DESIGN AND DEVELOPMENT OF THE UN VECTOR TILE TOOLKIT

H. Fujimura [1, *], O. Martin Sanchez [2], D. Gonzalez Ferreiro [2], Y. Kayama [3, 4], H. Hayashi [3, 5], N. Iwasaki [3,6], F. Mugambi [7], T. Obukhov [1], Y. Motojima [8], T. Sato [8]

[1] Geospatial Information Section, United Nations, New York, USA – fujimura.hidenori@gmail.com, obukhov@un.org
[2] Global Service Centre, United Nations, Brindisi Italy - (martinsanchez, gonzalezferreiro)@un.org
[3] OSGeo Foundation Japan Chapter, Kawagoe, Japan – yoichi.kayama@gmail.com, hayashi@apptec.co.jp, wata909@gmail.com
[4] Aero Asahi Corporation, Kawagoe, Japan – youichi-kayama@aeroasahi.co.jp
[5] Applied Technology Co., Ltd., Osaka, Japan – hayashi@apptec.co.jp
[6] Institute for Agro-Environmental Sciences, NARO, Tsukuba, Japan – niwasaki@affrc.go.jp
[7] United Nations Truce Supervision – UNTSO, Jerusalem, Israel – mugambi@un.org
[8] Geospatial Information Authority of Japan, Tsukuba, Japan – (motojima-y96st, satoh-t96b2)@mlit.go.jp

Commission IV, WG IV/4

KEY WORDS: Community, GeoJSON Text Sequences, Module, Software Development, Stream, Vector

ABSTRACT:

The UN Vector Tile Toolkit (https://github.com/un-vector-tile-toolkit/) is a package of open source tools designed under the UN Open GIS Initiative to enable public basemap providers, such as the UN geospatial information services or mapping organizations of governments, among others, to deliver their basemap vector tiles leveraging the latest web map technologies. The toolkit provides a set of Node.js open source scripts designed for developers to use with existing and proven open-source software such as Tippecanoe, Maputnik and vt-optimizer. The toolkit will help organizations to produce, host, style, and optimize fast and interoperable basemap vector tiles, making them available with various application frameworks. The talk will cover automatic and continuous updates of basemap vector tiles using a continuously updated PostGIS database which stores both the UN mission-specific basemap data and global OpenStreetMap data. The talk also focuses on how the project ensured interoperability with different existing enterprise geospatial software frameworks that use less-advanced web map libraries. The project aims to build a sustainable community of developers that support the provision of fast and interoperable basemap vector tiles.

1. BACKGROUND

Presentation of maps is transforming, due to the digitalization of information environment and the mobilization of information terminals. Responsive and real-time presentation of the maps in the field environment is becoming indispensable functions in modern web maps.

In this article, web maps refer to maps presented in information terminals including mobile ones by using Internet connections. In this section, we shortly review the need for vector tiles in web maps.

1.1 Need for Tiles

Fast and real-time map presentation in web maps has been studied from the last century. There were many examples in streaming mosaiced maps observing spatial locality of map data extraction and aiming to minimize the latency between each information request and information presentation.

Such mosaiced maps are called 'tiles.' A tile is defined as 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 (Open Geospatial Consortium, 2018).

Nowadays, all large-scale web maps with practical use employ tiles.

1.2 Need for Vector Tiles

From old days, maps often consist of points, lines, and polygons. In such cases, it is natural and reasonable to handle maps in vector form.

However, in reality, digital maps are first managed in image forms and then in vector forms, because of the graphics performance of computers or some constraints in information management. This compromise in using image forms has been seen in web maps, too.

Google Maps, which symbolizes Web 2.0, started the service with image tiles. It was in 2010 that Google Maps 5.0 for Android introduced vector tiles in dominant web map platforms (Google, 2010). The choice of Android as the first platform for vector tiles suggests that vector tiles require high-performance vector graphics.

Vector tiles enable dynamic map styling and hyperlinking from features because vector data are sent to the information terminal directly. Also, if well-designed, vector tiles tend to be smaller than image tiles. For these reasons, vector tiles are already a standard in dominant web map platforms.

* Corresponding author

However, not all technologies in dominant web map platforms are open source. Vector tile technology has been too expensive for other players including public organizations because they need to develop everything from scratch.

1.3 Rise of Open Source Vector Tile Technology

Open source web map technology seemed to be catalyzed by the advent of the OpenStreetMap project in 2004. Since the late 2000s, open source web map technology has been implemented in parallel with the progress of the dominant web map platforms.

