Data Structure, spatial data on the web

Stefan Steiniger[1] & Andrew J.S. Hunter[2]

[1] Pontificia Universidad Católica de Chile**;** sstein@geo.uzh.ch
[2] University of Calgary; ahunter@ucalgary.ca

To be published in: D. Richardson et al. (2015?): *International Encyclopedia of Geography: People, the Earth, Environment, and Technology*, Wiley-Blackwell.

Abstract
The Internet is now the main source of information. Most human activities take spatial information into account, and spatial data over the web are prominent to our daily life. We look at spatial data when we want to know about weather conditions and forecasts, for travel directions when we are heading off on vacation, or for a better grasp of online news that can be mapped, e.g. a map of average family income per county. Because geographic data is big, and users dislike waiting to see a weather map or a travel route, specific data structures for efficiently transferring spatial data have been developed. These data structures are implemented as web data formats in which the geographic data can be "stored, sorted and packed". Examples of such formats include GML, KML, GeoJSON, GeoRSS, etc., and web data service standards such as WMS, WCS and WFS.

Keywords
geographic technologies, standards, Geographic Information Systems (GIS), spatial data transfer, OGC web services, GeoJSON, Tiling, WFS

---

1. Spatial data on the web and in our daily life: Where do we need spatial data and what are spatial data?

Everyone who has a computer or television at home looks at or uses spatial data that has been transferred over the Internet. Weather forecasts on TV or weather maps on the Internet are based on spatial data: temperature, humidity, precipitation, etc., that are gathered and sent mostly automatically from weather observation networks. These data may arrive every 10 minutes or so at a weather office, then derived maps, forecasts and other information are generated or updated. Similarly, spatial data are analyzed and transferred over the web when one searches for the shortest or fastest route to a vacation destination, when one gathers statistics on one's weekly biking or running activities, or when consulting traffic and location services using Google Maps, for example. Sensing services used to monitor and analyze earthquakes or to forecast flood risk need to transfer spatial data over the web. Similarly, many public sector activities including transportation, planning, and development require large volumes of spatial data. Authorities that collect spatial data, e.g., a forestry service, land registries, or a transportation agency, often send data to other agencies to plan and undertake day-to-day activities. Some uses are tolerant of long (slow) data responses, but other applications are time critical.

"Data structures for spatial data on the web" denotes specialized *data formats* that are used to *transfer spatial data* from one place to another place via *communication networks* and allow (almost) *immediate* display, manipulation and perhaps further processing of the data.

Note that either a person or a computer can be at the receiving end of the data transfer. In the case of a computer, data are transferred between two *web servers*. Here one web server "serves" data to another server, a.k.a. "processing web server", which may generate maps, statistics, etc. In the case of a person requesting such maps and statistics, by using their home PC, mobile phone, or a tablet, data is transferred between a *server* and a *client*. A client displays the data and may even improve data presentation. But the client does not serve the (processed) data to someone else.

The types of geographic data that are transferred over the web are similar to information that we see on daily news sites, nytimes.com for example, or social networks such as facebook.com. i.e., *images* and *text*. What makes the data "spatial data", however, is the fact that the data contains additional information, called "geographic coordinates", that allows us to relate the information to a place.

Spatial data are stored and transferred in two very distinct data formats: as *vector* and *raster data*. Both of these formats are described in detail below. However, to summarize: if we transfer raster data over the web, then we transfer some type of *image* format (e.g. a *.jpg or a *.png file), which stores "continuous" information in a grid structure — raster data examples include satellite images, terrain elevations, or temperature surfaces for a region. One or more discrete real-world objects, such as a bridge, a building or a forest, are transferred as vector data. When using vector data, each object is described by a *geometry*, a set of *attributes* that characterize the real-world object (see the example in Fig. 2), and often *styling information* for presentation of the information.

In the following paragraphs we will discuss important considerations when sending spatial data over the web, how data can be transferred from one location to another, and what data formats and standards exist for storing and transferring spatial data.

