Data Integration Project – A.Y. 2019/2020

Discover Zurich: the exploration map

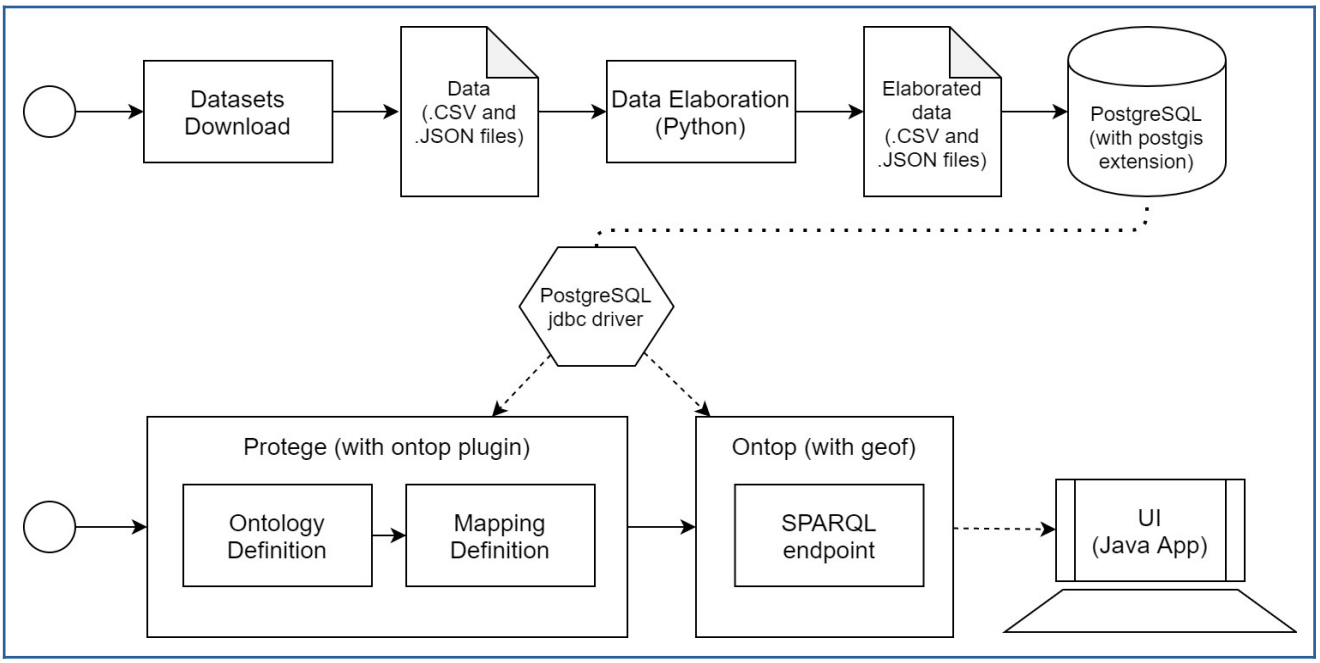
Silvia Fracalossi
Student ID: 13180
June 2020

Link to unibz GitLab Repository:
https://gitlab.inf.unibz.it/Silvia.Fracalossi/visit_zurich

“Discover Zurich” is an offline application in which a user can search for several types of entertainment offered in the famous Swiss city. In the main page, a map of the city is displayed, showing the 12 district areas. On the left side of the screen, a filter panel allows the user to select and visualize on the map several places such as restaurants, bars, museums, attractions, shops, bus stops, train stations, car and bike parking areas and bike rentals.

1) Data Flow

To develop the project, several steps were followed, which are represented in the image below.



Datasets Download.
The datasets were downloaded. More on this topic in “Section 2 – Data Sources”.

Data Elaboration (Python).
The data contained geographical coordinates, represented into two different systems. The python data elaboration script converts all the coordinates into the latitude-longitude system.

PostgreSQL (with postgis extension).
In order to handle geographical data and execute its functions, Denodo has been substituted with the relational database PostgreSQL, extending the postgis plug-in.

Ontology Definition.
Based on the User Interface requirements and on the available data, the ontology has been defined directly in Protégé. More on this topic in “Section 3 – Ontology Definition”.

Mapping Definition.
Based on the ontology and on the database tables, the mapping has been defined in Protégé. More on this topic in “Section 4 – Mappings Definition”.

