OGC Vector Tiles Pilot

WMTS Vector Tiles Extension Engineering Report

Table of Contents

Publication Date: 2019-02-11

Approval Date: 2018-12-13

Submission Date: 2018-11-14

Reference number of this document: OGC 18-083

Reference URL for this document: http://www.opengis.net/doc/PER/vtp-wmts1

Category: Public Engineering Report

Editor: Panagiotis (Peter) A. Vretanos

Title: OGC Vector Tiles Pilot: WMTS Vector Tiles Extension Engineering Report

---

OGC Engineering Report

COPYRIGHT

WARNING

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.

LICENSE AGREEMENT

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

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

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

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

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to

indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Chapter 1. Summary

The tiling of feature data is an approach that can be used to optimize the delivery vector feature data over the web to create maps. The approach provides a pre-defined shape (i.e. tile) to package vector data. Tiling of vector data enables faster map loads (due to reduced size) and offer flexible styling on the client side with modern, easy-to-use tools.

This Engineering Report (ER) describes the work done by participants during the Vector Tiles Pilot (VTP) to add Mapbox and GeoJSON vector tile support to Web Map Tile Servers. A summary of other work done in the VTP is presented in the VTP Summary Engineering Report [1].

NOTE	This engineering report interchangeably uses both 'tiled feature data' and the colloquial term 'vector tiles'.

1.1. Requirements & Research Motivation

The Web Map Tile Service (WMTS) standard currently offers predefined tiles in a fixed set of graphical representations with no provision, from the perspective of a client, for adding additional styles or even dynamically styling tiles on the fly. However, some implementations, like GeoServer, can be configured to allow style selection, dynamic styling and even on-the-fly content filtering, at the price of either extra storage or no caching).

This ER addresses this deficiency by exploring the addition of tiled feature data support to a WMTS as another supported output format. This addition to WMTS follows lessons learnt from the recent OGC Testbed-13 initiative [2]. Tiled feature data can be efficiently transmitted to the client where the data can be styled in a manner chosen by the client using a number of easy-to-use tools before being finally rendered onto the display.

1.2. Prior-After Comparison

The following Standards Working Groups (SWGs) are most directly involved in OGC work related to map rendering and would be interested in the work described in this ER:

- Web Mapping Service 1.4 SWG (also responsible for WMTS)
- Styled Layer Descriptor & Symbology Encoding SWG

The GeoPackage SWG, being involved in work related to packaging formats that support map rendering in Denied, Degraded, Intermittent or Limited bandwidth (DDIL) environments would also be directly interested in the work described in this ER.

The following SWGs, are involved in work that is redefining the underlying architecture of OGC Web Services (OWS). As such, they would also be interested in the work described in this ER as it explores how far the new paradigms can be extended to support the resource type of most interest to OGC members:

- Web Feature Service/Filter Encoding Specification SWG
- Web Processing Service SWG

1.3. Recommendations for Future Work

Having developed a set of abstract test suites in the Vector Tiles Pilot, future work should look to develop executable test suites for compliance testing of implementations of this specification.

Further investigation is also needed towards allowing WMTS servers and clients to operate in denied, degraded, intermittent or limited bandwidth (DDIL) networks. The DDIL clause in this document discusses some avenues of investigation.

1.4. Document contributor contact points

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

Contacts

Name	Organization
Panagiotis (Peter) A. Vretanos	CubeWerx Inc.
Kalimar Maia	Mapbox

1.5. Foreword

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.

Chapter 2. References

The following normative documents are referenced in this document.

- OGC: OGC 07-057r7, OGC® OpenGIS Web Map Tile Service Implementation Standard, 2010 [http://portal.opengeospatial.org/files/?artifact_id=35326]
- OGC: OGC 17-069, OGC® Web Feature Service 3.0: Part 1 - Core Candidate Standard, 2018 [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html]
- OGC: OGC 12-128r15, OGC® GeoPackage Encoding Standard - with Corrigendum, 2018 [https://portal.opengeospatial.org/files/12-128r15]
- OGC: OGC 17-083, OGC® Tile Matrix Set Candidate Standard, 2018 [https://portal.opengeospatial.org/files/?artifact_id=79767&version=1]

Chapter 3. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard OGC 06-121r9 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2] shall apply. In addition, the following terms and definitions apply.

- dataset

```
identifiable collection of data [ISO 19115-1:2014]
```

NOTE The use of 'collection' in the definition from ISO 19115-1 is broader than the use of the term 'collection' in this specification. See the definition of 'feature collection'.

- distribution

```
represents an accessible form of a dataset [W3C DCAT]
```

EXAMPLE: a downloadable file, an RSS feed or a web service that provides the data.

- feature

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

NOTE If you are unfamiliar with the term 'feature', the explanations in the W3C/OGC Spatial Data on the Web Best Practice document may help, in particular the section on Spatial Things, Features and Geometry [https://www.w3.org/TR/sdw-bp/#spatial-things-features-and-geometry].

- feature collection; collection

```
a set of features from a dataset
```

NOTE In this specification, 'collection' is used as a synonym for 'feature collection'. This is done to make, for example, URI path expressions shorter and easier to understand for those that are not geo-experts.

- coordinate reference system

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

- coordinate system

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

- feature info

```
information related to a particular pixel of a map that refers to the geographic
data portrayed on that area
```

- layer

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

- map

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

- portrayal

```
graphical presentation of information to humans
```

- procedure oriented architectural style

```
platform-independent design approach that is focused on operations, their
parameters and their results, that can be defined in an abstract level
specification. Concrete platform-dependent specifications can be derived from the
abstract level, allowing, for example, KVP or SOAP messaging.
```

- resource oriented architectural style

```
platform-independent design approach that is focused on resources, representations
and actions, that can be defined in an abstract level specification. Concrete
platform-dependent specifications can be derived form the abstract level, allowing,
for example, a RESTful architecture.
```

- theme

```
a group of layers that can be nested hierarchically
```

- tile

```
a tessellated representation of geographic data, often part of a set of such
elements, covering a spatially contiguous extent which can be uniquely defined by a
pair of indices for the column and row along with an identifier for the tile matrix
(adapted from OGC 07-057r7)
```

- tile matrix

```
a collection of tiles for a fixed scale
```

- tile matrix set

```
a collection of tile matrices defined at different scales
```

3.1. Abbreviated terms

Table 1. Abbreviations

JPEG	Joint Photographic Experts Group (image format)
JVT	GeoJSON Vector Tile
MVT	Mapbox Vector Tile
PNG	Portable Network Graphics (image format)
REST	Representational State Transfer
SLD	Styled Layer Descriptor
WFS	Web Feature Service
WMTS	Web Map Tile Service

Chapter 4. Overview

Clause 5 describes the client and server implementations deployed for the Vector Tiles Pilot.

Clause 6 presents a discussion on some of the potential requirements for a Vector Tiles extension to WMTS.

Chapter 5. WMTS Implementations

A number of WMTS services and client applications took part in the Pilot. The following subsections describe those services and client applications.

5.1. Pilot Architecture

To support experimentation, an architecture was designed to cover various client-server relationships that are common in OGC use cases as illustrated in Figure 1.

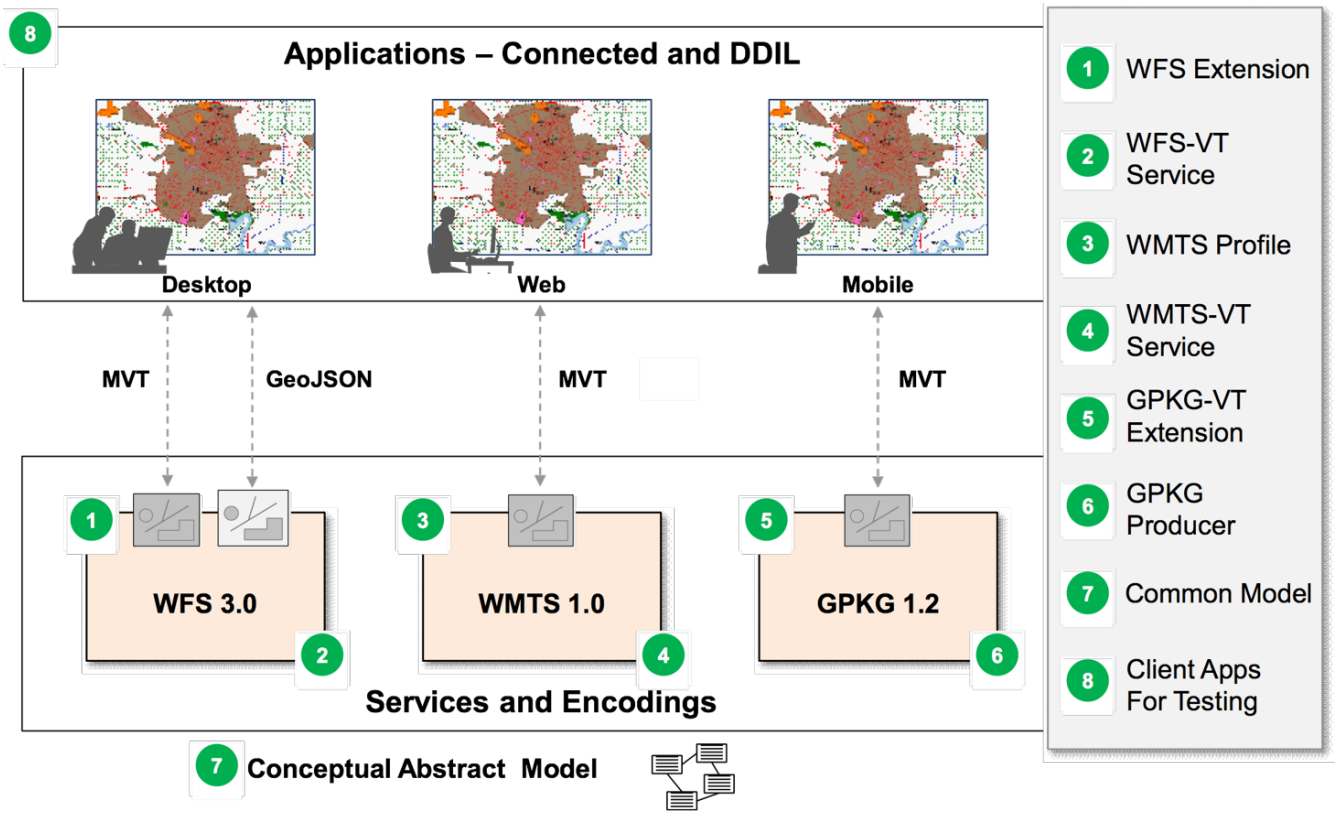

Figure 1. Vector Tiles Pilot Architecture

The architecture therefore addresses the following client-server relationships to simultaneously enable tiled feature data across the relevant suite of OGC standards.

- Desktop Client → Web Feature Service (WFS) 3.0
- Web Client → Web Map Tile Service (WMTS) 1.0
- Mobile Client → GeoPackage 1.2

The following sections focus only on WMTS, as it is the subject of this engineering report.