In the 2010s, several companies made progress in developing open source vector tile technology. Release of Mapbox Vector Tile Specification (Agafonkin et al., 2014) and Mapbox GL (Mapbox, 2014) initiated the convergence of vector tile technology used by the players other than the most dominant web map platforms.

In 2015, Esri adopted the Mapbox Vector Tile Specification (Andrew Turner, 2015). Azure Maps, which was released to the public from Microsoft in 2018, uses Mapbox GL (Microsoft, 2018). Open Geospatial Consortium conducted in 2018 a Vector Tile Pilot to propose draft standards to support tiled feature data based on the Mapbox Vector Tile Specification (Open Geospatial Consortium, 2018).

As described above, the transition to vector tiles is inevitable in web maps. It is worth noting that the transition involves the use of open source methodology especially among players other than dominant web map platforms to share the key vector tile technology. Thanks to the open source methodology, the vector tile specification is loosely converging, which benefits both map providers and map users.

2. PURPOSE

2.1 Vector Tiles in Public Organizations

Although open source vector tile technology is emerging since the mid-2010s, public organizations, such as international organizations, governmental organizations, and non-governmental organizations, has not fully adopted the technology. This delay of adoption results in the use of more conventional web map technology than dominant web map platforms.

Web maps of public organizations are used not because their performance or functionalities are better than dominant web map platforms. In many cases, they are used because of their information content that fit into respective administrative purposes. This gap allows public organization web maps to make clients' duties more efficient, by catching up with dominant web map platforms using open source vector tile technology. The adoption of open source vector tile technology is also an opportunity to make their mapping efforts more cost-effective.

2.2 Prior Works

Transition to vector tiles in public organizations is pioneered by several geospatial information authorities using open source software and methodology.

Geospatial Information Authority of Japan has been conducting vector tile experiment since 2014 (Geospatial Information Authority of Japan, 2014).

Institut Cartogràfic i Geològic de Catalunya provides vector tile basemap within OpenICGC initiative (Institut Cartogràfic i Geològic de Catalunya, 2018).

Ordnance Survey officially released vector tile basemap titled OS Open Zoomstack on 28 January 2019 (Ordnance Survey, 2019).

2.3 Purpose of the UN Vector Tile Toolkit

Geospatial information service in the United Nations is also an early adopter of the vector tile technology by prototyping vector tiles.

In 2017, we initiated a project for UN Vector Tile Toolkit within Sprial 4 of the UN Open GIS Initiative.

The purpose of the UN Vector Tile Toolkit is to enable all players to catch up with the vector tile technology of dominant web map platforms.

In implementing the Toolkit, we make maximal use of existing fast and stable open source software while develop and share some mission functions and integrations in an open source manner.

We focus on public organizations as expected uses of the Toolkit.

3. STRATEGY

The base map is a most fundamental and a biggest data component with a longest lifespan. Therefore, the stability and sustainability of the toolkit is the most fundamental value. Another important value is the speed both in production and application use.

From this viewpoint, we decided to use existing reliable open source software such as Tippecanoe, Maputnik, MapboxGL, Vector Tile optimizer, and Osmium Tool. We also decided to implement automated production and vector tile server in Node.js, a single JavaScript runtime platform. Also, we decided to publish these Node.js scripts under open source licenses. This strategy led us to two major challenges as below.

3.1 Software challenges

We set four challenges to both to promote producing and application of basemap vector tiles inside the UN and also sustain the project by meeting the requirements of various public organizations:

1. To keep software flexible so that the software can be applied to various data sources.
2. To update global basemap vector tiles continuously and automatically.
3. To host vector tiles in a way where mobile field applications can use them.
4. To interoperate with existing enterprise geospatial frameworks by a single vector tileset.

Public organizations do not necessarily have enough resources for information technology infrastructure and expertise. Also, this project does not have plenty of resources for information technology infrastructure and expertise. Therefore, we reduce the use of middleware as possible to minimize the running and maintenance cost. Also, for flexibility and cost reduction, we eliminate packaging efforts and decided to expose internal modules transparently.

3.2 Community challenges

As a cost-neutral project started inside public organizations, sustainment of the project was a challenge.

We needed to continue to learn open source culture and needed to join the open source geospatial software community for the sustainment of the project.

4. DESIGN

4.1 Identification of tasks

We have identified four major tasks for the toolkit in Figure 1 and the list below.