2. Important technical considerations for sending spatial data on the Web

Previously, spatial data formats have often been developed for use with desktop geographic information system (GIS) software to store data in some practical and convenient way on hard disks. File format development has focused on re-use of data with desktop GIS programs. That is, developers have been concerned about the speed of data read and write operations to disk, accessibility of file content, file size, etcetera. However, many initial formats were not developed to support interoperability that is necessary when data needs to be transferred from one computer to another. Hence, in the following we will have a look at requirements and constraints that are important when transferring spatial data via the Internet:

(1) Probably the most important thing for the average Internet map user is that they can see the map or map data as soon as possible after sending their request to a web server. Rapid *display* and *update* of online maps depends, among other minor factors, on available network bandwidth and the amount of map data to be transferred. Reducing data size is possible by restructuring the data, by generalizing the data (see *Generalization*), by using compression methods, or by sending only the data that the user really needs or wants at that very moment.

However, there are also other requirements: (2) the data's format should not be *prone* to transmission *failure* due to network *interruptions*. That is, if there is a network interruption, then the data do not need to be resent from the beginning, but can be restarted from where the interruption occurred. (3) Data formats need to be *independent of the computer platform*, i.e. a format can be read regardless of the operating system that the sending or receiving computer uses. This independence is also termed *interoperability* and discussed further below.

Depending on the particular use case of the data, it may be useful for the user to preview the data, choose the level of detail needed (which is similar to viewing maps at different cartographic scales), and select the themes and geographical areas that they are interested in — before sending a data request. This means on the one hand that (4) the spatial data format should contain or link to *metadata* that describe the datasets spatial extent, the phenomena included in the dataset (e.g. roads, or forests), the purpose of its generation, the year of creation, etc. — so that the user can decide if a dataset is of interest. (5) On the other hand, a data transfer service, which answers a user's request for data, should allow data browsing, as well as *content* and *spatial filtering* of the data to avoid sending out data that is irrelevant.

(6) Data formats should ensure *data consistency*. That means, when compression methods are used to reduce the size of the data, then the compression method should not change the data in any way. If we look at the geometry of two geographic objects such as a field and a forest that are adjacent to each other, then the relationship between their borders, their topology, before compression and after de-compression should be the same. (7) As final important criterion for spatial data web formats and services is the *separation* of *content* and *styling*. That is, a user may be interested in knowing about all the forests near a certain city for an environmental assessment; but the user may not be interested how the forest should be portrayed on a map, i.e. its fill color and pattern, line style, etc. In this case, sending map styling information is not important.

3. The importance of standards for sending spatial data over the Internet

When data are transferred between two computers the data server needs to understand what data is requested, and the client needs to understand the format used to send the spatial data so that the data can be stored, processed and displayed. *Syntactical interoperability* deals with information exchange between systems through the use of common data formats. Several organizations such as the *World Wide Web Consortium* (W3C), the *International Organization for Standardization* (ISO), and the *Open Geospatial Consortium* (OGC) have developed and published many XML-based formats, along with their data schemas, as well as standardized image formats such as JPEG, TIFF and PNG to support interoperability. W3C is mainly concerned with general Internet standards, whereas the OGC has a particular focus on standards for web mapping applications and spatial data exchange. In particular, the OGC has released standards to describe spatial data and data format encodings (see below). OGC has produced numerous XML Schemas for web applications and services, for instance the Web Feature Service (WFS), data encoding languages such as the Geography Markup Language (GML), style languages such as the Styled Layer Descriptor (SLD) for visualization, and filter encoding (FE) specifications. However, several *de facto* spatial data formats are also frequently used that have not been standardized by ISO, OGC or W3C. Examples include ArcGIS' Rest API, the GeoJSON format (see below), and the GPX format.

4. How to send spatial data over the Internet

4.1 Solutions for sending data: Files vs. Services

To send spatial data over the web, one can distinguish between two options: *sending files*, and sending data via a *data service* (see Figure 1). Sending files is convenient if the dataset is small or the user understands the data structure. Using a data web service makes sense when the data's content is unknown, when data sets are large and only a subset is needed, and when data should be updated. It is important to note that sending data via a service is very often a file-based process, with the files being created ad-hoc for transfer.

Sending Files — if data are sent by file, then they are often compressed first to reduce file size, for instance using GZIP compression. Transferring data by (compressed) file, however, does not allow the user to see and browse the data before they receive it completely. Furthermore, the send process is not responsive in that it may fail during network interruptions, and users are not able to select a subset of the data. Probably even more importantly, the user needs to be sure that their software can read the data format. For instance, the user may download geometry data in Autocad's DWG format, but few GIS are able to read this format. Here, syntactic interoperability is not ensured. Converting the data into a standardized geospatial file format, such as GML or KML, before making them available to others can resolve this interoperability problem. In comparison to the DWG format, OGC's GML and KML format specifications are open to the public, and hence, can be implemented freely by software developers. However, in practice only a few (desktop) GIS support the GML standard fully.

