Transforming RDF data into maps: RDF2Map Library

Ana Cristina Trillos Ujueta[1], Jaime Manuel Trillos Ujueta[1] and Luis Daniel Fernandes Rotger[1]

Master of Computer Science, Universität Bonn, Germany
trillos@informatik.uni-bonn.de, trillosj@informatik.uni-bonn.de, s6lufern@uni-bonn.de

Abstract

The Web provides a large number of geospatial information. Recently the information can be represented using Resource Description Framework (RDF), which allow interlinking data. However, the geospatial data must be processed before being displayed on a map. In this paper, we will present RDF2Map library, a developing tool that will help developers to automatically display geospatial data in a map.

1 Introduction

One of the main problems developers encounter when using Linked Data technologies lies in the fact that the tools provided to encourage and help develop processes are still restricted. Nevertheless, Linked Data and Semantic Web provide powerful properties that could be greatly valuable for a wide range of use cases; e.g., it is quite helpful when dealing with distributed data that are linked to each other, it maintains the veracity of each concept since they are unique in the Web, and it is universally accessible [4]. Few examples could improve almost any application or software product development, especially for geospatial software. Rather than having a big dataset that specifies geospatial resources with all the properties and respective values; it would be much easier to have only the identifier (URI) of the concepts and look their geospatial properties (latitude, longitude) into Linked Open datasets like DBpedia[1][1].

Currently, there are several approaches that use semantic data on maps but there is not an easy way for transforming these days. As for applications for displaying RDF concepts enriched with geospatial properties, Map4RDF [9] is a tool for faceted exploring, visualizing and interacting with RDF geospatial datasets. QMap application [7] is able to extract information from HTML selected data sources, create RDF instances using wrappers and RDF Schemas, and with a SPARQL engine extract relevant data to be displayed on a map. LinkedGeoData Browser [2] allows to browse the world, analyze nodes and create facets to allow filtering. Each facet will display related elements on the map. On the other hand, there is no evidence of an already developed, working and tested library that helps developers display their RDF datasets on a Web page.

RDF2Map library focus on the problem of a developer that wants to make use of Linked Data to provide to its final user an interface with a map displaying the representation of geospatial concepts, using an RDF file written in Turtle format. RDF2Map is very straightforward to use, as importing a JavaScript library inside an HTML Web page. The library is flexible in the sense that it allows the developer to create its own RDF geospatial concepts representations in a Turtle file, taking into account that the concepts do not necessarily exist on the Semantic

[1] `https://dbpedia.org/sparql`

Web; and the library will display them in the map.

The paper is structured in 6 sections. Section 2 presents concepts for understanding the problem. In Section 3, we cover an overview of related work and then we present RDF2Map library, its implementation, and use cases, in Sections 4 and 5; finally, in Section 6, conclusions and future work are presented.

2 Background

Semantic Web [6] is a technology that promotes common data formats (i.e. RDF) for integrating data taken from different sources, as well as a language for describing how the data is related. RDF (Resource Description Framework)[2] is the standard model used in Semantic Web, where data and their relationships are represented by means of triples. A triplestore is optimized for the storage and retrieval of triples, it is composed of subject, predicate, object. Linked Data is data which is published on the web. This data is specifically defined in RDF and linked with other datasets [4]. A dataset is a collection of RDF graphs where are associated with a URI (Uniform Resource Identifier) [8], a Unicode string of characters, generally, a dataset contains more than one property and concerns a single topic. Each geospatial concept is described using geospatial vocabularies; Basic Geo Vocabulary [3] offers properties `geo:lat` and `geo:long`, for denoting latitude and longitude of a geospatial resource; it can also be used more general vocabularies like, Friend of a Friend[4], that has properties such as `foaf:name` to denote the name, and `foaf:homepage` to denote the Web page link of a resource.