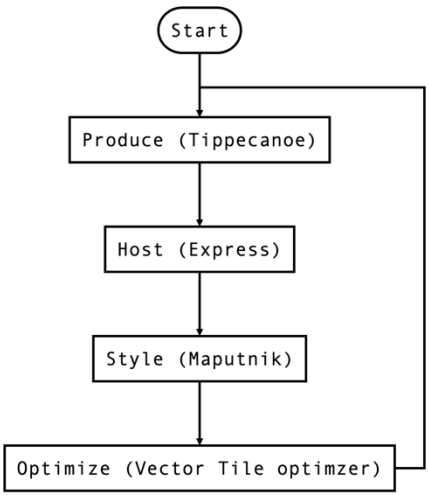

Figure 1. Tasks covered by the UN Vector Tile Toolkit

1. To produce vector tiles from source data, in such form as PostGIS database, Shapefile, or OpenStreetMap PBF, using Tippecanoe (Mapbox, 2014).
2. To host produced vector tiles using Express (Holowaychuk et al., 2009), a web framework for Node.js so that we can exploit existing geospatial libraries in JavaScript.
3. To style vector tiles in Mapbox Style using Maputnik (Martinelli, 2015).
4. To optimize the size of vector tiles by actively updating vector tile schema with assistance from Vector Tile optimizer (Besora Vilardaga, 2018).

4.2 Implementation strategies

To implement the tasks identified in 4.1, we have identified four implementation strategies as below.

4.2.1 Stream-oriented Module-based Production: Considering the size of global basemap data which is currently in the order of 100GB, we thought it was critical to reduce the footprint of the data and also divide the tasks to easily manageable parallel modules. In this view, we decided to use GeoJSON Text Sequences (Gillies, 2017) for processing geospatial data and interfacing between software components for vector tile production. We also decided to divide the globe into z=6 modules so that each vector tile package takes up to several gigabytes. Here a z=6 module means an extent that covers the area of a single tile of zoom level 6 of a slippy map tilename (OpenStreetMap Wiki contributors, 2019). This modularization also raises the opportunity for the parallel production of modules when we have enough computing resources.

4.2.2 On-the-fly Vector Tile Schema Modification: It was critically important for the project to encapsulate and separate dataset-specific details so that we can maximize the reusability of the code in addition to enforcing proper geospatial information management of secured UN Mission critical data. For that purpose, we devised a simple yet programmable interface called modify(f) to modify the vector tile schema on the fly. The data-specific codes are written in a JavaScript function named modify(f) that takes a GeoJSON Feature, and that returns a GeoJSON Feature. In this function, a vector tile designer can modify all three properties, i.e. geometric properties, thematic properties, and Tippecanoe properties which feed such as the layer name, minimum zoom, and maximum zoom. When we need to drop a specific given feature, the modify(f) function shall return null. In the vector tile production code, modify(f) is called after a feature is imported from the data source and before the feature is piped to Tippecanoe. By this on-the-fly modification method for vector tile schema, we could dramatically reduce disk access while keeping the memory usage small.

4.2.3 Module-wise Hosting: Static vector tile hosting from MBTiles (Fischer et al., 2011) is a great way for hosting vector tiles fast. However, merging MBTiles databases takes a significant amount of time and also reduces modularity of the vector tile product. In this view, we decided to host vector tiles from separate MBTiles databases for each z=6 modules.

4.2.4 Real-time Server-side Image Tile Rendering: Existing enterprise geospatial application frameworks are large-scale and therefore complex. As a result, the front-end web map libraries used in such frameworks are not state-of-the-art. Sometimes the front-end libraries used are aged a few years older than the newest one. In such case, direct use of vector tiles in the browser applications makes the application rather slow. In the history of web mapping, similar cases were there when the browser-side vector graphic rendering is in general weak. The was a generation of web maps called "2.75 generation" (MacWright, 2015) where tiles are stored as vector tiles, but sent to the client after rendering to the image tiles. We implemented real-time server-side image tile rendering using Mapbox GL Native for this 2.75 generation solution where necessary.

5. DEVELOPMENT

5.1 Production

We have developed several variations of vector tile production software depending on different data source. Our main focus remained in producing basemap vector tiles to be used internally inside the UN. In this primal use case, we make use of data from OpenStreetMap, Natural Earth, SRTM, GLC30, and UN internal geospatial data which includes data from UN Missions.