Using Data Web Services — if a data web service, such as OGC's WFS and Web Coverage Service (WCS) are used, then several of the problems previously outlined can be avoided. For instance a data service, such as a Spatial Data Infrastructure (SDI), enables browsing of the data, selection of subsets, can cope with network interruptions, and ensures interoperability. HTTP GET and DESCRIBE communication commands, such as GetCapabilities, GetMap, GetFeature, DescribeFeatureType, etc., are used to inform the client about the offerings from the data services and to select data subsets. The use of standardized methods and data descriptions enable the client application to identify in which data format the requested data can be sent so that the client can read the data correctly. Spatial data are mostly transferred using one of four data serialization standards: URLs that contain Key-Value pairs, eXtensible Markup Language (XML), JavaScript Object Notation (JSON), and Protocol Buffers - that are detailed in Table 1. Some data serialization formats, such as XML and Protocol Buffers, provide a schema in addition to the data. The schema contains a description of the data structure and data types for the data during data transfer. This allows easier parsing and validation of the data received.

While we will discuss particular spatial data formats in more detail below, we note that XML and JSON-based data formats are not designed to minimize the amount of information to be sent. Hence, standard HTTP compression capabilities are often employed (e.g. gzip) to reduce the transmission time. Furthermore, choosing a subsets of the data can significantly reduce the size of the dataset that needs to be transferred. Several of the OGCs spatial data service standards allow the selection of subsets using filter encoding standards. One can select data within a particular geographic area of interest, by geographic theme, image band of interest, by level of detail (similar to map scale), and one can select certain geographic objects or raster images based

on client specified criteria. Standards that allow subset selection include OGCs WFS, WCS and the Web Map Tiling Standard (WMTS).

