Dipartimento di Ingegneria e Scienza dell'Informazione

– KnowDive Group –

KGE 2022 - Project Report
Trentino Urban Transportation

Document Data:	Reference Persons:
January 10, 2023	Diego Barquero Morera, Vahan Petrosyan

Index:

1 Introduction **1**

2 Purpose and Domain of Interest (DoI) **1**
2.1 Purpose 1
2.2 Domain of Interest 1

3 Knowledge and Data Resources **1**
3.1 Knowledge sources 1
3.2 Data sources 2

4 Purpose Formalization **3**
4.1 Scenarios 3
4.2 Personas 3
4.3 Competency Questions (CQs) 3
4.4 Entities identified 5

5 Inception **6**
5.1 Data collection 6
5.2 Data processing (filtering and cleaning) 6
5.3 Resources knowledge definition/extraction 7

6 Informal Modeling **10**
6.1 Teleology foundations 10
6.2 ER model description 12

7 Formal Modeling **14**
7.1 ETG generation 14
7.2 Language alignment 16
7.3 Formal data alignment 16

8 KGC **17**

9 Outcome Exploitation **18**
9.1 KG information 18
9.2 Coverage 18
9.3 KG exploitation through SparQL 19

10 Conclusions & Open Issues **23**

Revision History:

Revision	Date	Author	Description of Changes
0.1	14.11.2022	Diego Barquero Morera, Vahan Petrosyan	First report progress - Phase 0: inception
0.2	23.11.2022	Diego Barquero Morera, Vahan Petrosyan	Second report progress - Phase 1: informal modeling
0.3	06.12.2022	Diego Barquero Morera, Vahan Petrosyan	Third report progress - Phase 2: formal modeling
0.4	05.01.2023	Diego Barquero Morera, Vahan Petrosyan	Fourth report progress - Phase 3: data integration
1.0	09.01.2023	Diego Barquero Morera, Vahan Petrosyan	Final report progress - Closing phase: exploitation, conclusions

1 Introduction

Reusability is one of the main principles in the Knowledge Graph Engineering (KGE) process defined by iTelos. The KGE project documentation plays an important role in order to enhance the reusabiltiy of the resources handled and produced during the process. A clear description of the resources and the process developed, provides a clear understanding of the KGE project, thus serving such an information to external readers in order to exploit that in new projects.

The current document aims to provide a detailed report of the *"Trentino Urban Transportation"* KGE project developed following the iTelos methodology.

2 Purpose and Domain of Interest (DoI)

2.1 Purpose

The goal of this project is to build a Knowledge Graph (KG) which satisfies the following purpose:

> "A person currently in an urban area of Trentino region wants to easily move from one place to another by means of public transportation."

This will be obtained by integrating static transportation data from different sources and producing a reusable ontology. The KG will be built under the assumption that its first use will be an application for easily and conveniently accessing information about bus stops, train stations and timetables of the trips inside the urban area of Trentino.

2.2 Domain of Interest

The KGE project considers the public transportation of urban areas of the region of Trentino (Italy) over a period of time of 10 months (between September 2022 and June 2023).

3 Knowledge and Data Resources

3.1 Knowledge sources

Due to the fact that schemas play a crucial role in the knowledge and data integration, they were taken as the initial knowledge resources. For this reason schema.org has been considered for the entity types, which are presented in the Table 1.

Name	Reference
Agency	Organization
Stops	TrainStation BusStop Trip
Trips and Routes	TravelAction TrainTrip BusTrip
Calendar and Timetable	Schedule Event
Locations	Place

Table 1: Schemas used for the knowledge integration

Even though the used schemas ensure the fundamental concepts for the creation of ontology, and hence, for the development of the further stages, it is still not enough for covering the specific properties available in the data. The latter can be solved by using GTFS documentation.

3.2 Data sources

The main source of urban transportation data for the Trentino region is Trentino Transporti website. In the Open Data section, it's possible to find curated files with information about the bus stops (position, name, etc.) as well as timetables of bus routes and how the routes change according to the weekday or calendar date. These files cover information for one year, and follow the guidelines of the GTFS specifications mentioned before.

For the urban routes regarding the train service, unstructured data (i.e. timetables in PDF format) is available in the Ferrovia section of said website. Furthermore, the geolocation of the train lines of interest can be obtained using the Google Maps service, either by manual extraction or usage of its API.

4 Purpose Formalization

The knowledge layer of the KG can be extracted from the purpose by formalizing a set of scenarios, personas and competency questions. For this project, they are described as follows:

4.1 Scenarios

The following scenarios were defined in order to cover the main cases addressed.