5.2. Servers

The pilot included WMTS services from CubeWerx, Ecere, GeoSolutions and MapBox. The following subsections describe the services.

5.2.1. CubeWerx

The CubeWerx WMTS service implements version 1.0 of the WMTS standard. The service offers the following operations:

- GetCapabilities
- GetTile
- GetFeatureInfo

The service also offers the following vendor-specific operations:

- GetDescription
- GetLegendGraphic
- GetTileJson
- GetAccessibility
- GetAssociations

Server deployment

The CubeWerx WMTS has been deployed on an Amazon instance at the following endpoint:

- https://tb14.cubewerx.com/cubewerx/cubeserv/vt

It offers the standard spherical mercator (smerc) tile matrix set and supports the following tile formats:

- image/x-jpegorpng
- image/jpgpng
- image/png
- image/jpeg
- application/vnd.mapbox-vector-tile

The complete capabilities document of the server can be found here:

- https://tb14.cubewerx.com/cubewerx/cubeserv/vt/wmts/1.0.0/WMTSCapabilities.xml

Implementation Efforts for Adding MVT Support to WMTS

No changes were necessary to the WMTS standard. It simply advertises that vector tiles data is offered as another available tile format. To add Mapbox Vector Tile (MVT) specification support, the server:

- integrated the third-party protobuf 3.6.0 and protobuf-c 1.3.0 libraries to handle the Google Protobuf encoding requirements.
- implemented the generation of MVTs as per the Mapbox Vector Tile specification [3].

Some of the challenges encountered during this implementation were:

- supporting the various geometry types and property value types
- implementing proper (and efficient) simplification and clipping of geometries
- honoring any filters and scale rules that may be present in the specified style
- making the buffer zone size configurable
- making the ratio of internal resolution to display resolution configurable
- making whether or not properties should be emitted configurable
- implementing a low-level mechanism that will generate an MVT tile of any size, in any coordinate system, at any resolution, with any number of layers, from any supported source
- implementing a higher-level tile-oriented mechanism that generates vector tiles in accordance with the WMTS-defined tile-matrix sets
- implementing GeoJSON Vector Tile support in parallel
- performing appropriate testing

Testing

In order to test the CubeWerx MVT implementation, a bespoke test client that is based on OpenLayers was deployed at the following endpoint:

- https://tb14.cubewerx.com/mvtTest/

This client is hardcoded to test the CubeWerx server specifically.

5.2.2. Ecere

The Ecere WMTS service implements version 1.0 of the WMTS standard. The service offers the following operations:

- GetCapabilities
- GetTile

Server deployment

The Ecere WMTS has been deployed at the following endpoint:

- http://maps.ecere.com/wmts?

Tiles for any layers are generated on-the-fly for any supported encoding and tiling scheme from a GNOSIS tiled data store.

It supports the following vector tile formats:

- text/xml;subtype=gml/3.1.1 - GML
- application/vnd.geo+json - GeoJSON [https://tools.ietf.org/html/rfc7946]
- application/vnd.mapbox-vector-tile - Mapbox vector tiles [https://www.mapbox.com/vector-tiles/specification/]

- application/vnd.geo+econ - GeoECON [http://docs.opengeospatial.org/per/17-041.html#GeoECON]
- application/vnd.gnosis-map-tile - GNOSIS Map Tiles [http://ecere.com/gmt.pdf] (initial publication [http://docs.opengeospatial.org/per/17-041.html#_gnosis_compact_vector_tiles_representation])

The complete capabilities document of the server can be found here:

- http://maps.ecere.com/wmts?SERVICE=WMTS&REQUEST=GetCapabilities

The service supports the following Tile Matrix Sets:

- GlobalCRS84Pixel - EPSG:4326 (geographic); Based on the WMTS Well Known Scale Set with convenient pixels/degrees values (not a quad tree, resolution does not double at each level)
- GlobalCRS84Scale - EPSG:4326 (geographic); Based on the WMTS Well Known Scale Set with convenient scale denominator values (not a quad tree, resolution does not double at each level)
- GoogleCRS84Quad - EPSG:4326 (geographic); Based on the WMTS Well Known Scale Set, a quad tree with convenient scale denominator values
- GoogleMapsCompatible - EPSG:3857 (spherical pseudo-Mercator projection); Based on the WMTS Well Known Scale Set compatible with GoogleMaps and others (also a quad tree)
- GNOSISGlobalGrid - EPSG:4326 (geographic); A global grid with fewer columns of tiles closer to the pole than at the equator, approximating equal area tiles, a quad-tree except for starting with eight 90deg x 90deg tiles at level 0, and tiles touching a pole splitting in 3 rather than 4 [http://docs.opengeospatial.org/per/17-041.html#_global_gnosis_tiling_scheme_adapted_to_polar_regions]

Implementation Efforts for Adding MVT Support to WMTS

No changes were necessary to the WMTS specification. MVT was simply included as an additional output format offered by the WMTS. Support for encoding vector features according to the MVT specification, using Google Protocol Buffers, was implemented during the course of this pilot within the Ecere's GNOSIS software libraries and had to be integrated within Ecere's GNOSIS Map Server.

Testing

Mapbox vector tiles generated by Ecere's WMTS were tested using 'vtvalidate', which performs the following tests [1: https://github.com/mapbox/vtvalidate]:

- Tiled feature data consistent with the Mapbox Vector Tile specification - Version 2.
- Read tile layer(s) and feature(s)
- Decode properties
- Decode geometries

Tiles were also tested successfully with OGR software, which required a fix [https://github.com/OSGeo/gdal/commit/04952b2a05036e4b2d32d378b67e60c35d13912a] for an extent greater than 16384, as well as with QGIS 3.2.3.

Support for visualizing Mapbox vector tiles was implemented in Ecere's visualization client (see section below), which was also used to test the tiles delivered by the WMTS service.

The Ecere WMTS content was also successfully rendered by Compusult, thereby confirming a successful Technology Integration Experiment (TIE).

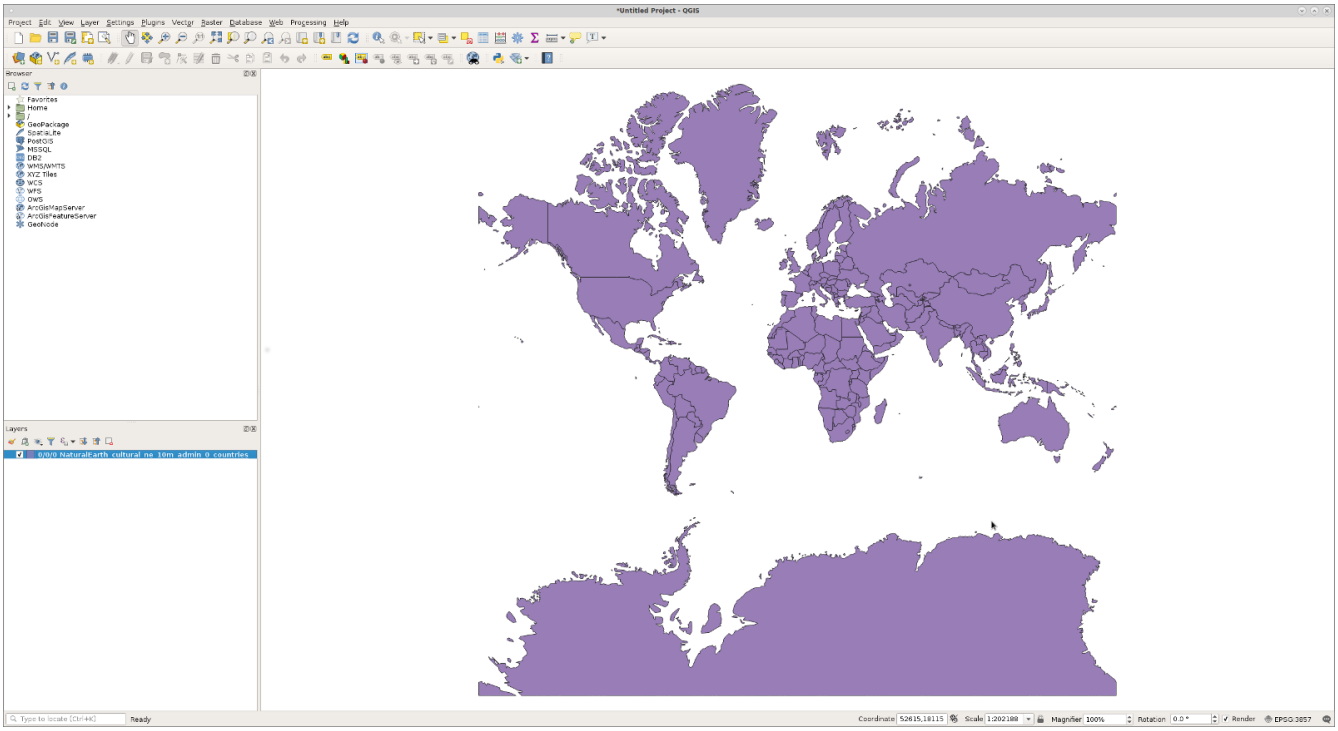

Figure 2. Mapbox vector tile served by Ecere WMTS rendered in QGIS 3.2.3

5.2.3. GeoSolutions

The GeoSolutions WMTS service implements version 1.0 of the WMTS standard. The service offers the following operations:

- GetCapabilities
- GetTile
- GetFeatureInfo

Server deployment

The GeoSolutions WMTS has been deployed at the following endpoint:

- https://maps.geo-solutions.it/geoserver/vtp/gwc/service/wmts?

It supports the following tile formats:

- application/vnd.mapbox-vector-tile

The complete capabilities document of the server can be found here:

https://maps.geo-solutions.it/geoserver/vtp/gwc/service/wmts?Service=WMTS%20&Request=GetCapabilities&Version=1.0.0

The vector tiles are served in the following Tile Matrix Sets:

- EPSG:4326 (WGS84 world coverage, starting with two side by side tiles at zoom level zero)

- EPSG:4326_512 (same as above, but having tiles of 512 pixels)
- EPSG:900913 (Google maps compatible alias)

Implementation Efforts for Adding MVT Support to WMTS

No changes were necessary to the WMTS specification. MVT was simply included as an additional output format offered by the WMTS, both as in Key-Value Pair (KVP) and Representational State Transfer (REST) service styles.

The GeoServer MVT implementation has the following features:

- MapBox 2.1 specification (acquired during pilot development, previous supported version was 1.0.1)
- Tile contents are driven by the style associated to the layer in GeoServer, the filters or rules active at the specified zoom level are used to decide which features get into the tile
- Single or multi-layer support
- Automatic determination of buffer zone based on the symbolizers included in the reference style (can be manually configured too)
- Efficient rectangular clipping, followed by a topology preserving geometry simplification
- All attributes available are included in the output tile

Testing

Tiled feature data generated by the GeoSolutions WMTS was successfully rendered in a Mapbox vector tiles client deployed by GeoSolutions, as well as other clients

5.2.4. Mapbox

The Mapbox WMTS service implements version 1.0 of the WMTS standard. The service offers the following operations:

- GetCapabilities
- GetTile

Server deployment

In the spirit of interoperability, the Mapbox WMTS was implemented using Klokan Tech's tileserver-gl https://tileserver.readthedocs.io/en/latest/

The Mapbox WMTS has been deployed at the following endpoint:

- http://ec2-54-172-96-205.compute-1.amazonaws.com/

The service also offers the following:

- Style viewer (both raster and vector)
- GL Style viewer

- TileJSON
- WMTS
- XYZ
- mbtiles inspector

Implementation Efforts for Adding MVT Support to WMTS

No changes were necessary to the WMTS specification. MVT was simply included as an additional output format offered by the WMTS.