SPARQL endpoint.
The SPARQL endpoint has been created by Ontop “geof”, using the mapping file, the ontology file, the properties file containing the connection configurations and the PostgreSQL JDBC driver.

User Interface (Java Application).
The user interface connects to the SPARQL endpoint to retrieve the needed data. More on this topic in “Section 5 – Java Functionalities”.

2) Data Sources

Below, a table representing the name of the objects present in the project, along with the name of the source website and the link to access the dataset. The links listed are either the download link or the link to the visualization page of the raw data.

Entity No.	Name	Source Name	Link
1	District	Stadt Zürich	https://opendata.swiss/en/dataset/stadtkreise1
2	Neighbourhood	Wikipedia	https://en.wikipedia.org/wiki/Subdivisions_of_Z%C3%BCrich
3	Bus stop	OpenData.swiss	https://data.stadt-zuerich.ch/dataset/ktzh_haltestellen_des_oeffentlichen_verkehrs_
4	Bus line	OpenData.swiss	"
5	Train station	Transport Open Data	http://transport.opendata.ch/v1/locations?query=Zurich
6	Car parking	OpenData.swiss	https://data.stadt-zuerich.ch/dataset/geo_oeffentlich_zugaengliche_parkhaeuser
7	Bike parking	OpenData.swiss	https://data.stadt-zuerich.ch/dataset/geo_zweiradparkierung
8	Bike rental	Zürich	https://www.zuerich.com/en/data/place/shopping/bike+hire
"	"	OpenData.swiss	https://data.stadt-zuerich.ch/dataset/geo_veloverleih
9	Restaurants	Stadt Zurich	https://www.zuerich.com/en/data/gastronomy/restaurants
10	Bars	Stadt Zurich	https://www.zuerich.com/en/data/gastronomy/nightlife/bars+%7C%7C+lounges
11	Museums	Stadt Zurich	https://www.zuerich.com/en/data/place/culture/museums
12	Attractions	Stadt Zurich	https://www.zuerich.com/en/data/place/attractions
13	Shops	Stadt Zurich	https://www.zuerich.com/en/data/place/shopping

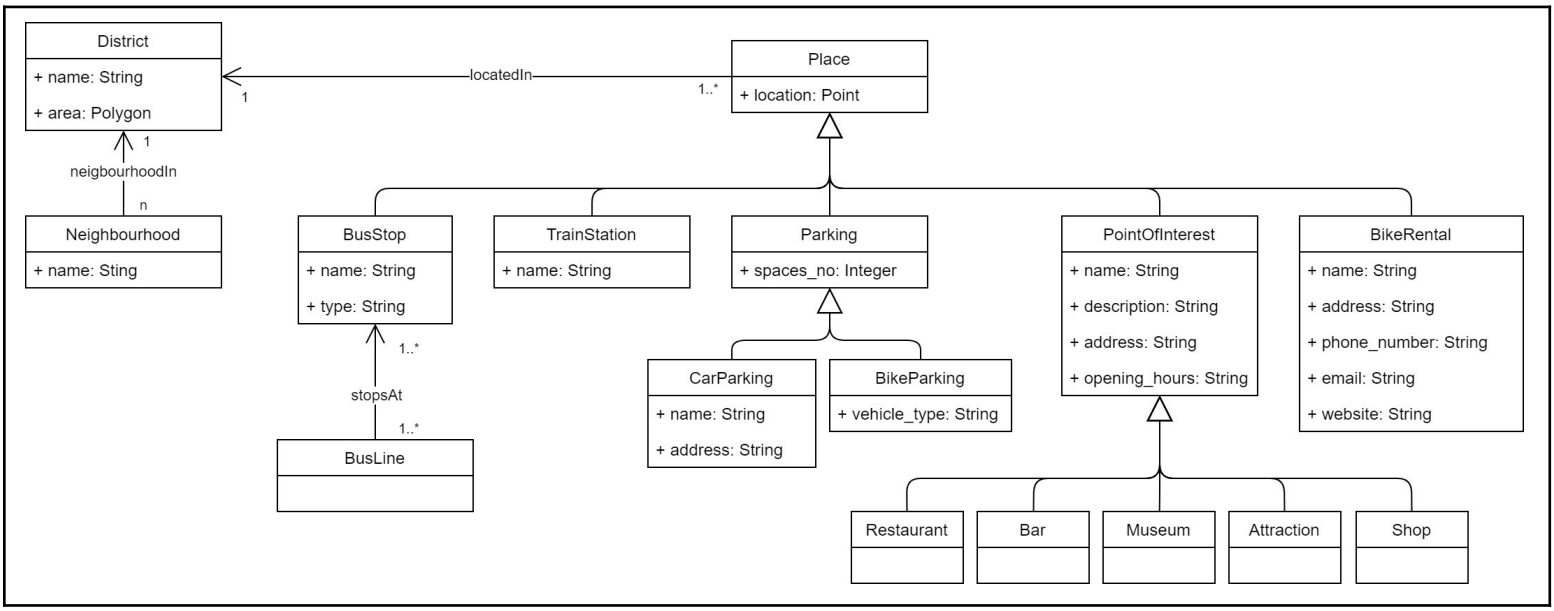
The following list adds information on some of the object types; The ones not listed are assumed to be self-explanatory:

