space)

	id	format	content	uri
1	1	image/png	PNG IHDR A CW… [27690]	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TLM-SATELLITE-3395/sprite
2	2	image/png	PNG IHDR A CW… [27690]	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-JOG-SATELLITE-3395/sprite
3	3	image/png	PNG IHDR û ª ¼… [137121]	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TLM-3395/sprite
4	4	image/png	PNG IHDR û ª ¼… [137121]	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-TPC-3395/sprite
5	5	image/png	PNG IHDR û ª ¼… [137121]	https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/styles/RBT-JOG-3395/sprite

Figure B.16 — RBT GeoPackage gpkgext_symbol_content table

The `gpkgext_symbol_content` and `gpkgext_symbol_images` tables are then populated based on the sprite metadata found at `sprite.json`.


```
{
    "aerial-tower-communication": {
        "height": 76,
        "pixelRatio": 1,
        "width": 48,
        "x": 65,
        "y": 0
    },
    "aerial-tower-communication-sm": { … }, // 5 items
    "cable-nonObstruction-communication-powerDistribution": { … },
    "cable-nonObstruction-powerTransmission": { … }, // 5 items
```


Figure B.17 — MBStyle Sprite Metadata gpkgext_symbols | Enter a SQL expression to filter results (use Ctrl+Space)

	id	symbol	title	description	uri
1	1	aerial-tower-communication	aerial-tower-communication	[NULL]	aerial-tower-communication
2	2	dot-8	dot-8	[NULL]	dot-8
3	3	dot-5	dot-5	[NULL]	dot-5
4	4	dot-4	dot-4	[NULL]	dot-4
5	5	railway-1track	railway-1track	[NULL]	railway-1track
6	6	dot-7	dot-7	[NULL]	dot-7
7	7	circle-dot-9	circle-dot-9	[NULL]	circle-dot-9
8	8	dot-6	dot-6	[NULL]	dot-6
9	9	circle-dot-8	circle-dot-8	[NULL]	circle-dot-8

Figure B.18 — RBT GeoPackage gpkgext_symbols table

gpkgext_symbol_images | Enter a SQL expression to filter results (use Ctrl+Space)

	id	symbol_id	content_id	width	height	offset_x	offset_y	pixel_ratio
1	1	1	1	48	76	65	0	1
2	2	2	1	8	8	261	255	1
3	3	3	1	5	5	295	255	1
4	4	4	1	4	4	300	255	1
5	5	5	1	56	18	24	231	1
6	6	6	1	7	7	276	255	1
7	7	7	1	9	9	252	255	1
8	8	8	1	6	6	289	255	1
9	9	9	1	8	8	512	309	1

Figure B.19 — RBT GeoPackage gpkgext_symbol_images table

The `gpkgext_fonts` tables can also be populated by retrieving the `glyphs` endpoint from the Mapbox stylesheet and zipping the content. Uploaded TrueType fonts are also supported and are gzipped before inserting.

id	name	font	glyphs
1	NGATopo_Cn_bld	[NULL]	PK +rzX 38... [117764]
2	NGATopo_Cn_ita	[NULL]	PK 3rzX 64... [128226]
3	NGATopo_Cn_reg	[NULL]	PK :rzX 55... [115541]
4	NGATopo_LtCn_ita	[NULL]	PK CrzX 31... [124686]
5	NGATopo_LtCn_reg	[NULL]	PK KrzX 30... [112031]
6	NGATopo_barcode	[NULL]	PK RrzX 55... [44047]
7	NGATopo_bi	[NULL]	PK YrzX 29... [138602]
8	NGATopo_bld	[NULL]	PK brzX 42... [123574]
9	NGATopo_ita	[NULL]	PK irzX 43... [134411]
10	NGATopo_reg	[NULL]	PK przX 46... [121473]
11	NotoSansBold	ì½iwÛH²6ø½~ ^öé ¾o öÅ... [31551]	PK }rzX 55... [21184053]
12	NotoSansItalic	ì½{wÛFÒ7ø > ^Î Ù 7 ûÅ... [31544]	PK rzX 62... [20340265]
13	NotoSansRegular	ì½{wÛFÒ7ø > ^Î Ù w hÜ... [31578]	PK rzX 32... [20300271]

Figure B.20 — RBT GeoPackage gpkgext_fonts table

To appropriately link the `gpkgext_styles` and the `gpkgext_vt_layers` the `gpkgext_semantic_annotations` table is populated with any `GeoDataClass` semantic annotations found in the Mapbox stylesheet sources, or defined in the `gpkg_contents` table.

```
{
    "version": 8,
    "name": "RBT-TPC-3395",
    "owner": {…}, // 2 items
    "metadata": {…}, // 2 items
    "sources": {
        "CULTURAL": {
            "url": "https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/data/CULTURAL.json",
            "type": "vector"
        },
        "PHYSICAL": {
            "url": "https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/data/PHYSICAL.json",
            "type": "vector"
        },
        "HILLSHADE": {
            "tileSize": 256,
            "type": "raster",
            "url": "https://tileserver-rbt-agc-dev.apps.kubic.dev.ngaxc.net/data/HILLSHADE.json"
        }
    },
```

Figure B.21 — MBStyle Source Metadata

gpkgext_semantic_annotations | Enter a SQL expression to filter results (use Ctrl+Space)

	id	type	title	description	uri
1	1	GeoDataClass	physical	Physical features as specified by AGC for Releasable Basemap Tiles (RBT)	http://www.opengis.net/def/geodataclass/NSG/0/rbt-physical
2	2	GeoDataClass	hillshade	Grayscale hillshaded elevation data as specified by AGC for Releasable Basemap Tiles (RBT)	http://www.opengis.net/def/geodataclass/NSG/0/rbt-hillshade
3	3	GeoDataClass	cultural	Cultural features as specified by AGC for Releasable Basemap Tiles (RBT)	http://www.opengis.net/def/geodataclass/NSG/0/rbt-cultural
4	4	GeoDataClass	imagery	Global Imagery data as specified by AGC for Releasable Basemap Tiles (RBT)	http://www.opengis.net/def/geodataclass/NSG/0/rbt-imagery

Figure B.22 — RBT GeoPackage gpkg_semantic_annotations table

Once the `GeoDataClass` semantic annotations have been established they are linked to the layers or styles that reference them by populating the `gpkgext_sa_reference` table.

gpkgext_vt_layers	id	43	3
gpkgext_vt_layers	id	44	3
gpkgext_vt_layers	id	45	3
gpkgext_vt_layers	id	46	3
gpkgext_vt_layers	id	47	3
gpkgext_vt_layers	id	48	3
gpkgext_vt_layers	id	49	3
gpkgext_vt_layers	id	50	3
gpkgext_vt_layers	id	51	3
gpkgext_vt_layers	id	52	3
gpkgext_vt_layers	id	53	3
gpkgext_vt_layers	id	54	3
gpkgext_vt_layers	id	55	3
gpkgext_vt_layers	id	56	3
gpkg_contents	rowid	4	4
gpkgext_styles	id	1	1
gpkgext_styles	id	1	3
gpkgext_styles	id	1	4
gpkgext_styles	id	2	1
gpkgext_styles	id	2	3
gpkgext_styles	id	2	4
gpkgext_styles	id	3	1
gpkgext_styles	id	3	2
gpkgext_styles	id	3	3
gpkgext_styles	id	4	1
gpkgext_styles	id	4	2

Figure B.23 — RBT GeoPackage gpkgext_sa_reference table

With all data and relations populated in the RBT GeoPackage the last thing the producer does is register the extensions that are used in its construction. This is done be populating the `gpkg_extensions` table.

gpkg_extensions | Enter a SQL expression to filter results (use Ctrl+Space)

	table_name	column_name	extension_name	definition	scope
1	gpkgext_styles	[NULL]	nsg_rbt	OGC 24-010	read-write
2	gpkgext_stylesheets	[NULL]	nsg_rbt	OGC 24-010	read-write
3	gpkgext_symbol_content	[NULL]	nsg_rbt	OGC 24-010	read-write
4	gpkgext_symbol_images	[NULL]	nsg_rbt	OGC 24-010	read-write
5	gpkgext_symbols	[NULL]	nsg_rbt	OGC 24-010	read-write
6	gpkgext_semantic_annotations	[NULL]	nsg_rbt	OGC 24-010	read-write
7	gpkgext_sa_reference	[NULL]	nsg_rbt	OGC 24-010	read-write
8	gpkgext_vt_layers	[NULL]	nsg_rbt	OGC 24-010	read-write
9	gpkgext_vt_fields	[NULL]	nsg_rbt	OGC 24-010	read-write
10	gpkgext_content_types	[NULL]	nsg_rbt	OGC 24-010	read-write
11	gpkgext_fonts	[NULL]	nsg_rbt	OGC 24-010	read-write
12	physical	tile_data	nsg_rbt	OGC 24-010	read-write
13	cultural	tile_data	nsg_rbt	OGC 24-010	read-write

Figure B.24 — RBT GeoPackage gpkg_extensions_ table

Once the GeoPackage has been produced, the GeoPackage becomes discoverable in a WES Portfolio, and can be viewed on the web-browser based client using the automatically generated GeoPackage WMS/WMTS service. It can now also be discovered by the GO Mobile product line which allows for discovery, consumption, display and analysis of the RBT GeoPackage.

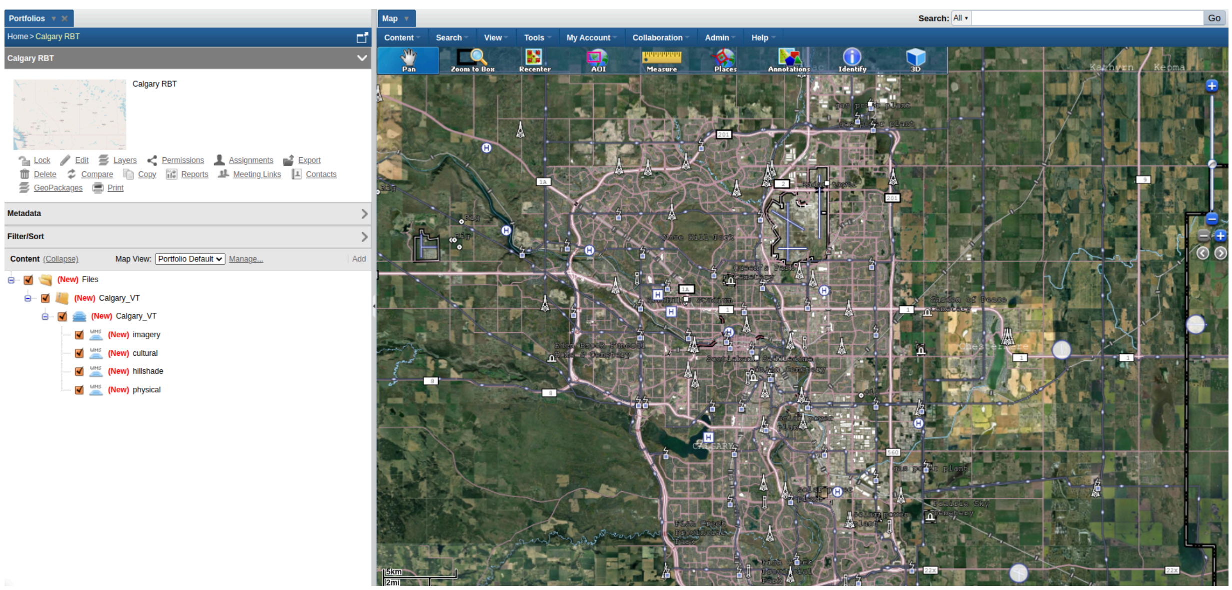


Figure B.25 — WES RBT GeoPackage in Portfolio

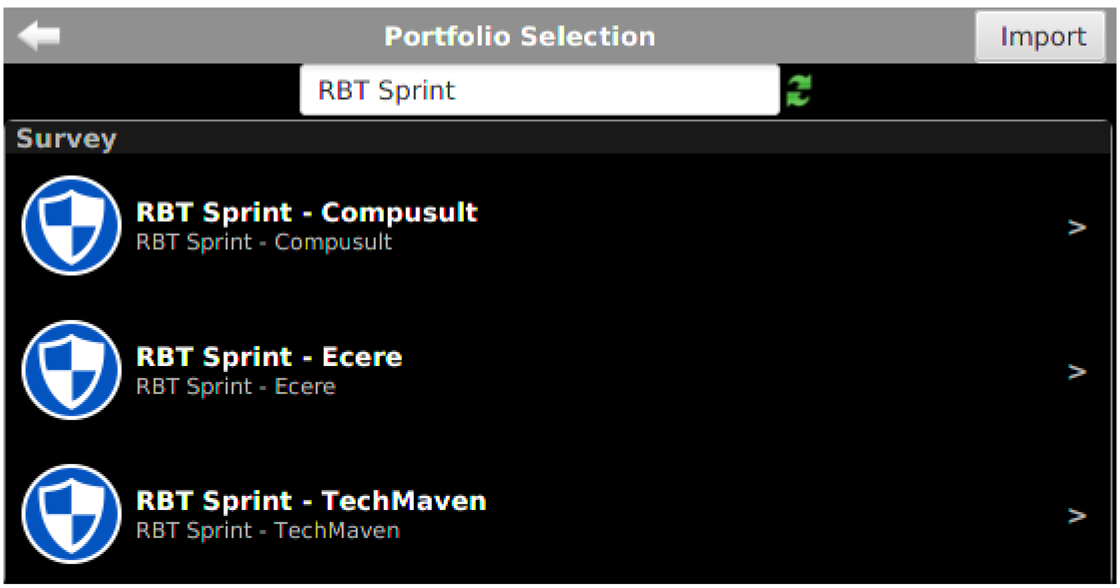
B.1.3. RBT GeoPackage Consumer/Viewer

GO Mobile is a mobile app for iOS, Android and Windows-based smartphones, tablets and PCs that provides and supports superior situational awareness for military users, first responders and other field users that use and collect geospatial information in connected and disconnected modes of operation. It allows users to easily discover and access geospatial content and other data and services through a variety of communication networks. GO Mobile provides the collection, sharing and synchronization of information and content among other mobile users, command/operational posts and centralized operations, such as regional headquarters.

The Compusult GO Mobile product line has been updated to support the rendering and analysis of RBT GeoPackages. This required updates to the map rendering engine to support/enhance the following concepts:

- GeoDataClass semantic annotations for style association
- Decoding of Gzipped Vector Tiles using `gpkgext_content_types`
- Mapbox Style Specification parsing and expression handling
- Mapbox Style Specification rendering rules, labeling and collision detection

To view any created GeoPackages the user must first login to a deployment of WebEnterpriseSuite where the GeoPackage has been created/uploaded to its catalog. Once logged in the user is given the option to open a Portfolio which contains the relevant content.



NOTE: If the user has an external GeoPackage that has not been added to a Portfolio, they are able to side-load the GeoPackage from a local/remote drive, USB device, or accessible URI.

Figure B.26 — GO Mobile Portfolio Selection

Once the GeoPackage has been loaded into the application the user is able to view the layers contained within the GeoPackage, allowing them to turn on/off individual layers, view metadata, as well as specify which stylesheet to use when rendering the content.

The user is also able to manually modify the drawing order by clicking the `Edit` button and dragging the layers up or down in the list.

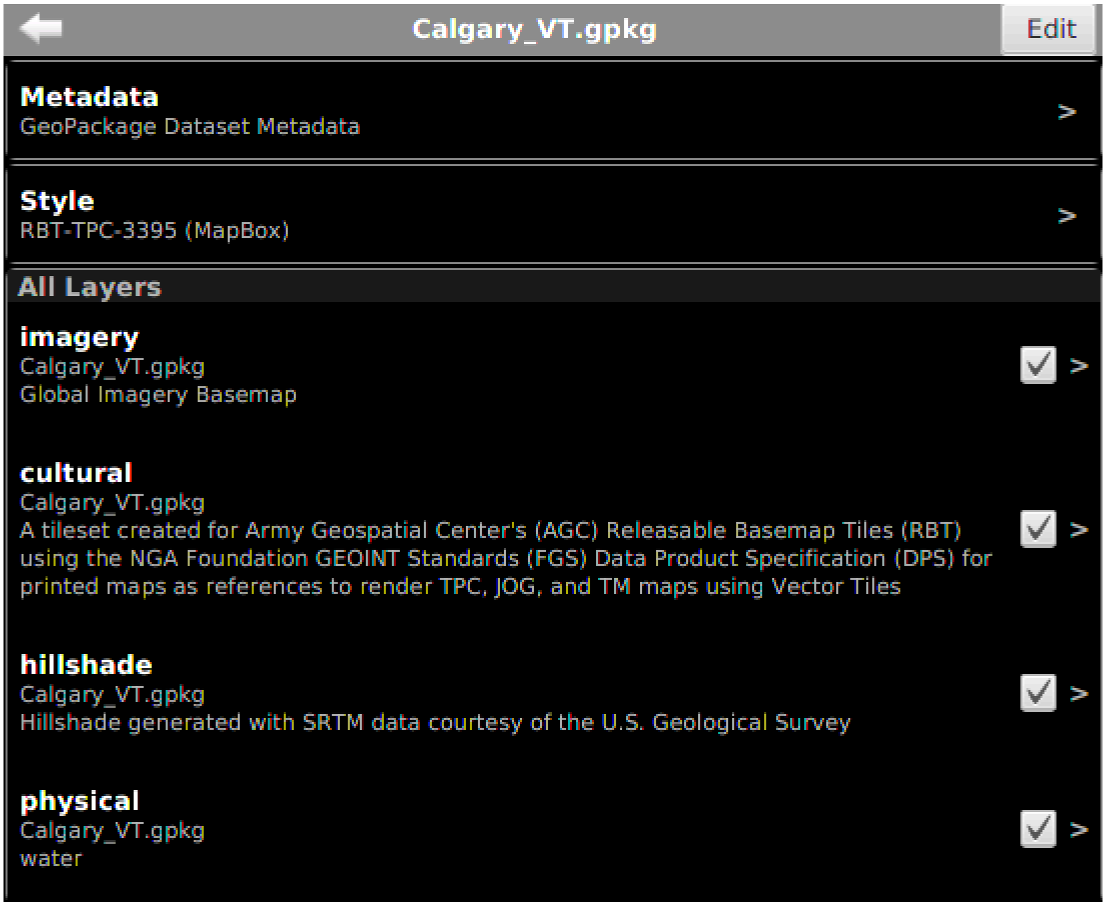


Figure B.27 — GO Mobile Layer Selection

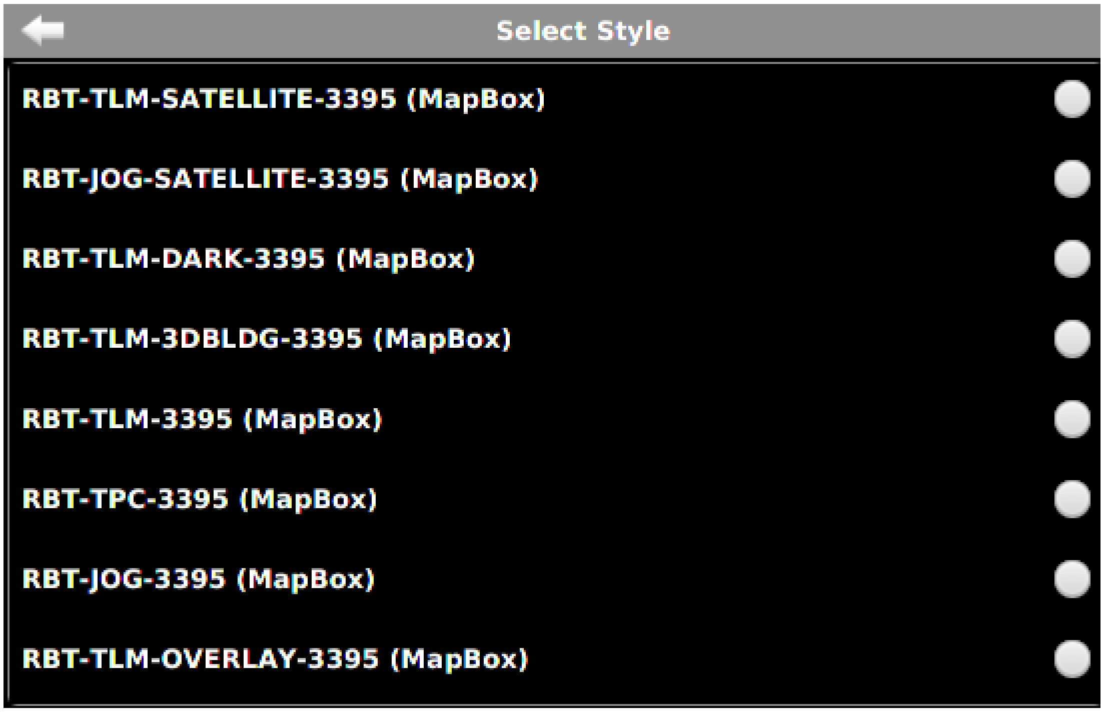


Figure B.28 — GO Mobile Style Selection

The GO Mobile map client is able to apply the selected stylesheet against the Vector Tiles to visualize the features found within them. The client is also able to re-project the content to different map projections to support overlaying them on other WMS,WMTS and OGC API services that do not support EPSG:3395, but support other common projections like EPSG:4326 and EPGS:3857.