5.3. Clients

The pilot included WMTS client applications from CubeWerx, Compusult, Ecere, GeoSolutions and MapBox. The following subsections describe the clients.

5.3.1. CubeWerx

The CubeWerx WMTS client runs as a web-browser based application. The client application allows an end-user to select the layers to display through the interface shown in Figure 3.

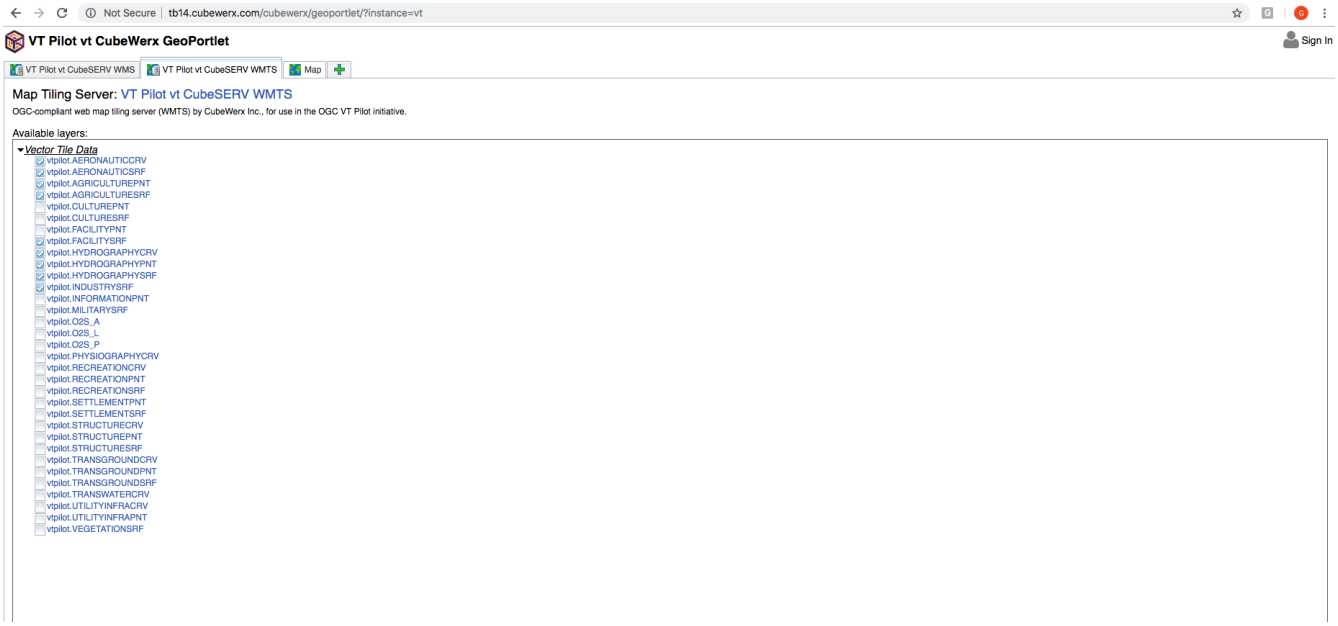

Figure 3. Vector Tiles Layer Selection on the CubeWerx Client

Once the end user has selected one or more layers to display, a separate tab is opened to display the contents of that layer. A screenshot showing MVT data rendered on a map is shown in Figure 4.

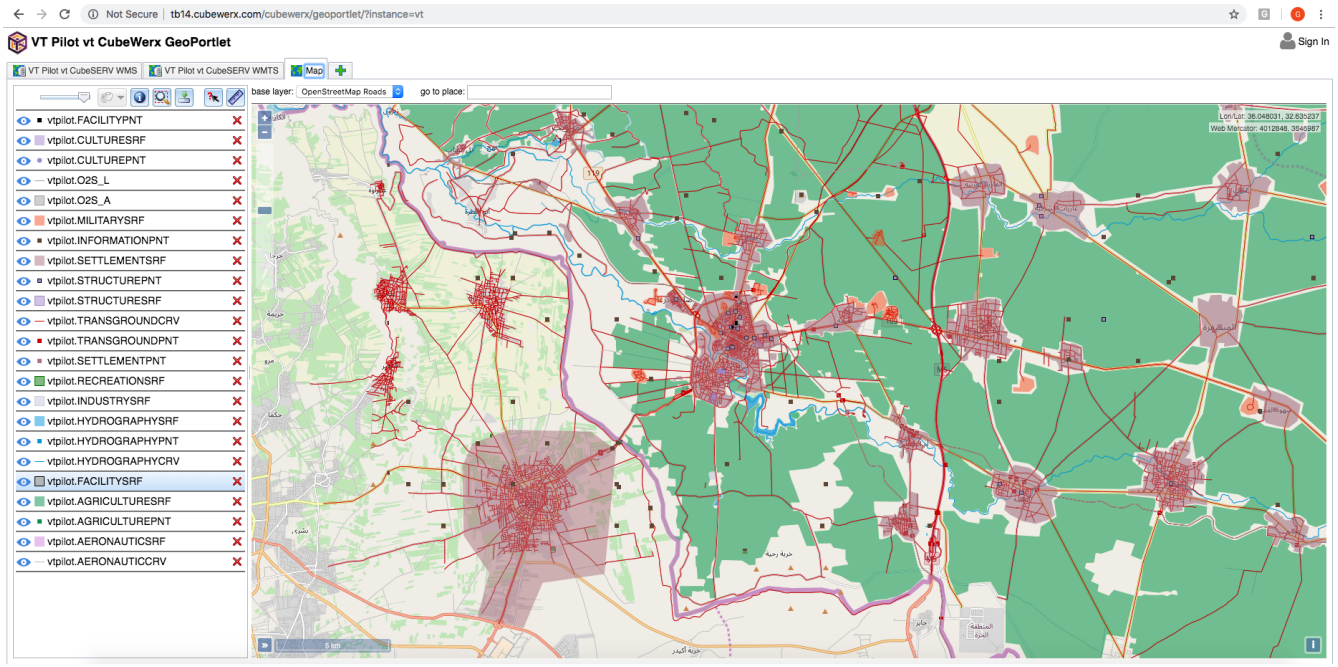

Figure 4. Vector Tiles rendered on the CubeWerx Client

5.3.2. GeoSolutions

The GeoSolutions WMTS client runs as a web-browser based application:

http://vtp2018.s3-eu-west-1.amazonaws.com/wmts.html

The client application presents 6 maps, with integrated controls between them (that is panning and zooming one map, adjusts the other maps as well). A screenshot showing the client application is presented in Figure 6 and Figure 5.

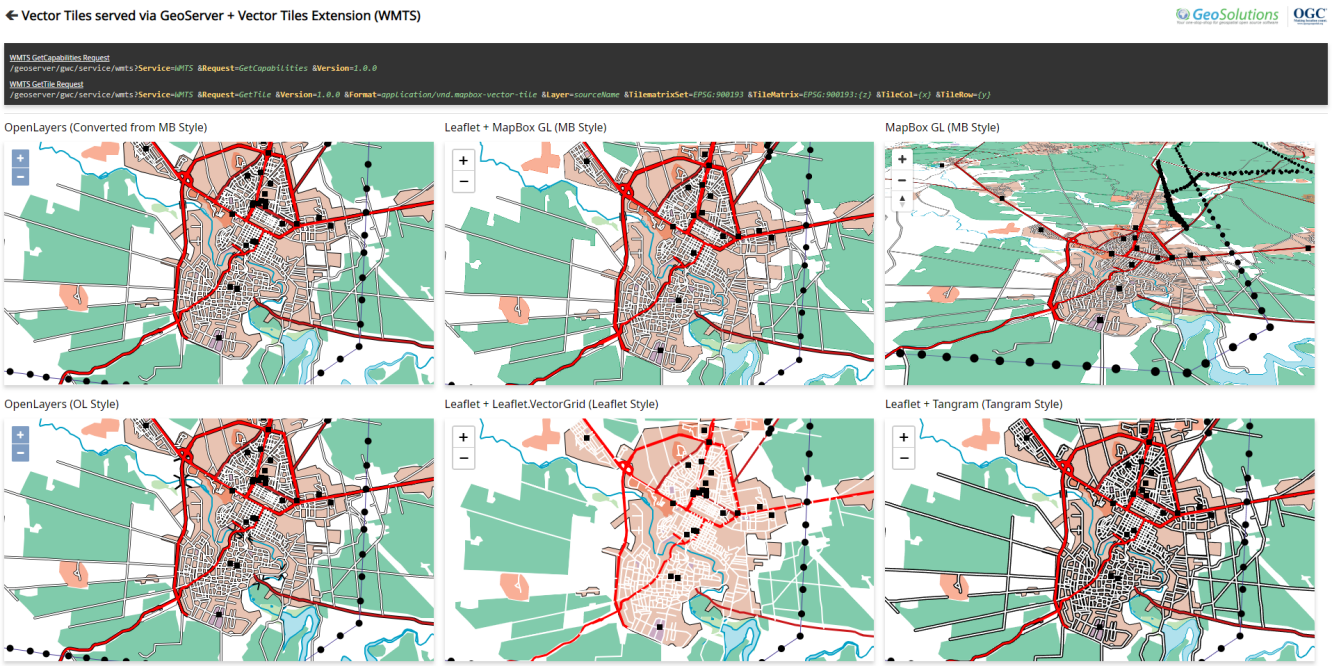

Figure 5. Vector Tiles rendered on the GeoSolutions Client showing styles applied client side