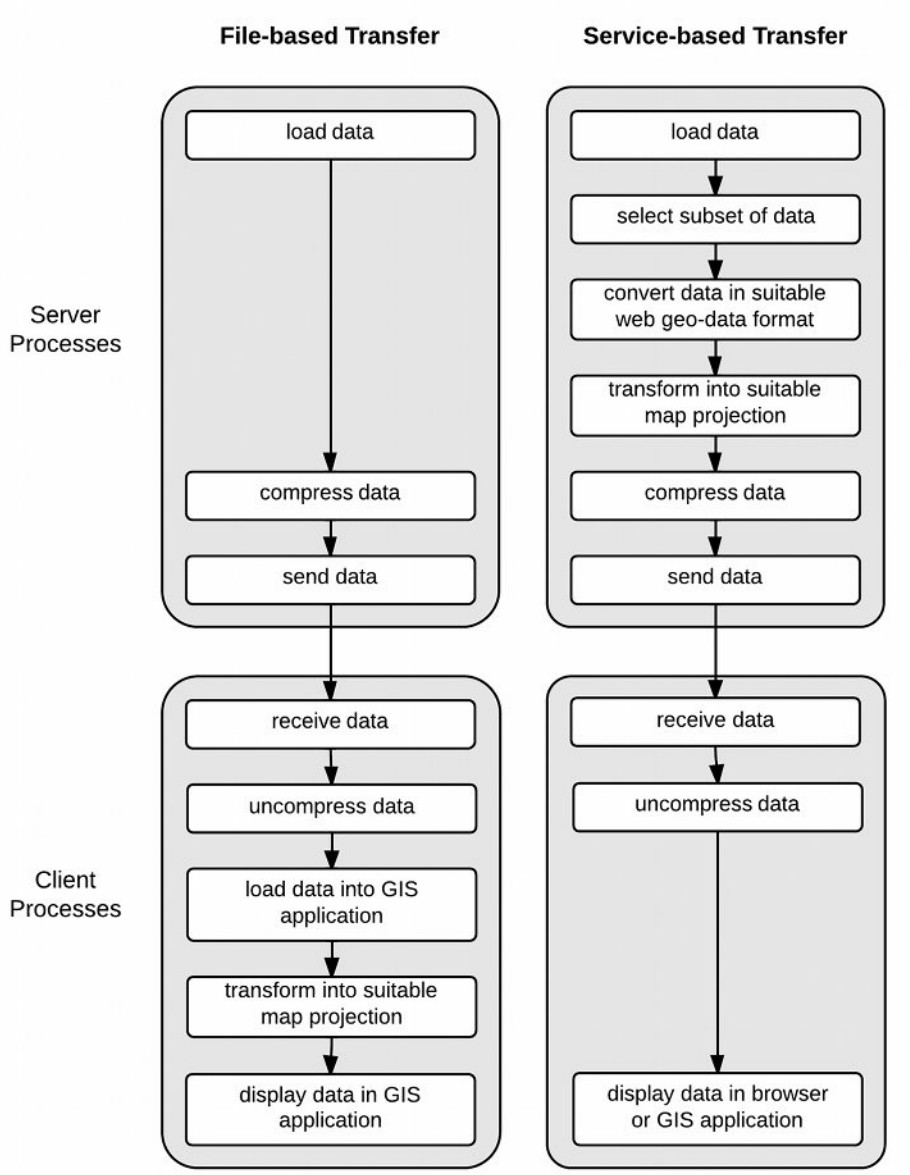


Figure 1 – Two general approaches to send spatial data via the Internet

Table 1 – Internet encoding/serialization standards used for transferring spatial data.

Name (Specification)	Purpose	Elements and Example	Characteristics
Key-Value pair (IETF RFC 2616)	For data requests and data submission, i.e. HTTP GET, HTTP POST	- Form: "keyword=value", separated by ampersands (&) appended to an URL - Example: `http://example.com?name=Smith&initial=JS`	Human readable, keyword-value pairs

XML (IETF RFC 3023)	For description of data, including documents, to request and send data	- Version and character encoding declaration
		- Elements declared by a start tag and end tag, e.g. `<step>` and `</step>`
		- Element attributes, e.g. `<step number="3">`
		- Example:
		`<person>`
		`  <name>Smith</name>`
		`  <initial>JS</initial>`
		`</person>`
			Human readable; can be used for hierarchical structured data; self-describing, extensible.

JSON (IETF RFC 4627)	For sending data objects (not for documents)	- Array and record based - Curly brackets to indicate object start and end - Attribute-value pairs - Example: `{"name":"Smith","initial":"JS"}`	Human readable, attribute-value pairs. Simpler than XML (smaller grammar); can be used for hierarchical structured data; self-describing.

Protocol Buffers (Google)	For sending and storing structured data objects	- Using a .proto schema file to describe data first, followed by serialization of one or many data objects
		- Proto schema example:
		`message Person {`
		`  required int32 id = 1;`
		`  required string name = 2;`
		`  optional string initial = 3; }`
			Binary: i.e. small size but not human readable; name-value pairs, can be used for hierarchically structured data; not self-describing. – requires .proto files, extensible, forward and backward compatible

4.2 2D - Vector Data Formats and Services

As outlined vector data can be transferred as files or via a data web service. Common vector data formats include ESRI's *Shapefile* format and AutoCAD's DXF format. Both data formats are proprietary and pre date the geospatial web. Hence, data transfer via the Internet was not likely a design objective for Shapefile and DXF formats. The *Shapefile* format, although fairly small in size because of its binary form, allows only one geometry type (i.e. polygons or lines or points) per file, all features must have the same attribute set within a file — so we cannot mix trees with buildings in the same dataset. Furthermore, one *Shapefile* consists of at least four files that need to be packaged together. Similar problems exist for the popular *DXF format*. DXF does not allow user defined attributes. This means that DXF files cannot carry information to characterize a tree

as an oak tree with a height of 10.5 m and a stem diameter of 80 cm. Hence, new web-specific spatial data formats were developed to address such requirements.

Based on XML serialization, the OGC developed and standardized two spatial data formats: the Geography Markup Language (GML) and the Keyhole Markup Language (KML). Another existing XML-based format not adopted as a standard by OGC is OpenStreetMap XML (OSM). GeoJSON, derived from JSON, has not been formally recognized as a standard yet (in 2014), but GeoJSON is widely supported by browser-based mapping applications (e.g. Leaflet and OpenLayers). Finally, spatial data formats based on the Protocol Buffer File (PBF) standard are the OpenStreetMap PBF format and MapBox's Vector Tile format. Table 2 presents the characteristics of several vector data formats.

