MAPA BASE DE INFORMACIÓN GEOGRÁFICA OFICIAL CON TESELAS VECTORIALES

1. Qué son las teselas vectoriales:

En primer lugar hacer una breve introducción sobre lo que son las teselas vectoriales.

Las teselas vectoriales son un formato de datos liviano que nos permite almacenar datos vectoriales geoespaciales, como puntos, líneas y polígonos. Las teselas vectoriales codifican información geográfica de acuerdo con la especificación de teselas vector de Mapbox. La especificación de Mapbox es un estándar abierto bajo una licencia Creative Commons Attribution 3.0 US.

Una tesela vectorial (vector tiles) contiene datos vectoriales georreferenciados (puede contener múltiples capas), recortados en teselas para facilitar su recuperación. Son equivalentes a las teselas raster tradicionales (WMTS, TMS) pero retornan datos vectoriales en lugar de una imagen.

Cada conjunto de teselas vectoriales tiene su propio esquema. Un esquema consiste en nombres de capas, atributos, selección de elementos, etc.

No existe un esquema que sirva para todo. Existen varios esquemas como por ejemplo: OpenMapTiles, Mapbox Streets, etc.

Las teselas vectoriales no son un formato de datos vectoriales estilo Shapefile, GeoJSON, etc. pensado para trabajar (hacer análisis, explotación de datos, etc.) sino que está pensado y enfocado principalmente para la visualización.

Otra definición:

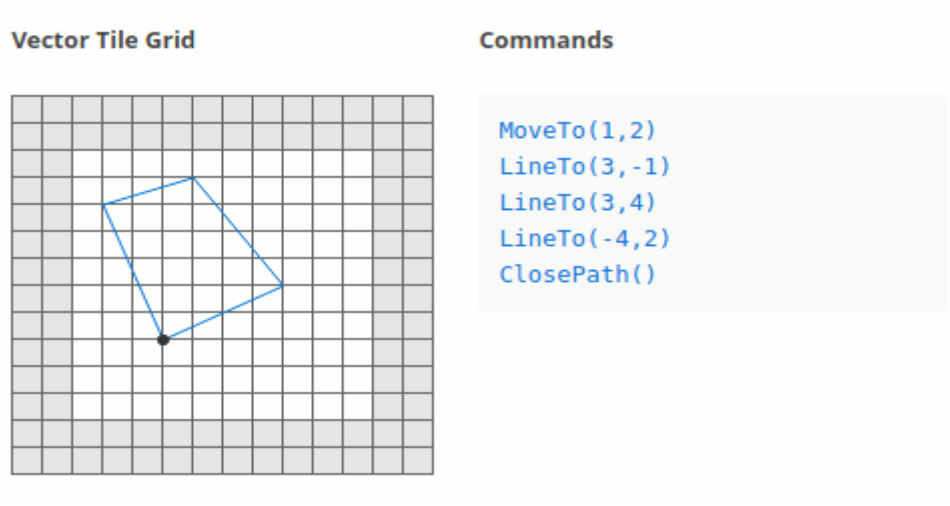
2. Cómo están hechas por dentro:

Las geometrías y los atributos se codifican como datos binarios de Google Protobuf (PBF).

3. Cómo codificar las geometrías:

Para codificar información geográfica en una tesela vectorial, una herramienta debe convertir las coordenadas geográficas, como la latitud y la longitud, en coordenadas vectoriales de cuadrículas. Las teselas de vectoriales no tienen ningún concepto de información geográfica. Codifican puntos, líneas y polígonos como pares x/y relativos a la esquina superior izquierda de la cuadrícula de forma descendente.

Las geometrías son transformadas a una sola tesela, con un sistema de coordenadas de píxel local, que por defecto va de la esquina superior izquierda (0,0) a la esquina inferior derecha (4096,4096).



Codificar geometría. Fuente https://www.mapbox.com/vector-tiles/specification/#encoding-geom

4. Cómo codificar los atributos:

Los atributos se codifican como un conjunto único de claves (algo así como un esquema de campos de capa) y la lista de sus valores.

Los atributos están codificados en una serie de etiquetas que existen dentro de un elemento en el vector que tienen valores enteros que hacen referencia a las claves y los valores que provienen de la geometría. Esto elimina la redundancia de los atributos para geometrías que tienen las mismas claves y valores similares.

Original geojson

```
{
  "type": "FeatureCollection",
    "features": [
    {
      "geometry": { ... },
      "type": "Feature",
      "properties": {
         "hello": "world",
         "h": "world",
         "count": 1.23
      }
    },
    {
      "geometry": { ... },
      "type": "Feature",
      "properties": {
         "hello": "again",
         "count": 2
      }
    }
  ]
}
```

Final vector tile

```
layers {
  version: 2
  name: "points"
  features: {
    id: 1
    tags: 0
    tags: 0
    tags: 1
    tags: 0
    tags: 2
    tags: 1
    type: Point
    geometry: ...
  }
  features {
    id: 2
    tags: 0
    tags: 2
    tags: 2
    tags: 3
    type: Point
    geometry: ...
  }
  keys: "hello"
  keys: "h"
  keys: "count"
  values: {
    string_value: "world"
  }
  values: {
    double_value: 1.23
  }
  values: {
    string_value: "again"
  }
  values: {
    int_value: 2
  }
  extent: 4096
}
```

Codificar atributos. Fuente https://www.mapbox.com/vector-tiles/specification/#encoding-attr

5. Especificaciones y conceptos relacionados con las teselas vectoriales.

pbf

Protocol buffers desarrollado por Google (para uso interno) es un método para serializar datos estructurados. Es *language-neutral*, *platform-neutral* y en cuyo objetivo de diseño enfatizaron la simplicidad y el rendimiento.