1. The urban areas of Trentino during working days (i.e. feriali).
2. The urban areas of Trentino during weekend / national holidays (i.e. festivi) and maintainance.
3. The urban areas of Trentino during the nighttime.

4.2 Personas

For the purpose of describing the competency questions, six personas were defined with different backgrounds, e.g. age groups and occupation.

1. Vahan and Diego are international students living in Trento. They reside in the **San Bartolomeo student residence** and attend lectures at **Povo**.
2. Chiara is a 45 year old person who had a car accident and now needs a wheelchair to move around the city.
3. Claudio is a 17 year old high school student living in Rovereto (**Lizzanella**). He uses the urban transportation for attending the classes at school.
4. Giulia is a 50 year old woman living in the northern part of Trento (**Gardolo 4 Novembre**). She usually goes to the office in the southern part of the city (**Mattarello Via Nazionale**).
5. Rodrigo is a 56 year old worker living outside of Trento. He works in the city, but far from the main train station (**Mattarello Gotarda**) and needs to use public means of transportation in order to reach the workplace.

4.3 Competency Questions (CQs)

The following CQs were developed as combinations of proposed scenarios and personas and describe specific situations, such as the wheelchair availability for the stops and routes.

1. Vahan needs to go to classes at Povo on a normal Monday morning before 11:30 am. He takes his lunch at noon so he is wondering if it's better to take the bus or the train (whichever departs last).
2. Diego has to go from Povo back to his room at 4:30pm each workday after classes. He feels car-sick when taking the bus, so he highly prefers taking the train.

3. Vahan finishes classes at 4:00 pm in Povo. He knows it's possible to reach his dorm (Verona Questura) using two different routes (line 13 and line 5 + line 3), and is wondering which way is faster.
4. Rodrigo works at the restaurant during the weekend and needs to find the best time for reaching his workplace at 11 am.
5. Giulia is going to a birthday party next Saturday in the city centre (Trento). The party tends to end at midnight, so she is wondering when is the last bus to return back home.
6. Claudio has booked a train from Rovereto to Bolzano on 1st of November, 9:00 am. He is curious about which bus he should take from his apartment in order to reach the train station on time as it is on a national holiday.
7. Giulia finished her work later than usual due to a project deadline and managed to arrive to the city center at 11pm thanks to a friend. Now she needs to go back to her house in the north.
8. Claudio should prepare for the exams during the summer exam session and decides to go to the central library (Biblioteca Universitaria Rovereto). He needs to know when the first bus is after 9 am.
9. Diego will be volunteering during a sport festival and needs to go from the dormitory to the city centre before 9 am.
10. Vahan must go from Povo to his dorm (12:30 pm) and wants to try to take a train. However, he heard that there's some maintenance on the rails and he wants to check which is the next available train.
11. Chiara wants to go from the city centre of Trento to the north, however, she has to get a bus for people with reduced mobility.
12. Diego was shopping in the city center and it started raining, so now he's looking to go back home. He would prefer waiting on a bus stop that allows to cover from the rain.
13. Rodrigo is tired after working all day and is heading back to Trento main station. He thinks he has to wait a lot for the next bus, and would like to know which bus stops have some place to sit.

4.4 Entities identified

CQ	Scenario	Common	Core	Contextual
1	1	Location, datetime	Stop, trip	Workday, morning
2	1	Location, datetime	Stop, trip	Workday, afternoon, train
3	1	Location, datetime	Stop, trip	Workday, afternoon
4	2	Location, datetime	Stop, trip	Weekend, morning
5	3	Location, datetime	Stop, trip	Workday, night time
6	2	Location, datetime	Stop, trip	Holiday
7	3	Location, datetime	Stop, trip	Workday, night time
8	2	Location, datetime	Stop, trip	Morning, bus
9	2	Location, datetime	Stop, trip	Holiday, morning
10	2	Location, datetime	Stop, trip	Workday, afternoon, maintenance, train
11	1	Location, datetime	Stop, trip	Bus, wheelchair
12	1	Location, datetime	Stop, trip	Bus, stop covering
13	1	Location, datetime	Stop, trip	Bus, stop seats

Table 2: Entities identified from the CQs

The **common** entity types *location* and *datetime* were prevalent in all CQs as well as the **core** entity types *stop* and *trip*. On the other hand, different **contextual** entity types were defined from the CQs, namely: *workday*, *holiday*, *morning*, *afternoon*, *night time*, *bus*, *train*, *maintenance*, *wheelchair*, *stop covering*, *stop seats*.

5 Inception

5.1 Data collection

The data was recovered in different formats from the different sources mentioned before. These are summarized in table 3. Regarding the geolocation of the train lines of interest, they were manually extracted from the Google Maps service.