GML, KML and GeoJSON formats have adopted the OGC Simple Feature Specification (OGC SFS) to describe the geometry of geographic objects. This specification, among other things, defines the basic geometry elements: Point, LineString, and Polygon, as well as GeometryCollections of those (see *data structures, vector data*). GeoJSON files and Protocol Buffers are compact formats in comparison to XML-based data formats. However, GeoJSON and XML formats (i.e., GML, KML or OSM), can be compressed. Some of the vector formats, in particular KML and GML, can include styling information for point symbols, labeling, line width, color, etc., for display. Each of the formats has its advantages and disadvantages with respect to size, extensibility and flexibility. Hence, choosing a transfer format should be based on the characteristics of the particular use case. Since we discuss formats for 2D vector data only, we refer readers with an interest in 3D vector data to OGCs CityGML standard and the American Society for Photogrammetry and Remote Sensing (ASPRS) organization's LAS format.

There are two *web service* standards for transferring vector data: the OGC Web Feature Service (WFS) and ISO 19142: 2010 Geographic Information – Web Feature Service. The OGC standard forms the basis of the ISO WFS standard. The standards utilize the GML standard to send vector data. Operations described by the WFS standards such as *DescribeFeatureType* and *GetFeature*, allow a client to browse, retrieve (get) a subset of a dataset, or even a single geographic object only using their respective Filter Encoding specifications, and also create, delete, and update features in a dataset using a transaction service.

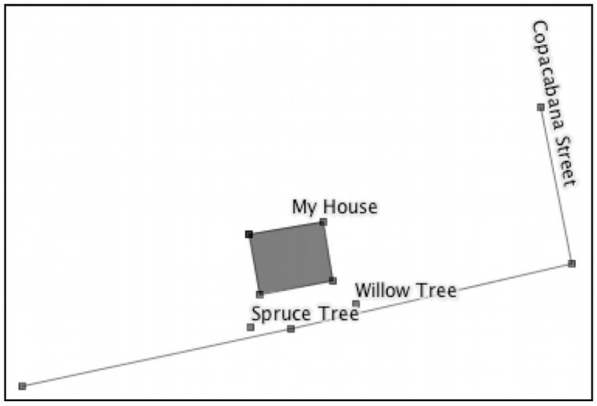
Figure 2 presents a vector data *example* (see image on the bottom left) showing how different geographic objects can be described in GeoJSON and KML formats. The dataset contains: a building with the attributes "name" and "address"; a road with the attributes "name" and "road type"; and two trees, with the attributes "tree type" and "height". As can be seen, the geometry type of the building is Polygon, the road geometry is of type LineString and the trees are described as Points. In both formats the locations, i.e. coordinates, are geographic coordinates, latitude and longitude. GeoJSON coordinates are assumed to be CRS84, (urn:ogc:def:crs:OGC:1.3:CRS84) unless specified otherwise, whereas KML is WGS84 (EPSG:4326). GML, not shown here, expects that the Coordinate Reference System (CRS) will be defined explicitly as an attribute of the feature, and requires a schema (.xsd) describing the feature types included in the GML document. In the given example, no styling information (KML) is included for the display of the objects. When compared, the GeoJSON format is more compact (1,788 Bytes) than KML (4,343 Bytes). GML is generally more verbose (3,358 Bytes plus a 2,408 Byte schema).

Table 2 – Geospatial data formats used for online data transfer.