El método implica un lenguaje de descripción de interfaz que describe la estructura de algunos datos y un programa que genera código fuente a partir de esa descripción para generar o analizar una secuencia de bytes que representa los datos estructurados.

mvt

Formato binario basado en la especificación de Mapbox que usa **pbf** para serializar datos geográficos. Los archivos deberían tener extensión .mvt pero no es obligatorio

así que se pueden encontrar archivos con extensión .pbf, .vector.pbf o .mvt.gz (compresión gzip)

Por ejemplo un conjunto de teselas mvt almacenadas en un SQLite siguiendo una esquema específico formaría un MBTiles.

MBTiles

MBTiles es un **formato de archivo para almacenar conjuntos de teselas**. Está diseñado para que pueda empaquetar los potencialmente miles de archivos que componen un conjunto de teselas y moverlos, usarlos en una aplicación web o móvil. **MBTiles es una especificación abierta y se basa en la base de datos SQLite**. MBTiles puede contener conjunto de teselas reaster y/o vector.

MBTiles es una especificación compacta y restrictiva. Sólo admite datos teselados, incluidas teselas vectoriales o de imágenes y UTFGrid (hasta la versión 1.2). Sólo la **proyección esférica de Mercator** es soportada para la presentación (visualización) y sólo se admiten coordenadas de latitud y longitud para metadatos, como límites y centros.

Es una especificación mínima, que soló específica las formas en que los datos deben ser recuperables. Por lo tanto, los archivos MBTiles pueden comprimir y optimizar datos internamente, y construir vistas que se adhieren a la especificación MBTiles. Dentro del MBtiles los vectores están almacenados comprimidos (gzip) y en formato pbf.

A diferencia de Spatialite, GeoJSON y Rasterlite, MBTiles no es un almacenamiento de datos sin formato. Es almacenamiento de datos en teselas.

Formato de MBtiles: El formato MBTiles es un formato pensado en origen para web, donde la cartografía está estructurada por niveles de zoom y cortada en teselas para poder servirse con rapidez.

Se trata en realidad de una base de datos SQLite. Este formato también se ha evidenciado eficaz para el consumo de cartografía *offline* en dispositivos móviles, donde el conjunto de teselas queda almacenada en un solo archivo.

6. CÓMO ESTÁ COMPUESTO UN VECTOR TILE: (MBTILES) estructura de la base de datos

Tablas:

Metadata: campos Name y value

Bounds, center, description, format, generator, generator_options, json, maxzoom, minzoom, name, type, version

Tiles: campor zoom_level, tile_column, tile_row, tile_data

OGC – VECTOR TILES:

OGC está trabajando en la estandarización de un estándar de vector tiles.

OGC propone un Geopackage 1.2 para una extensión de vector tile.

Mapbox Vector Tiles Extension

The Mapbox Vector Tiles extension for GeoPackage defines the rules and requirements for encoding vector tiles in a GeoPackage Data Store, it is expressed as a draft standard. A key aspect to this extension is that the data are encoded using Google Protocol Buffers, which has many advantages discussed elsewhere in this document. This provides noticeable performance enhancements over other vector data formats and encodings.

GeoJSON Vector Tiles Extension

The GeoJSON Vector Tiles extension defines the rules and requirements for encoding GeoJSON Vector Tiles in the GeoPackage Data Store. It is not the focus of this pilot but may be utilized in future OGC initiatives. GeoJSON vector tiles are considered here as the GeoJSON format is useful for exchange with other services and vendors.

OWS Context Extension

The context document is a method of storing context of vector tiles as part of a GeoPackage file. OWS Context is an OGC standard that has encodings in Atom and GeoJSON. An OWS Context extension for GeoPackage is not yet adopted by OGC and remains a candidate standard. There are several requirements for this extension that are explored in the VT GeoPackage Extension ER.

Vector Tile Attribute Extension

Attributes can be stored as part of Vector Tiles Binary Large Objects (BLOBs) in GeoPackage, however enabling this extension allows vector tiles objects to be queried without having to

GeoPackage Producers

GeoPackages with Vector Tiles extensions were produced by Compusult, CubeWerx and Ecere.

GeoPackage Clients

There are several GeoPackage clients from a variety of vendors.