File names	Description	Format	Domain	Source
agency.txt calendar.txt calendar_dates.txt feed_info.txt routes.txt shapes.txt stops.txt stopslevel.txt stop_times.txt transfers.txt trips.txt	Data under the GTFS standard regarding the bus stops and lines in the urban area of Trentino (don't include trains).	CSV	bus	Trentino Transporti
FTMAndata.pdf FTMRitorno.pdf LineaValsuganaAndata.pdf LineaValsuganaRitorno.pdf	Timetables for the Trento-Malé-Mezzana and Trento-Valsugana-Bassano_del_Grappa train lines.	PDF	train	Trentino Transporti
stops.csv	Locations of the urban train stations (Trento), manually extracted from the Google Maps service.	CSV	train	Google Maps

Table 3: Characteristics of the data collected from different sources

5.2 Data processing (filtering and cleaning)

The data that needed the most processing was the timetables of train routes, as they were in an unstructured format (PDF). The data for the train stations of interest (i.e. the ones inside the urban area of Trento) was manually extracted as a text file, and then processed with a Python script to obtain an intermediary and better structured file in JSON format.

The intermediary JSON file was further processed to generate a set of CSV files compliant with the GTFS standard and analogous to the CSV files already obtained for the bus transportation. This allowed to merge the bus and train GTFS datasets into a single one, and discard the information that was considered not required for the next stages of the project. Both of these steps were performed using Python scripts as well.

Finally, additional information was needed to further describe the bus stops and train stations, following what was obtained from the purpose formalization: **seats**, a boolean value that specifies whether there is place to seat or not while waiting for the bus or train; and **covered**, a boolean value that describes whether the stop offers a place to be covered from the rain or snow. For the sake of this project and having in mind its educational purpose, this data was simulated following a normal random distribution by means of a Python script as a viable alternative to actually spending time and money to produce the real data from scratch.

5.3 Resources knowledge definition/extraction

For the purpose of knowledge and data integration several tools were used. First of all, an ontology was constructed using *Protege* development tool. This was done based on the main entities extracted from the data structure and afterwards the schemas, available on the *schema.org*, were considered for modelling the common concepts. As a result, the following entities were obtained (Figure 1).

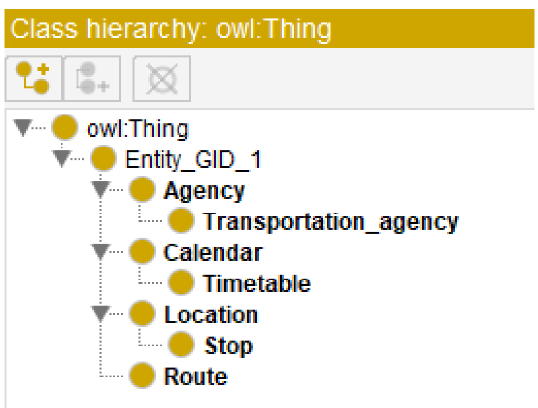

Figure 1: Ontology created using Protege development tool

Data properties were initialized based on the data structure while there were only four main object properties defined. All the properties are shown in the Table 4.

Property name	Property type	Domain(s)	Range(s)
has_phone	Data property	Agency	xsd:string
has_url	Data property	Agency	xsd:string
has_name	Data property	Agency, Route, Location	xsd:string
has_id	Data property	Agency, Timetable, Route, Stop	xsd:int
has_timezone	Data property	Agency, Location	xsd:string
has_langauge	Data property	Agency	xsd:string
has_weekdays	Data property	Calendar	xsd:dateTime
has_start_date	Data property	Calendar	xsd:dateTime
has_end_date	Data property	Calendar	xsd:dateTime
has_sequence_number	Data property	Stop	xsd:int
has_seats	Data property	Stop	xsd:boolean
has_covering	Data property	Stop	xsd:boolean
has_wheelchair_accessibility	Data property	Route, Stop	xsd:boolean
has_type	Data property	Route, Stop	xsd:int
has_start_location	Data property	Route	xsd:string
has_end_location	Data property	Route	xsd:string
has_direction	Data property	Route	xsd:boolean
has_longitude	Data property	Location	xsd:double
has_latitude	Data property	Location	xsd:double
has_departure_time	Data property	Timetable	xsd:dateTime
has_arrival_time	Data property	Timetable	xsd:dateTime
has_route	Object property	Transportation Agency	Route
has_calendar	Object property	Route	Calendar
has_stops	Object property	Route	Stop
has_timetable	Object property	Route	Timetable

Table 4: Data and Object Properties created in the ontology

Finally, the obtained ontology was uploaded in *Karma* data integration tool along with the filtered data. Based on the available information, the *links* were established and the final results were extracted in RDF format. An example for Transportation Agency is shown in the figure below (figure 2). The content of the corresponding RDF file is provided as well (figure 3).