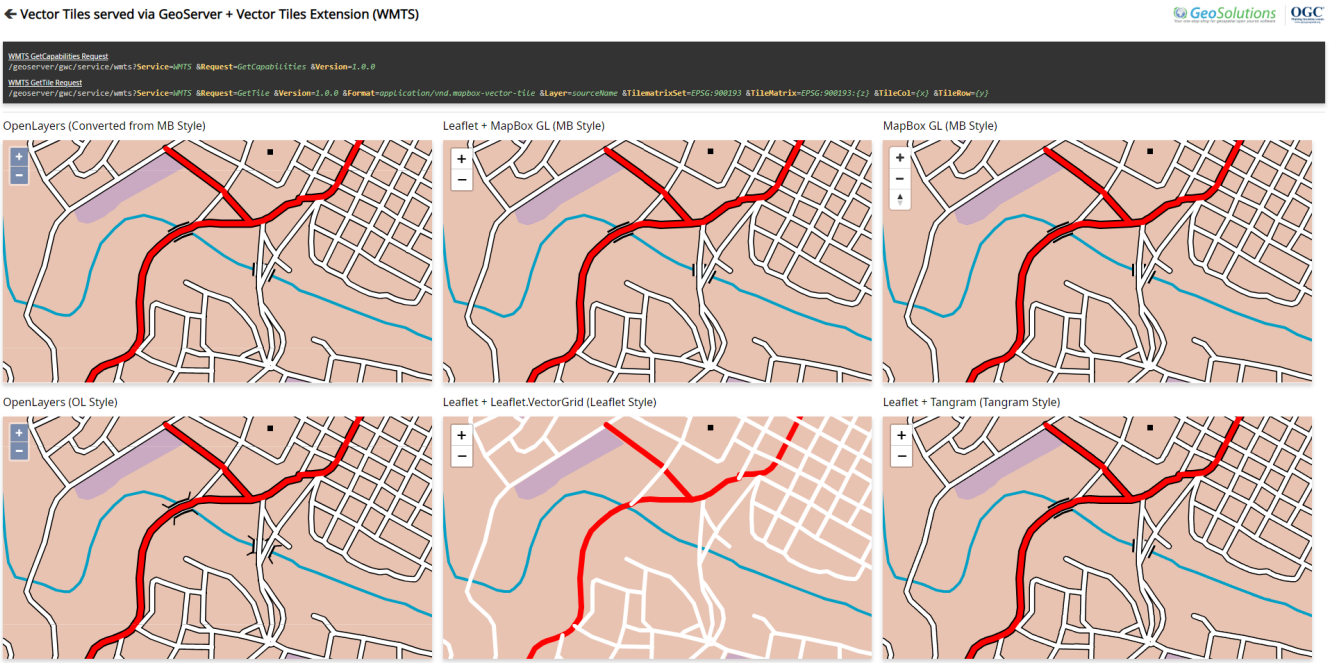

Figure 6. Vector Tiles rendered on the GeoSolutions Client, bridge detail

The 6 maps are implemented using:

- OpenLayers (Converted from Mapbox Styles)
- Leaflet with Mapbox GL (using Mapbox Styles)
- Mapbox GL (using Mapbox Style), OpenLayers (using OpenLayers styling)
- Leaflet with Leaflet.VectorGrid (using Leaflet styling)
- Leaflet with Tangram (using Tangram styling)

The second client instead shows all available datasets in the GeoServer implementation, giving an idea of performance extracting data on the fly, without caching, over large areas, and :

http://vtp2018.s3-eu-west-1.amazonaws.com/datasets.html

The lack of zoom based automatic feature selection, coupled with the lack of caching, shows that MVT can also deliver relatively poor performance when mis-used. Just like the other client, clicking on a feature shows a popup with the attributes, as fetched from the vector tile contents.

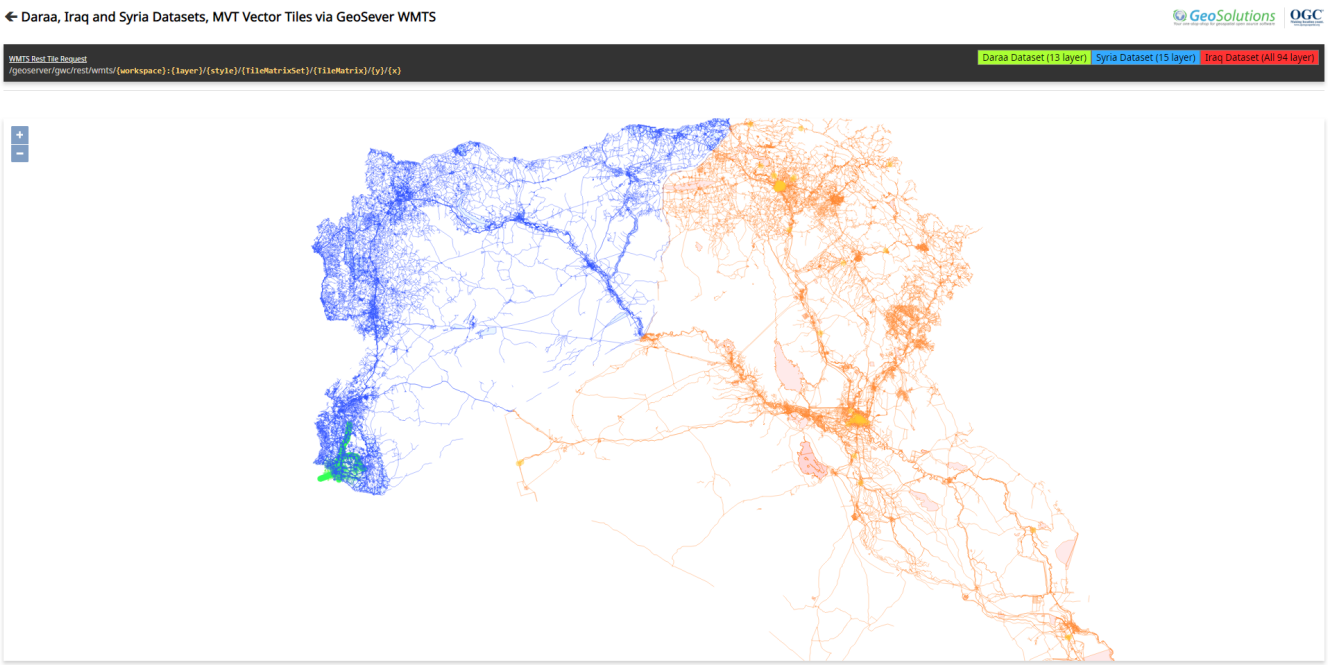

Figure 7. Vector Tiles rendered on the GeoSolutions Client showing datasets of Daraa, Syria and Iraq

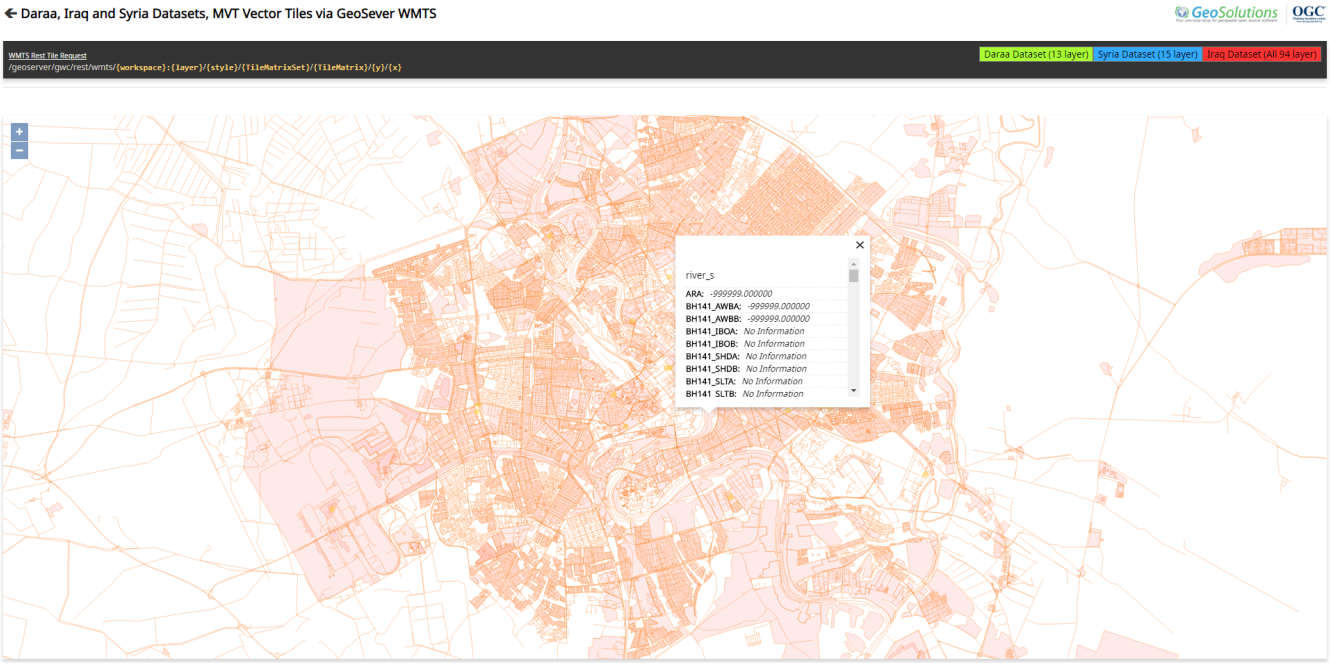

Figure 8. Vector Tiles rendered on the GeoSolutions Client showing features requested client side, directly from vector tiles

5.3.3. Compusult

The Compusult WMTS client runs as a web-browser based application, as well as a desktop/android based application. The WMTS client supports both Mapbox Vector Tile and GeoJSON tile formats. The client application allows users to interactively enable/disable specific WMTS layers, as well as overlay any specified base map for visualization. Styles are randomly generated for each layer. The screenshots below show the WMTS client rendering different WMTS server providers. The client also allows for native GetFeatureInfo requests. Supported by the server, or implemented by the client.