In this primal use case, the enterprise architecture is organized to centralize all the data into a PostGIS database. Therefore, we implemented data import from PostGIS using node-pg. Also, for benchmarking and testing purpose, we implemented data import from Shapefile using node-shapefile and ogr2ogr. We also implemented data import from OpenStreetMap PBF using osmium-tool.

For benchmarking and reference purposes, we released vector tile production software from planet.osm.pbf at github.com/un-vector-tile-toolkit/produce-320. With this software, we tried to focus on benchmarking the speed of vector tile production from a global dataset. In order to ensure a good speed, we have devised the following two techniques.

5.1.1 no-feature-module: It was said that 70% of the Earth surface is the ocean. Therefore, there are modules without any data. We defined such no-feature-modules (nfm) that are calculated from the actual OpenStreetMap data, which sums to 1669 modules out of 4096 modules that cover the whole globe. In Figure 2, the area with blue squares is identified as no-feature-modules. We released this data on github.com/un-vector-tile-toolkit/nfm. Thanks to no-feature-module, production of global vector tileset were reduced to production of 4096 - 1669 = 2427 vector tile modules.

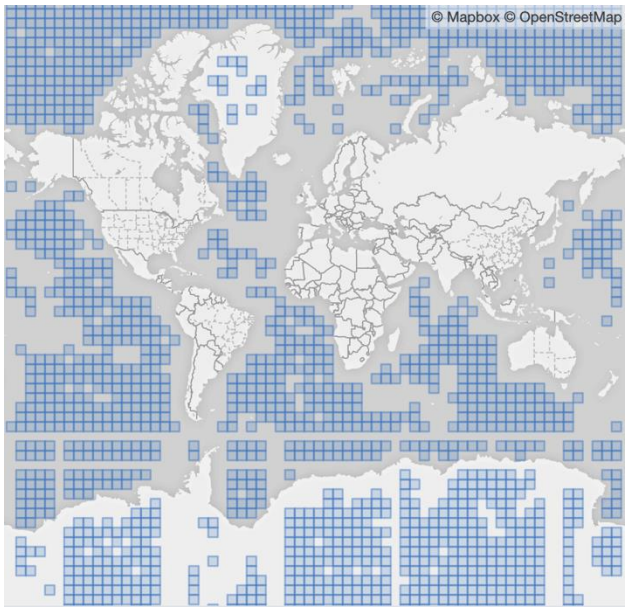

Figure 2. No-feature-modules

5.1.2 Division of the globe into 12 areas: As of April 2019, the size of planet.osm.pbf was as big as 45GB. Extracting geospatial data that covers a module took time because parsing a 45GB file was still a heavy-lifting for a PC. Extracting 2427 modules directly from a 45GB file was found inefficient. Therefore, we devised a set of areas each of which takes at most around ten gigabytes. A small study on the distribution of OpenStreetMap data found that the OpenStreetMap data is unevenly distributed. For example, continental European area took almost half of the OpenStreetM dataset in data size, while Arctic and Antarctic area are very sparse especially in slippy map tilenames which is based on Mercator projection. From our study and experiment, we have developed a rule to split the globe into 12 areas each of which takes a similar amount of disk space. Fig. 3 shows our division rule which we called 'duodecim.' We shared this division rule on github.com/un-vector-tile-toolkit/duodecim though we believe there is much space for improvement.

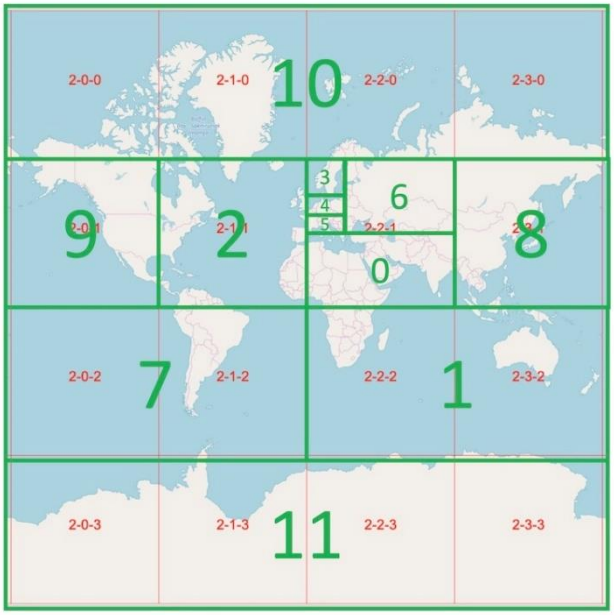

Figure 3. Division of globe into 12 areas based on the distribution of OpenStreetMap data

5.2 Hosting

We have also developed several variations of vector tile servers. A standard implementation is published at github.com/un-vector-tile-toolkit/onyx. The implementation makes use of HTTP/2 protocol to accelerate the access to vector tiles.