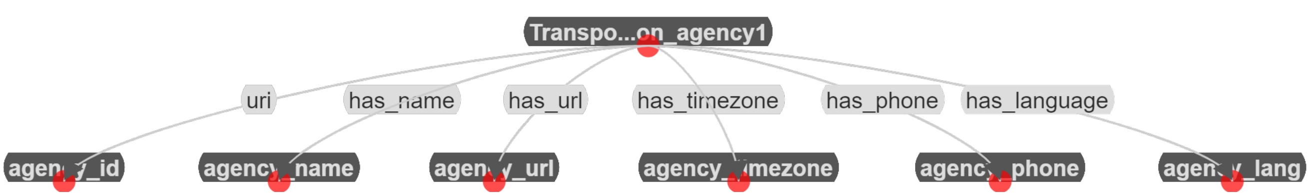

Figure 2: Transportation agency graph constructed using Karma data integration tool

```
<http://localhost:8080/source/12> <http://knowdive.disi.unitn.it/etype#has_url>
"https://www.trentinotrasporti.it/" .
<http://localhost:8080/source/12> <http://knowdive.disi.unitn.it/etype#has_name>
"Trentino trasporti S.p.A." .
<http://localhost:8080/source/12> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://knowdive.disi.unitn.it/etype#Transportation_agency> .
<http://localhost:8080/source/12>
<http://knowdive.disi.unitn.it/etype#has_language> "it" .
<http://localhost:8080/source/12>
<http://knowdive.disi.unitn.it/etype#has_timezone> "Europe/Rome" .
<http://localhost:8080/source/12> <http://knowdive.disi.unitn.it/etype#has_phone>
"+39 0461 821000" .
```

Figure 3: Content of the RDF file for Transportation agency graph

6 Informal Modeling

6.1 Teleology foundations

A set of components of the foundational teleology were extracted from the CQs described in the inception phase, with the aim of defining the entity types for the ER model. These are listed in table 5

Components	Common	Core	Contextual
Object	location date time person	stop line schedule	bus train wheelchair workday weekend morning afternoon night holiday
Action	decision making	-	maintainance raining
Function	student worker	trip	departure arrival cover available seats

Table 5: Teleology components classified as object, action or function, according to the information extracted from the CQs.

Using this information as reference, the main entity types that were chosen for the next step were: **Transportation_agency**, **Line**, **Trip**, **Schedule**, **Calendar**, **Stop**. These were used to upgrade the ontology (previously defined in figure 1), and are summarized in figure 4. Furthermore, the object and data properties were defined as explained in tables 6 and 7.

Property name	Description	Domain(s)	Range(s)
has_id	Unique identifier of an instance/entity	Agency Stop Line Trip Calendar	xsd:int
has_name	Name of an instance/entity	Agency Line Location	xsd:string
has_phone	Phone number of an organization	Agency	xsd:string
has_language	Official language of an organization	Agency	xsd:string
has_timezone	Time zone of an organization	Agency	xsd:string
has_url	Website of an organization	Agency	xsd:string
has_headsign	Short description of the trip	Trip	xsd:string
has_direction	Direction of the trip/line	Trip	xsd:boolean
has_wheelchair_accessibility	Availability of facilities for people with reduced mobility	Trip Stop	xsd:boolean
has_covering	Availability of a covering in a stop	Stop	xsd:boolean
has_seats	Availability of seats in a stop	Stop	xsd:boolean
has_latitude	Latitude of the location	Location	xsd:double
has_longitude	Longitude of the location	Location	xsd:double
has_altitude	Altitude of the location	Region	xsd:double
has_arrival_time	Arrival time of the transportation mean in a stop	Schedule	xsd:dateTime
has_departure_time	Departure time of the transportation mean in a stop	Schedule	xsd:dateTime
has_stops_sequence	Sequence of stops which are included in the current trip	Schedule	xsd:int
has_weekdays	Weekdays when the current trip runs	Calendar	xsd:string
has_end_date	End date of a specific trip	Calendar	xsd:dateTime
has_start_date	Start date of a specific trip	Calendar	xsd:dateTime
has_long_name	Full name of the line	Line	xsd:string
has_type	Type of a transportation mean, i.e., train or bus	Line	xsd:int

Table 6: Data Properties extracted from the CQs.

Property name	Description	Domain(s)	Range(s)
has_location_in	Urban area where a stop is located	Stop	Region
has_stops	Sequence of stops which are included in a trip	Schedule	Stop
has_lines	Lines operated by a transportation organization	Agency	Line
has_trips	Trips included in the route	Line	Trip
has_calendar	Weekdays when the current trip runs	Trip	Calendar
has_schedule	Timetable of transportation means	Stop, Trip	Schedule

Table 7: Object Properties extracted from the CQs.