3 Related Work

Few approaches focus on the representation of RDF geospatial resources on a map. One of these approaches is Map4RDF [9], an application that allows the user to search facets and visualize geospatial information from a selected dataset. The backend of the application is a Web Server which connects to a given triplestore through the SPARQL endpoint for retrieving instances of the chosen facet. The faceted browsing interface gathers the information and displays the points on a map. In contrast to RDF2Map library, it is an application which has no signs that it can be integrated into an HTML Web page, for displaying relevant data to a developer's client.
Another approach is LinkedGeoData Browser [2], an application for browsing and editing LinkedGeoData[5] points. It behaves similarly to Map4RDF, however, LinkedGeoData Browser displays markers and polygons on the map; Map4RDF claims that it displays, LineStrings, Polygons, Markers, etc.

In contrast to these two applications, QMap [7] claims that information is extracted from HTML data sources and not from RDF datasets. The information is translated into RDF instances using wrappers and RDF Schemas. RDF instances are queried by a SPARQL engine providing the query results and finally being displayed as markers in a map.
A similar approach is a platform for the Municipality of Catania [5] which extract the data

[2]`https://www.w3.org/RDF/`
[3]`https://www.w3.org/2003/01/geo/`
[4]`http://www.foaf-project.org/`
[5]`http://linkedgeodata.org/Datasets`

from different data sources (GIS, JSON format, XML file, MySQL) and convert all the data provided by the GIS into RDF in order to show the information in a map. The platform works with Google Maps and it displays lines, markers, and polygons.
Sextant is a web-based tool for browsing, creating, sharing and visualizing linked geospatial data [3]. The tool is based on KML file used for expressing geographic annotation and visualization (2-dimensional and 3-dimensional space). The tool uses GeoSPARQL for querying RDF data and displays polygons and lines on the map.

As can be seen, the problem of having a library that will help developers to automatically display RDF geospatial information on a map, without further processing, is still an open issue.

4 Proposed Approach

In this section, we describe our solution. RDF2Map library aims to help developers to illustrate/display RDF geospatial information on a map.

4.1 Problem Statement and Proposed Solution

After evaluating related work, we noticed there is a lack in libraries for processing and displaying RDF geospatial concepts in a map. For this reason, we developed RDF2Map library[6]. A JavaScript library for processing geospatial concepts, from a Turtle file, and displaying them in a map.

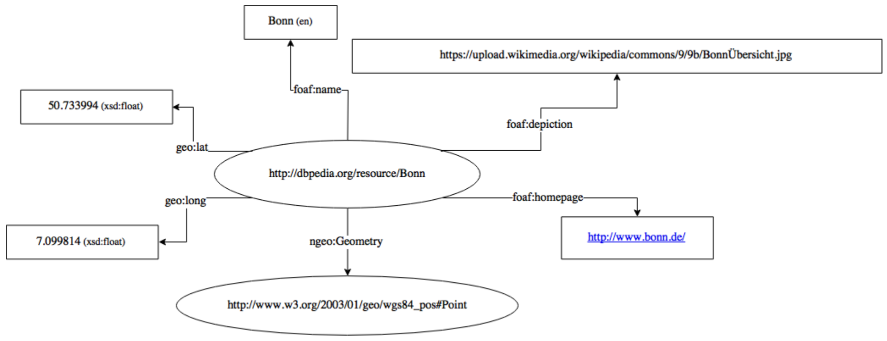
RDF2Map works with two open-source JavaScript libraries, Leaflet[7] for interactive maps and RDFStore[8] for processing information. RDFStore supports SPARQL querying version 1.0 and data manipulation.
One of the main features of RDF2Map is the capability to extract remote information from the DBpedia SPARQL Endpoint, using the URI of a geospatial concept, and then with the extracted coordinates (latitude and longitude) display them in the map. This information can be displayed in different shapes and forms, hence Leaflet allows working with markers, polylines, polygons, and pop-ups.