5.2.1 Real-time Server-side Image Tile Rendering: On the top of the architecture of the 'onyx' above, we prototyped real-time server-side image tile rendering at github.com/un-vector-tile-toolkit/carbon using node-mapbox-gl-native. The first implementation of the server that provides vector tiles and real-time rendered image tiles is at github.com/un-vector-tile-toolkit/moai. The server is implemented as a simple Node.js script which is less than 140 lines of code.

5.3 Styling and Optimization

For styling and optimization, there was no development required. We used Maputnik for assisting configuration of the

Mapbox Style description. We used Vector Tile optimizer for assisting the measurement of the size of vector tiles.

With assistance from these tools, we repeatedly updated style.json and modify.js to obtain optimal vector tiles and its styling settings.

6. PRODUCTION PERFORMANCE

6.1 Around the World in 80 Hours with OpenStreetMap

We have measured the performance of the production of vector tiles with the specification in Table 1.

Table 1. Specification for the production time measurement

Source data	planet-190429.osm.pbf
Computer	MacBook Pro (13-inch, 2017, Two Thunderbolt 3 ports) with 2.3GHz Intel Core i5 and 8GB 2133MHz LPDDR3
Storage	Sandisk Extreme 900 (480GB)

Table 2 shows the measured production time for each area in Fig. 3 and the whole world as a sum.

Table 2. Production time of global vector tiles divided by 'duodecim' areas

Area	**OSM PBF size**	**Number of modules**	**Production time** (d: day, h: hour, m: minutes)
#0	4.9GB	128	6h 53m
#1	4.9GB	512	13h 19m
#2	12GB	256	19h 36m
#3	1.5GB	16	1h 9m
#4	9.3GB	8	4h 0m
#5	8.0GB	8	3h 24m
#6	5.4GB	96	5h 54m
#7	2.0GB	512	4h 21m
#8	4.8GB	256	8h 25m
#9	6.0GB	256	8h 36m
#10	510MB	1024	2h 42m
#11	29MB	1024	1h 3m
World	45GB	4096	79h 22m (3d 7h 22m)

According to github.com/openmaptiles/openmaptiles/issues/242, OpenMapTiles requires 37 days, which is as long as around 900 hours, to produce global vector tileset with its default production script. Figure 4 shows rendering result of produced vector tiles.

Figure 4. Produced Vector Tiles rendered on a Browser

6.2 Production of Other Layers

Although not for public use, we also produced vector tiles from data other than OpenStreetMap. Table 3 shows the size of the produced MBTiles packages.

Table 3. Size of produced MBTiles packages

theme	**MBTiles size**
Hill shade	218MB
Landcover	4.8GB
Ocean	4.9GB
Other UN data	1.5GB
OpenStreetMap	77GB

7. COMMUNITY DEVELOPMENT

7.1 OSGeo.JP Workshop for UN Vector Tile Toolkit

We conducted a workshop, titled the OSGeo.JP Workshop for the UN Vector Tile Toolkit, to introduce the methodology used in the UN Vector Tile Toolkit with FOSS4G experts with support from the OSGeo Foundation Japan Chapter in December 2018 as a pre-conference workshop for FOSS4G Asia 2018. The workshop materials and software are published with open source license at github.com/un-vector-tile-toolkit.

In a keynote presentation of FOSS4G Asia 2018, we have announced the release of the UN Vector Tile Toolkit on December 3, 2018.

7.2 Application by the Community Partners

In addition to the application for the basemap vector tiles for the internal use in the UN, the UN Vector Tile Toolkit is used for the following applications.

7.2.1 Geospatial Information Authority of Japan (GSI): GSI has been prototyping its vector tiles from several years ago. As a partner of the UN Vector Tile Toolkit, they use the Toolkit to produce their prototype vector tiles. Their feedback was useful to improve the codebase of the UN Vector Tile Toolkit.

7.2.2 CartoTiles: UN Geospatial service is planning to release vector tiles for geospatial visualization in very small map scale. UN Vector Tile Toolkit is used for the production and update of such vector tiles.