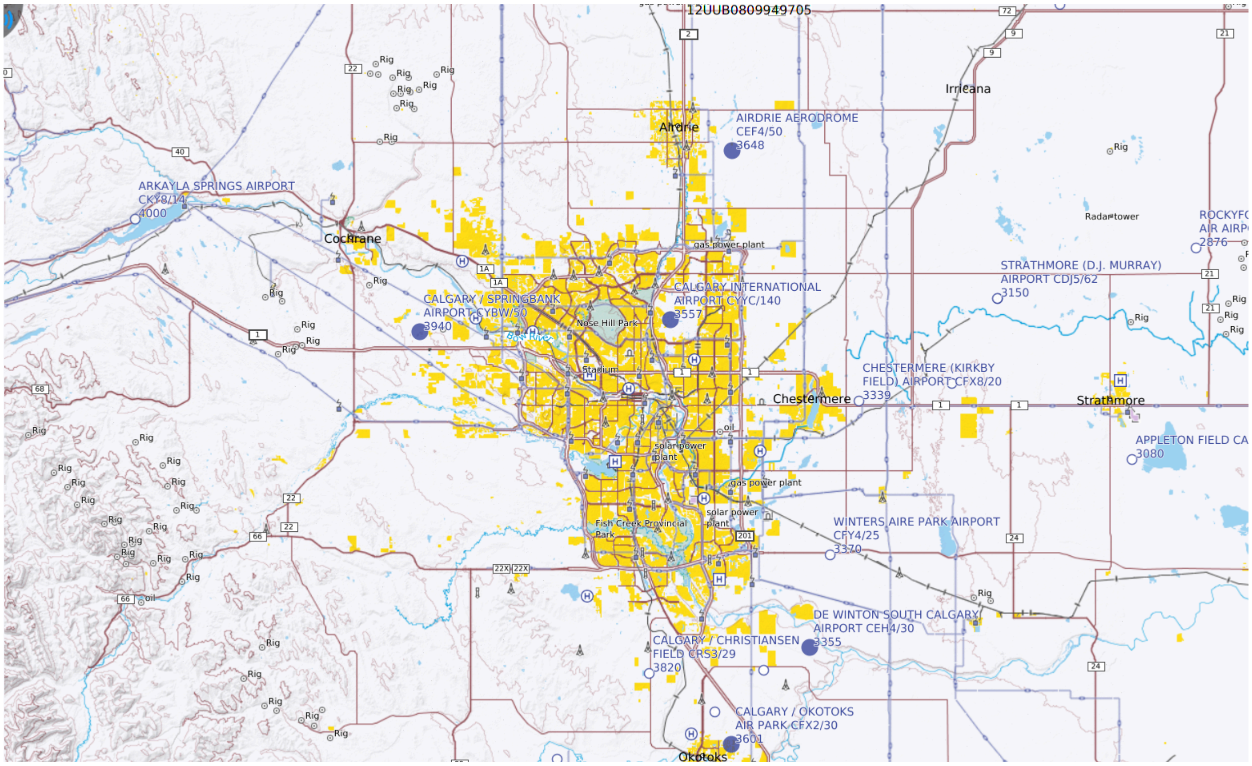


Figure B.29 — GO Mobile RBT GeoPackage Rendering Overview

To demonstrate the ability to use a single RBT GeoPackage with multiple different styles, a snapshot of each style for the Calgary region can be found below.

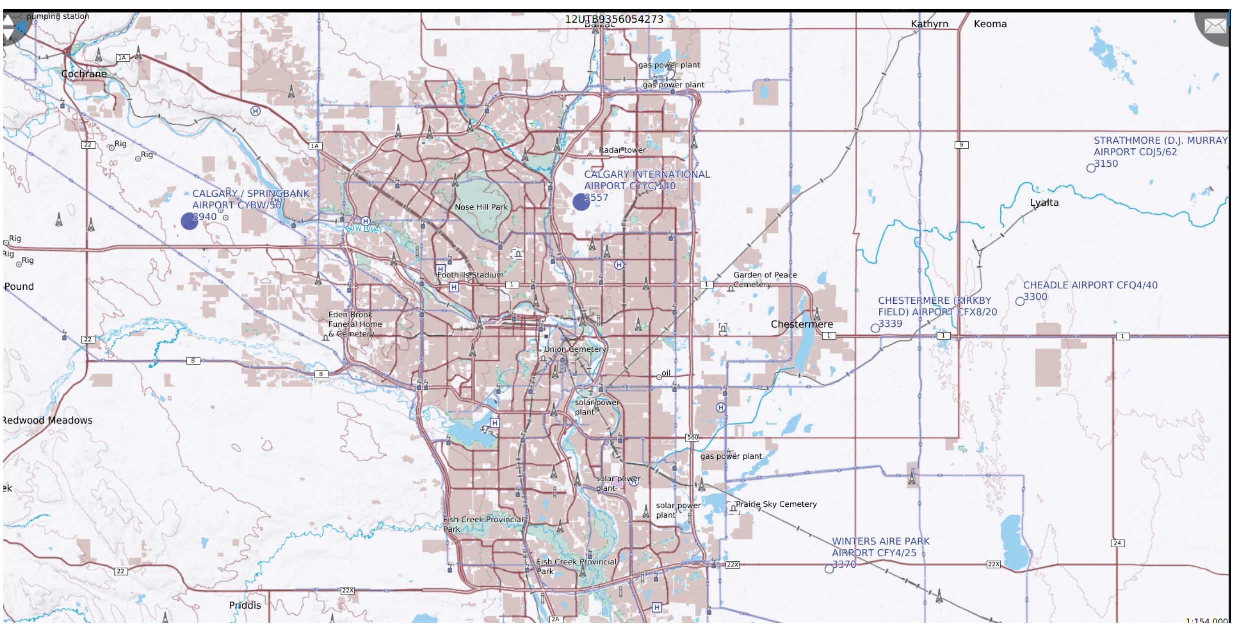


Figure B.30 — GO Mobile RBT GeoPackage RBT-TPC-3395

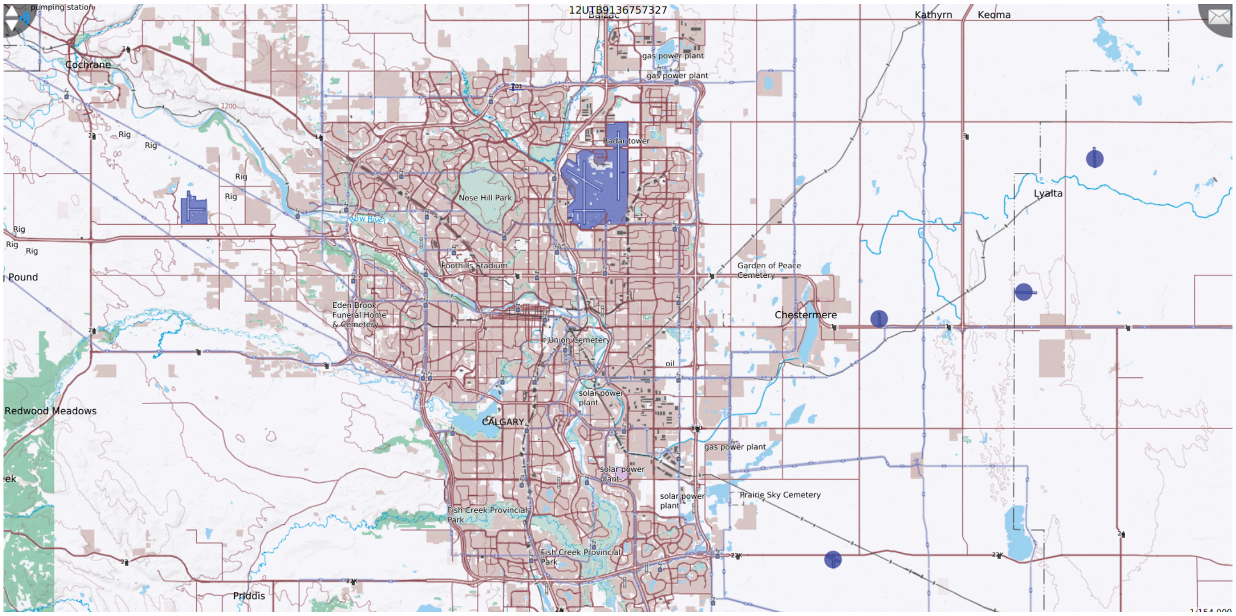


Figure B.31 — GO Mobile RBT GeoPackage RBT-TLM-3395

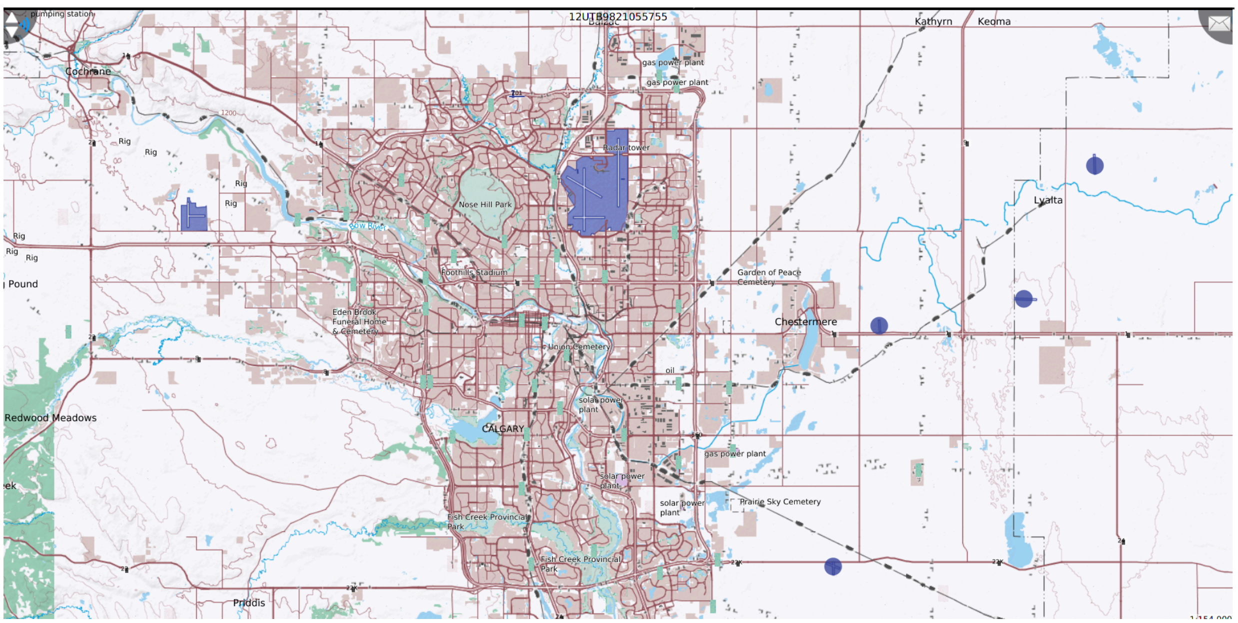


Figure B.32 — GO Mobile RBT GeoPackage RBT-JOG-3395

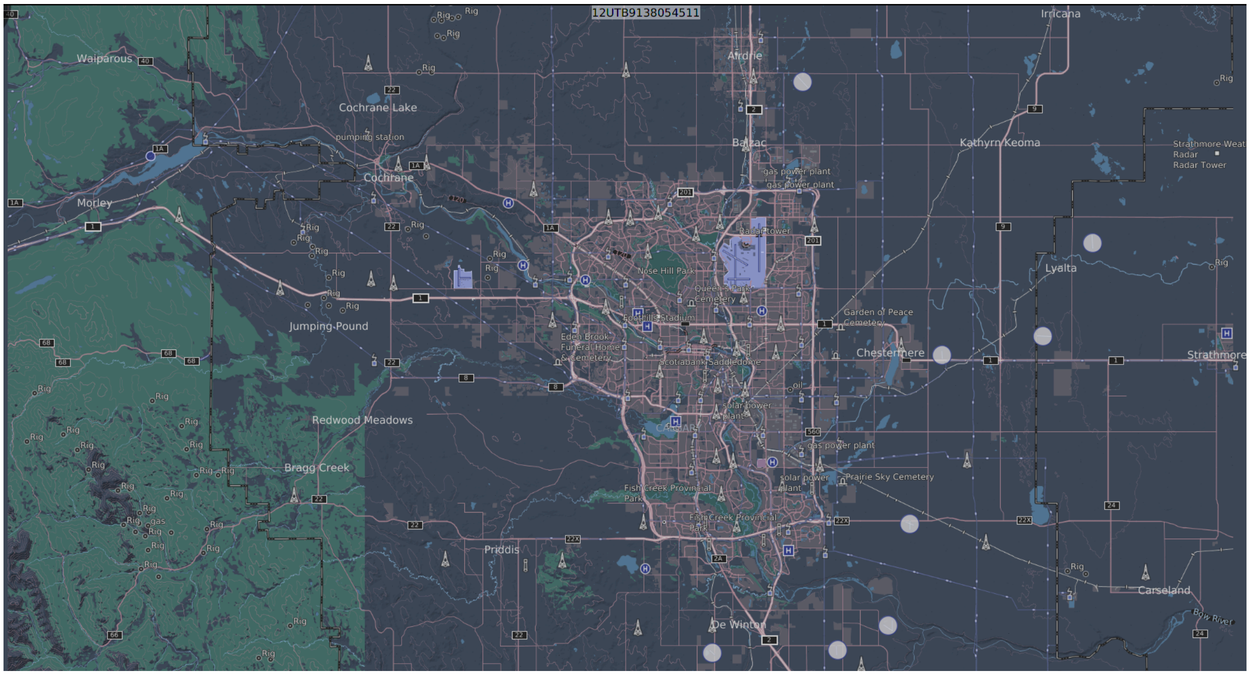


Figure B.33 — GO Mobile RBT GeoPackage RBT-TLM-DARK-3395

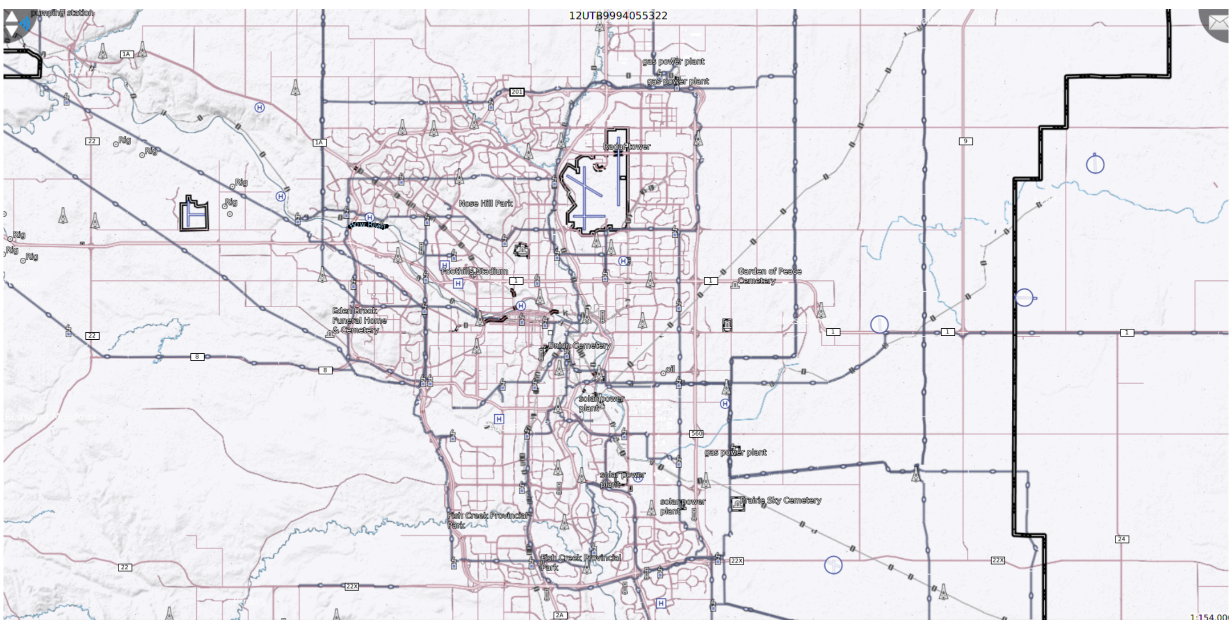


Figure B.34 — GO Mobile RBT GeoPackage RBM-TLM-OVERLAY

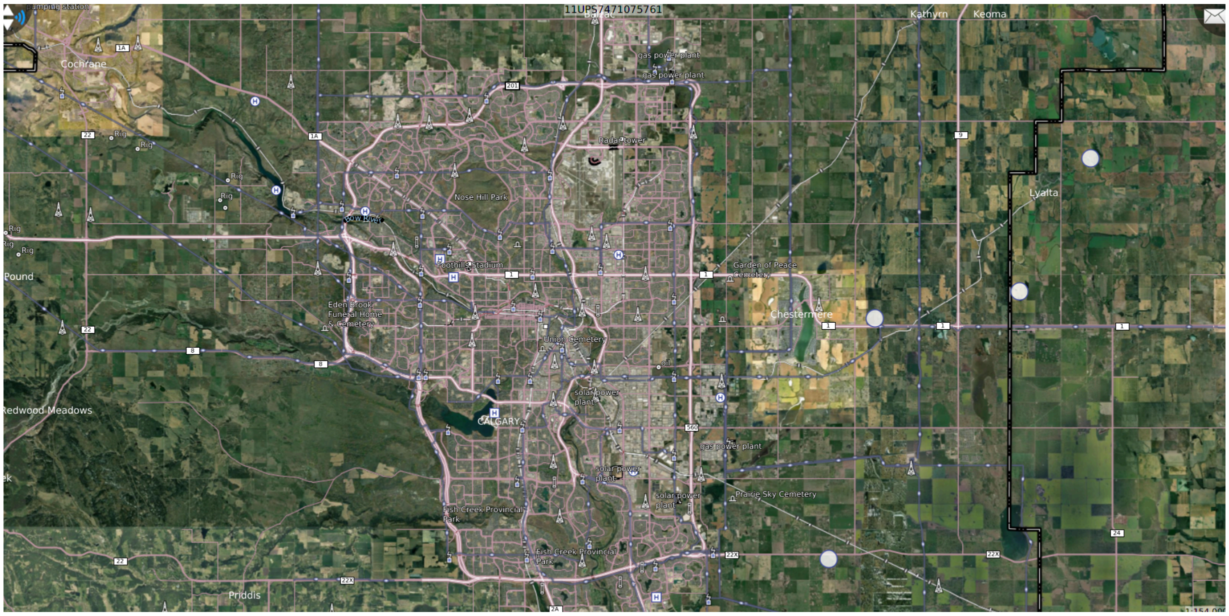


Figure B.35 — GO Mobile RBT GeoPackage RBT-TLM-SATELLITE-3395

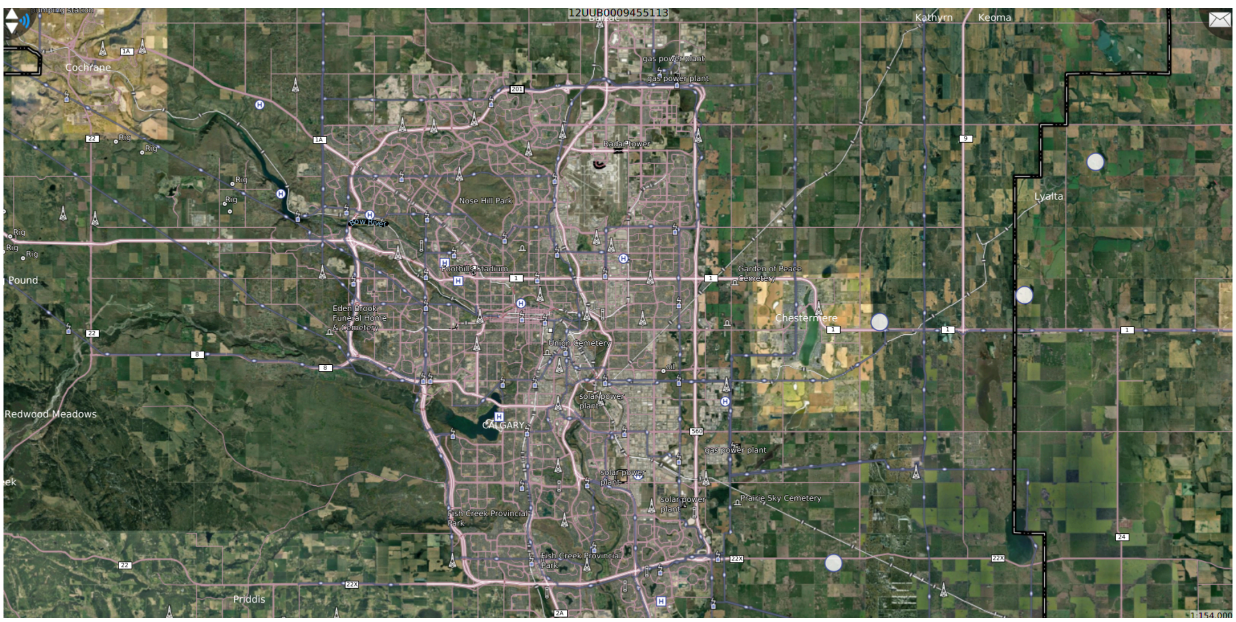


Figure B.36 — GO Mobile RBT GeoPackage RBT-JOG-SATELLITE-3395

Not only can the GO Mobile product line render the vector tile content, it also allows for dynamic feature analysis by clicking a location on the map. The software dynamically retrieves all visible features which intersect that point on the map, and displays a list of features to chose from. Features are grouped by the internal vector layers they are associated with.