- District, containing the 12 districts of Zurich; for each District, it is included the geographical area represented by a polygon of coordinates;
- Neighbourhood, belonging to a specific District;
- Bus Stops, representing public transportation stops for buses, trams or both;
- Bus Line, originated from the same dataset of the Bus Stops, containing the line code of the bus and tram lines;
- Bike Parking, representing parking spots for bikes, motorbikes or both;
- Bike Rental, which will then be the integration of two datasets, a CSV one and a JSON one;

All datasets, except the Neighbourhood one, contain geographical references of the location of the instances.

3) Ontology Definition

Here, the UML diagram represents the ontology of the web application.



The main association that can be found in the ontology is the “locatedIn” one. It represents where the location point of Place is located in the polygon area of District. The association is computed dynamically at the UI level, using the GeoSPARQL function called “sfWithin”.

Another interesting aspect of the ontology is the hierarchy created from “Place”. It was created in order to have a single class containing all the geographical point locations and a unique link to the District class. As it can be seen, its direct children are BusStop, TrainStation, Parking, PointOfInterest and BikeRental.

The same parent-child structure has been applied to the class Parking, with its subclasses CarParking and BikeParking, since the two classes are a specialization of the Parking one, containing information on the number of available parking spots. Also, the class PointOfInterest has subclasses, namely Restaurant, Bar, Museum, Attraction and Shop, since they share the same information.

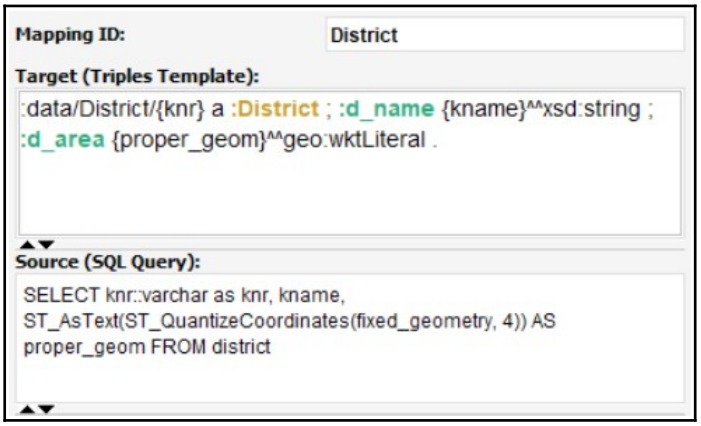
By using the parent-child structure, the retrieval of the data is enhanced, especially from the speed point of view.

4) Mappings Specification

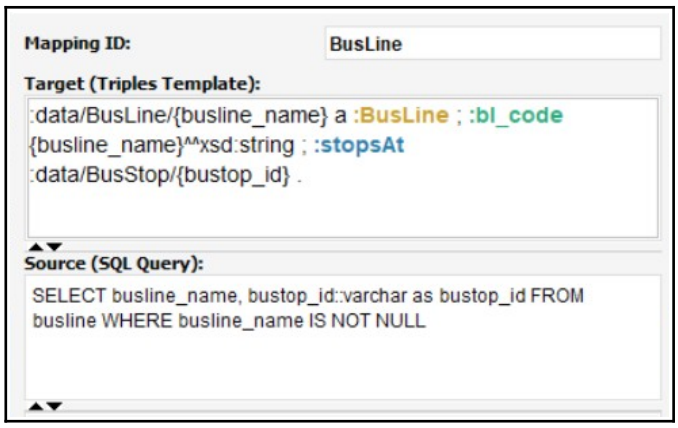
The mappings between the PostgreSQL datasets and the ontology has been defined inside the Protégé application with the Ontop plug-in and the PostgreSQL JDBC driver. All the mappings can be found in the file called “discover_zurich-protege_ontology_postgres.obda”. The most particular mappings include the following ones.