7.2.3 Tabular Maps: The project for Tabular Maps is external to the UN Vector Tile Toolkit, aiming for visualizing administrative bodies by an array of simple squares rather than the realistic administrative area polygons. The UN Vector Tile Toolkit is used to produce vector tiles to visualize tabular maps using Mapbox GL JS.

7.2.4 Institute for Agro-Environmental Sciences (NARO): In NARO one of the authors developed vector tiles of agricultural plot polygons, which were provided as Open Data from the Ministry of Agriculture, Forestry and Fisheries (MAFF) of Japan, by making use of the UN Vector Tile Toolkit.

8. CONCLUSION

We started the UN Vector Tile Toolkit inside the UN Open GIS Initiative, aiming for supporting public organizations, such as

the UN and government organizations, to provide basemap vector tiles.

Future extensions of the UN Vector Tile Toolkit will include:

- delta update using expiretiles
- deployment with nginx
- Docker implementation

It remains a crucial challenge for the UN Vector Tile Toolkit to formulate, develop, and sustain a community of developers by appropriate documentation. Also, we need to keep the following two principles to sustain the UN Vector Tile Toolkit.

1. We need a good networking mechanism of vector tile experts distributed in different organizations and different geographical areas. Vector tile experts tend to isolate in their respective work areas because vector tile technology is highly specialized.
2. We need a focus on technical topics by carefully choosing appropriate common challenges in the project.

ACKNOWLEDGMENTS

The authors would like to acknowledge various advice from the UN Open GIS Initiative, especially from Professor Maria Antonia Brovelli, Professor Ki-Joune Li and Mr Kyoung-Soo Eom. The authors would also like to acknowledge kind support from the UN Global Service Centre regarding computing and networking infrastructure. The lead author also would like to thank Dr Isaac Besora Vilardaga for a fruitful discussion about how best the size statistics of vector tiles shall be presented.

REFERENCES

Agafonkin, V., Firebaugh, J., Fischer E., Käfer K., Loyd C., MacWright T., Pavlenko A., Springmeyer D., Thompson B., 2014. Mapbox Vector Tile Specification, github.com/mapbox/vector-tile-spec (May 1, 2019).

Besora Vilardaga, I., 2018. Vector Tile optimizer, github.com/ibesora/vt-optimizer (May 1, 2019).

Fischer, E., Kaefer, K., MacWright, T., Miller, J., Norman, P., Thompson, B., White, W., 2011. MBTiles Specification, github.com/mapbox/mbtiles-spec (May 1, 2019).

Geospatial Information Authority of Japan, 2014. GSI Vector Tile experiment, github.com/gsi-cyberjapan/vector-tile-experiment (May 1, 2019).

Gillies, S., 2017. GeoJSON Text Sequences, doi.org/10.17487/RFC8142 (May 1, 2019).

Google, 2010. Under the hood of Google Maps 5.0 for Android, googleblog.blogspot.com/2010/12/under-hood-of-google-maps-50-for.html (May 1, 2019).

Holowaychuk, TJ., Shtylman, R., Wilson, D.C., 2009. Express, github.com/express/expressjs (May 1, 2019).

Institut Cartogràfic i Geològic de Catalunya, 2018. OpenICGC, openicgc.github.io (May 1, 2019).

MacWright, T., 2015. How we got here. tmcw.github.io/presentations/jsgeo (May 1, 2019).

Mapbox, 2014. Mapbox GL, docs.mapbox.com/help/glossary/mapbox-gl (May 1, 2019).

Mapbox, 2014. Tippecanoe, github.com/mapbox/tippecanoe (May 1, 2019).

Martinelli, L., 2015. Maputnik, github.com/maputnik/editor (May 1, 2019).

Microsoft, 2018. Microsoft Software License Terms - Microsoft Azure Maps SDK Preview, atlas.microsoft.com/sdk/js/atlas.min.js?api-version=1.2 (May 1, 2019).

Open Geospatial Consortium, 2018. OGC Vector Tiles Pilot: Summary Engineering Report, docs.opengeospatial.org/per/18-086r1.html (May 1, 2019).

OpenStreetMap Wiki contributors, 2019. Slippy map tilenames, wiki.openstreetmap.org/wiki/Slippy_map_tilenames (May 1, 2019).

Ordnance Survey, 2019. OS Open Zoomstack released today, www.ordnancesurvey.co.uk/blog/2019/01/os-open-zoomstack-released-today (May 1, 2019).

Turner, A., 2015. Vector Tiles preview, www.esri.com/arcgis-blog/products/arcgis-enterprise/mapping/vector-tiles-preview (May 1, 2019).