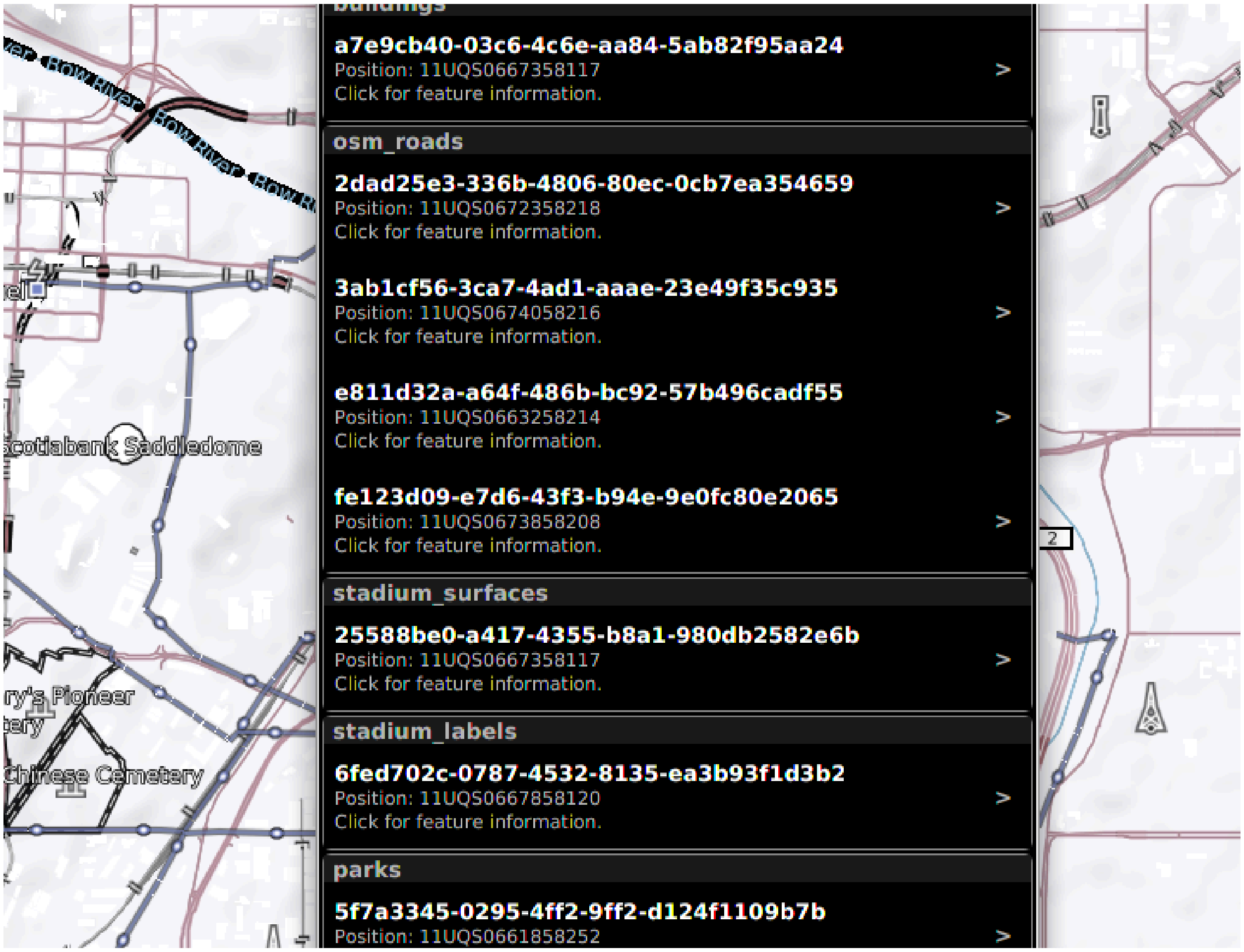


Figure B.37 — GO Mobile RBT GeoPackage Feature List

Once a user identifies the feature they are looking for they are able to select that feature and get a complete view of that features metadata, along with any field data associated with that feature. A user is also given the option to zoom to the bounds of that feature, to get a more detailed view of its surroundings.

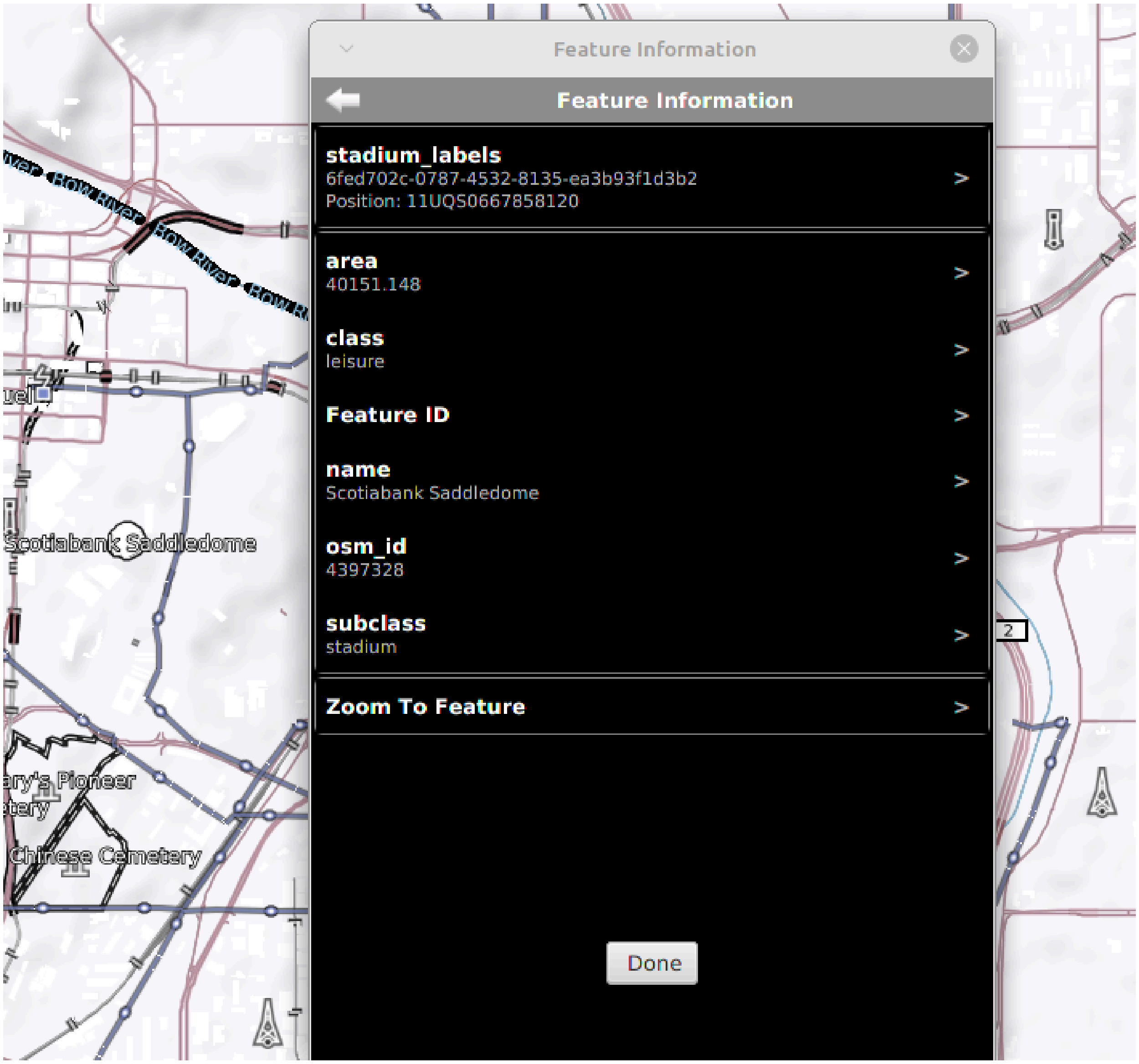


Figure B.38 — GO Mobile RBT GeoPackage Feature Info

B.1.4. Technology Integration Experiments

Throughout and after the sprint, Compusult attempted TIEs against any produced GeoPackages by the other participants. A WES Portfolio was created for each participant and the GeoPackages were published to those portfolios to be discovered and accessed by the GO Mobile client.

B.1.4.1. Ecere

The Compusult GO Mobile software was able to visualize and retrieve feature information from the Ecere Europe GeoPackage without modification. Below are some screenshots of different areas and resolutions using the RBT-TPC-3395 stylesheet.

NOTE: The GeoPackage produced by Ecere used an older version of the RBT styles, where the same sprite sheet was used for all styles, therefore the same sprites are used regardless of the style.

Figure B.39 — GO Mobile Ecere GeoPackage Europe (Low Resolution)

Figure B.40 — GO Mobile Ecere GeoPackage Europe (Medium Resolution)

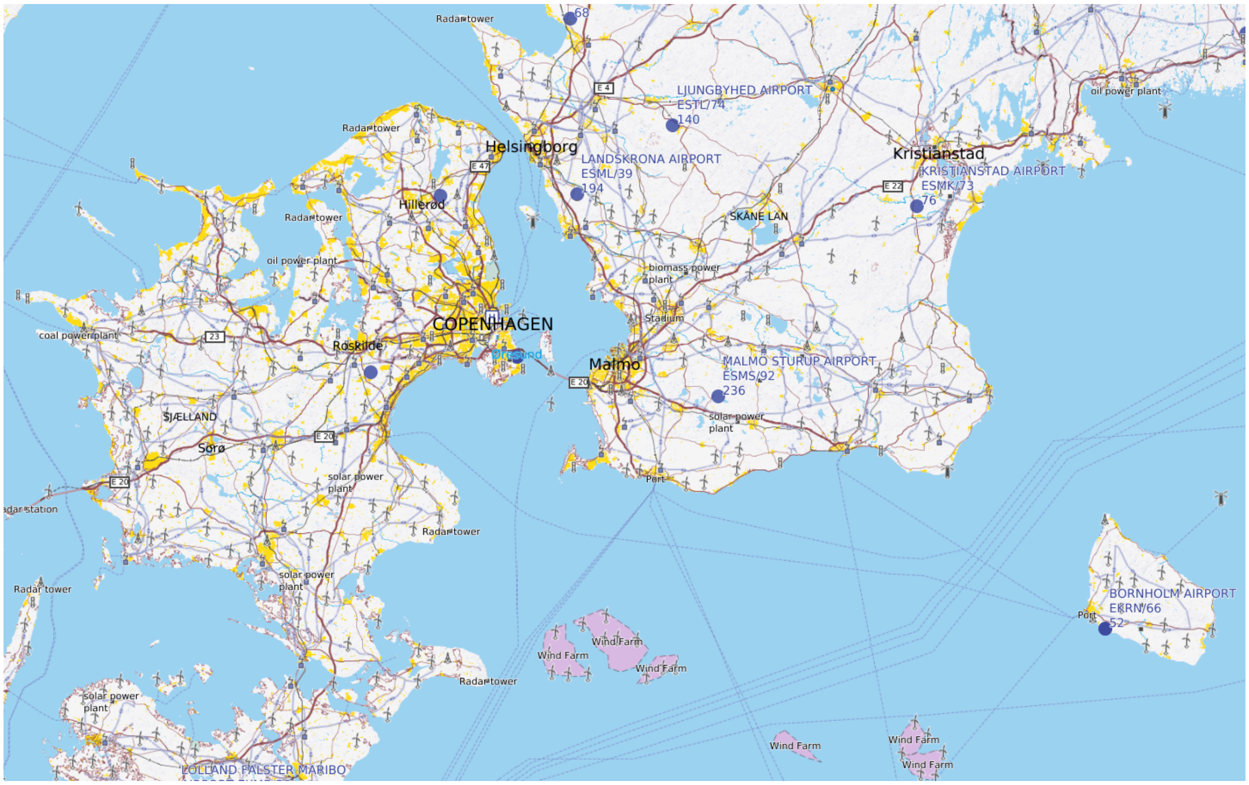


Figure B.41 — GO Mobile Ecere GeoPackage Copenhagan (Medium Resolution)

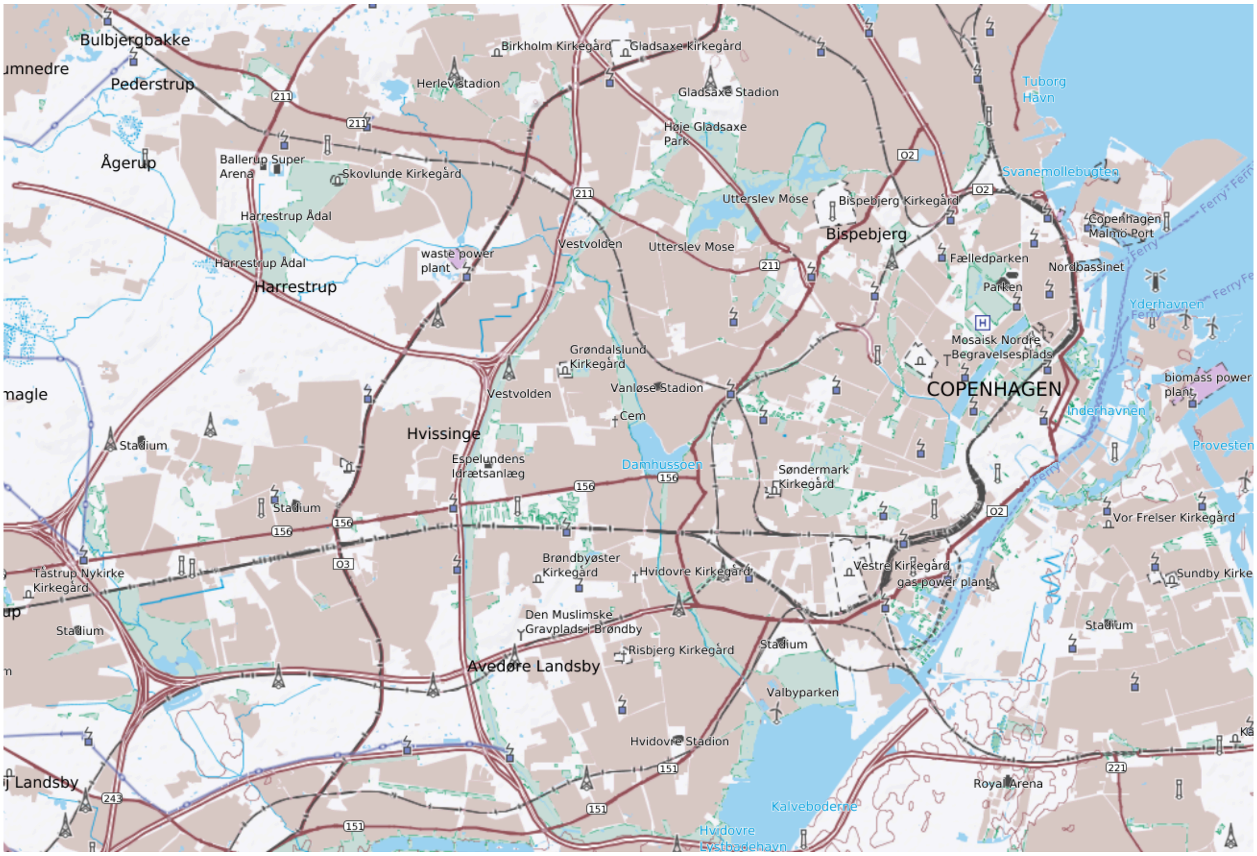


Figure B.42 — GO Mobile Ecere GeoPackage Copenhagan (Low Resolution)

Figure B.43 — GO Mobile Ecere GeoPackage Karistad (Low Resolution)