4.2 Implementation

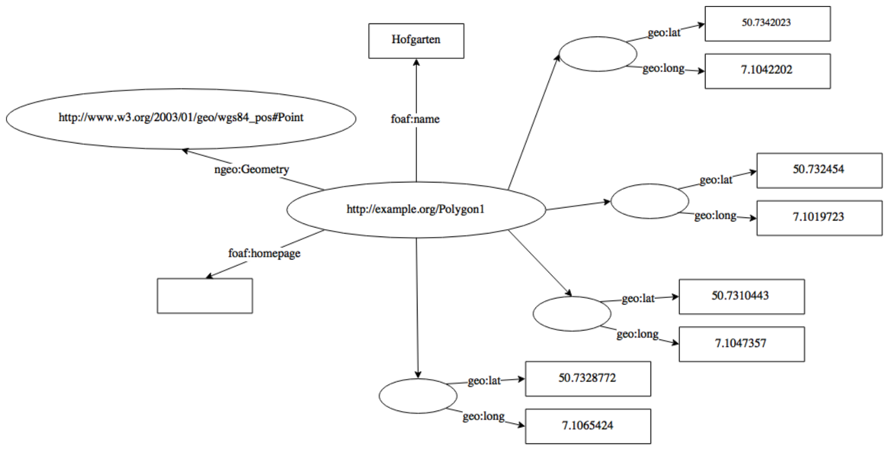
RDF2Map library uses a subset of RDF properties, described on two data models. Figure 1a shows the data model for representing geospatial concepts displayed as markers and icons. And figure 1b represents the data model for polygons and polylines geospatial concepts.

As an introduction to a detailed explanation of the algorithm 1, a brief overview follows: RDF2Map library receives a Turtle file with all geospatial concepts to be shown on the map. The library creates a store using RDFStore, where the concepts will be stored; then the concepts are extracted and those which data can be requested remotely, will be requested and loaded into the store, consequently, markers, icons, paths and/or polygons will be generated and saved in a JSON object. Leaflet API converts the JSON object in order to update the map and all geospatial information will be displayed. As you can see in figure 2, the activity diagram.

[6]`https://github.com/atrillos/Rdf2Map_library`
[7]`http://leafletjs.com`
[8]`https://github.com/antoniogarrote/rdfstore-js`


Bonn (en)
foaf:name
https://upload.wikimedia.org/wikipedia/commons/9/9b/BonnÜbersicht.jpg
foaf:depiction
50.733994 (xsd:float)
geo:lat
http://dbpedia.org/resource/Bonn
geo:long
foaf:homepage
7.099814 (xsd:float)
ngeo:Geometry
http://www.bonn.de/
http://www.w3.org/2003/01/geo/wgs84_pos#Point

(a) RDF2Map library data model.



(b) RDF2Map library data model for Polygons and Polylines.

Figure 1: RDF2Map data models.

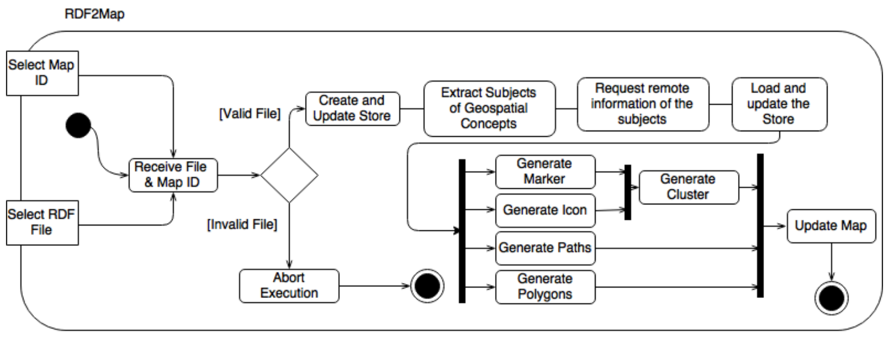


Figure 2: Activity diagram of the RDF2Map library.