Name	Purpose	Serial.	Key elements	Characteristics
GML	To describe application schemas and to transport and store geographic information	XML	- Data schema file (*.xsd) to describe data types, and a data file (.gml) - FeatureCollections with FeatureMembers with geometries, or coverages, and attributes, Spatial Reference System (SRS) information, display styles	Human readable, (very) flexible, for raster data, vector data, and triangulated irregular networks, permits specification of application schemas (e.g. CityGML, netCDF)
KML	To display geographic data in a (Google) Earth browser	XML	Placemarks, ground overlays, paths, polygons, network links (e.g. for dynamic data), display styles	Human readable, fixed schema, a specialisation of GML, for raster and vector data, coordinates only in WGS 84
OSM	To transfer OpenStreetMap data	XML, PBF, JSON	Nodes, ways, relations	XML: Human readable, fixed schema
GeoJSON	To transfer geospatial data based on JSON	JSON	FeatureCollection with Features with geometries and properties	Human readable, not very practical for image or raster data, coordinates in WGS84, no style information
TopoJSON	Extension of GeoJSON to encode topology	JSON	Like GeoJSON plus additional topology object type with an “arcs” member	See GeoJSON; files should be smaller than GeoJSON files
MapBox VectorTile	To encode tiled geospatial data to be shared across clients	PBF	- Data schema defined in vector-tile.proto, data file - Layers with features with geometries (described by drawing commands) and attributes	Binary; for squared tiles in Spherical Mercator projection
GeoRSS / Atom	To extend existing feeds with geographic information	XML	Feed data: title, author, entry, updated, etc.; with a geometry element as part of an RSS entry element,	GeoRSS Simple coordinates are in WGS84; GeoRSS GML is a formal GML application profile which allows to specify the spatial reference system

GeoJSON

```
{
"type": "FeatureCollection",
"features": [
  { "type": "Feature",
    "properties": { "name": "My House",
                    "address": "25 Copacabana Street"},
    "geometry": { "type": "Polygon", "coordinates": [ [
        [ -70.60296097, -33.42574372, 0.0 ],
        [ -70.60289771, -33.42593201, 0.0 ],
        [ -70.60265559, -33.42589031, 0.0 ],
        [ -70.60272146, -33.42569695, 0.0 ],
        [ -70.60296097, -33.42574372, 0.0 ] ] ]} },
  { "type": "Feature",
    "properties": { "name": "Copacabana Street",
                    "type": "Neighbourhood Road"},
    "geometry": { "type": "LineString", "coordinates": [
        [ -70.60199492, -33.42532401, 0.0 ],
        [ -70.60185070, -33.42583767, 0.0 ],
        [ -70.60280898, -33.42604940, 0.0 ],
        [ -70.60368679, -33.42623646, 0.0 ] ] }  },
  { "type": "Feature",
    "properties": { "tree_type": "willow",
                    "height": 6 },
    "geometry": { "type": "Point", "coordinates":
        [ -70.60259833, -33.42596654, 0.0 ]}  },
  { "type": "Feature",
    "properties": { "tree_type": "spruce",
                    "height": 12.3 },
    "geometry": { "type": "Point", "coordinates":
        [ -70.60293927, -33.42604344, 0.0 ] } }
  ]
}
```