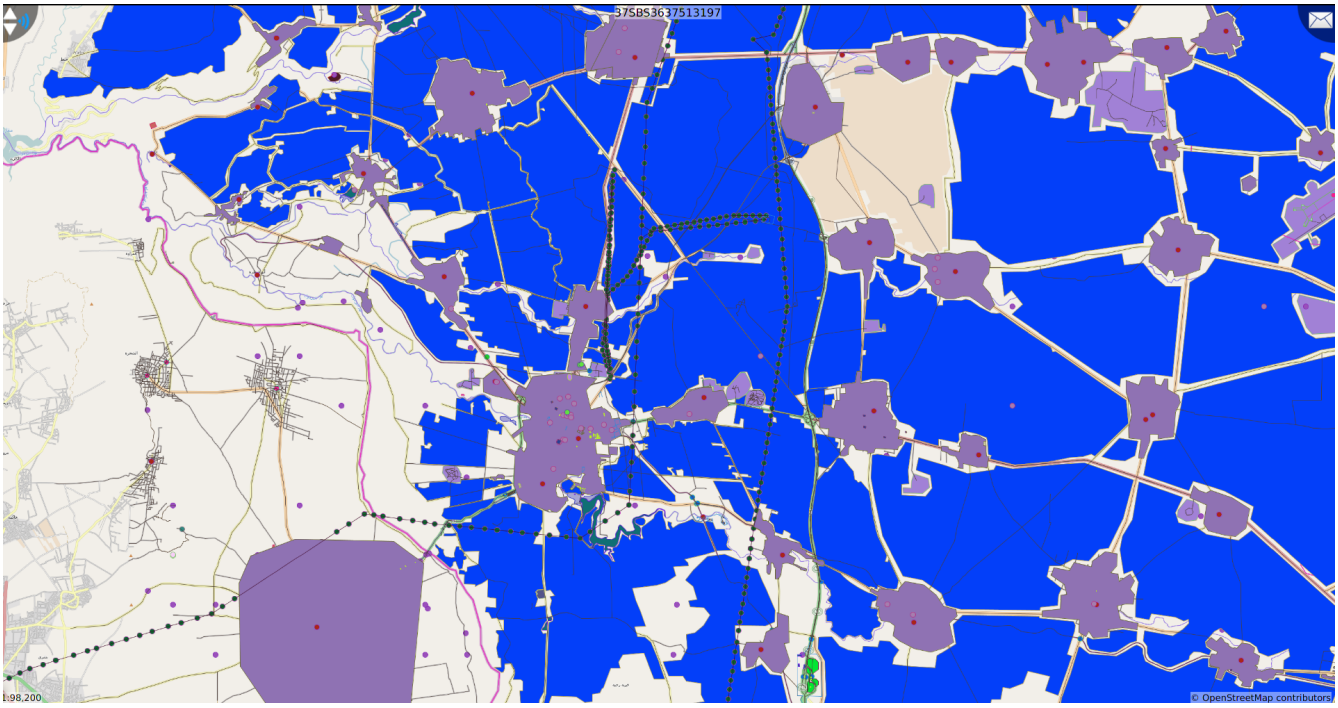

Figure 9. Vector Tiles from CubeWerx Server rendered on the Compusult Client

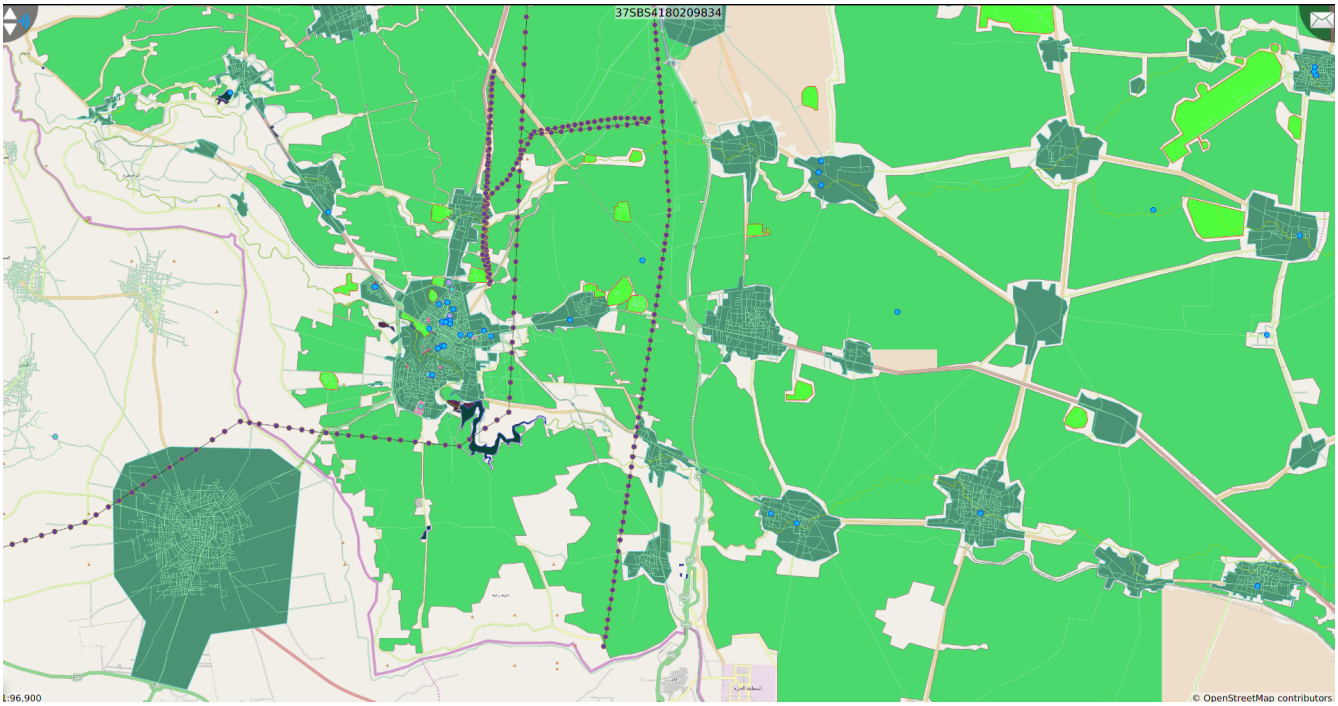

Figure 10. Vector Tiles from GeoSolutions rendered on the Compusult Client

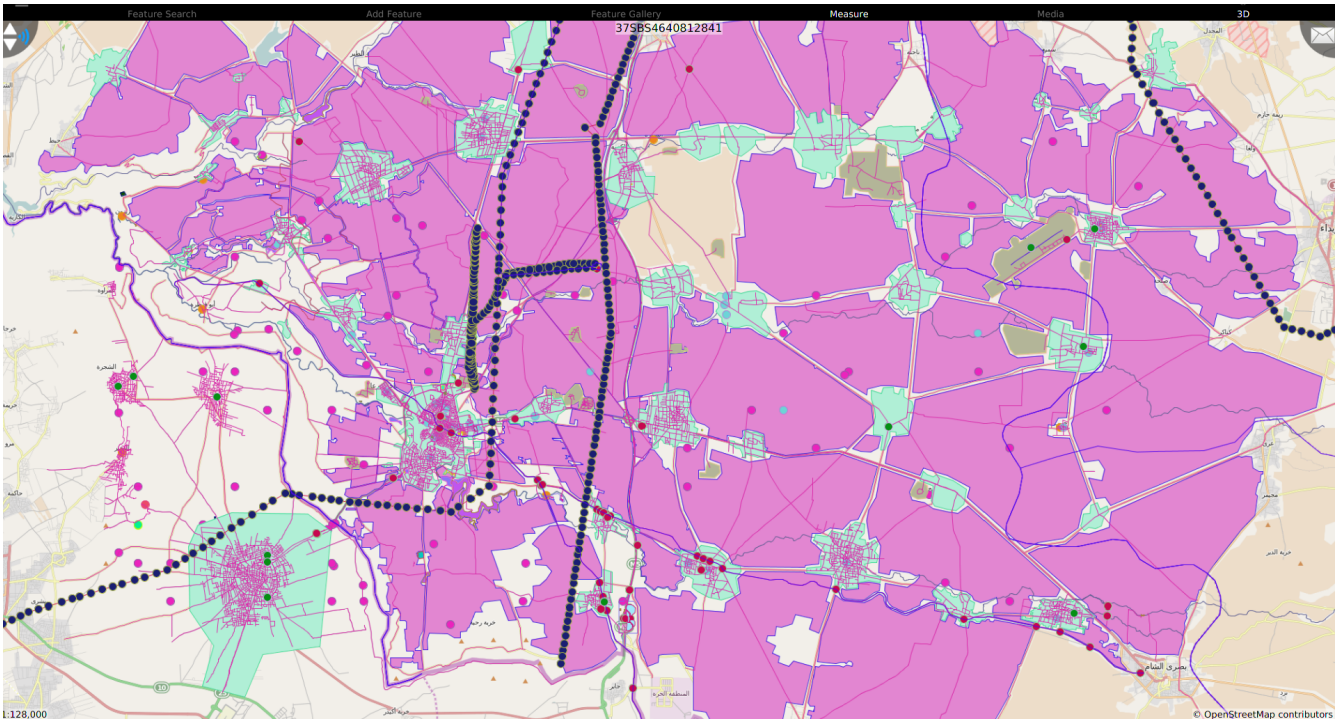

Figure 11. Vector Tiles from Ecere rendered on the Compusult Client

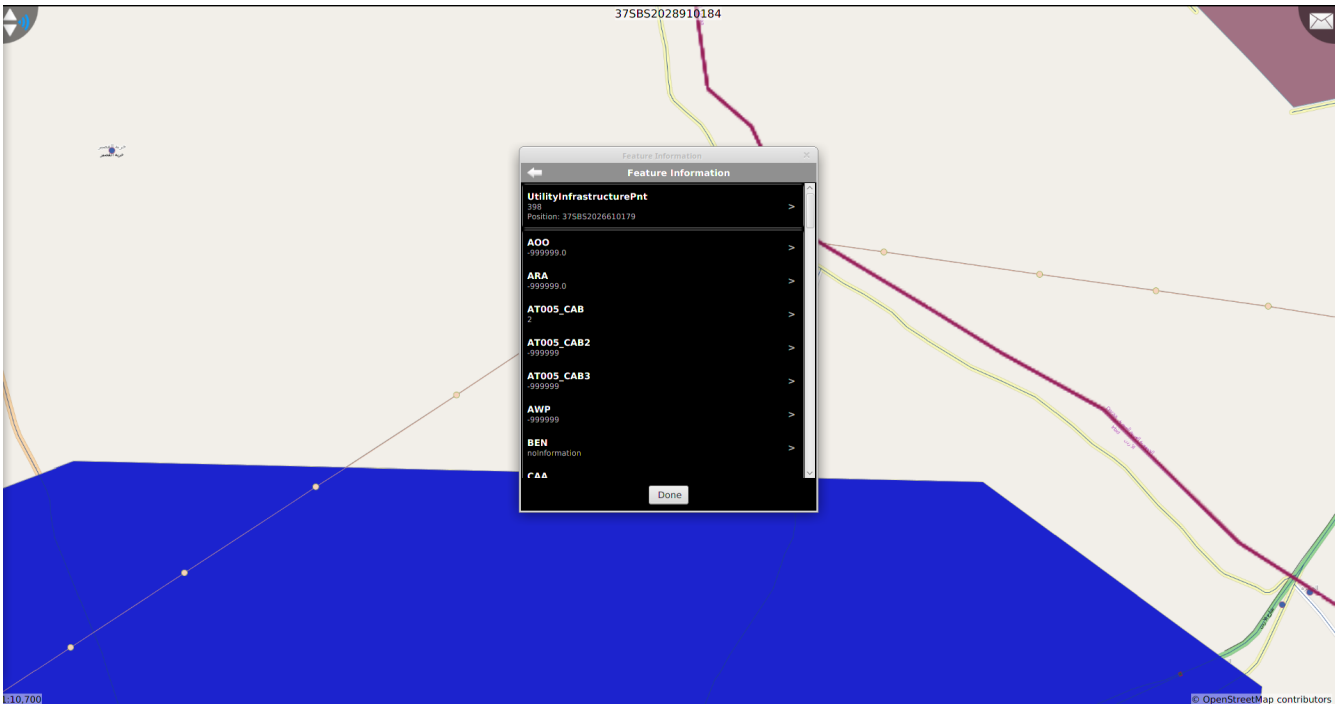

Figure 12. Feature Info Response Compusult Client

There were some minor issues noted:

- WMTS services displayed with all feature types as individual layers resulted in slow rendering times on a web-based client due to the number of requests and browser request limitations.
- Services with MVT tiles that have polygons that are not buffered will result in intermittent tile boundaries when trying to render polygon strokes. Same issue occurs rendering points with images that cross tile boundaries
- Services with multiple layers did not specify the correct drawing order to properly layer all feature types

Recommendations:

- OGC should consider including a WMTS extension that allows for a user to filter the layers for an MVT that has a multi-layer composition. It makes sense to only make 1 request for all layers of a MVT, but the user may want to filter those layers on demand to be recognized by the server. WMS allows for a layer list, while WMTS does not. Since an MVT will only be used as a WMTS, but does not fit the standard WMTS base map type response, OGC may want to allow selecting multiple layers per request, or filtering layers inside of a single MVT layer.
- Clients should not have to depend on sibling tiles to perform proper rendering operations.