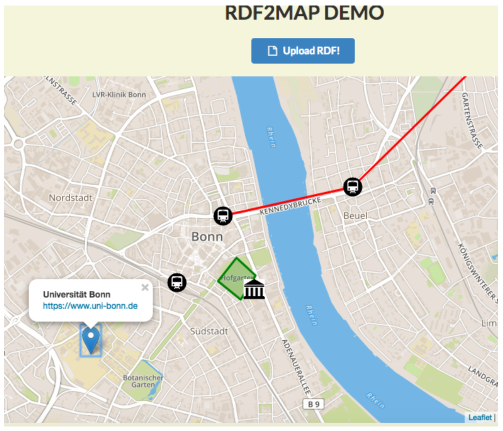
Once RDF2Map finishes processing all geospatial concepts from the file, the map will be updated presenting the geospatial concepts, see figure 3a for an example. RDF2Map library also offers to cluster only for two types of geospatial concepts, markers, and icons. Figure 3b is an example of how clustering will be shown on the map. The final user can interact with the

```
input  : $T$, set of tupples in RDF
output: $J$ set of JSON objects for the map
$Store \leftarrow \emptyset$ // Set of RDF tuples
$S \leftarrow \emptyset$
$SPQ1 \leftarrow$ SPARQLQuery to extract subjects from $T$
$S \leftarrow$ Executed $SPQ1$
$RDFInfor \leftarrow \emptyset$ // Set de RDF tuples
for every s subject in S do
    $SPQ2 \leftarrow$ SPARQLQuery to extract latitude, longitude, name, homepage,
     depiction and color // remotely from DBpedia
    $RDFInforInstances \leftarrow$ executed $SPQ2$
    $RDFInfor.add(RDFInforInstances)$
end
$Store \leftarrow T \cup RDFInfor$
$SPQPoint \leftarrow$ SPARQL to extract subject, name, latitude, longitude, homepage(?)
 from geo:Point
$SPQIcon \leftarrow$ SPARQL to extract subject, name, latitude, longitude, homepage(?),
 depiction from lgd:Icon
$SPQPolygon \leftarrow$ SPARQL to extract subject, name, (latitude,longitude: postion []),
 homepage(?), color from ngeo:Polygon
$SPQPath \leftarrow$ SPARQL to extract subject, name, (latitude,longitude: postion []),
 homepage(?), color from lgd:Path
$P \leftarrow$ executed $SPQPoint$
$I \leftarrow$ executed $SPQIcon$
$Poly \leftarrow$ executed $SPQPolygon$
$Pa \leftarrow$ executed $SPQPath$
$J \leftarrow$ JSON( $P \cup I \cup Poly \cup Pa$)
return J
```

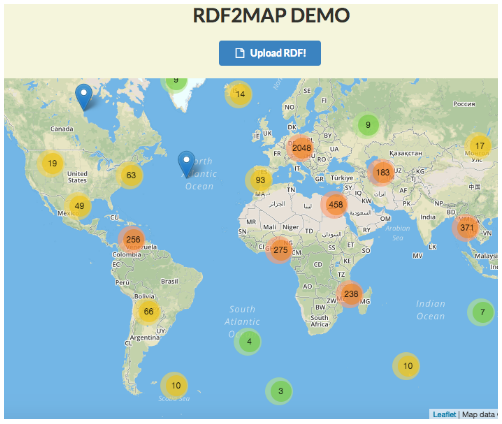
Algorithm 1: Algorithm of RDF2Map

markers, icons, polygons, and paths; when selected, a pop-up will appear displaying name and homepage for each concept.
When a geospatial concept is described as an icon, it will be shown on the map with the image extracted using the property `foaf:depiction`, as shown in figure 3a.
For polygons and paths, the property `dbc:Color` describe a specific color to personalize them; in case this property is not specified the Leaflet default color will be used.

RDF2Map was tested to evaluate its performance when receiving a file with different quantity of input concepts. The tests were performed on a laptop with Intel Corei7 6700HQ processor and 4GB RAM.
Table 1 shows the times (in millisecond) that each process performed by RDF2Map takes. *loadTime* is the time that the RDFStore takes to load the data from the turtle file. *requestTime* represents the time that the library takes when requesting information remotely, and *ProcessingTime* is the time, RDF2Map takes to process and display all geospatial concepts (markers, icons, polygons, and paths) in the map. Figure 4 represents the execution relative times for the test. As we can see, when the number of concepts is increasing, the time is going to increase too. In other words, each time a new concept is called the library needs to identify it (*loadTime*), find the missing information on DBpedia (*requestTime*) and finally create the concept



(a) A sample of the map displaying markers, icons, polygons and paths.