KML

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
  <name>kmlvectorexample.kml</name>
  <Folder>
    <name>kmlvectorexample</name>
    <Placemark>
      <name>My House</name>
      <ExtendedData><Data name="address">
          <value>25 Copacabana Street</value></Data>
      </ExtendedData>
      <Polygon><outerBoundaryIs><LinearRing>
          <coordinates>
            -33.42574372, -70.60296097, 0
            -33.42593201, -70.60289771, 0
            -33.42589031, -70.60265559, 0
            -33.42569695, -70.60272146, 0
            -33.42574372, -70.60296097, 0
          </coordinates></LinearRing></outerBoundaryIs>
      </Polygon></Placemark>
    <Placemark>
      <name>Copacabana Street</name>
      <ExtendedData><Data name="type">
          <value>Neighbourhood Road</value> </Data>
      </ExtendedData>
      <LineString><coordinates>
          -33.42532401, -70.60199492, 0
          -33.42583767, -70.60185070, 0
          -33.42604940, -70.60280898, 0
          -33.42623646, -70.60368679, 0
        </coordinates></LineString></Placemark>
    <Placemark>
      <name>Willow Tree</name>
      <ExtendedData>
        <Data name="tree_type">
          <value>willow</value> </Data>
        <Data name="height">
          <value>6</value> </Data></ExtendedData>
      <Point><coordinates>-33.42596654, -70.60259833, 0
        </coordinates></Point></Placemark>
    <Placemark>
      <name>Spruce Tree</name>
      <ExtendedData>
        <Data name="tree_type">
          <value>spruce</value> </Data>
        <Data name="height">
          <value>12.3</value> </Data></ExtendedData>
      <Point><coordinates>-33.42604344, -70.60293927, 0
        </coordinates></Point></Placemark>
  </Folder>
</Document>
</kml>
```

Figure 2. Example for a geographic dataset stored in GeoJSON and KML format.

4.3 2D - Raster Data Formats and Services

Examples of spatial raster data include satellite images, gridded elevation models, and temperature or wind speed snapshots of a region. Spatial raster data also includes image tiles used by web mapping services to render a base map, such as Google Maps or Here.com.

Raster data and raster maps are usually transferred in an *image format*, such as JPG, TIFF or PNG. Supplemental information is added to each raster image indicating the real-world coordinates of the image corners, the map projection used, and perhaps information describing any image transformations applied (e.g. stretching and rotation). Spatial data web services offer the same image formats. Since JPG, TIFF and PNG have built-in data *compression* methods, HTTP compression is usually not applied when transferring image formats. Which type of image format is best to use depends on whether the user is looking for a ready-made map (with styling), or the actual raster data set. In the case of a map, it makes sense to use JPG or PNG formats. JPG employs lossy data compression, whereas with raster data the TIFF format is recommended as it offers lossless LZW compression. The image formats, JPG, TIFF and PNG, do not allow separation of data and styling information. However, newer formats may support this (see for instance the OGC Symbology Encoding specification).

There are at least three Open Geospatial Consortium (OGC) *standards* for distribution of raster data using a *data web service*: the OGC Web Mapping Service (see also ISO 19128), the OGC Web Coverage Service (WCS), and the Web Map Tiling Service (WMTS). However, other standards address distribution of raster data as well, such as OGC GML in JPEG 2000, OGC GML version 3, OGC NetCDF, and ISO 19123.

An *OGC WMS* returns complete maps of (dynamic) spatially referenced data in an image format. The WMS service may offer its own styling, or an OGC Styled Layer Description (SLD) file can be sent with the map request. A fully conforming OGC WMS service returns PNG, GIF, JPEG and TIFF formatted images. The bounding box of the area of interest, image dimensions, themes of interest, and the map projection can be specified in the WMS request. The service does not allow updates, so-called transactions, to the data on the server.

An *OGC WCS* service is not used for retrieving completely styled maps but allows detailed access to raster data of space/time-varying phenomena. Supported output formats may include GeoTIFF, GRD, NetCDF and JP2000. However, metadata about the raster data are sent using OGC GML. The user request to the service can specify parameters such as the bounding box for the area of interest, the time - to select data for a particular date, and a particular image band (e.g. near infrared) to ensure that only data of interest are received. The Map projection, image dimensions, image resolutions and interpolation methods for the rescaling of an image can be specified as well. Version 2 of the standard extends capabilities to work with multi-dimensional and time series raster data as commonly encountered in meteorology and climatology. Extensions that allow a WCS to be transactional, i.e. to update datasets (WCS-T), and to process and filter data (WCPS) exist as well.

The introduction of a *Web Map Tiling Service* (WMTS) resulted from deficiencies (response time) encountered when working with a WMS service. Often WMS services require a few seconds to respond to an image request, because the map image is generated on-the-fly. When concurrent users are high, e.g. serving many users at the same time, these CPU-intensive services become impractical. Google Maps and later OpenStreetMap were the first to use the so-called “slippy map” approach to address this processing problem. Here, one may imagine that a big map image is rendered first and then cut into many small images that are called “tiles”. Only a few of those tiles are of interest to the user and need to be sent to the client. Individual tiles for a complete map can be pre-computed over a range of map scales (different levels of detail) so that when a user zooms in and out tiles can be retrieved more rapidly. This technique was formalized in the OSGeo TMS standard and later in the OGC WMTS standard. A restriction of