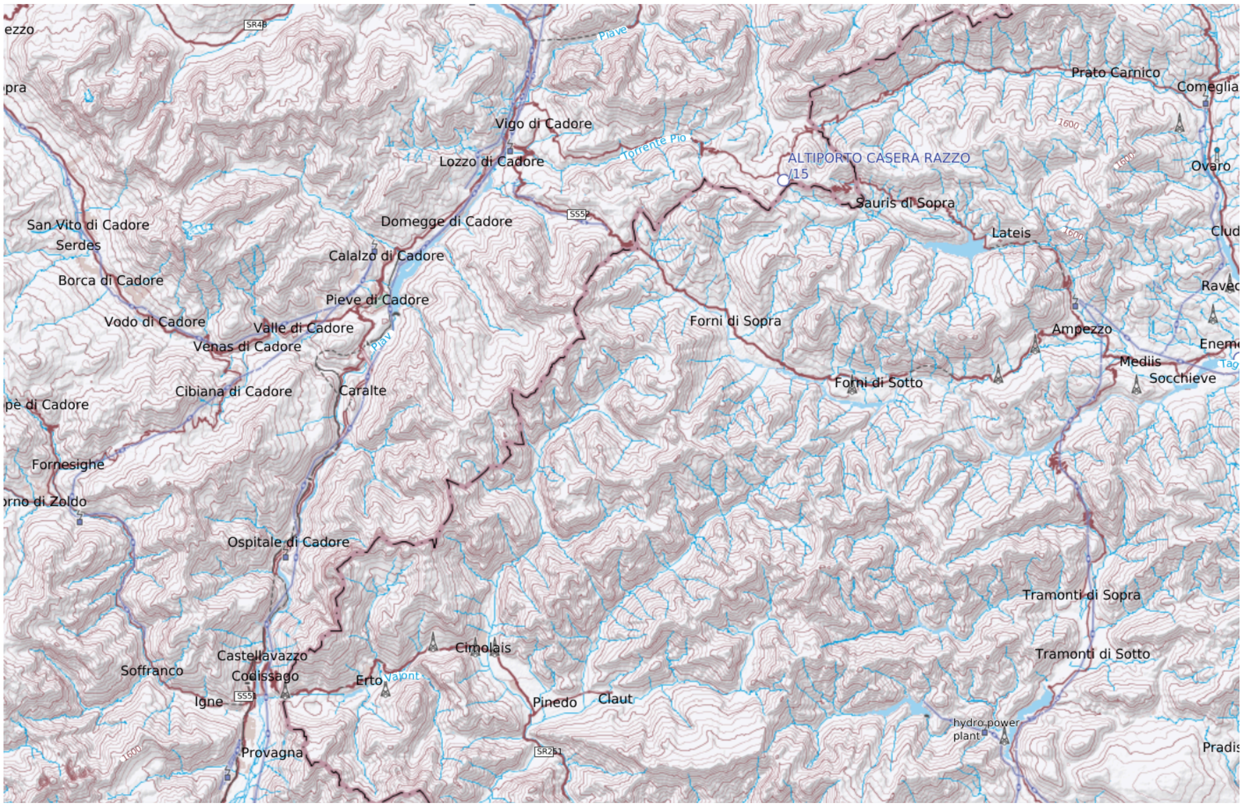


Figure B.44 — GO Mobile Ecere GeoPackage Hillshade

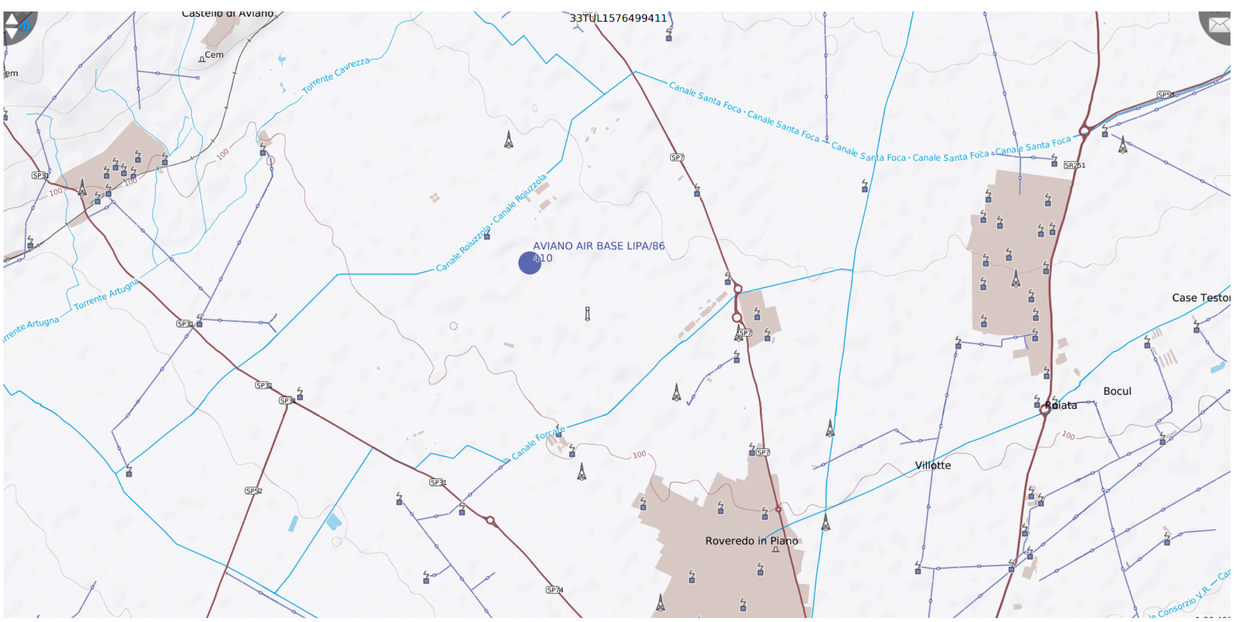


Figure B.45 — GO Mobile Ecere GeoPackage RBT-TPC-3395

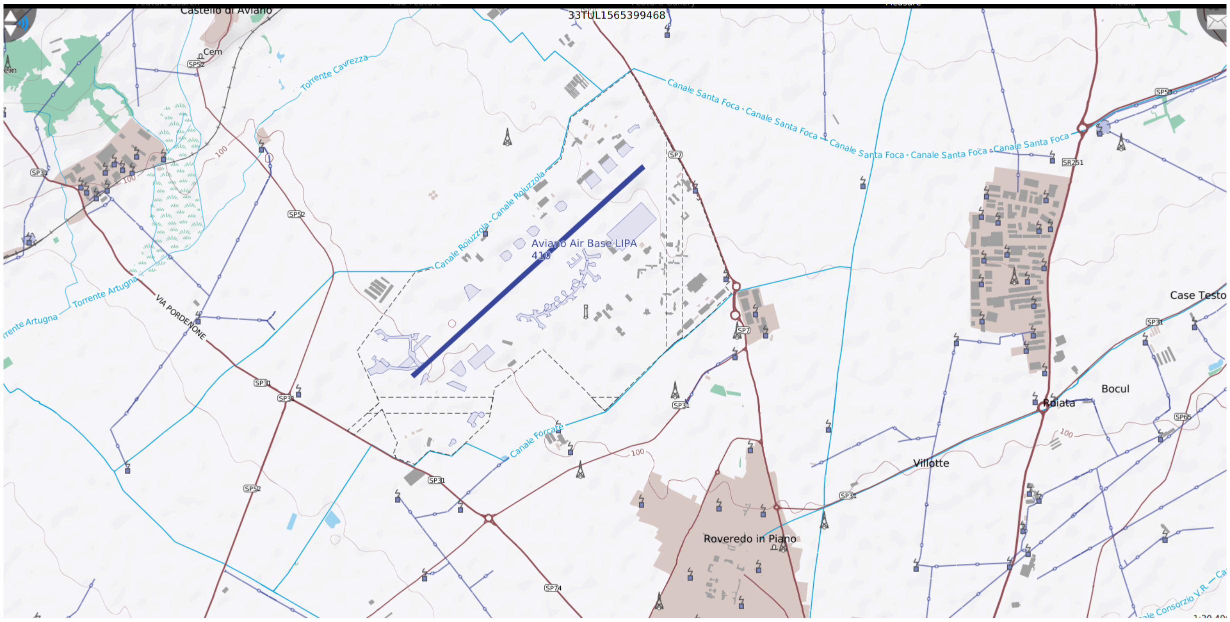


Figure B.46 — GO Mobile Ecere GeoPackage RBT-TLM-3395

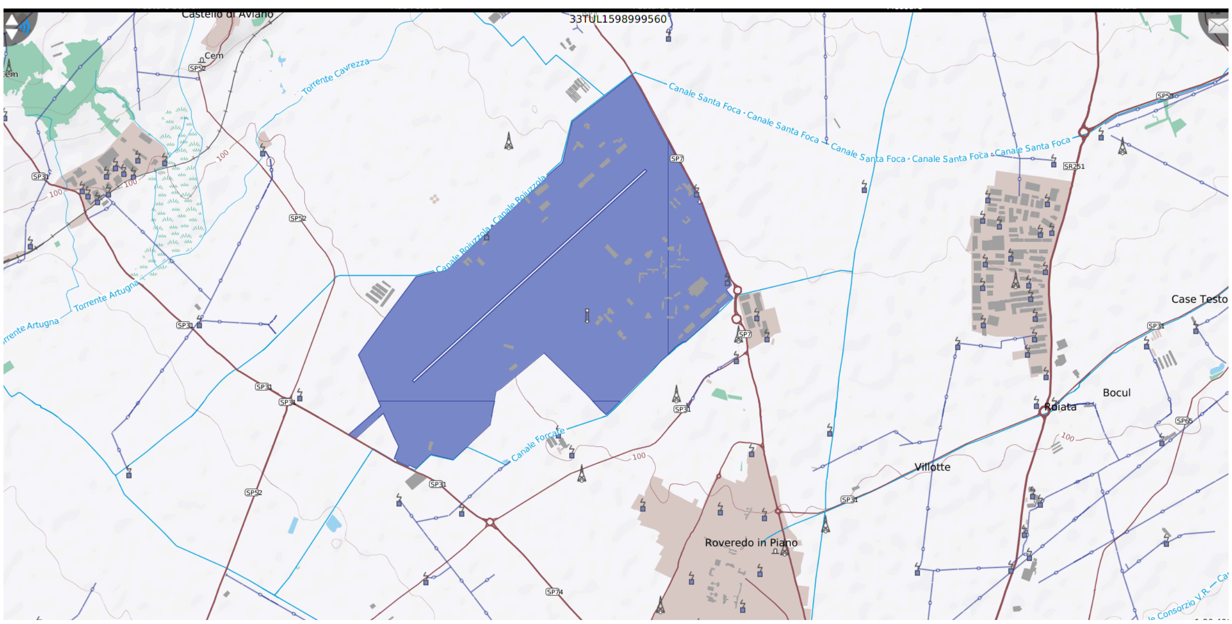


Figure B.47 — GO Mobile Ecere GeoPackage RBT-JOG-3395

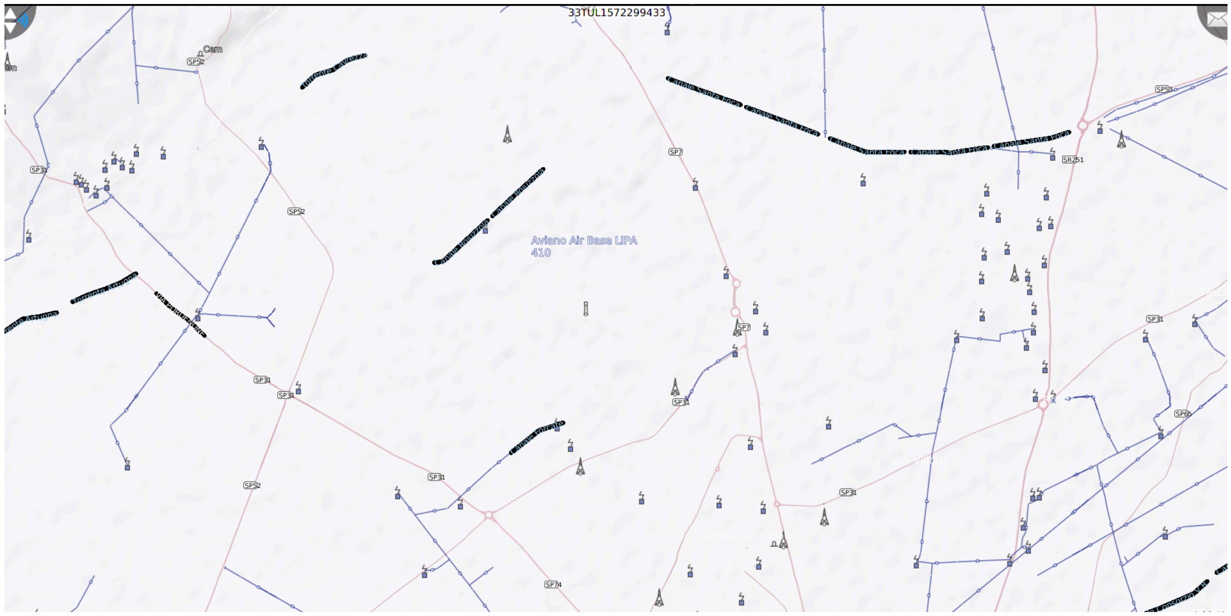


Figure B.48 — GO Mobile Ecere GeoPackage RBT-TLM-SATELLITE-3395

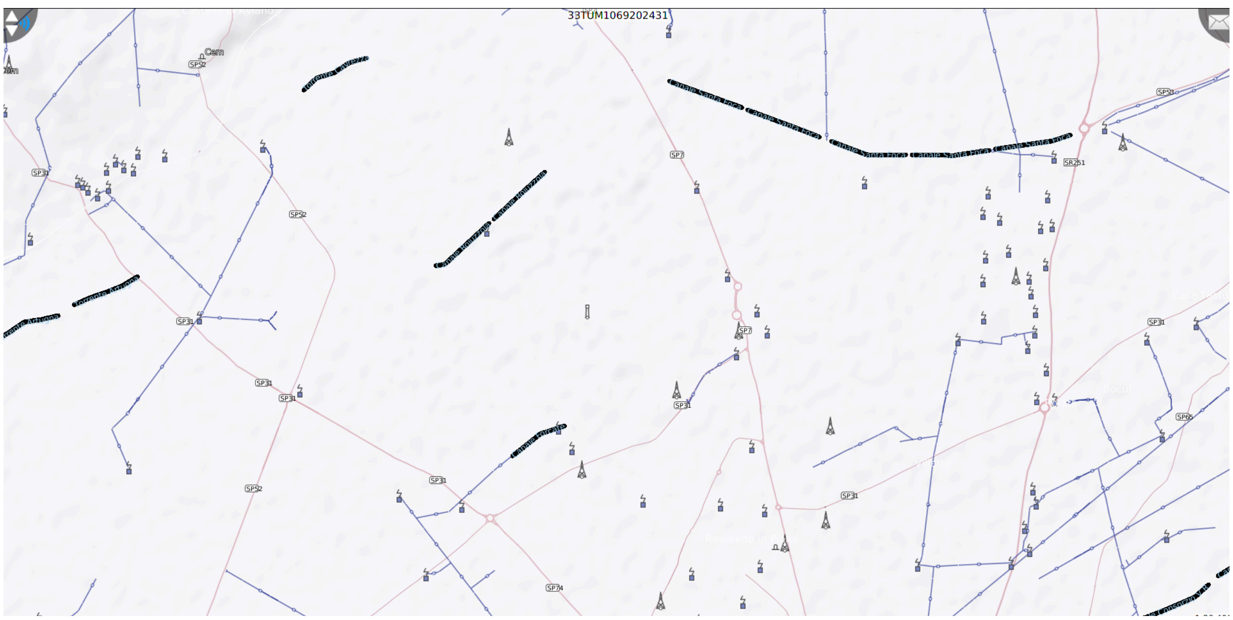


Figure B.49 — GO Mobile Ecere GeoPackage RBT-JOG-SATELLITE-3395

B.1.4.2. Tech Maven Geospatial

The Compusult GO Mobile software was able to visualize and retrieve feature information from the Europe GeoPackage produced by Tech Maven Geospatial without modification. Below are some screenshots of different areas and resolutions using the RBT-TPC-3395 stylesheet.

NOTE: The Tech Maven Geospatial's GeoPackage `gpkgext_stylesheets` values reference a sprite endpoint that is not found in `gpkgext_symbol_content` uri field. The sprite endpoint was also not accessible online so no sprites were able to be retrieved for visualization.