District. As it can be seen from the picture, the geometry that were stored in the PostgreSQL database did not keep their shape. Therefore, a method called “ST_QuantizeCoordinates” had to be applied on the data to convert them back to the original form.

BusLine. BusLine is one of the few classes having the definition of an object property. First, it was defined the IRI, then the code of the representing bus line, and finally the association to the BusStop class.

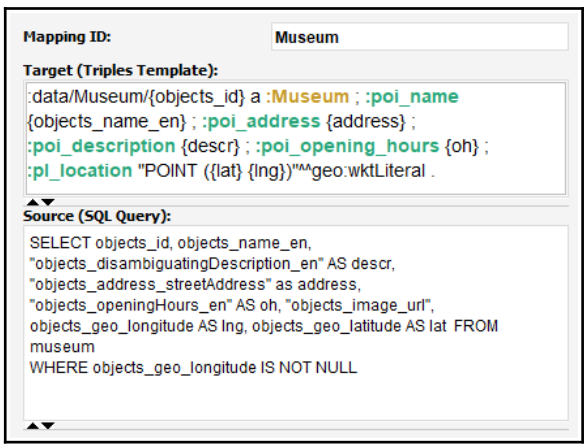


District Mapping



BusLine Mapping

Museum. The classes contains the definition of several data properties, including the location one. As it can be seen, the PostgreSQL table contains the latitude and longitude information into two separated fields. In order to utilize the geographical location, latitude and longitude have been merged into a single field, whose type was set to “geo:wktLiteral” in order to make the geographical functions work.

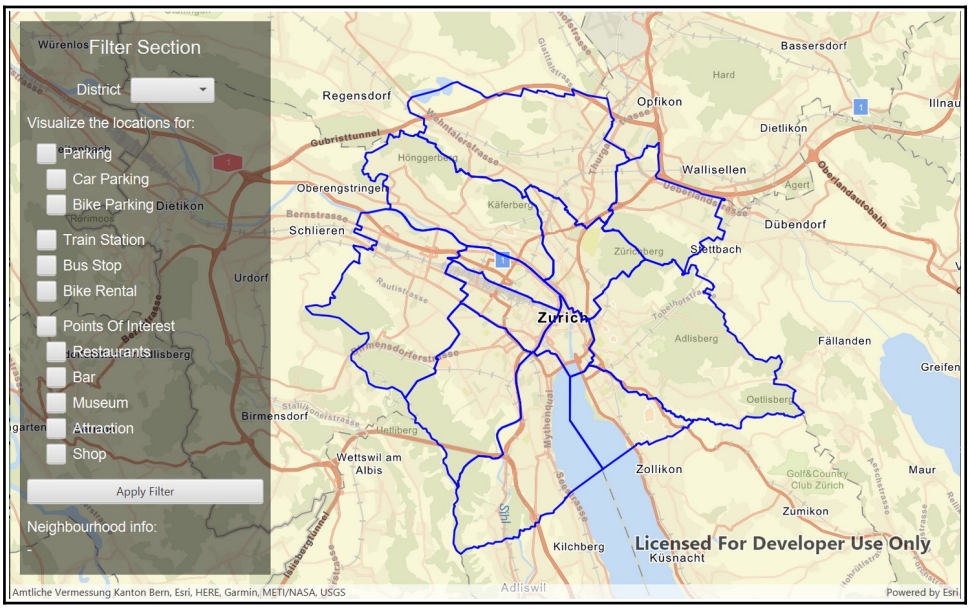


Museum Mapping

5) Java Functionalities

The User Interface has been implemented as a Java Application.

From the technical point of view, in order to access the data, RDF4J has been utilized to open the connection to the SPARQL endpoint generated by Ontop. Furthermore, in order to handle and display geographical data in a map, the library ArcGIS has been utilized. Moreover, to show graphical components like buttons and labels, the library JavaFX was chosen.