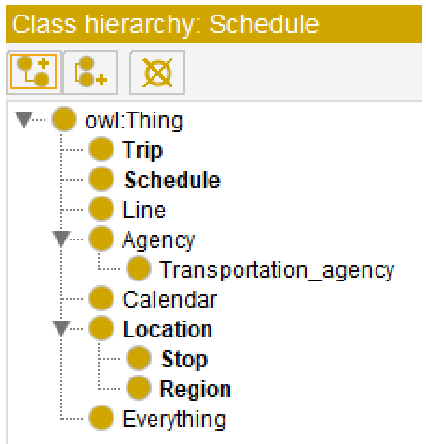

Figure 4: Updated ontology after the informal modelling

6.2 ER model description

With the insights obtained from the teleology foundation sub-phase, the chosen entity types and the object and data properties were schematized into an ER model (figure 5). This diagram summarizes the main aspects that should be covered by the knowledge layer of the final KG, and will be of help for the next stages.

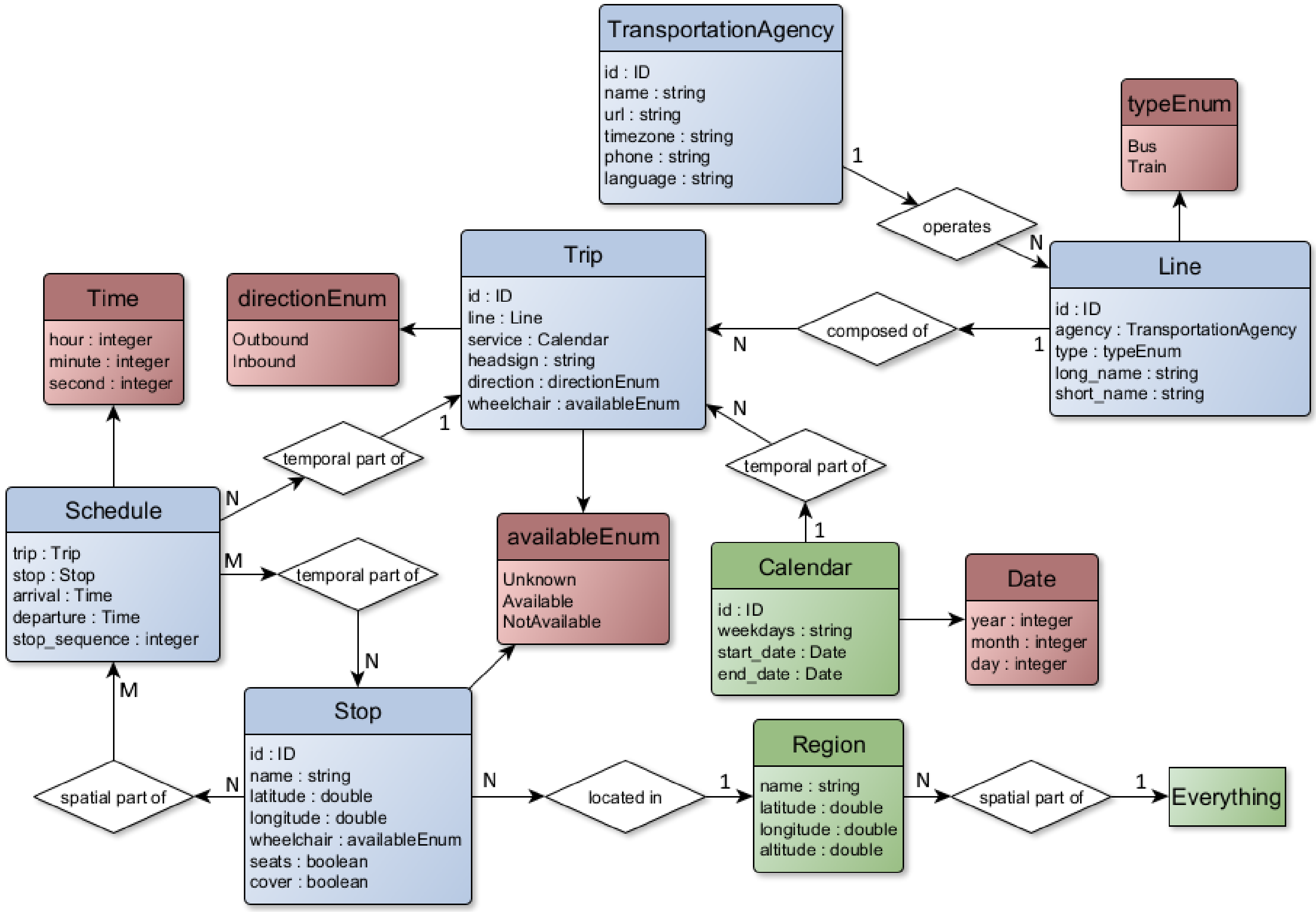

Figure 5: Entity relationship diagram.