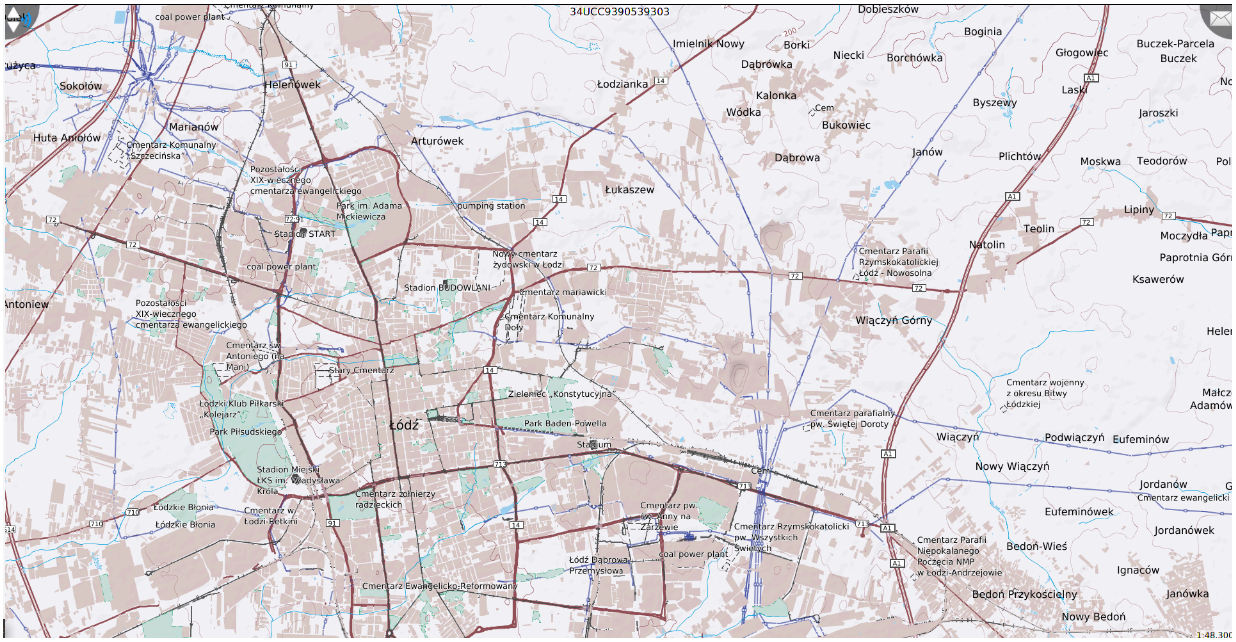


Figure B.50 — GO Mobile TechMaven GeoPackage RBT-TPC-3395

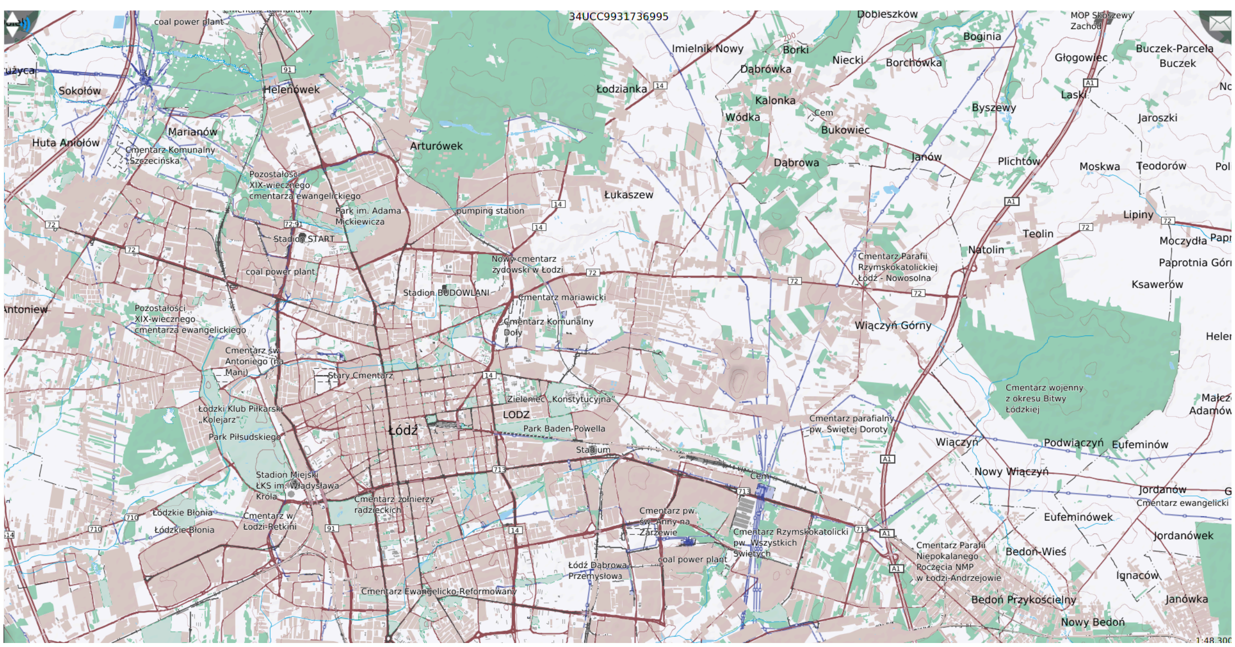


Figure B.51 — GO Mobile TechMaven GeoPackage RBT-TLM-3395

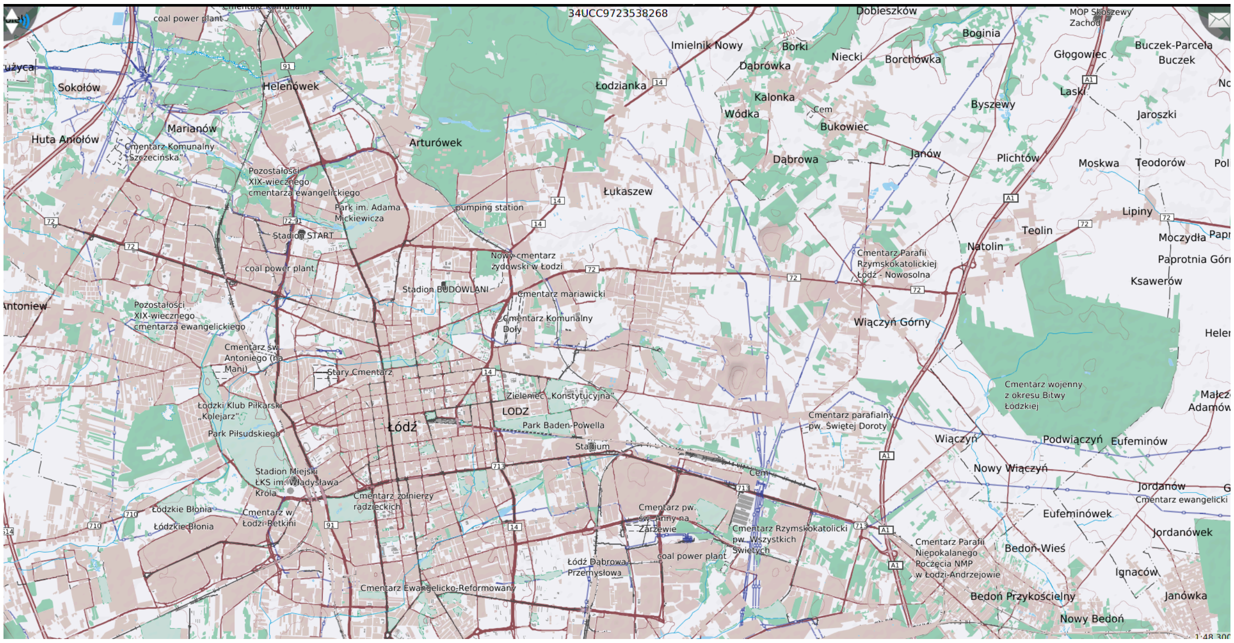


Figure B.52 — GO Mobile TechMaven GeoPackage RBT-JOG-3395

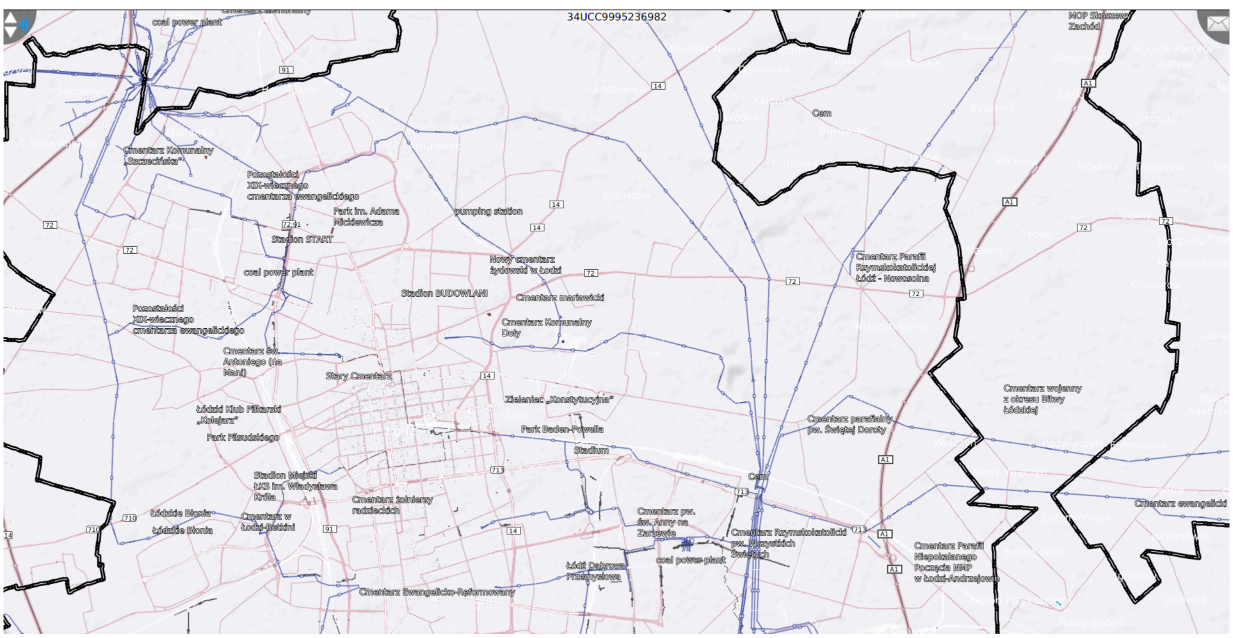


Figure B.53 — GO Mobile TechMaven GeoPackage RBT-TLM-SATELLITE-3395

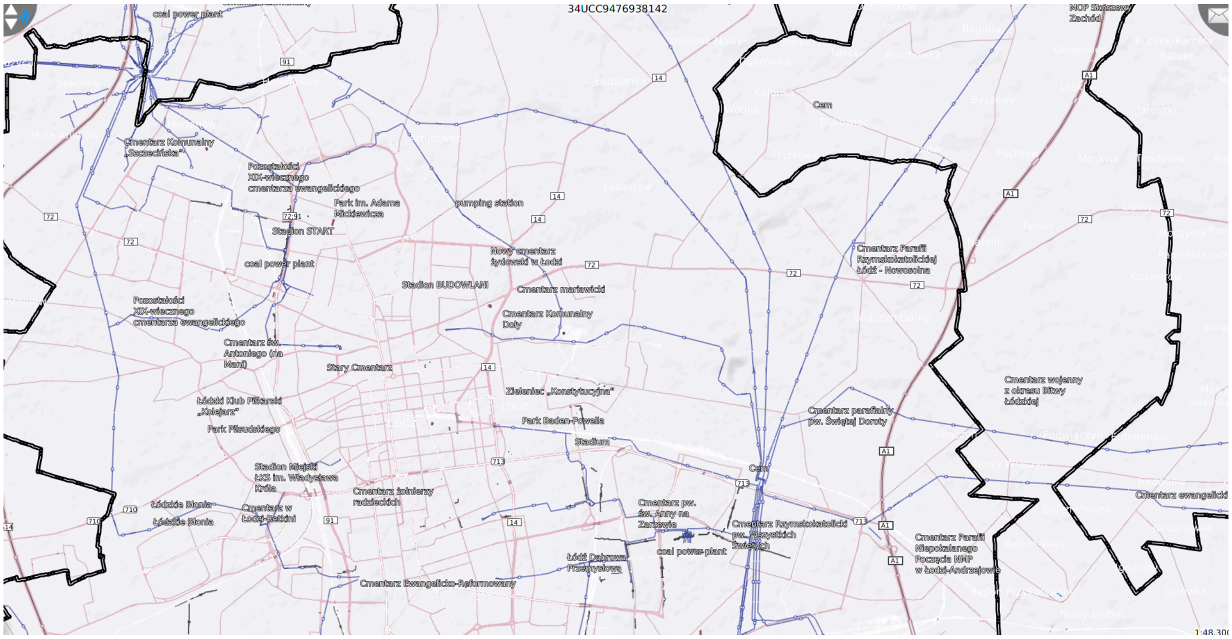


Figure B.54 — GO Mobile TechMaven GeoPackage RBT-JOG-SATELLITE-3395

B.2. Tech Maven Geospatial

B.2.1. About

Tech Maven Geospatial is a small geospatial data and development services firm based in South Florida, USA. The firm specializes in offline degraded, intermittent, and low-bandwidth (DDIL) environment workflows, and have been heavily involved in the use of the GeoPackage format. Tech Maven Geospatial has implemented support for RBT GeoPackages in many of their solutions, including support for both producing and visualizing content.

B.2.2. RBT GeoPackage Producer

Tech Maven Geospatial has developed a custom .NET console application, and enhanced its open source python package for building GeoPackages to support producing RBT GeoPackages.

B.2.2.1. Python RBT GeoPackage producer

The Python RBT GeoPackage producer tool consists of the main *tiles_to_rbt_gpkg.py* script, and can be obtained from its GitHub repository.

The content can be sourced from either MBTiles, or a directory of Mapbox Vector Tiles and/or PNG/JPEG raster tiles. The application also supports including styles and associated resources.

Tech Maven Geospatial produced the following RBT GeoPackages using this tool and compared them against the GeoPackages produced by other participants to validate them.

- Combined RBT GeoPackage
- RBT GeoPackage for Slovakia
- RBT GeoPackage for Poland
- RBT GeoPackage for Hungary

The command-line tool supports the following options:

- `-h, --help`: Show this help message and exit
- `-i INPUT, --input=INPUT`: Path of input MBTiles
- `-o OUTPUT, --output=OUTPUT`: Path for output GeoPackage
- `-p PROJECTION, --proj=PROJECTION`: Specify the CRS / Common 2D Tile Matrix Set, where the possible values are 3395, 3857, 4326 for the associated EPSG CRS codes
- `-r RESOURCE, --resource=RESOURCE`: Path of the resources directory
- `-t TABLE, --table=TABLE`: Table name in GeoPackage

```
> python tiles_to_rbt_gpkg.py
  --input "input file path of mbtiles"
  --output "output gpkg path"
  --proj 3395
  --resource "Resource folder directory path"
  --table "tbl_cultural"
```

Listing B.3 — Example usage of tiles_to_rbt_gpkg.py

The Python RBT GeoPackage tool depends on the following packages which can be installed via the *pip* utility:

- optparse (`pip install optparse`)
- pyproj (`pip install pyproj`)
- sqlite3 (`pip install sqlite3`)

B.2.2.2. Tile Conversion API

Tech Maven Geospatial also updated its Tile Conversion API, Tile Downloading API and Tile Clipping API to support producing RBT GeoPackages and World Mercator / EPSG:3395 tiles.

These APIs are powered by custom .NET Console applications, and use pygeoapi to support OGC API — Processes, including support for job queuing, orchestration and monitoring.

The API is available from this end-point, using the username `army_geospatial_center` and the password `abcd1234`.

The following steps can be used to produce an RBT GeoPackage including styles, fonts and glyphs from a source Mapbox Vector Tiles layer:

- Navigate to *Package Downloader* from the side menu
- Enter the URL of source Mapbox Vector Tiles layer (z/x/y)
- Select *Function — Download as GeoPackage*
- Enter metadata, description, format, input srs, output srs, min zoom, max zoom and bounding box
- Press *Convert* and a file will be created on the server, which can then be downloaded by the user

The following steps can be used to retrieve an RBT GeoPackage produced from a source MBTiles file, clipped to a region of interest:

- Navigate to *Package Clipper* from the side menu
- Select Input MBTiles file
- Select *Function — Clip as GeoPackage*
- Enter metadata, description, format, input, srs/output, srs, min zoom, max zoom and bounding box
- Press *Convert* and a file will be created on the server, which can then be downloaded by the user

B.2.2.3. Additional data processing and preparation tools

TechMaven Geospatial provides some additional tools for data processing and preparation, including the following:

- Tile Utilities which supports clipping data to an area of interest, converting MBTiles or ESRI Tile Package to GeoPackage,
- Map Tiling which supports converting vector or raster data to GeoPackage or MBTiles,
- Offline Map Data Generator for iOS, Android, Windows — the Windows version has been updated to support GeoPackage output,
- Vector To Raster Tiles allows to render MapboxGL JSON stylesheet referencing a URL for vector tiles or raster tiles to map tiles for a given area of interest. This application supports

iOS, Android, Windows and macOS, and can be paired with Tile Server or Geo Data Server for sourcing data from a local GeoPackage.

B.2.3. RBT GeoPackage Consumer/Viewer

Tech Maven Geospatial enhanced some of its solutions with support for visualizing RBT GeoPackages.

B.2.3.1. Map Discovery (viewer)

The Android version of Map Discovery is based on an Android Webview and the OpenLayers JavaScript Mapping Engine. Map Discovery Android supports working offline in a disconnected environment, and can directly visualize RBT GeoPackages.

A demonstration Android package (APK) is available here.

The following videos demonstrate visualizing an RBT GeoPackage from local offline storage using Map Discovery Android and displaying it in a particular style.

- Map Discovery Android RBT GeoPackage demonstration video #1
- Map Discovery Android RBT GeoPackage demonstration video #2

The Map Discovery solution is also available for the iOS and Windows platforms.

However, the Windows version requires pairing with the Geo Data Server or Tile Server Windows products to render the RBT tiles content.

The following screen captures of Map Discovery Android demonstrates the capability to visualize the combined RBT GeoPackage produced by Tech Maven Geospatial from the MBTiles shared by the US Army Geospatial Center.

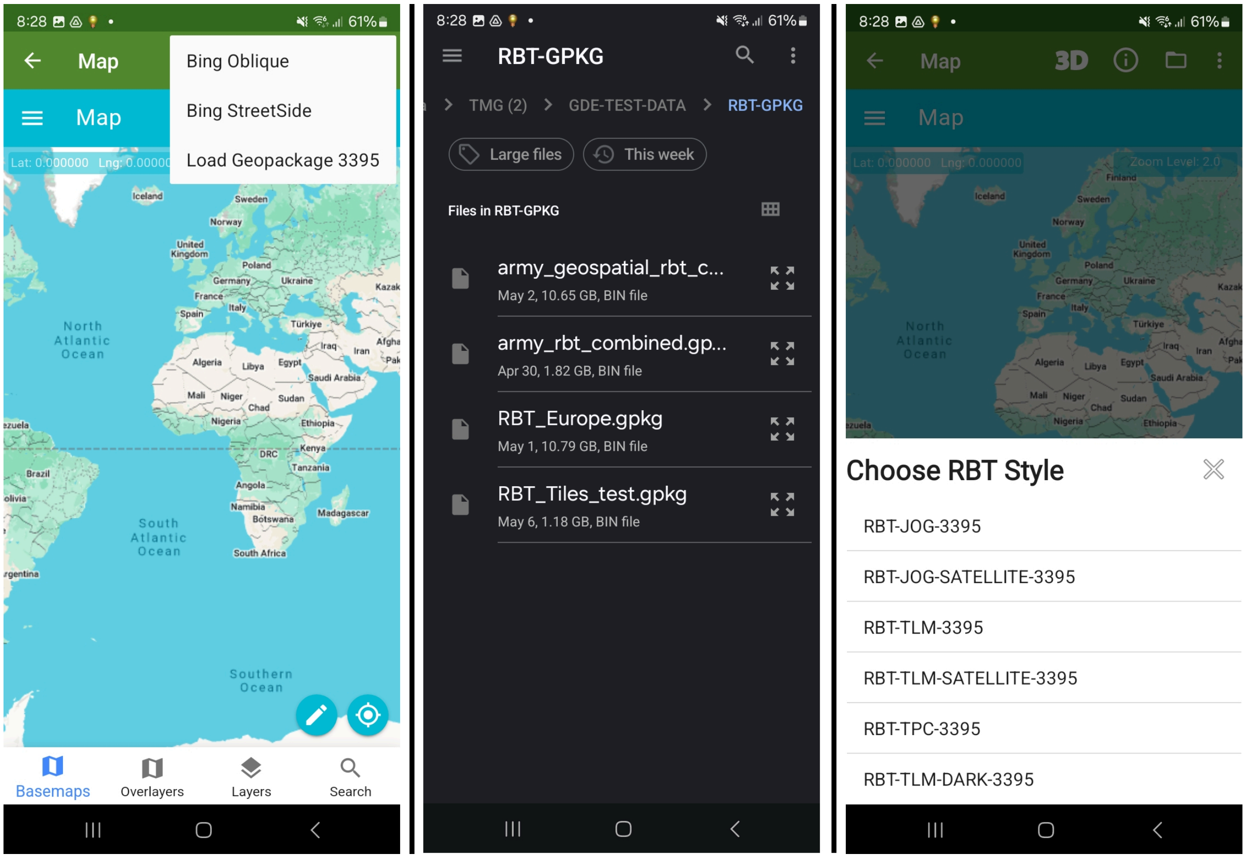


Figure B.55 — Loading an RBT GeoPackage (using World Mercator / EPSG:3395 2D Tile Matrix Set) and selecting a style in Map Discovery Android

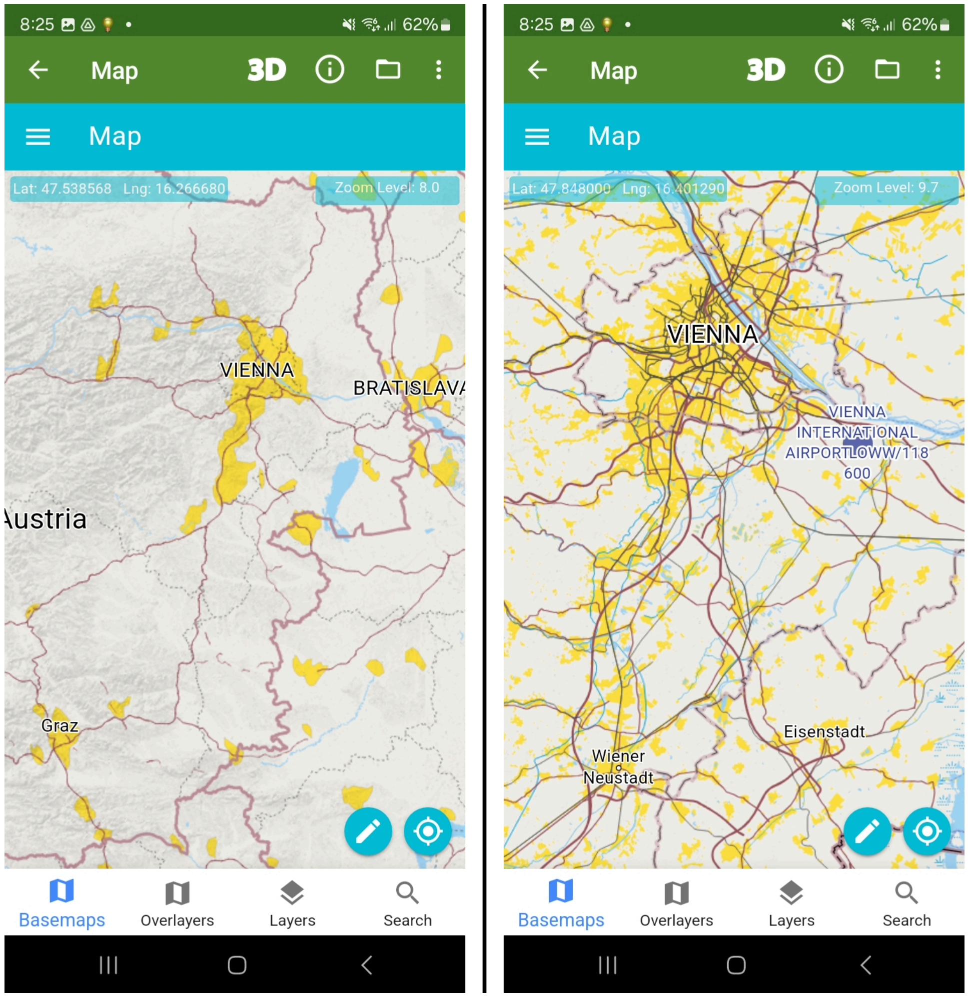


Figure B.56 — Visualizing an RBT GeoPackage in Map Discovery Android (Vienna, Austria)

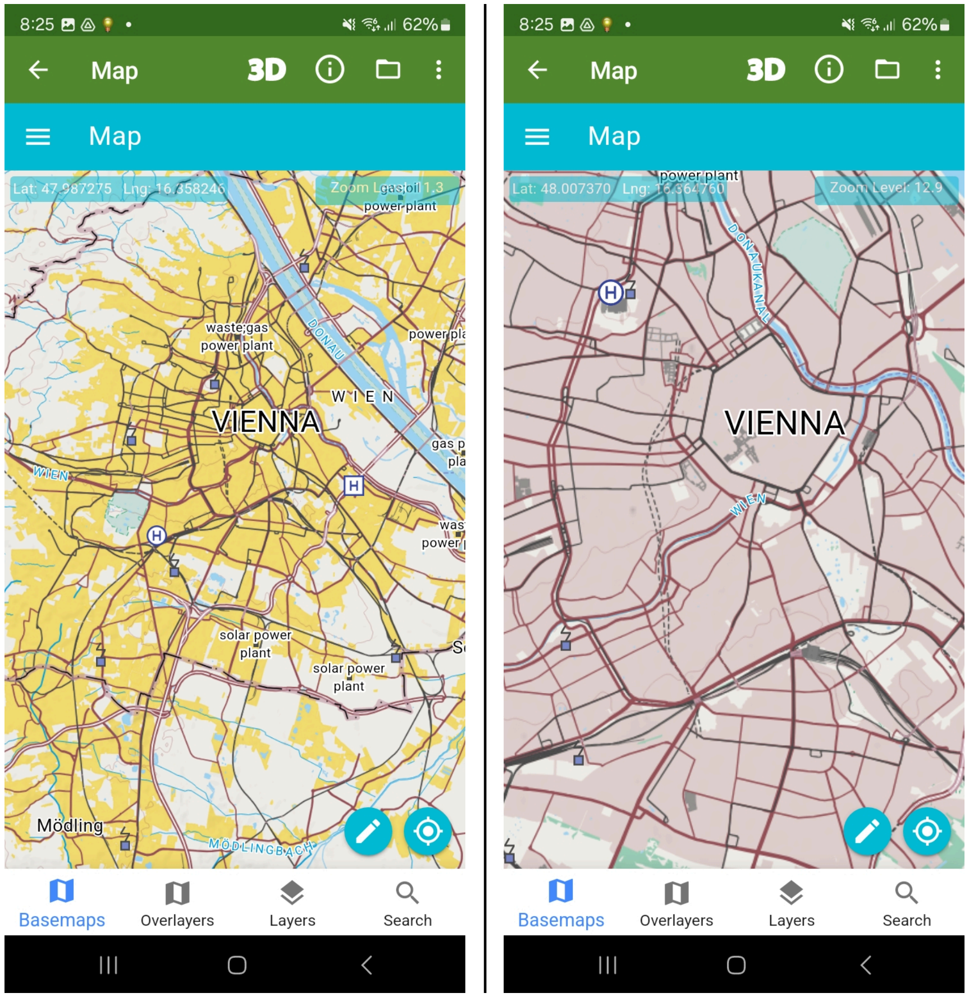


Figure B.57 — Visualizing an RBT GeoPackage in Map Discovery Android (Vienna, zooming in)

The following screen captures of Map Discovery on Windows demonstrates the capability to visualize RBT GeoPackages, when pairing with Tile Server.