Geopackage: GeoPackage es un formato estándar, y según se define en el propio sitio web de la OGC (Open Geospatial Foundation), se trata de un “estándar abierto, multiplataforma y compacto, para la transferencia de información geoespacial”. Así pues, a efectos prácticos se trata de un formato de datos espaciales que, montado y extendido sobre la base de datos espacial SQLite, permite el almacenado de datos vectoriales, matrices de datos raster, atributos alfanuméricos y posibles extensiones.

7. PONER EN MARCHA UN SERVICIO DE TESELAS VECTORIALES

Para la construcción de teselas vectoriales se dispone de una máquina:

Linux CentOS – 90Gb RAM – 400Gb SSD

Compilar e instalar GDAL

(https://www.gdal.org)

Compilar e instalar Tippecanoe

(https://github.com/mapbox/tippecanoe)

CAPAS A TILEAr:

1. Se define el conjunto de capas a tilear. Se ha tomado como referencia la información geográfica oficial siguiente:

- Red de transporte (CNIG)
- Red hidrográfica (CNIG)
- Edificios (Catastro)
- Callejero: portales y viales urbanos (CNIG)
- Nomenclátor (CNIG)
- Líneas límite municipales (CNIG, IHM)
- Mapa Forestal (MITECO)
- Espacios naturales protegidos (MITECO)

2. Preparar la información para su tileado.

Conversión a .GeoJSON – Seq (ndjson)

- Convertimos con OGR2OGR las capas a GeoJSON en EPSG:4326 (Coordenadas Geográficas WGS84)
- Obtención de capas en formato .GeoJSON – Seq, con el comando OGR2OGR
- Zipeado de las capas resultantes

3. A continuación las capas se cortan en teselas. Se indica que capas en cada nivel de zoom. Mediante un archivo de configuración, se define que capas en cada uno de los niveles de zoom.

Tileado de capas

- Hacemos 2 tileados independientes:
    - Capas sin etiquetas
    - Capas con etiqueta
- Las capas con etiqueta requieren de un buffer mayor para que no se corten a la hora de representarlas
- Así se ahorra tamaño en el tile
- Utilizamos el comando Tippecanoe
- Obtenemos un fichero .**MBTiles** por cada nivel de Zoom
- Los ficheros GeoJSON de entrada están en EPSG:4326
- Si son GeoJSON – Seq, Tippecanoe utiliza todos los cores disponibles de la máquina (multiproceso)
- Se lanza del nivel 0 al 18
- Nivel 19 en adelante se utiliza la técnica del overzoom (*Overzooming* es una técnica específica de teselas vectorial que permite que una tesela se represente más allá de su nivel de zoom previsto, por lo que continúa siendo visible en el mapa. Si un conjunto de teselas tiene un minzoom de 6 y un maxzoom de 12, esos son los rangos válidos calculados por el generador de teselas. Si ampliara el mapa más allá del nivel de zoom 12, el renderizador del mapa puede seguir utilizando los datos del zoom 12 escalando los datos del vector hacia arriba. Las teselas raster pierden claridad si ocurre overzoom. Por ejemplo, si está visualizando un conjunto de teselas raster con una extensión de zoom entre z0 y z6, si hace un zoom a un nivel de zoom más alto después de z6, las imágenes se volverán borrosas y difíciles de ver. Los efectos del overzoom no son tan notables con las teselas vector, ya que los datos vectoriales no se almacenan en un formato basado en píxeles, sino que se codifican y calculan a partir de una serie de puntos, líneas y polígonos.
- Los tiles internamente no tienen sistema de coordenadas pero están pensados para representarlos en el EPSG:3857 (Proyección WGS84 / Pseudo-Mercator)
- La estructura de los tiles es TMS: ZYX (zoom, columna, fila)

Tenemos una segunda máquina, que es nuestro Servidor de Vector Tiles. En la máquina está instalado NodeJS, Mapnik y SQLite.

Los servicios que se ofrecen con esta máquina:

- Tiles en formato PBF, MVT (ts y tms)
- Tiles en formato GeoJSON (ts y tms)
- Tiles renderizados con Mapnik en formatos diversos (png, jpeg, webp, etc.) y diversos estilos (xml de Mapnik)

- Estilos .json para Mapbox GL
- Glyphs (tipos de letra) en formato PBF
- Sprites (iconos o imágenes) para Mapbox GL
- Metadatos del servicio en formato JSON (QGIS y Maputnik)

Vector Tiles – Cliente

Tipos de clientes:

- Leaflet, OpenLayers… tiles ráster
    - Poco consumo de CPU
    - Máxima compatibilidad
    - Todos los navegadores
    - Tiles muy pequeños: unos 10-20Kb
- MapBox GL, QGIS, Maputnik… tiles vector
    - Mucho consumo de CPU y GPU
    - Chrome y Firefox
    - Tile recomendable con máximo de 500Kb
    - Carga secuencial (se visualiza información durante la carga)

Página Web: https://vts.larioja.org

- Servicios
    - Servicio de teselas vectoriales de ámbito estatal
    - Servicio de teselas vectoriales de ámbito regional
    - Servicio de teselas vectoriales de ámbito global
    - Otros servicios
- Visualizadores
    - Leaflet, Mapbox GL, Here Maps, Cesium
- Editores
    - Maputnik, QGIS Desktop