(b) Sample of clustering working on a map.

Figure 3: Examples.

on the map (*processingTime*). The time most spend when it has more than 4000 concepts is in *loadTime* because the library needs to read the file, then extract each concept with its corresponding information with the help of RDFStore using SPARQL, to then do the request part. In the other hand, when the concepts are less than 4000, the *requestTime* spend more time than the others. The reason is the library needs to connect to the internet and download the data from DBpedia.

# Concepts in file	loadTime	requestTime	ProcessingTime	# Concepts displayed
10	35.95	474.78	25.835	10
100	175.395	610.33	60.449	100
1000	1369.775	1854.108	271.885	1000
2000	2936.82	2598.87	572.34	2000
4000	6030.48	2650.354	1158.064	4000
8000	12512.215	4383.68	2812.709	8000
10000	14772.42	6123.982	2937.875	10000

Table 1: Performance test results in ms.

4.3 Use Cases

RDF2Map library can be used downloading `rdf2map.js` from RDF2Map on GitHub, and importing it into an HTML file; at the same time, it is important to download and import RDFStore and Leaflet libraries, as shown below.

```
<script type="text/javascript" src="resources/rdfstore.js"></script>
<script type="text/javascript" src="resources/leaflet.js"></script>
<link rel="stylesheet" type="text/css" href="resources/leaflet.css">
<link rel="stylesheet" type="text/css" href="resources/styles.css">
<link rel="stylesheet" type="text/css" href="resources/MarkerCluster.css">
<link rel="stylesheet" type="text/css" href="resources/MarkerCluster.Default.css">
```

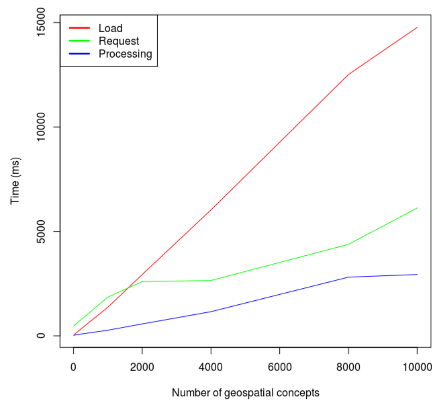


Figure 4: Activity diagram of the RDF2Map library.

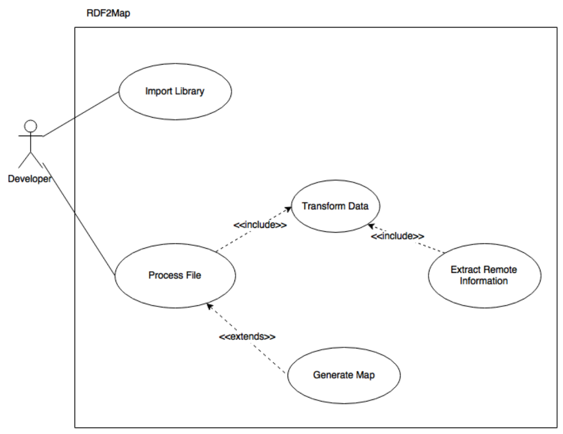


Figure 5: Use case diagram for RDF2Map library.

```
<script type="text/javascript" src="resources/leaflet.markercluster.js"></script>
<script type="text/javascript" src="resources/rdf2map.js"></script>
```

The general use case of RDF2Map (Figure 5), a user imports the library and then calls RDF2Map function `bindFileInput` with two parameters, the input file and the map ID. Next, we will describe two possible use cases for using RDF2Map. The first use case is related for a a specific company and the second is for didactic use.