7 Formal Modeling

7.1 ETG generation

First, an ontology translation of the GTFS format (figure 6) was considered as reference and the entity types *ServiceRule* and *Agency* were obtained from it. An additional entity type was added to this ontology, namely *TravelAction*, which was obtained from schema.org. This intermediate ontology is presented in figure 7.

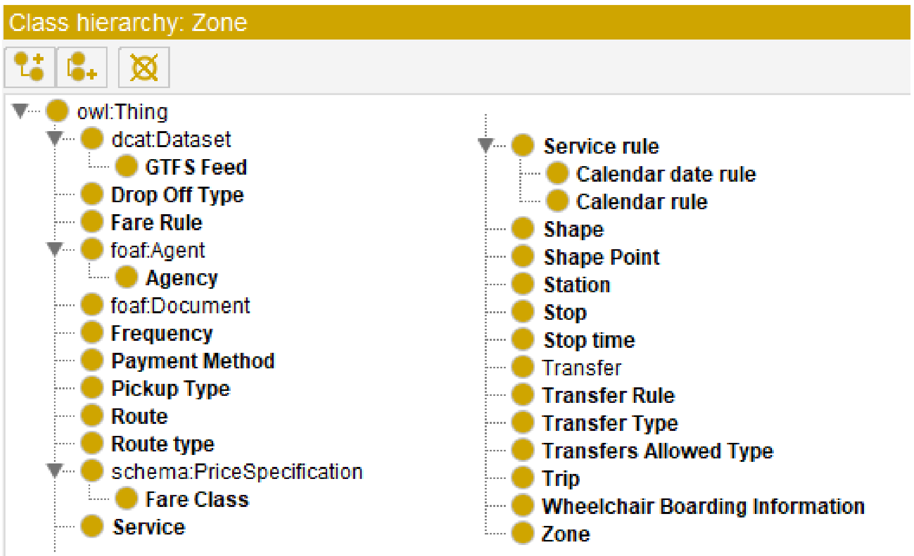

Figure 6: Ontology translation of the GTFS format, used as part of the formalization of this project's schema. Source.

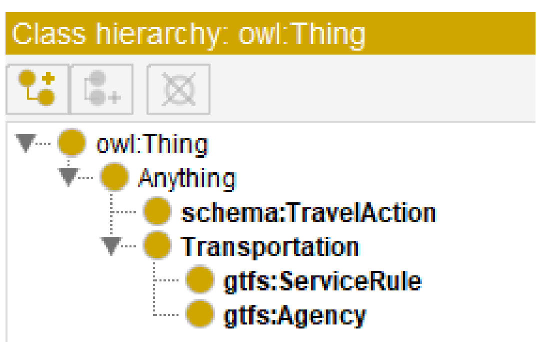

Figure 7: Intermediate ontology with e-types extracted from *schema.org* and the ontology translation of the GTFS format.

Then, the ontology (figure 4) and teleology (figure 5) proposed in the informal modelling were merged together into a single teleontology. This structure was further integrated with the intermediate ontology described before, to obtain a combined structure represented in figure 8.

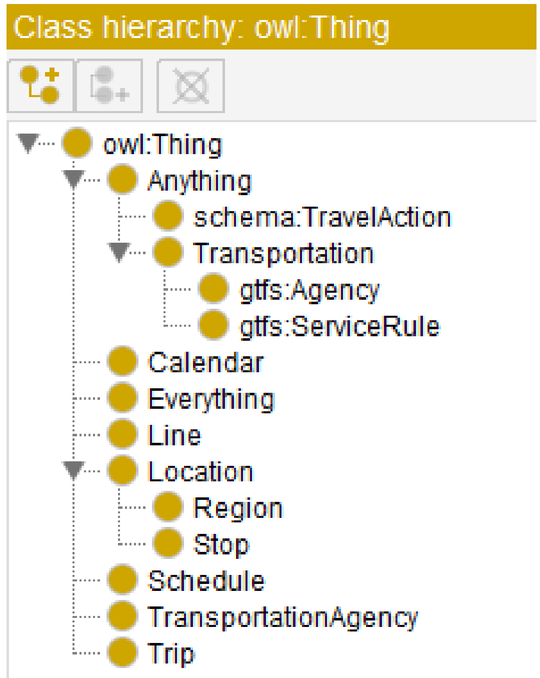

Figure 8: Intermediate teleontology.