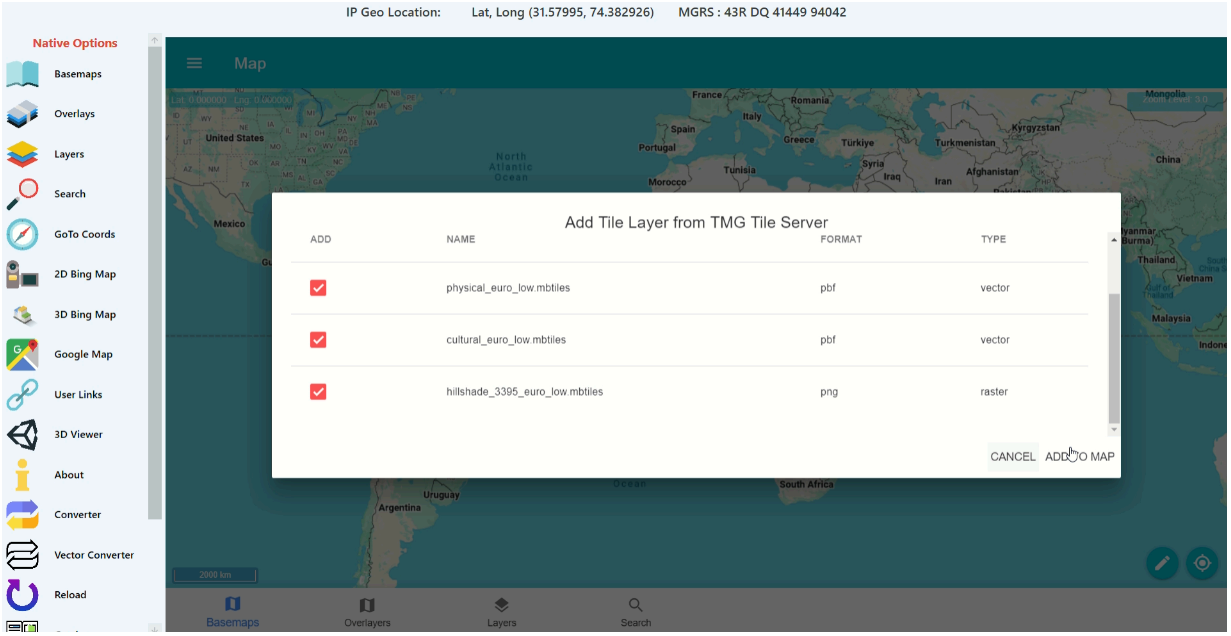


Figure B.58 — Loading RBT content in Map Discovery Windows for visualization

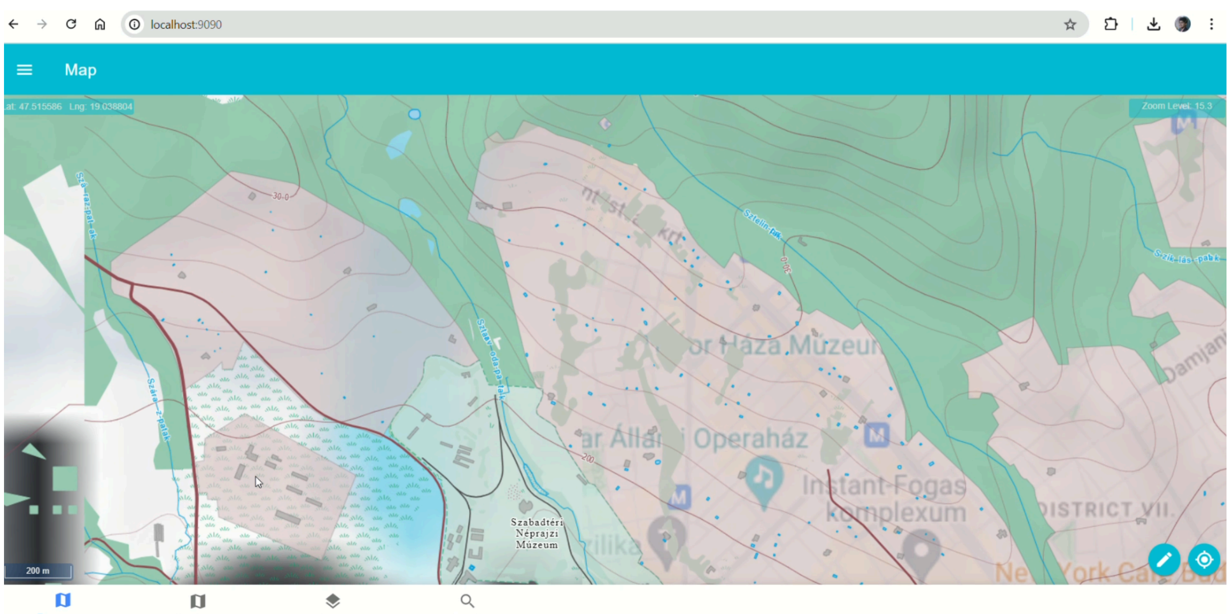


Figure B.59 — Visualizing RBT content in Map Discovery Windows

B.2.3.2. Serving data and map tiles from RBT GeoPackages

Tech Maven Geospatial provides a number of solutions to serve map tiles rendered from source RBT GeoPackages. These solutions can serve data as dynamic Mapbox Vector Tiles, as rendered styled map tiles and through *OGC API — Features*. The data can be sourced from GeoPackage Vector Tiles, GeoPackage Raster Tiles, GeoPackage Gridded Coverage Extension (elevation terrain tiles), or GeoPackage Vector Features Tables. The solutions include:

- Portable Tile Server Android,
- Tile Server Windows, and
- Geo Data Server Windows.

Portable Tile Server Android supports rendering styled map tiles dynamically sourced from GeoPackage vector features. In addition to cached map tiles, Portable Tile Server Android supports returning feature data through *OGC API — Features*.

Tile Server Windows and Portable Tile Server Android can render MapboxGL JSON style sheets referencing vector tiles or raster tiles sources to styled map tiles.

Tile Server Windows supports either dynamic or cached area of interest, while the Android version only supports a cached area of interest.

These solutions also provide a built-in 2D client using OpenLayers available from a `/map` end-point to any end user device via the Web Browser (not as an Android app) and support loading EPSG:3395 Vector Tiles, Raster Tiles and MapboxGL JSON Styles. A 3D Map, which only supports Web Mercator / EPSG:3857 tiles, is also available from `/3dmap`.

B.3. Ecere

B.3.1. About

Ecere is a small software company located in Gatineau, Québec, Canada. Ecere develops the GNOSIS cross-platform suite of geospatial software, including a map server, a Software Development Kit and a 3D visualization client. These tools are all built using Ecere's Free and Open Source eC programming language, 2D/3D graphics engine, cross platform SDK and Integrated Development Environment.

As an OGC member, Ecere actively contributes to several Standard Working Groups and participates in testbeds, pilots and code sprints. Ecere contributed to the initial design of the GeoPackage extensions forming the basis of the specification described in this Engineering Report as a participant in the Vector Tiles Pilots.

Ecere has been a regular contributor and an early implementer for several OGC API standards in its GNOSIS Map Server and GNOSIS Cartographer client. Ecere is also an editor of the OGC CDB 2.0 GeoPackage data store, which relies on the vector tiles extension. In addition, Ecere is leading the development of the OGC Cartographic Symbology 2.0 candidate standard, aiming to achieve portrayal interoperability, and providing the basis for efforts to render the RBT MapboxGL styles in Ecere's GNOSIS visualization engine.

B.3.2. RBT GeoPackage Producer

Ecere enhanced the exporting capabilities of its GNOSIS Cartographer tool for producing conforming RBT GeoPackages. These enhancements included new support for:

- exporting GeoPackages consisting of multiple tile sets (such as the *physical*, *cultural* and *hillshade* RBT tile sets),
- loading Mapbox Vector Tiles and map tiles from an MBTiles input,
- processing the TileJSON from the MBTiles to populate the layers metadata,
- transferring loaded tiles directly to the output GeoPackage,
- optionally compressing vector tiles using the *gzip* encoding,
- the description of media types and encoding in new `gpkgext_content_types` table,
- using the new proposed *GeoDataClass* semantic annotation (a *GeoDataClass* is the updated term for what was previously called a *StylableLayerSet*) to identify content and associate styles,
- including fonts and glyphs in the output GeoPackage.

In addition, several fixes and improvements to the pre-existing support for GeoPackage vector tiles, styles and semantic annotation extensions were made.

Exporting an RBT GeoPackage from GNOSIS Cartographer is performed using the following steps:

1. organize all source MBTiles (physical, cultural and hillshade tile sets) in one directory,
2. create a `.styles/` sub-directory within that MBTiles directory and populate it with the MapboxGL style sheets to include,
3. create a `symbols/` sub-directory within that `.styles/` directory and populate it with the sprite sheets, sprite JSON description and optionally the source SVG of the symbols used by those styles,
4. create a `fonts/` sub-directory within the MBTiles directory and populate it with the TrueType fonts and zipped Mapbox PBF glyphs for the fonts used by those styles,
5. launch GNOSIS Cartographer and add the MBTiles directory as a data source to the Map Library by using the *Add…* button in the bottom-left corner of the application,
6. with the data source selected in the *Map Library*, click the rubber band/cursor icon next to the *Extent* panel above the *Map Library* and select an area of interest

to extract in the map view (skip this step, or click the blue cycling arrows icon reset button, to export the full content),

7. right-click the data source row of the *Map Library* and select the “Export this layer to GeoPackage…” menu item to bring up the GeoPackage Export dialog,

8. select a destination file path for the output RBT GeoPackage,

9. configure the GeoPackage exporting options as such (see picture below):

   - select the *WorldMercatorWGS84Quad Tiling Scheme*,
   - select *Multiple layers per tile*,
   - select *Embed attributes*,
   - select the *png Image Format*,
   - uncheck *Features/Tiles Mapping*,
   - uncheck *R-Tree Spatial Index*,
   - leave *Tile Vector* selected,
   - leave the *mvt Vector format* selected,
   - leave the default *Max Zoom* option selected to export at the maximum available resolution,
   - leave *Include styles* selected,

10. finally, press the *Export* button.

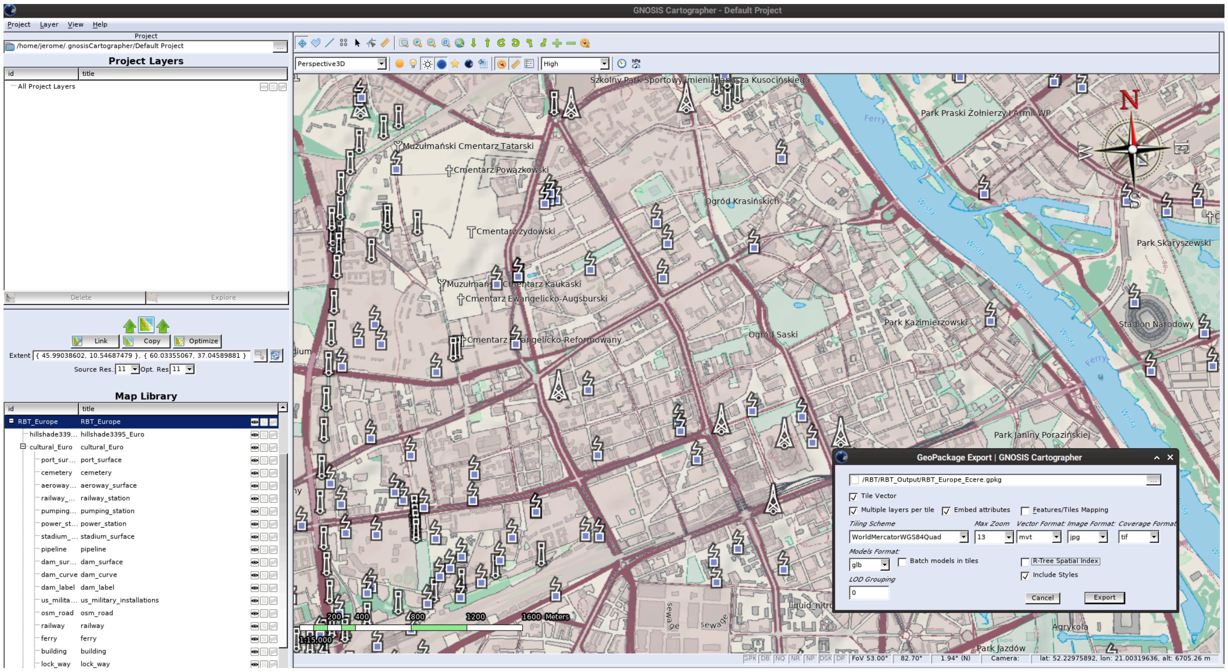


Figure B.60 — Producing an RBT GeoPackage using GNOSIS Cartographer from RBT MBTiles (also showing visualized MBTiles)

Initial efforts to produce an RBT GeoPackage were focused on attempting to process vector tiles through the clipping and generalization pipeline already in place for outputting vector tiles. These efforts ran against several challenges and had to be abandoned in favor of passing through the source vector tiles directly. The main challenge with the original approach was the large amount of data in the source tiles, often more than intended for the tile zoom level. Existing issues with the vector pipeline, some of which were partially addressed during the sprint, also contributed to these difficulties. Once these issues are fully resolved, a future version will likely support optionally applying this processing pipeline, which would allow correcting both the improper clipping and the insufficient generalization of the source RBT MBTiles content, at the cost of a longer processing time to generate the RBT GeoPackage.

Ecere also planned to develop a command-line tool to produce RBT GeoPackage, but this was not yet done by the end of the sprint due to limited time.

Ecere provided an RBT GeoPackage, based on the source MBTiles provided by the US Army Geospatial Center (AGC) for a subset of Europe, for other participants to attempt visualization Technology Integration Experiments.

B.3.3. RBT GeoPackage Consumer/Viewer

Ecere enhanced the GNOSIS SDK with the ability to access content from RBT GeoPackages. The ability to access vector tiles encoded using the *gzip* encoding was the main new capability to be introduced. In addition, some fixes and improvements were made, as well as updates to reflect the minor changes to the extensions agreed upon during the sprint.

Following these enhancements, RBT GeoPackages can then be visualized in GNOSIS Cartographer, and can also be used as data sources to publish them using GNOSIS Map Server, including as vector tiles, rendered styled map tiles or vector features. Additionally, GNOSIS Cartographer is able to visualize the RBT content directly from the MBTiles, although the lack of proper tileset metadata makes it ambiguous whether tile sets should be interpreted as using a Web Mercator (EPSG:3857) or World Mercator (EPSG:3395) 2D Tile Matrix Set.

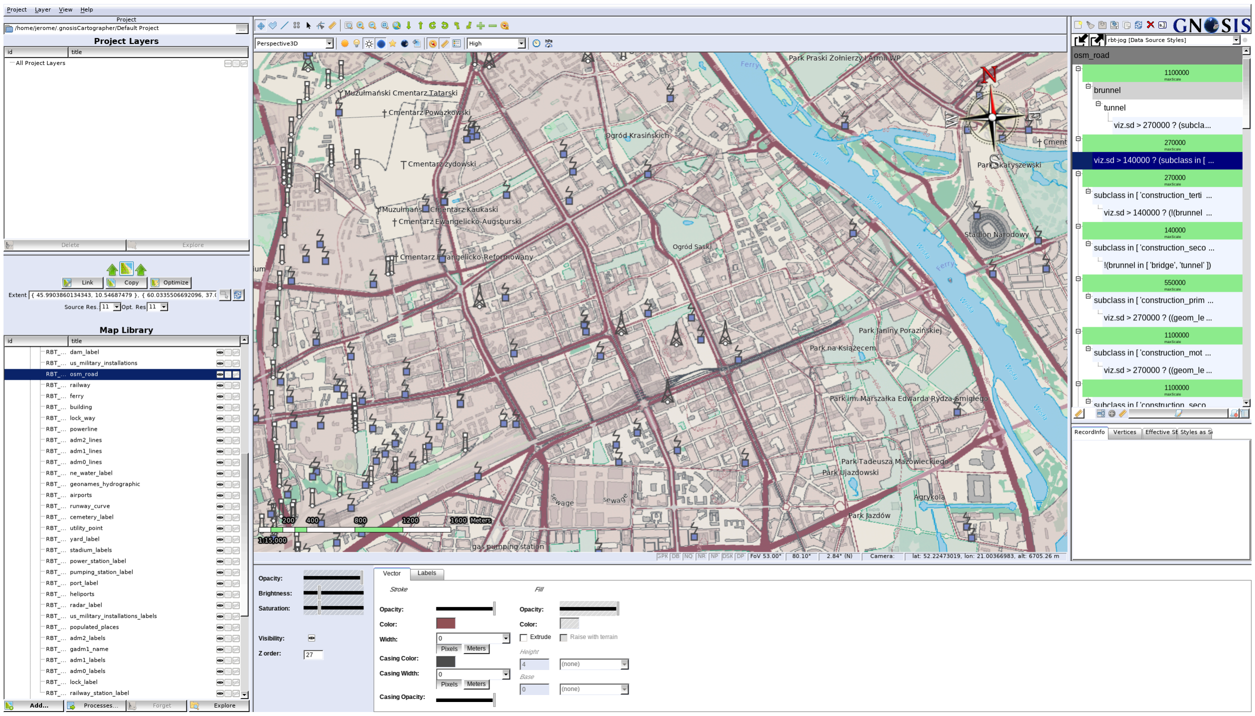


Figure B.61 — Visualizing the RBT GeoPackage produced using GNOSIS Cartographer

The GNOSIS SDK supports the development of visualization applications which can be deployed across multiple platforms, including Windows, Linux, Android, WebAssembly as well as Virtual and Augmented Reality headsets.

GNOSIS Map Server is a certified implementation of OGC API — Tiles, OGC API — Features and OGC API — Processes, and implements support for several other OGC API standards, including OGC API — Maps, OGC API — Styles, OGC API — Coverages and OGC API — DGGS. Ecere provides a stable persistent demonstration end-point for its GNOSIS Map Server, which is also a reference implementation for OGC API — Tiles and OGC API — Processes.