5.3.4. Ecere

The Ecere WMTS client is implemented as a capability of Ecere's GNOSIS software libraries, available from within Ecere's GNOSIS Cartographer GIS tool, as well as from any application built using the GNOSIS SDK (whether for desktop, web or mobile). Support for accessing vector tiles through WMTS has been added for this pilot, with support for GNOSIS Map Tiles, GML, GeoECON, GeoJSON and Mapbox vector tiles. Visualizations of Mapbox vector tiles and GeoJSON tiles were entirely new capabilities developed during the pilot.

Figure 13. Vector tiles served by Ecere WMTS rendered in Ecere's GNOSIS client

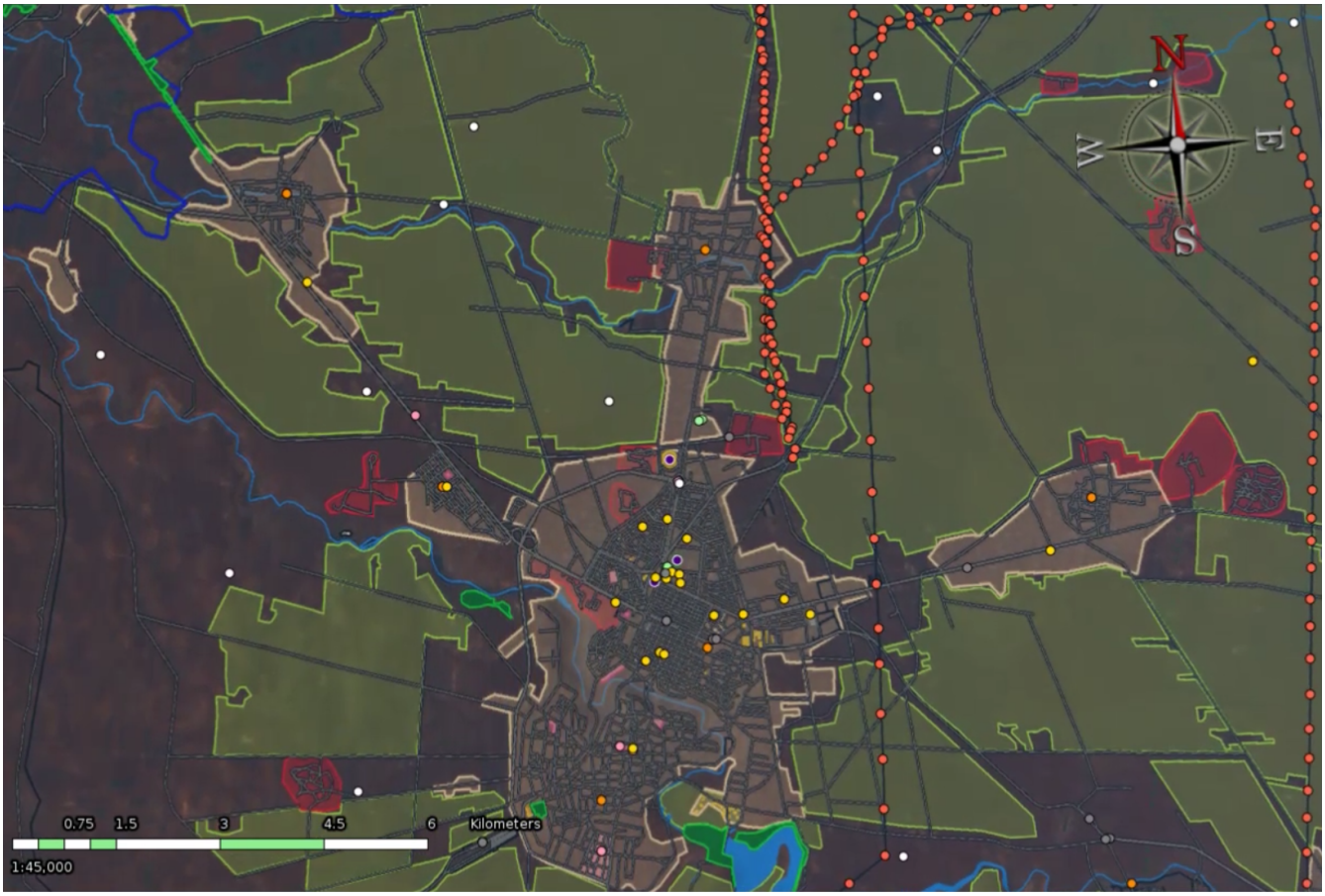

Figure 14. Vector tiles served by CubeWerx WMTS rendered in Ecere's GNOSIS client

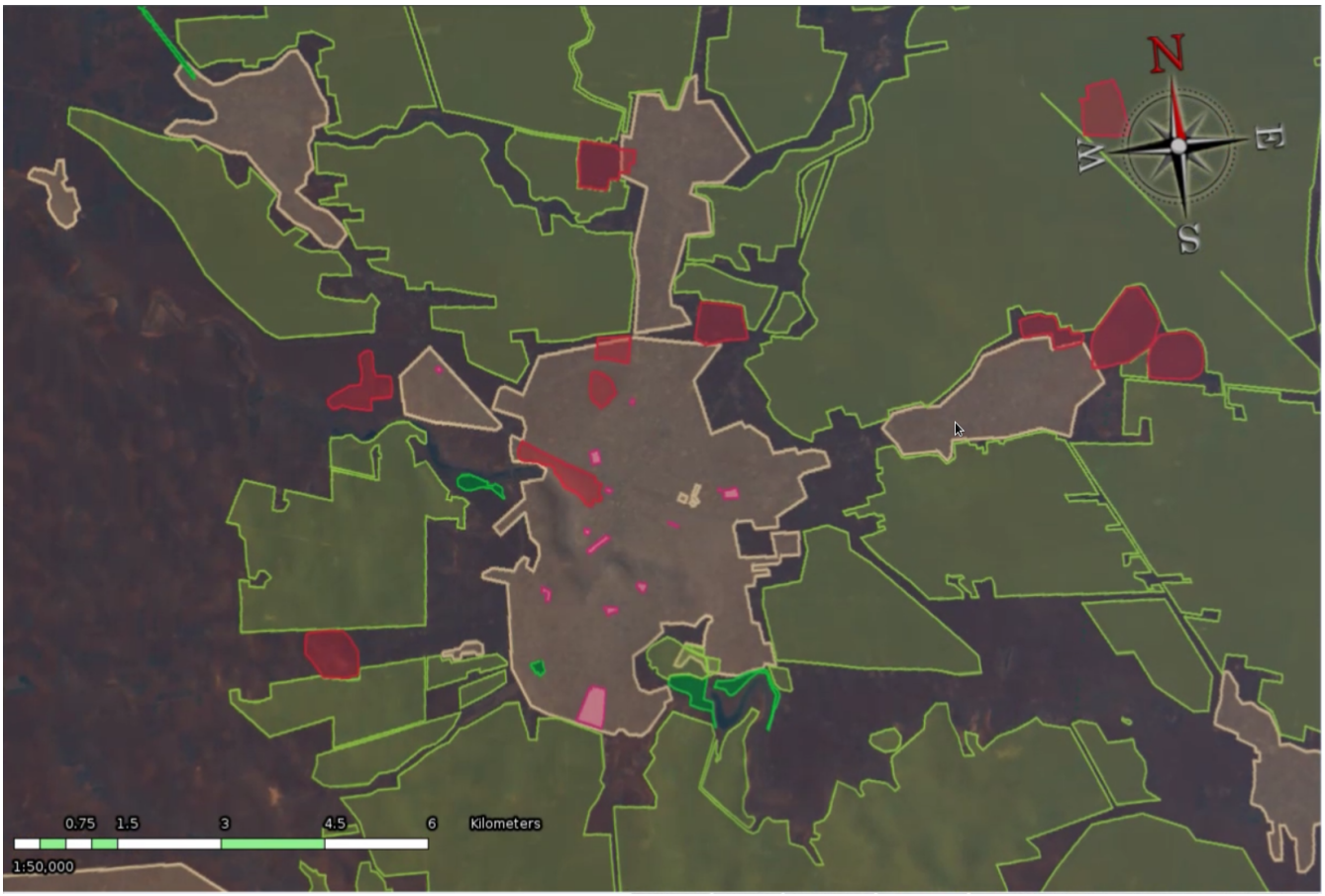

Figure 15. Vector tiles served by GeoSolutions WMTS rendered in Ecere's GNOSIS client (surfaces only)