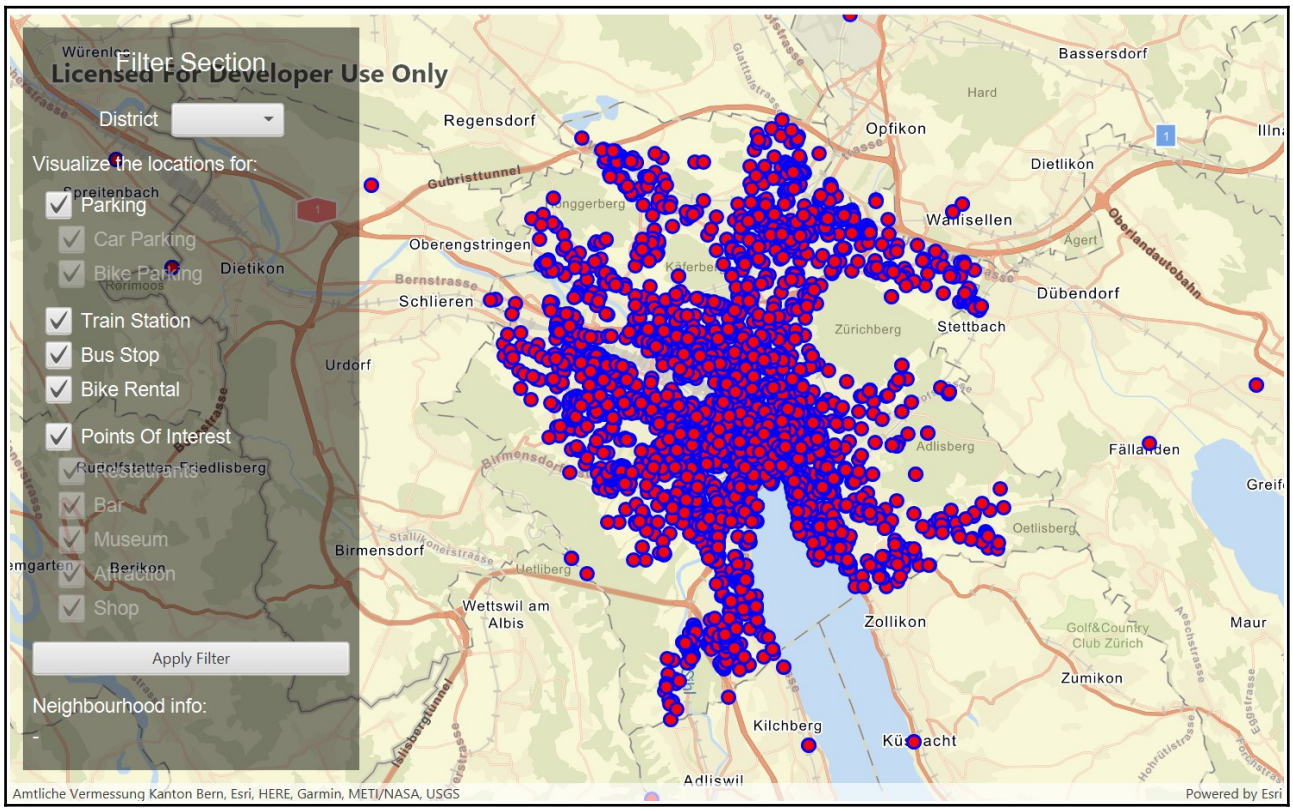
Main window of application

The window shows the map of Zurich, along with blue polygons, representing the 12 districts of the city. On the left side, there is the filter section. It contains check boxes and option boxes in order to select the district and decide whether to display the location of Parking, like Car Parking and Bike Parking, Train Stations, Bus Stops, Bike Rentals and Point Of Interest, like Restaurants, Bars, Museums, Attractions, Shops.

After the display request has been entered in the filter section and the “Apply Filter” has been clicked, the window will update and display the filtered data as points on the map. Furthermore, by clicking a point in the map, a pop-up will open, containing information about the selected point. Moreover, under the filter section, a label will display the name of the neighbourhoods contained in the selected district.

Pop-up of clicked point on the map

In order to see all the datapoints of the city, the user has to check all the places check boxes in the filter section and not select a district.



Visualization of all the data points