From this combination, the e-types *Everything*, *Anything* and *Region* were removed, as they were too vague and already fulfilled their purpose in the previous stage. Furthermore, the e-types were reorganized to better fit accordingly. As a result of these two steps, a new schema was obtained, namely a formalized entity type graph (ETG) (figure 9).

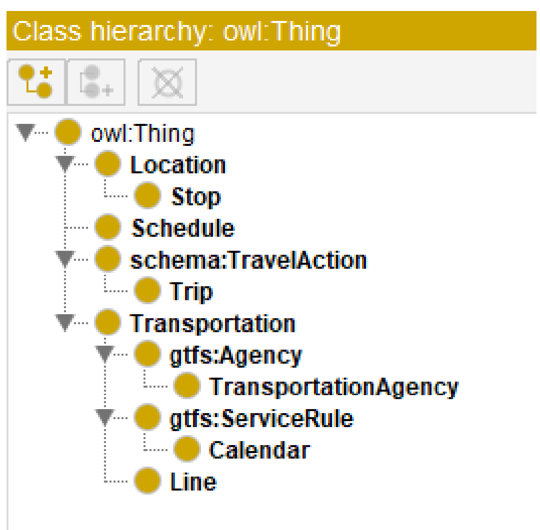

Figure 9: Formalized schema.

7.2 Language alignment

To finalize the formalization of the ETG, the entity types needed to be standardized under non ambiguous identifiers, which will aid in the future reusability of the schema. The KOS tool was used to achieve this language alignment by assigning unique global identifiers (GIDs) to each entity type, following the Universal Knowledge Core (UKC) standard. The aligned schema is represented in figure 10.

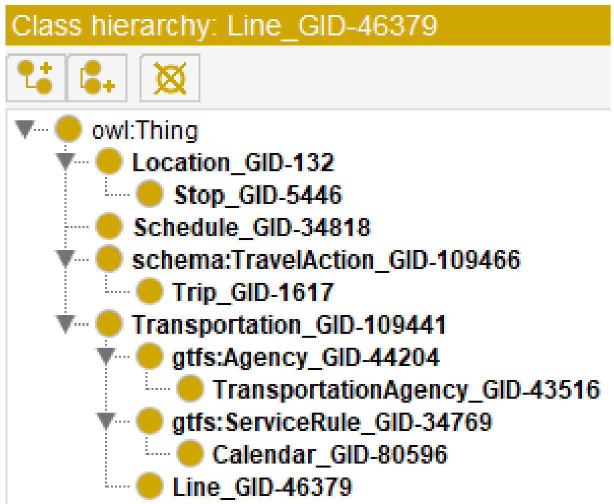

Figure 10: Language aligned ETG.

7.3 Formal data alignment

As preparation for the next stage, the datasets were processed so that the data is adjusted according to the final ETG schema. This will allow a more seamless data integration during the Knowledge Graph Construction. The polishing performed on the datasets is summarized as following:

- Renamed the dataset files to match their correspondent e-type (e.g. *stop_times.csv* renamed to *schedules.csv*).
- Renamed some columns to better fit their correspondent e-type (e.g. *calendar_id* instead of *service_id*).
- In the original dataset file *calendar.csv*, the availability of the routes during the week was represented by separate columns, i.e., one for each weekday as a boolean value representing availability. However, for the *calendar* e-type, *weekdays* is a data property defined as a string of seven characters, either 0 or 1, which encodes this information exactly in the same way but in a more compact representation. For the aligned *calendar.csv* dataset, this was taken into consideration and these columns were merged accordingly.
- For the aligned *stops.csv* dataset, the missing columns for the *seats* and *covered* data properties were incorporated by merging the incomplete *stops.csv* dataset with the simulated dataset containing this information.

8 KGC

The final stage of the iTelos methodology considers the development of the EG. For this purpose, the filtered datasets and the final language-aligned schema have been given as inputs to the KarmaLinker data linking tool. This procedure required finding the appropriate URIs in the datasets for each Etype. The results have been extracted as RDF files which represent the links created between data properties and the data variables as well as the connections between different Etypes in the datasets, i.e. object properties. The examples of the connections are presented in the figures 21, 12, 13.