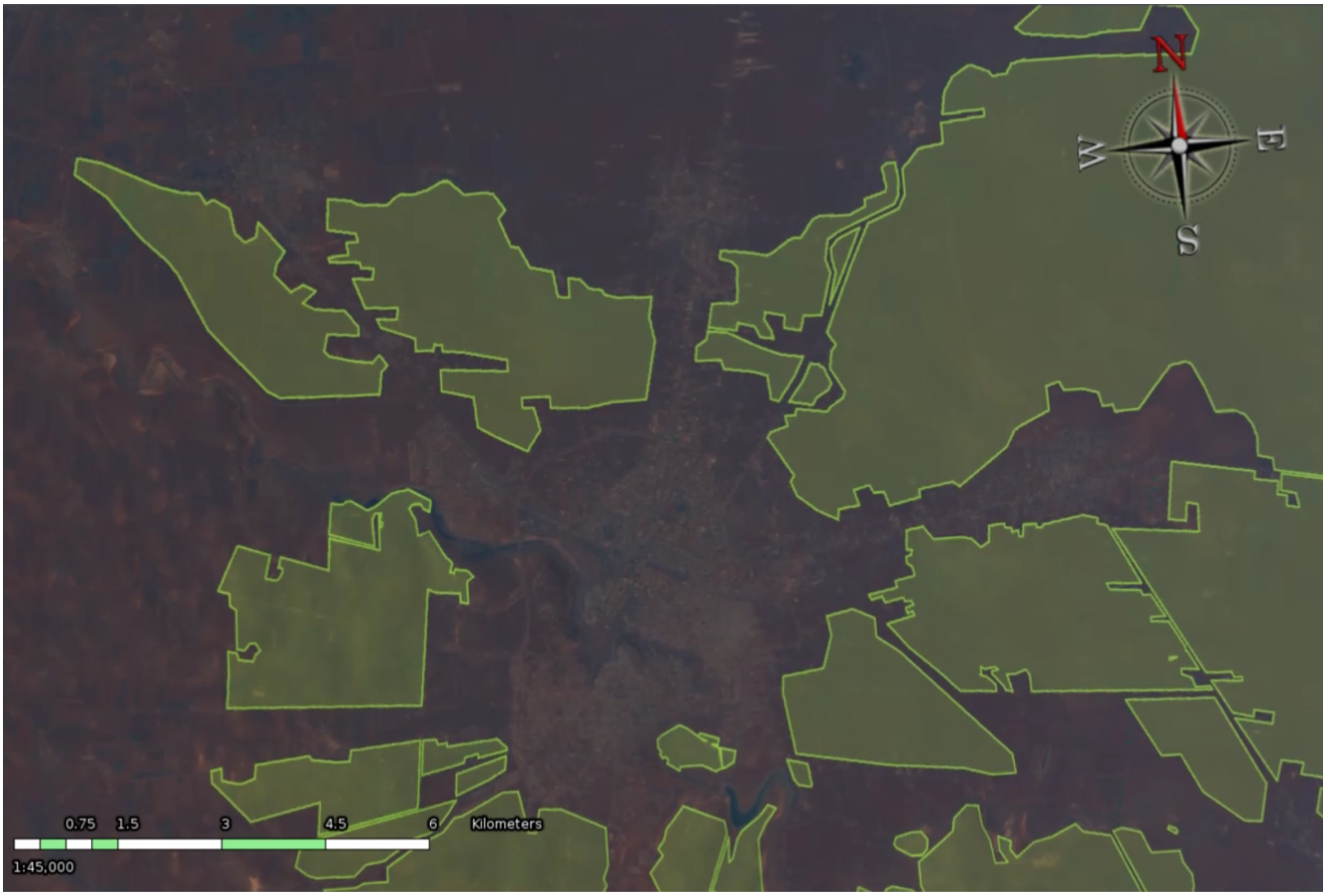

Figure 16. Vector tiles served by Mapbox WMTS rendered in Ecere's GNOSIS client (agricultural surfaces)

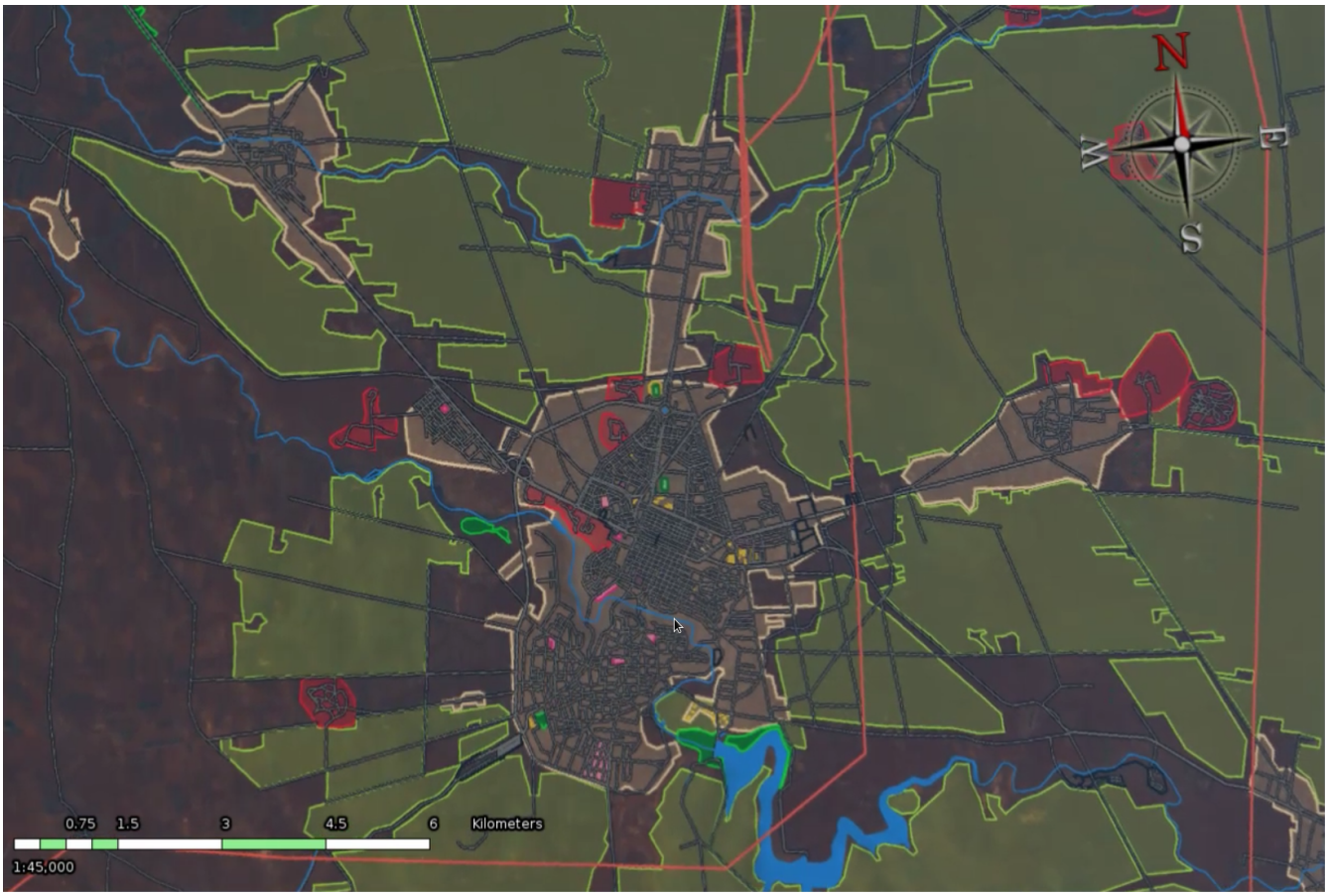

Figure 17. Vector tiles served by Compusult WMTS rendered in Ecere's GNOSIS client

Mapbox Vector Tiles inner v. outer contours:

- For Mapbox vector tiles not adhering to the 2.1 specifications (with specific ordering rules: outer contours appear clockwise, inner contours appear counter-clockwise), it is tricky to determine whether a given contour of a polygon is inner or outer. The MVT polygon encoding has no construct either separating the different polygons, or marking a given contour as inner or outer. There is only 'ClosePath' command, which identifies the end of a contour. One cannot use 'first contour is outer' rule, because it is not known where one polygon ends and the next one begins. The decoder must check whether a contour has an area (absolute value) smaller than the previous contour, as the outer contour would have an area greater than the sum of all inner ones.

Recommendations:

- For single MVT tilesets containing multiple layers, a mechanism should be defined to allow listing the individual layers, along with their geospatial primitive type (points, lines or polygons). There is currently no way to do this without requesting every single possible tile, as empty layers are typically not included if the features list is empty for a given tile.
- The possibility of selecting one of two possible approaches to avoid unwanted edges of polygons when drawing strokes, as well as allowing feature reconstruction should be offered: an extra border as is used in Mapbox vector tiles, or marking artificial segments (e.g. http://ogc.standardstracker.org/show_request.cgi?id=515). It should be possible for clients and services to implement either or both approaches (the Ecere WMTS and clients support both, but no parameters have been defined to handle negotiation, and no standard mechanism for marking edges within GML and GeoJSON have been agreed upon).
- Vector tiles should define an ID for features (including making use of the currently optional and ambiguous 'id' of the Mapbox vector tile specifications) which should match the feature ID before the vector data was tiled. This facilitates recombining the geometry, without having to rely on the attributes (e.g. MVT tags).
- The upcoming candidate for describing tiling schemes (tile matrix sets) across OGC standards should offer additional flexibility, such as the ability to describe tile matrices of different widths at different latitudes which would support both the GNOSIS Global Grid [http://docs.opengeospatial.org/per/17-041.html#_global_gnosis_tiling_scheme_adapted_to_polar_regions], as well as the CDB [https://portal.opengeospatial.org/files/?artifact_id=72714] grid, as previously raised in Testbed 13 (CR 518 [http://ogc.standardstracker.org/show_request.cgi?id=518]).

5.3.5. Mapbox

The Mapbox WMTS client runs as a web-browser based application. The client application allows users to visualize a number of feature collection layers and toggle between base maps. The client also displays information about layers found under the mouse pointer. A screenshot showing the client application is presented in [img_mapbox_client.png].

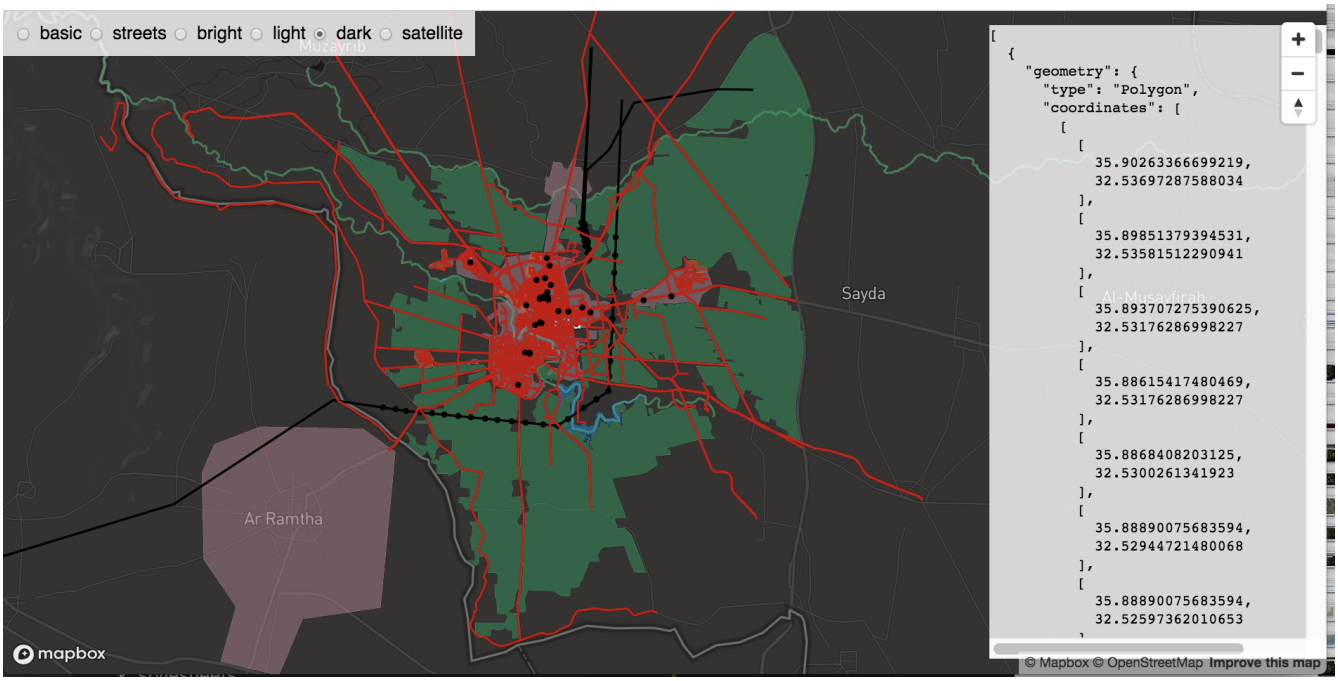

Figure 18. Vector Tiles rendered on the Mapbox Client

5.4. TIE Matrix

Table 2. TIE Matrix

CLIENTS	SERVERS			
	CubeWerx	Ecere	GeoSolutions	Mapbox
Compusult	X	X	X	X
Ecere	X	X	X	X
GeoSolutions			X	
MapBox				

Chapter 6. Discussion

6.1. Specification changes

The WMTS services deployed in this pilot demonstrated that no changes to the WMTS standard were necessary to support MVT. This is due to the extensibility of the WMTS standard. The GetCapabilities response of a WMTS allows for several 'Format' elements to be identified in a 'Layer' element through a one-to-many multiplicity. This means that any number of formats could be supported by a WMTS. MVT was therefore simply included as an additional output format offered by the WMTS.