this service is that each tile-map can have only one map projection and so-called "faces" are needed to cover the earth's polar regions. It is recommended that tiles be sent in JPG or PNG formats and tile sizes are 256 x 256 or 512 x 512 pixel large. The number of zoom-levels (scales) available usually depends on the map projection used and the WMTS data provider. The storage structure generally takes the following form: "www.domainname.com/mapfolder/z-zoom level/tile-column/tile-row".

4.5 Sending Event Data: Web-Updates and Sensors

So far, we have discussed formats best suited for data requests for a particular geographical object or an entire regional dataset. However, many real-time applications require event data with constant updates of spatial information. For example, when traveling by car the *car navigation* system may send information about the vehicles current position every 3 seconds. Another example of event data are seismometer data collected to record *earthquakes*. To map and analyze earthquake events over time the information from the seismometer should include the strength of the earthquake, the time stamp for the earthquake event, the geographic location of the seismometer, and perhaps a reference to calibration metrics for the instrument.

Formats that are commonly used to send event data include two XML-based formats: Rich Site Summary (RSS) and Atom, and JSON. Similar to GeoJSON the RSS format has a spatial equivalent called GeoRSS – described in Table 2, which is also used for Atom formatted data. Use of the *GeoRSS* or Atom format makes sense for updates on world news webpages. Since 2013 *GeoJSON* seems to have established itself as the *de facto* standard for transmitting location events. For instance, the U.S. Geological Surveys Earthquake Hazard Program makes real-time earthquake information available in GeoJSON and Atom formats. However, some webpages offer location event data in KML format.

The Open Geospatial Consortium has also defined a set of *"sensor web enablement" standards* for description, communication and data exchange with (environmental) sensors. One particular standard of this sensor web enablement set is the *OGC Sensor Observation Service* – OGC SOS. This service standard defines a *GetObservationService* request that returns sensor measurement data. Other, optional, commands allow the retrieval of information about the sensor itself (e.g. its location) and information about the object that is monitored. More information on this topic is to be found under the topic *sensor network*.

5. Current Developments

As with most technical fields there are a lot of development activities in the area of spatial data transfer techniques and standards. In particular we see work focusing on the delivery of background maps as vector tiles instead of raster tiles. In this context, new vector data encodings are being introduced. Among those is the TopoJSON format by M. Bostock (see Table 2). This format was developed to reduce the size of the datasets that are to be transmitted by exploiting their *topological relationships*. For instance, a common border of two touching polygons is stored only once, rather than twice, as in a purely object based geometric model.

A second format that may find wider application is MapBox' VectorTile format, which is based on the protocol buffer serialization (see Tables 1 and 2). Several objectives have led to the development of the format. Two of them are that vector-based tiles can be much smaller than image-based tiles for geographic regions with few map objects, such as a desert, a forest, or

water environments, and that using the vector format allows the immediate application of custom map rendering styles (e.g. chosen by the user) as opposed to pre-rendered image tiles. The development and testing of vector map tiling was and is not only performed at MapBox but also by others, and most often with a focus on effective rendering of OpenStreetMap data.

Finally, it should also be mentioned that substantial ongoing research investigates efficient data transmission of spatial data for use in multiplayer online games. If one thinks of online games as places where people explore cities and landscapes (such as in first person shooter games), then large volumes of terrain data need to be transferred and rendered by the game engine. Developments in this area are of importance to GIS developers because the gaming engines are not only used for online games, but they play also an increasing role in pilot and driver trainings, as well as in military and medical simulations.

SEE ALSO: Data structure: vector; Data structure: raster; Generalization; Interoperability of representations; Representations, Sensor network; Spatial Data Infrastructure; Topological relationships; Web cartography; Web mapping services;

Further Readings

Bertolotto, Michela, and Max J. Egenhofer. 2001. "Progressive transmission of vector map data over the world wide web." *GeoInformatica* 5: 345-373. DOI: 10.1023/A:1012745819426.

Gaffuri, Julien. 2012. "Toward web mapping with vector data." In *Geographic Information Science*, edited by Ningchuan Xiao, Mei-Po Kwan, Michael F. Goodchild, and Shashi Shekhar, 87-101. Berlin: Springer. DOI: 10.1007/978-3-642-33024-7_7.

Google. 2014. “KML Tutorial.” Accessed March 26, 2014. http://developers.google.com/kml/documentation/kml_tut

OGC - Open GeoSpatial Consortium. 2014. “OGC Standards and Supporting Documents.” Accessed March 26, 2014. www.opengeospatial.org/standards/

Yang, Bisheng, and Robert Weibel. 2009. "Editorial: Some thoughts on progressive transmission of spatial datasets in the web environment." *Computers & Geosciences* 35: 2175-2176. DOI: 10.1016/j.cageo.2009.07.001.

Youngblood, Brian. 2013. *GeoServer Beginner's Guide*. Birmingham, UK: Packt Publishing Ltd.