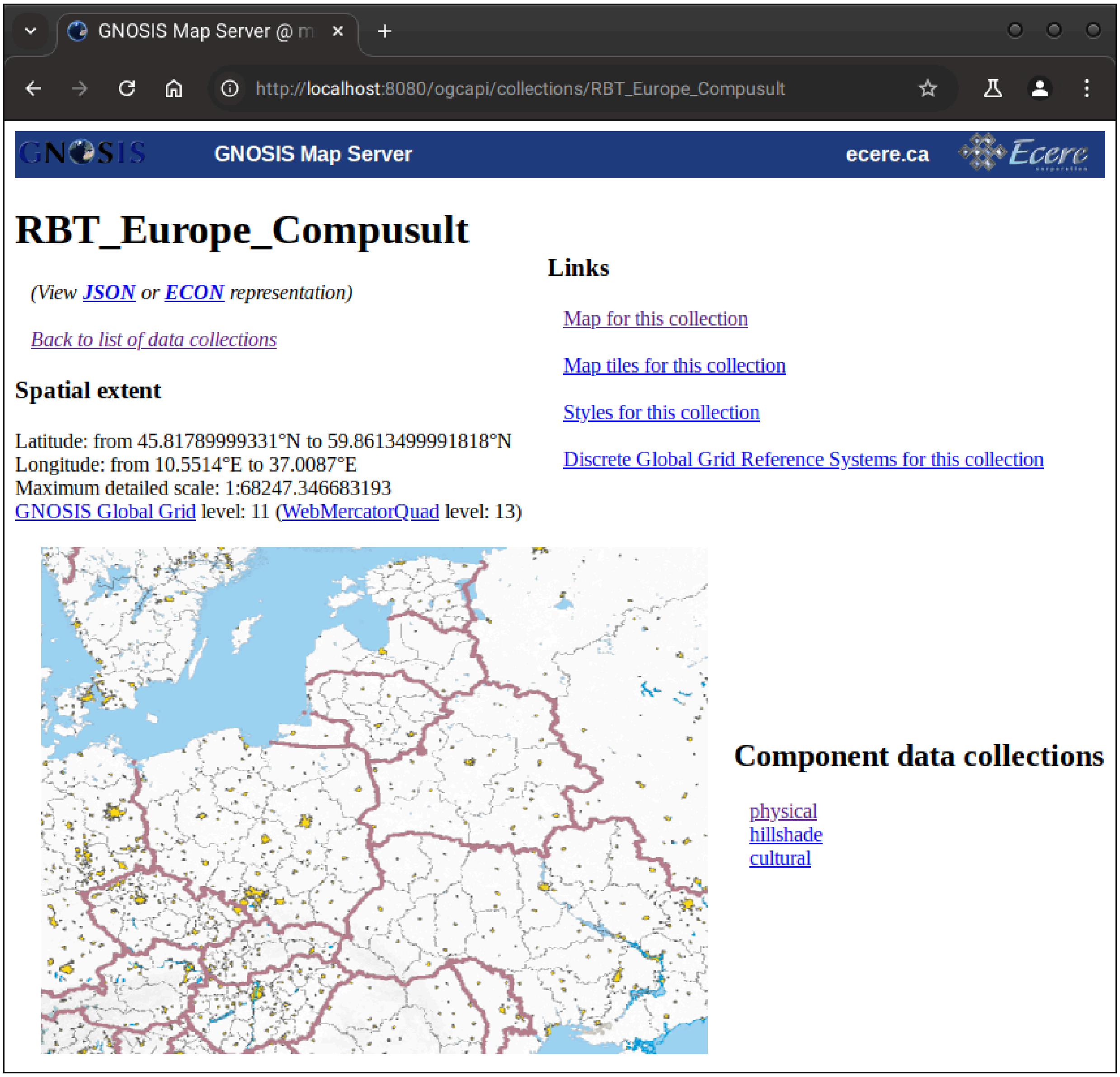


Figure B.62 — Serving data from RBT GeoPackage data source through OGC APIs using GNOSIS Map Server

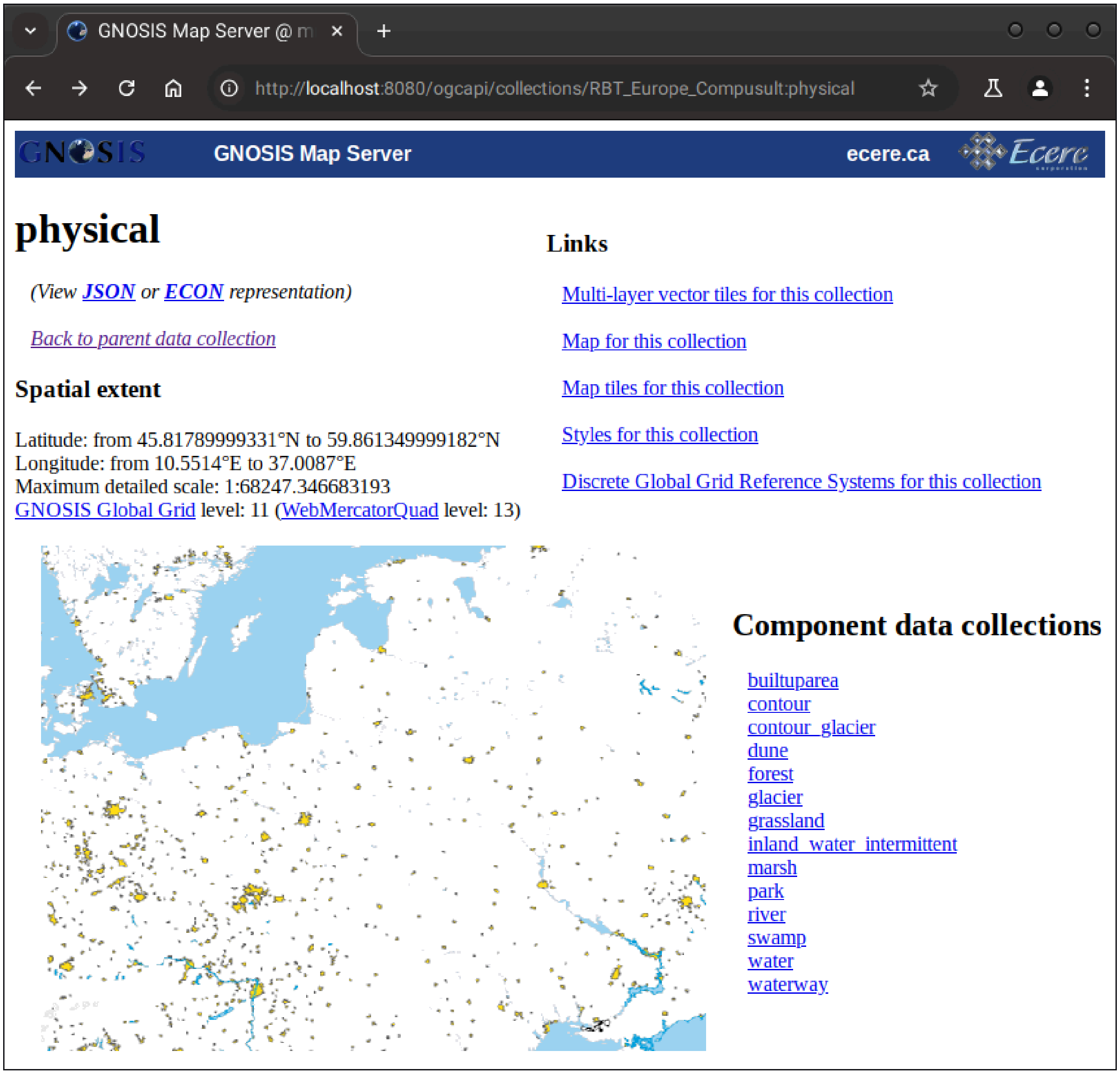


Figure B.63 — Serving the physical features tileset from RBT GeoPackage data source through OGC APIs using GNOSIS Map Server

Ecere spent major efforts attempting to improve its ability to portray the tilesets using styles described by the Mapbox GL style specification. The styles provided by AGC's RBT development team were considerably more complex than the styles used in the previous Vector Tiles Pilot initatives, making extensive use of advanced expressions for most symbolizer properties. These portrayal efforts were rooted in mapping the styling capabilities of MapboxGL styles to the conceptual and logical models defined in the draft OGC Cartographic Symbology 2.0 standard. Considerable work still remained to perfect this mapping to the extent possible, as well as to implement some rendering capabilities not yet available in the GNOSIS rendering engine.

Significant efforts were also spent attempting to improve performance accessing and displaying the RBT GeoPackage content. The initial performance was problematic, as a result of multiple reasons:

- the large number of layers in the RBT tilesets,
- the insufficient generalization of the vector data where more vertices are present than can be noticed visually,
- the complex symbolization rules and large number of data attributes involved,
- the additional processing applied in the GNOSIS engine to remap rendered content to its native GNOSIS Global Grid, which supports 3D visualization on a virtual globe, mixing tiles of multiple zoom levels in a single view.

Several important improvements were made to partially address how these factors negatively impact performance, but efforts were still ongoing at the end of the (extended) sprint to achieve satisfactory performance visualizing the RBT GeoPackages directly.

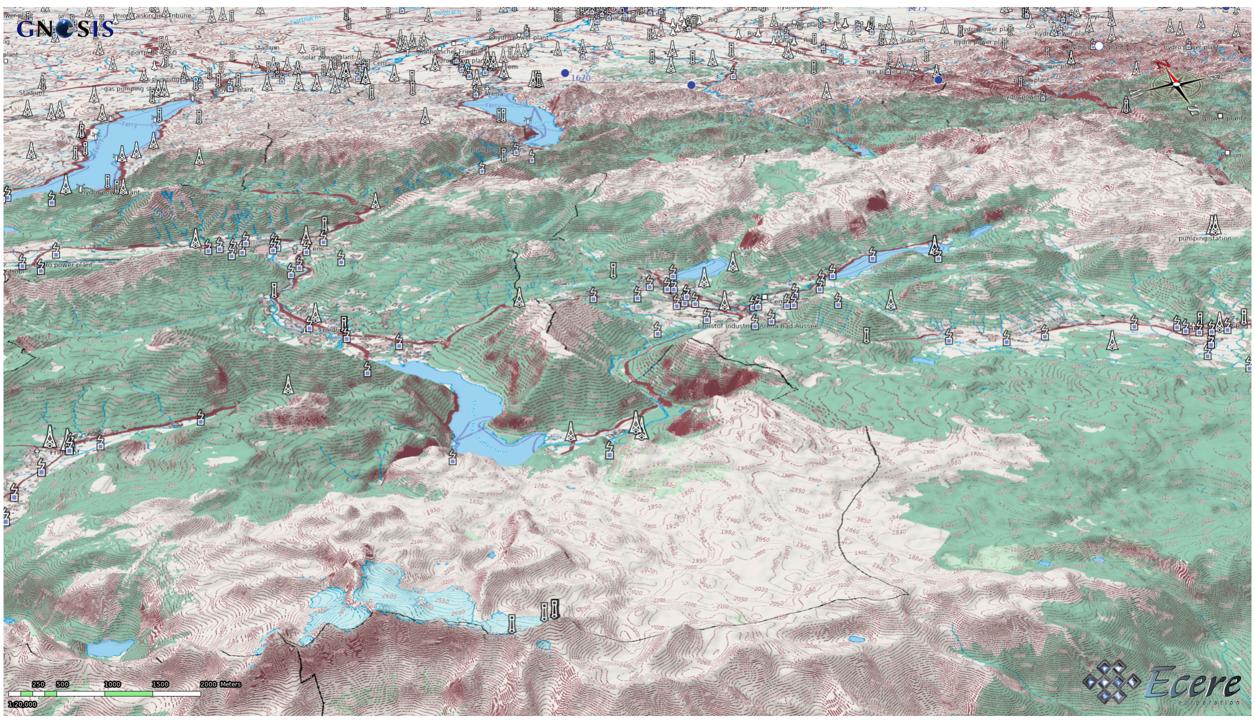


Figure B.64 — Visualizing the RBT GeoPackage produced by Compusult in a 3D perspective view (using Jonathan de Ferranti's Viewfinder Panoramas as a digital elevation model source for the 3D terrain)

B.3.4. Technology Integration Experiments

Ecere performed Technology Integration Experiments using its GNOSIS Cartographer product, attempting to visualize the GeoPackages produced by the tools of the other sprint participants.

B.3.4.1. Compusult

Ecere was able to visualize the RBT GeoPackage of Europe provided by Compusult in GNOSIS Cartographer, as demonstrated in the screenshots below.

Figure B.65 — Visualizing the RBT GeoPackage produced by Compusult (Berlin)

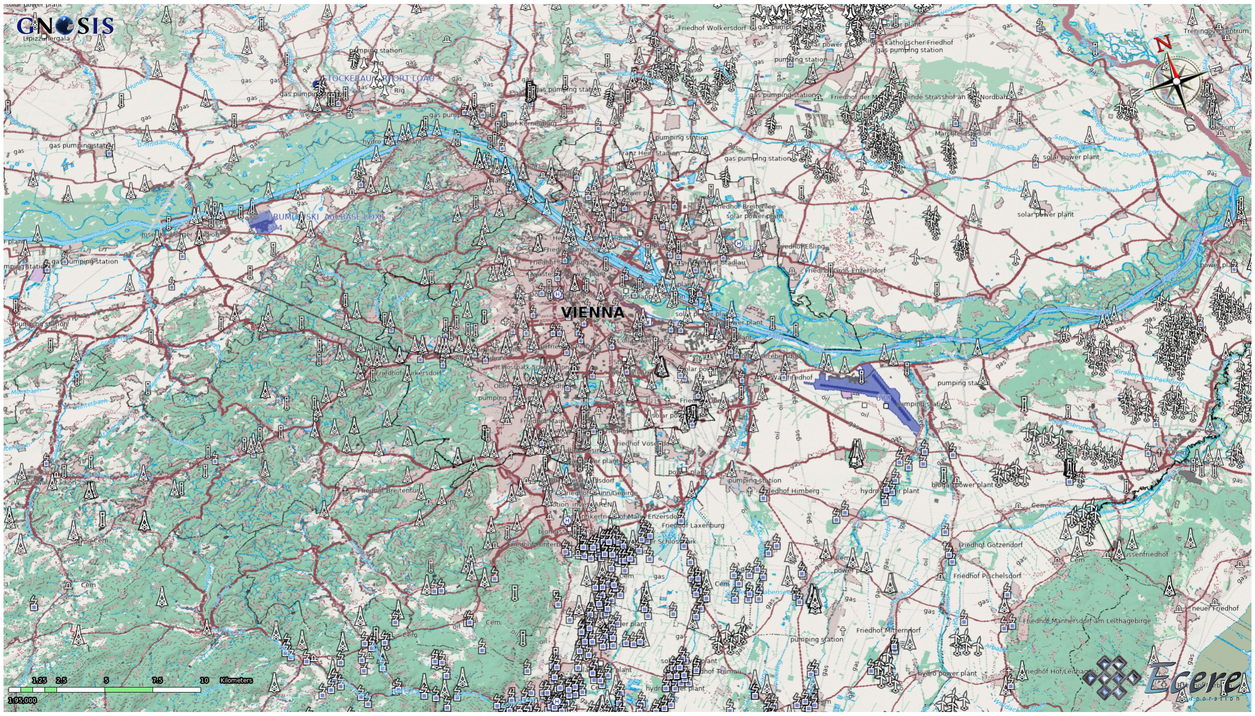


Figure B.66 — Visualizing the RBT GeoPackage produced by Compusult (Vienna)

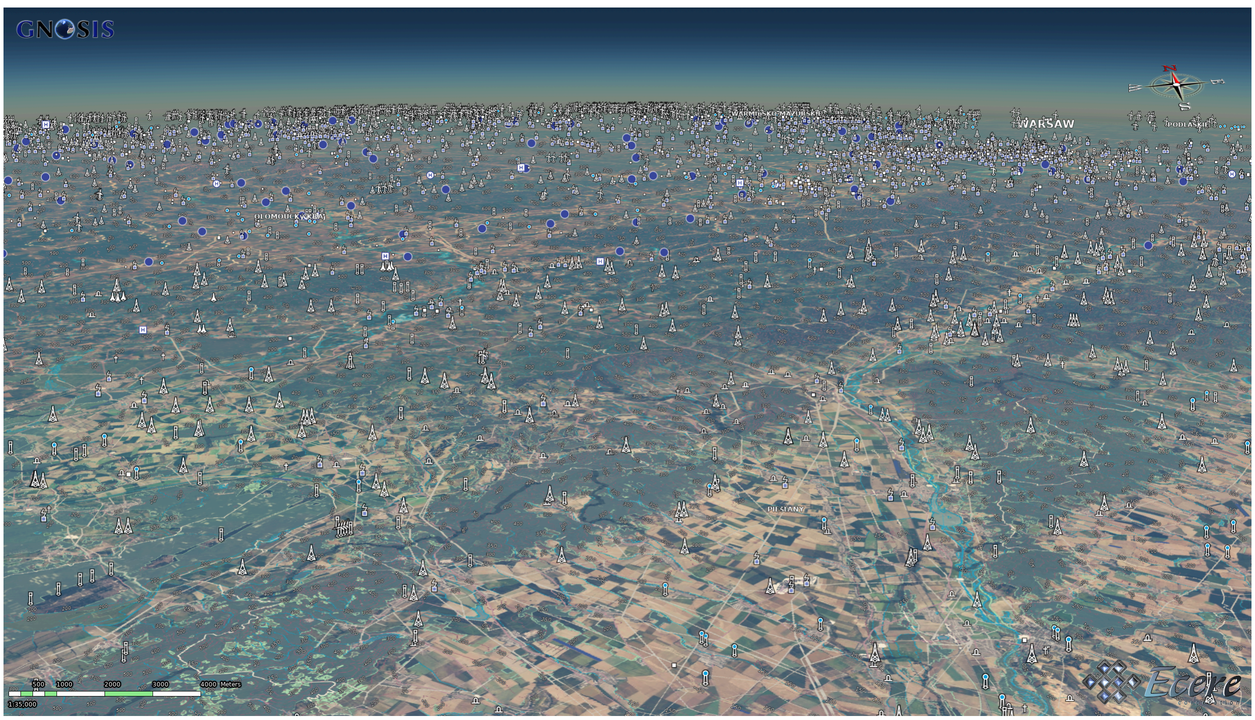


Figure B.67 — Visualizing the RBT GeoPackage produced by Compusult (Satellite overlay style)

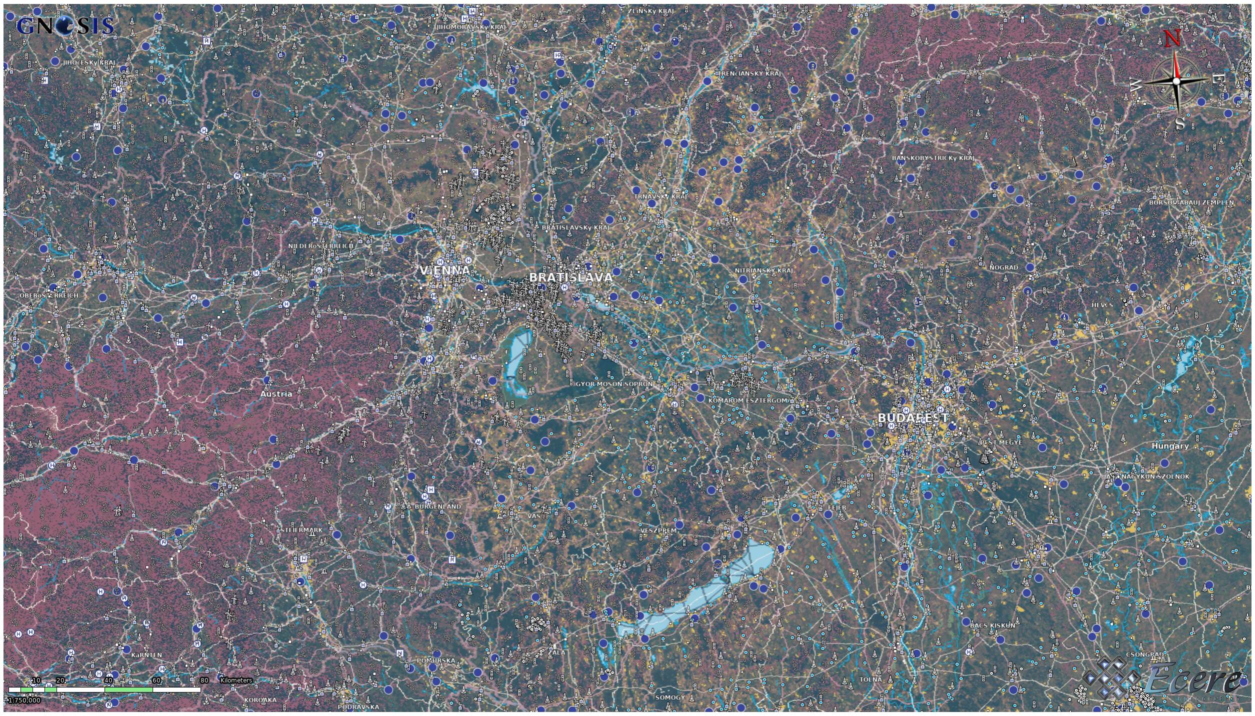


Figure B.68 — Visualizing the RBT GeoPackage produced by Compusult (Satellite overlay style, top-down)