Requirements Class: Core	
http://www.opengis.net/spec/wmts/1.0/vt/req/core	
Target type	
Requirement 1	http://www.opengis.net/spec/wmts/1.0/vt/req/core/getcapabilities
Requirement 2	http://www.opengis.net/spec/wmts/1.0/vt/req/core/gettile

The Pilot participants observed that there are different media types that potentially could be used to represent MVT. However, as the Internet Assigned Numbers Authority (IANA) has registered application/vnd.mapbox-vector-tile as the media type for MVT [2: https://www.iana.org/assignments/media-types/media-types.xhtml] OGC standards should use the IANA-registered media type to identify the MVT media type.

Requirement 1	http://www.opengis.net/spec/wmts/1.0/vt/req/core/getcapabilities Implementations of this WMTS extension should list the following as a supported media type in the GetCapabilities response application/vnd.mapbox-vector-tile .

The GetTile operation supports extensibility by allowing a client application to specify the format in which the tile being retrieved should be encoded. This is achieved through the 'Format' parameter. An implementation of a tiled feature data extension of WMTS would therefore need to allow client applications to specify the MVT media type or a future IANA-registered OGC media type as a value of the 'Format' parameter.

Requirement 2	http://www.opengis.net/spec/wmts/1.0/vt/req/core/gettile Upon receiving a GetTile request that has the MVT media type as a value of the Format parameter, an implementation of the Vector Tiles extension of WMTS shall respond with an MVT file containing features data for that tile

6.2. DDIL networks

6.2.1. Introduction

The purpose of this clause is to discuss the operation of WMTS servers and clients in denied, degraded, intermittent or limited bandwidth (DDIL) networks and how the work done in this pilot addresses some of the requirements of DDIL a well as highlight areas where additional work needs to be done.

DDIL networks are networks that have limited bandwidth, sporadic connectivity and/or no connection to the internet. Such environments pose a particular challenge to OGC web services which assume a reliable and constant connection to the internet.

However, several existing and emerging technologies being developed by OGC and others enhance the suitability of OGC web services (WMTS included) in DDIL networks.

6.2.2. Compression and serialization

In a DDIL network it is important to take best advantage of available connectivity and bandwidth. One aspect of this, is the use of compression and serialization to maximize the utility of the available bandwidth. The work of this pilot, with its use of Mapbox Vector Tiles, directly addresses this compression and serialization requirement. Mapbox Vector Tiles, that use Google Protobuf, are a highly compressed tile output format that is quite suitable for use in a DDIL network.

6.2.3. Asynchronous methods

Asynchronous methods, that do not rely on the request-reply paradigm of typical web services, are particularly suitable for DDIL networks. Service requests can be initiated by a connected client and then processed by the WMTS in the background while the client is disconnected. When the client achieves connectivity once again, a notification can be sent to the client and the response to the original request may be retrieved.

At the moment, WMTS does not support asynchronous requests and, in general, asynchronous requests do not make much sense in the "single-tile-at-a-time" architecture currently supported by the WMTS. A better approach would be to specify a "GetTiles" operation that can be viewed as a "batch" GetTile request that allows multiple tiles to be retrieved in a single request. An asynchronous GetTiles operation would thus allow a client to make requests for multiple tiles, disconnect and then retrieve the tiles once connectivity has again been established.

The short-form implementation details for adding asynchronous request processing to OGC services are:

- add a parameter named `responseHandler` to the WMTS API
- the value of the `responseHandler` parameter is a comma-separated list of URIs
- the URIs specify targets to receive notifications when the asynchronous job is completed
- examples include: mailto:pvretano@cubewerx.com, sms:4165551212, etc.
- the presence of the `responseHandler` parameter on a request triggers asynchronous processing

- if the server successfully accepts the asynchronous request, it generates an acknowledgement message indicating that the request has been accepted and optionally containing links that may — when connected — be used to monitor the execution status of the asynchronous job or to cancel the asynchronous job
- when the server completes processes the request in the background, a message is sent to each notification target
- in a DDIL network, the specified notification protocol (e.g. email, SMS, etc) handles the delivery of the notification

The specific implementation details of adding asynchronous request capability to an OGC service can, most recently, be found in AnnexA of OGC 18-045, "OGC Testbed-14: Next Generation Web APIs - WFS 3.0 Engineering Report".

6.2.4. PubSub methods and synchronization

Asynchronous methods are a on-time request and eventual response cycle. In some cases, however, longer life-time or even standing requests are required. Typically, such requests are handled using polling to make periodic service requests to detect and retrieve changes. In the limited and sporadic connectivity found in a DDIL network, polling is challenging.

A better approach is to use a publish/subscribe model. OGC has two standards related to publish/subscribe. The first is the OGC® Publish/Subscribe Interface Standard 1.0 - Core [http://docs.opengeospatial.org/is/13-131r1/13-131r1.html] standard built upon the Web Services Base Notification 1.3 (WS-BaseNotification) [http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf] model of the Organization for the Advancement of Structured Information Standards (OASIS) [https://www.oasis-open.org/].

The second, that uses the OGC® Publish/Subscribe Interface Standard 1.0 - Core [http://docs.opengeospatial.org/is/13-131r1/13-131r1.html], is the **draft** OGC Geo-Synchronization Standard [https://portal.opengeospatial.org/files/?artifact_id=76559&version=1](GSS).

Using a GSS, a client registers their interest in a "publication". A "publication" is a subset of data defined using a filter on some larger datastore. The filter used to define a topic is a general filter and can include spatial, temporal and scalar predicates to identify a specific subset of data of interest. Having registered a "topic" (i.e. having subscribed), the GSS then monitors the "publication" and whenever a change occurs, the GSS can passively or actively notify the client (once connectivity has been re-acquired) that new data is available. Passive notification can take the form of, for example, an email or SMS message. Active notification has the GSS pushing the new data to the client automatically.

With respect to the WMTS and this pilot, a GSS can be configured to push sets of new or updated tiles of interest to a client whenever connectivity is achieved. This capability is analogous to Google Map's Offline Map capability; the tiles in a region of interest are downloaded to a client. The client goes offline and uses the downloaded tiles on a mobile device; when the client reconnects, the offline tiles are synchronized with a source server.

6.2.5. Quality of service extensions

OGC has, since 2016, been developing extensions to service description metadata to include Quality-of-Service (QoS) metrics [https://portal.opengeospatial.org/files/?artifact_id=81584&version=1]. The metrics include such concepts as availability, capacity, performance and service profiles. The idea is that the declared QoS metadata will contribute to a complete picture of the QoS of the entire Spatial Data Infrastructure (SDI). With this QoS information, agile clients can be developed that react to limitations of a DDIL network. For example, using smaller page sizes for results or requesting compact formats from servers on unreliable and/or low-bandwidth networks.

Using some of the concepts discussed thus far in this clause, a DDIL profile of WMTS might be developed that includes:

- a "GetTiles" request for batch fetching of tiles
- support for MVT and/or GeoPackage as output formats
- support for asynchronous requests

6.2.6. Caching and local data stores

Most OGC web services are typically backed by a data store, a stand-alone subset of which — in a DDIL network — can be cached or stored locally on a device. Tiled feature data is particularly amenable to this treatment because a static WMTS can be easily set up by copying the desired subset of tiles to the device and creating a static WMTS capabilities document on the device that describes the necessary access templates. This pseudo-service arrangement allows existing WMTS clients installed on the device to continue to operate using the WMTS API.

Another stand-alone database format that can be used to cache data locally on a device is GeoPackage. For this Pilot, a GeoPackage extension was defined to also store tiled feature data (see OGC 18-74)(see OGC 18-074 [https://portal.opengeospatial.org/files/?artifact_id=81680&version=1]). Of course, access to tiled feature data stored in GeoPackage would require more capabilities from WMTS servers and/or clients — ones that know how to read a GeoPackage — than those required for the pseudo-service arrangement described in the previous paragraph.

Finally, most modern browsers support an "offline" operating mode. This is made possible because of the service worker API [https://www.w3.org/TR/service-workers-1/] specification that, among other capabilities, supports asset caching to allow browser apps to continue to operate even with the loss of network connectivity. In the case of WMTS and tiled feature data, service workers could be configured to cache tiles and allow web-based WMTS applications to operate without network connectivity.

Appendix A: Abstract Test Suite

A.1 Conformance class: Core

Table 3. A.1.1WMTS Vector Tiles - GetCapabilities

Test identifier	http://www.opengis.net/spec/wmts/1.0/vt/conf/core/getcapabilities
Test purpose:	To verify that an implementation of the Vector Tiles extension of WMTS correctly advertises support for the MVT format.
Test method:	Inspect the GetCapabilities response and confirm that it contains at least one Layer that has a 'Format' element with the value application/vnd.mapbox-vector-tile
Requirement:	http://www.opengis.net/spec/wmts/1.0/vt/req/core/getcapabilities
Test type:	Basic

Table 4. A.1.2WMTS Vector Tiles - GetTile

Test identifier	http://www.opengis.net/spec/wmts/1.0/vt/conf/core/gettile
Test purpose:	To verify that an implementation of the Vector Tiles extension of WMTS responds with an MVT file containing feature data for that tile when the service receives a Format parameter with the value application/vnd.mapbox-vector-tile
Test method:	Inspect the GetCapabilities response and confirm that it contains at least one Layer that has a 'Format' element with the value application/vnd.mapbox-vector-tile
Requirement:	http://www.opengis.net/spec/wmts/1.0/vt/req/core/gettile
Test type:	Basic

Appendix B: Revision History

Table 5. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
September 13, 2018	P. Vretanos	1.0	preface	got document number for the ER
September 22, 2018	P. Vretanos	1.0	various	added initial content to the ER
September 28, 2018	K. Maia	1.0	various	added Mapbox content to the ER
November 10, 2018	P. Vretanos	1.0	Clause 1,6	Added discussion about DDIL networks

Appendix C: Bibliography

1. Meek, S.: OGC Vector Tiles Pilot: Summary Engineering Report. OGC 18-086,Open Geospatial Consortium, http://www.opengeospatial.org/docs/er (2018).
2. Cavazzi, S.: OGC Testbed-13: Vector Tiles Engineering Report. OGC 17-041, Open Geospatial Consortium, http://docs.opengeospatial.org/per/17-041.html (2018).
3. Mapbox: Mapbox Vector Tile Specification - version 2.1, https://www.mapbox.com/vector-tiles/specification/, (2016).