4.4 Use Case 1

A company (e.g. Rewe) desires to inform its users the location of their stores on their Web page, but this information is not published on the Web.
The developer creates a RDF file, with the geospatial information for each store, as shown below.

```
@prefix ns0: <http://geovocab.org/geometry#> .
```

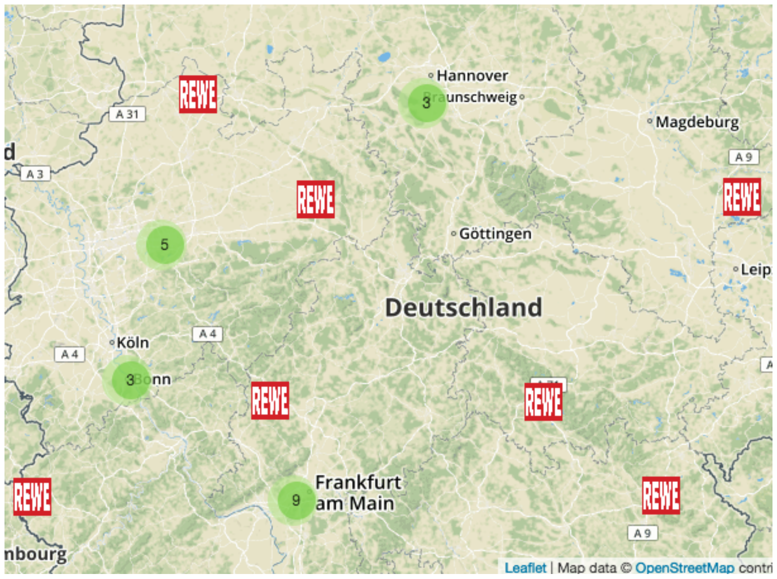


Figure 6: Map of Use Case 1.

```
@prefix ns1: <http://linkedgeodata.org/triplify/> .
@prefix ns2: <http://linkedgeodata.org/ontology/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> .
ns1:node1329330946 ns0:Geometry ns2:Icon ;
foaf:homepage "https://www.rewe.de/" ;
foaf:depiction "https://i.imgur.com/yH6LWPn.png" ;
geo:lat 53.133327100000002474 ;
geo:long 8.1904960000000013309 ;
foaf:name "REWE" .
ns1:node2545718183 ns0:Geometry ns2:Icon ;
foaf:homepage "https://www.rewe.de/" ;
foaf:depiction "https://i.imgur.com/yH6LWPn.png" ;
geo:lat 48.116935300000001519 ;
geo:long 11.525668500000000094 ;
foaf:name "REWE" .
ns1:node1574280669 ns0:Geometry ns2:Icon ;
foaf:homepage "https://www.rewe.de/" ;
foaf:depiction "https://i.imgur.com/yH6LWPn.png" ;
geo:lat 50.100445400000005236 ;
geo:long 8.6584660000000006619 ;
foaf:name "REWE" .
...
```

Then the developer imports RDF2Map, RDFStore and Leaflet libraries into his HTML file, and calls `RDF2Map.bindFileInput`. Every time the company's Web page is open, the map will display the stores (Figure 6)

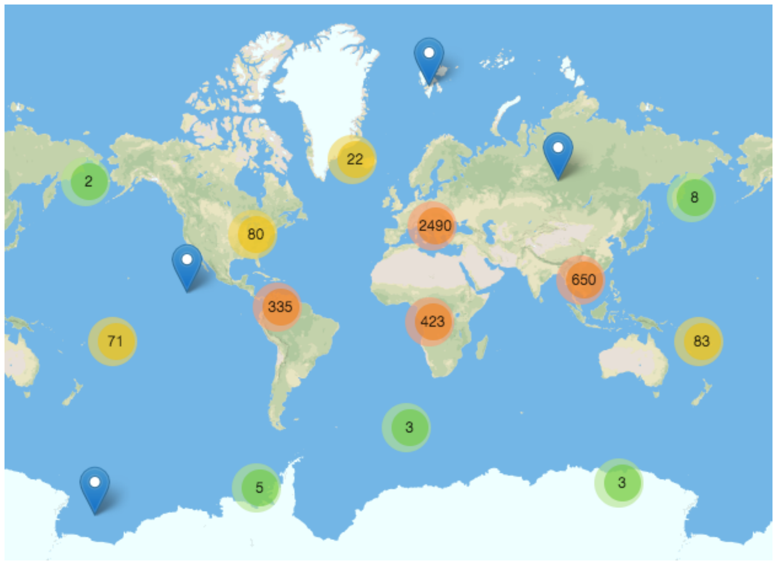


Figure 7: Map of Use Case 2.