Figure B.69 — Visualizing the RBT GeoPackage produced by Compusult (View of Europe)

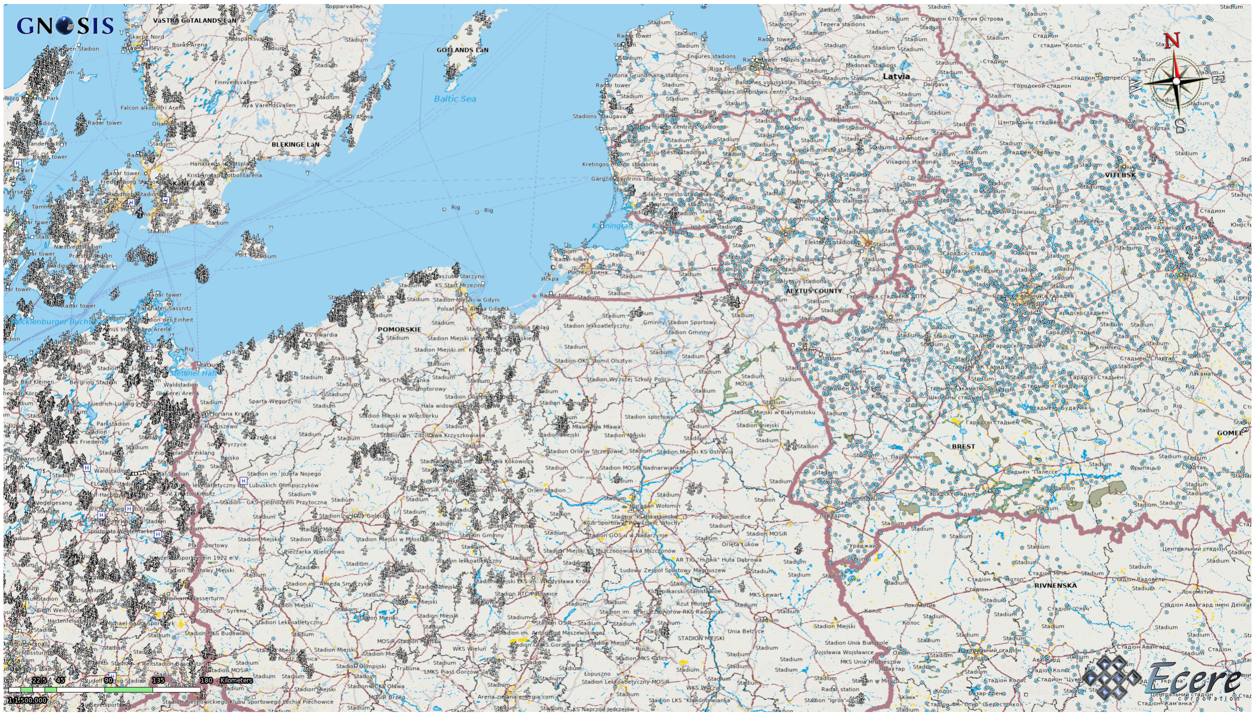


Figure B.70 — Visualizing the RBT GeoPackage produced by Compusult (View of Northern Europe)

Figure B.71 — Visualizing the RBT GeoPackage produced by Compusult (Contours)

Figure B.72 — Visualizing the RBT GeoPackage
produced by Compusult (View of Europe in Dark style)

Figure B.73 — Visualizing the RBT GeoPackage produced by
Compusult (Horizontal view in Dark style, Gaia Sky in Colour from ESA)

Figure B.74 — Visualizing the RBT GeoPackage produced by Compusult (Zoomed in view in Dark style)

B.3.4.2. Tech Maven Geospatial

Ecere was able to visualize the RBT GeoPackage of Europe provided by Tech Maven Geospatial in GNOSIS Cartographer, as demonstrated in the screenshot below.

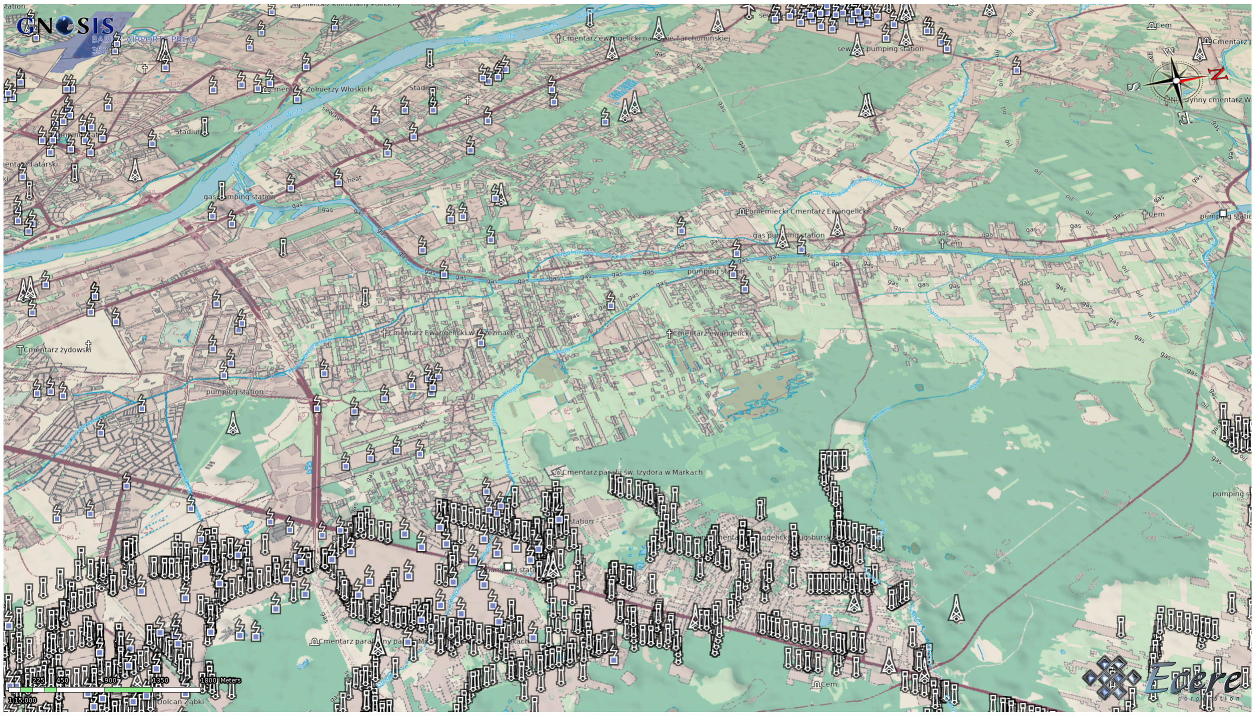


Figure B.75 — Visualizing the RBT GeoPackage produced by Tech Maven Geospatial

B.4. Challenges and Lessons learned

Some challenges were encountered by participants developing implementations producing and consuming RBT GeoPackages during this initiative.

B.4.1. Incorrect bounding boxes

One of these challenges was confusion regarding the bounding box of the data in the source MBTiles provided by AGC. The bounding box communicated, as well as the bounding box indicated within the source MBTiles metadata, did not reflect the actual limit of the content of the Mapbox Vector Tiles at the highest resolution (WorldMercatorWGS84Quad zoom level 13). This was due to some confusion using tools potentially dealing strictly with WebMercatorQuad (EPSG:3857) tilesets for clipping the source WorldMercatorWGS84Quad tilesets. Part of the confusion related to CRS units. Whereas the EPSG:3395 CRS uses coordinates in meters, the EPSG:4326 CRS uses latitude and longitude coordinates in degrees. In order to work around these challenges, participants tried to identify the actual bounding box of the source Europe MBTiles.

In addition, the MBTiles partial extract for Europe did not actually clip the data within the individual tiles at the clipping boundary, but either kept or discarded whole tiles. As a result, the RBT GeoPackages produced from these source tiles do not have a consistent boundary at different zoom levels, unless additional clipping is performed during the conversion to GeoPackage.

B.4.2. Insufficient generalization

Another challenge was the density of vector geometry, particularly at the lower zoom levels. The generalization process used for producing the source MBTiles resulted in tiles with more vertices than are noticeable at the zoom levels for which they are intended. This was particularly problematic in Ecere's viewer implementation, which performs a number of vector operations in realtime to reconstruct the geometry and re-tiling it to its native 2D Tile Matrix Set for the purpose of rendering the data on a 3D virtual globe.

Achieving performance with the large number of layers, overly dense geometry as well as the complex symbology rules defined in the style sheets posed some challenges, providing an opportunity to optimize visualization engines.

B.4.3. Lack of unique feature IDs

The Mapbox Vector Tiles in the source MBTiles did not make use of the optional unique feature identifiers field. Traditional geospatial engines rely heavily on the concept of unique feature identifiers. This allows for example to recombine the features spread across multiple tiles. This approach also supports the option of storing attributes in features table and associating the vector geometry in the Mapbox Vector Tiles with those attributes, which can result in significant storage space savings. As a work around for the lack of feature IDs present in the Mapbox Vector Tiles, the GNOSIS engine had to generate such IDs on-the-fly by hashing feature attributes. This introduces complications and potential for hash collisions, which are difficult to handle correctly. Generating unique feature identifiers and including them in the Mapbox Vector Tiles when generating the content is likely simpler, since such feature ID may already exist in the source data. However, care should be taken when exporting content from OpenStreetMap, as the the OSM identifiers are unique only among a particular entity type (nodes, ways and relations). The concept of an OSM permanent ID has also been proposed which would provide a more suitable identifier for this purpose.

B.4.4. Portrayal challenges

Associating data layers, styles and symbols encoded within sprite sheets with multiple symbols was a complex topic which would benefit from additional experiments, as described in the future work section.

Mapping the symbology rules of MaboxGL style sheets to the internal symbology rules of custom visualizing engines was also particularly challenging. Additional work on improving this mapping could also be part of future efforts focused on portraying RBT content as accurately as possible.

C

ANNEX C (INFORMATIVE) FUTURE WORK

C

ANNEX C (INFORMATIVE) FUTURE WORK

Due to the short duration for this RBT GeoPackage Sprint initiative and the inherent complexity of the topics covered, additional work efforts would be greatly beneficial for the purpose of finalizing OGC GeoPackage extension standards, an NSG profile of these standards, and improving the producer and consumer implementations.

C.1. OGC GeoPackage extensions

Although the draft specification in this Engineering Report is presented as a single GeoPackage extension, stakeholders agreed that it would be preferable to define an NSG profile of official OGC GeoPackage Extension standards, if this could be successfully achieved in a reasonable timeframe.

Three such extensions are proposed, corresponding to individual sections of this Engineering Report:

- Vector Tiles,
- Semantic Annotations, and
- Styling.

Ecere plans to lead the effort in converting these sections into separate candidate standard documents, present these to the GeoPackage Standard Working Group for consideration and moving them along the OGC standardization process.

While the extensions allow for additional flexibility, for example in terms of embedding attributes inside tiles or using attributes table, or selecting additional encodings for vector tiles and styles, the RBT GeoPackage profile introduces additional constraints in order to facilitate interoperability, as detailed in the main RBT GeoPackage extension of this Engineering Report.

Two other draft GeoPackage extensions may also be relevant to RBT GeoPackages:

- the Deterministic Hierarchical Tiles Grouping & Indexing Extension , developed during the SOFWERX 3D Tech Sprint II and the 2023 CDB 2.0 Summer Workshop, and
- the 2D Tile Matrix Set Extension, developed during the Vector Tile Pilots.

C.1.1. Vector Tiles

The proposed Vector Tiles extension presented in this Engineering Report regroup primarily the Vector Tiles Extension and the Mapbox Vector Tiles Extension developed during the Vector Tiles Pilots, as a single extension, but separate requirement classes. The proposed OGC GeoPackage extension would also re-incorporate the Vector Tiles Attributes Extension as a distinct requirement class, allowing to store attributes in features table instead of embedded directly within the tiles. Experimenting with storing RBT content using attribute tables instead of embedded attributes would allow quantifying how much storage savings could be achieved using this approach, as well as how it facilitates queries.

The many-to-many mapping of tiles and features when using attributes table, which depends on the Related Tables extension, would be defined in a separate requirement class, so as to simplify the basic use of attributes table. The Vector Tiles extension would likely also define requirement classes for additional encodings, such as the GeoJSON vector tiles Extension previously developed, and/or an encoding based on Well-known Binary which was developed during the 2023 CDB 2.0 Summer Workshop.

C.1.2. Semantic Annotations

The Semantic Annotations Extension is based on the extension developed during the Vector Tiles Pilots. Some requirements were clarified during this sprint, and examples on how to use a *GeoDataClass* semantic annotation for the purpose of identifying content and associating styles were added.

Alignment with the approach to identify content in the CDB 2.0 GeoPackage Data Store would be beneficial, which currently specifies a GeoDataClass URI directly within the hierarchical indexing extension (summarized below).

C.1.3. Styling

The Styling Extension is based on the Portrayal Extension developed during the Vector Tiles Pilots.

Although Technology Integration Experiments performed during this initiative were largely successful, additional experiments with different styles and symbol sets, as well as more testing of the new capability to include font resources, would be beneficial. In particular, the use of different symbol sets with identical symbol names in different styles presented some difficulties.

The RBT profile focuses on MapboxGL styles, which would be a specific requirement class of the extension. Requirement classes for styles defined using OGC SLD/SE and OGC Cartographic Symbology 2.0 would also be included in this extension, and would also benefit from further experiments.

C.1.4. Deterministic Hierarchical Tiles Grouping & Indexing

Although only briefly discussed during this initiative, the Deterministic Hierarchical Tiles Grouping & Indexing Extension developed during the SOFWERX 3D Tech Sprint II and the 2023 CDB 2.0 Summer Workshop offers a practical solution for dealing with arbitrarily large tilesets which would be applicable to RBT GeoPackages, in particular for very high resolution imagery.

The extension specifies how to define in a main GeoPackage a grouping configuration of tilesets, indicating to the consumer whether all tiles are stored in a single GeoPackage, or split into separate GeoPackages based on a pre-determined path hierarchy according to tile pyramids. By specifying a number of levels to be grouped together, a base tile (of lower resolution) and congruent tiles at the next few resolutions will be included together in the same GeoPackage. This mechanism provides a very efficient way to organize arbitrarily large content in a directory hierarchy in a deterministic manner which is simple to access directly, while limiting both the number of files per directory as well as a maximum desirable GeoPackage size, which facilitates storage and exchange of whole and partial tilesets.

Experiments with this approach for RBT content, which could be aligned with on-going CDB 2.0 efforts, would be of interest, especially in the context of testing with the whole world RBT dataset as well as for the RBT repository use case.

C.1.5. 2D Tile Matrix Sets

The 2D Tile Matrix Set Extension defined during the Vector Tile Pilots provides full alignment of GeoPackage with the 2D Tile Matrix Set standard. A minor update is likely needed to reflect the latest version 2.0 of the standard.

The extension notably allows to re-use the same tables for common 2D Tile Matrix Sets shared by multiple tile sets, while maintaining compatibility with the core GeoPackage standard by defining the traditional tables as views on the new tables defined in this extension.

The extension also enables the use of variable width 2D Tile Matrix Sets, such as the GNOSIS Global Grid, which facilitates storing global content in a single tileset (as opposed to requiring separate World Mercator, Antarctic Universal Polar Stereographic and Universal Arctic Polar Stereographic tilesets), while minimizing the overhead associated with storing content near the poles in a geographic CRS such as EPSG:4326.

C.2. GeoDataClass registry

The concept of a *GeoDataClass* is used in the specification described in this Engineering Report for the purpose of identifying the *cultural*, *physical* and *hillshade* tilesets, as conforming to particular schemas expected by consumers. This ensures that the data layers contained within these tilesets correspond to those expected by styles designed to portray RBT GeoPackages, including the attributes referenced by those styles. This association is done using the Semantic Annotation extension.

The use for GeoDataClasses is not limited to styling, but also benefits processing, where they can help determine the suitability of a dataset as an input to a process.

A registry of GeoDataClasses could be established on the OGC definition server, which would provide an authoritative definition with a stable URI, while also allowing to retrieve information about a particular GeoDataClass, such as its component layers, geometry dimension and associated attributes.

C.3. NSG Standard for RBT GeoPackage

The primary purpose of this initiative was to inform a future NSG standard for storing RBT in GeoPackages. Additional work is required to convert the specification defined in this Engineering Report into an NSG standard. Defining this standard as a profile of OGC GeoPackage extensions would simplify the document by focusing on requirements specific to the RBT profile, while facilitating interoperability by leveraging these extensions and avoiding potential divergence.

One important aspect of defining this NSG profile would be to clarify the definition of the GeoDataClasses for the *cultural* and *physical* tilesets. Ideally, the exact content of these tilesets in terms of the individual layers, their geometry dimension, and the required attributes that can be referenced by styles or queried, should be fully defined for a given version of the profile, and could be registered in the proposed GeoDataClass registry.

C.4. 3D Buildings

Regarding the cultural buildings data layer in particular, it might be useful to include additional attributes present in the source OpenStreetMap datasets to facilitate extrusion of building footprints, as done for example by the 3D building style. In the RBT dataset, the *height* attribute is not set on most buildings. In the OSM Simple 3D Buildings attribution model, the `levels`, `building:levels`, `building:height`, `min_level`, `min_height`, `building:min_level`, `building:min_height`, `building:part` are also useful in addition to the `height` attribute (along with others, for fancier 3D buildings with special roofs etc.). It would be valuable if the RBT importing process either preserved those additional attributes, or used them for the purpose of computing the *height* when that is not set. `min_height` allows for example to define a gap under the Arc de Triomphe.

C.5. Portrayal

Although not clearly within the original scope of this sprint, significant efforts by participants were spent on the ability to portray RBT GeoPackages as defined by the MapboxGL / MapLibre style sheets built by the AGC RBT development team. These efforts covered both clarifying the GeoPackage Styling extension, as well as implementing and improving support for these style sheets in custom rendering engines.

Due to the complexity of this topic and the limited time available during the sprint, additional work on validating the Styling extension as well as developing a best practice to portray RBT, and more generally to map the styling capabilities of Mapbox GL style sheets to the conceptual and logical model defined in the draft OGC Cartographic Symbology 2.0 standard, would be very useful.

C.6. Implementations

Three independent implementations of both producers and consumers of RBT GeoPackages were developed by the sprint participants in a very short timeframe. Additional work on these implementations would be beneficial to improve performance, improve rendering quality and portrayal accuracy, in particular better handling complex symbology labeling rules, as well as to polish the user interfaces to facilitate producing and visualizing RBT GeoPackages.

The development of additional implementations should also be encouraged, in particular in key libraries such as GDAL which power several tools such as QGIS, so as to increase the interoperability of RBT GeoPackages.

Additional testing could also be performed serving and accessing RBT content using OGC API standards, such as OGC API — Tiles, OGC API — Maps, OGC API — Features and OGC API — Styles, as there was a limited opportunity to do so during the short timeframe of this initiative.

D

ANNEX D (INFORMATIVE) REVISION HISTORY

D

ANNEX D (INFORMATIVE) REVISION HISTORY

DATE	RELEASE	EDITOR	PRIMARY CLAUSES MODIFIED	DESCRIPTION
2024-03-22	0.1	J. St-Louis	all	initial version
2024-05-30	0.2	J. St-Louis	all	final draft ready for review