4.4.1 Use Case 2

An institute wants to teach their students with an interactive map, all the countries that exist and have existed; so a developer creates an RDF file, with the URIs of all the countries from the DBpedia Endpoint, as shown below.

```
@prefix ns0: <http://geovocab.org/geometry#> .
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
dbr:Abbasid_Caliphate ns0:Geometry geo:Point .
dbr:Almohad_Caliphate ns0:Geometry geo:Point .
dbr:Arab_League ns0:Geometry geo:Point .
dbr:Cape_Colony ns0:Geometry geo:Point .
dbr:Central_Tibetan_Administration ns0:Geometry geo:Point .
dbr:Dacia ns0:Geometry geo:Point .
dbr:Democratic_Republic_of_Afghanistan ns0:Geometry geo:Point .
dbr:Duchy_of_Lorraine ns0:Geometry geo:Point .
dbr:Duchy_of_Mecklenburg-Strelitz ns0:Geometry geo:Point .
dbr:Fatimid_Caliphate ns0:Geometry geo:Point .
...
```

Later the developer, imports RDF2Map, RDFStore and Leaflet libraries into the HTML file and the information will be displayed. Since there have existed a lot of countries that have been divided or unified during history, the map will be clustered as shown in figure 7.

In both use cases, the implementation using RDF2Map in the development project is simple. The performance of the library for these cases takes some time (milliseconds) to process files containing more than 1000 geospatial concepts but RDF2Map ensures that all the concepts which are in the Turtle file have been inserted in the map.

4.5 Conclusions and Future Work

RDF2Map library is probably one of the first attempt to offer a library for displaying geospatial concepts on a map. It offers features like the ability to extract data remotely from the Web, or locally from a Turtle file and showing the results in a map. RDF2Map permits adding geo-points represented as markers, icons, polygons and or lines (paths).

RDF2Map library currently is limited to accept files written in Turtle format, and extract information from one source, DBpedia endpoint. Thus in the future, we plan to extend the library to include multiple SPARQL endpoints, as well as, accepting other RDF serializations formats (e.g. N3, RDFa, NQuads, etc.). At the same time, we plan to make RDF2Map more flexible so that through the API, the developer can choose which information about the geospatial concepts wants to obtain and display.

References

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. *The semantic web*, pages 722–735, 2007.

[2] Sören Auer and Jens Lehmann. Linkedgeodata–collaboratively created geo-information for the semantic web. *Semantic Web Challenge, ISWC*, 2009.

[3] Konstantina Bereta, Charalampos Nikolaou, Manos Karpathiotakis, Kostis Kyzirakos, and Manolis Koubarakis. Sextant: Visualizing time-evolving linked geospatial data. In *International Semantic Web Conference (Posters & Demos)*, pages 177–180, 2013.

[4] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227, 2009.

[5] Sergio Consoli, Misael Mongiovi, Andrea Nuzzolese, Silvio Peroni, Valentina Presutti, Diego Reforgiato Recupero, and Daria Spampinato. A smart city data model based on semantics best practice and principles. *ACM*, 05 2015.

[6] I Herman. The semantic web home page. `http://www.w3.org/2001/sw/`, 2006.

[7] Frederik Hogenboom, Alexander Hogenboom, Ricardo van Gelder, Viorel Milea, Flavius Frasincar, and Uzay Kaymak. Qmap: an rdf-based queryable world map. In *Third international conference on knowledge management in organizations (KMO 2008)*, pages 99–110. Vaasan Yliopiston Julkaisuja, 2008.

[8] J. J Klyne, G. Carroll. Rdf semantics. `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`, 2004.

[9] Alexander de Leon, Filip Wisniewki, Boris Villazón-Terrazas, and Oscar Corcho. Map4rdf-faceted browser for geospatial datasets. *UPM*, 2012.