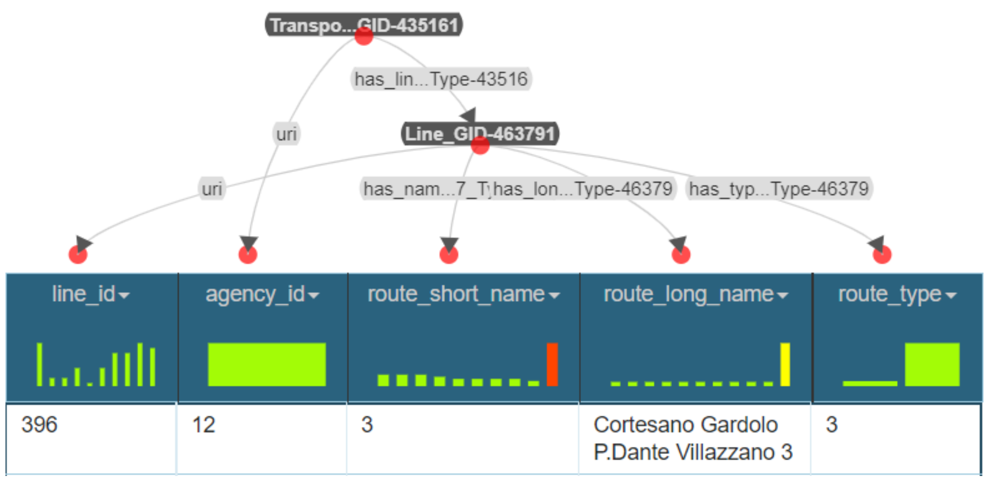

Figure 11: Data Integration in KarmaLinker: Lines

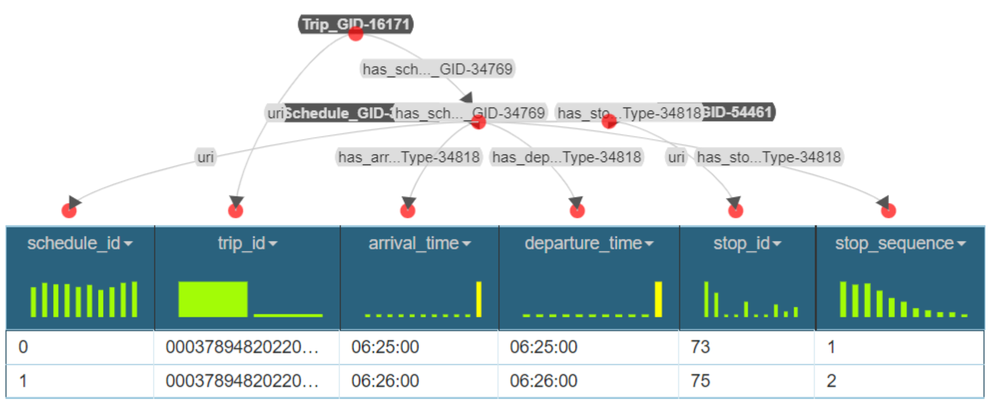

Figure 12: Data Integration in KarmaLinker: Schedule

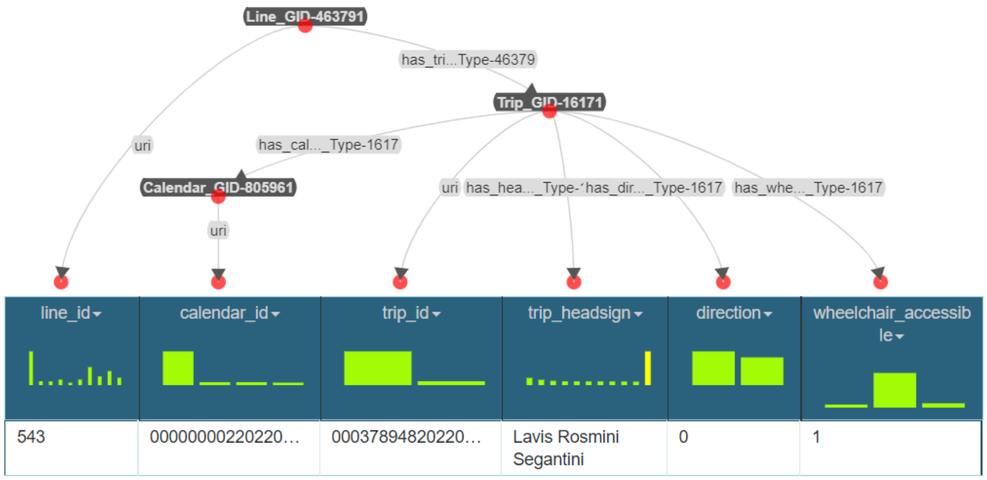

Figure 13: Data Integration in KarmaLinker:Trips

9 Outcome Exploitation

9.1 KG information

Following are some specifications of the final Knowledge Graph produced, obtained with the aid of the GraphDB tool.

- 6 etypes, namely:
  1. TransportationAgency_GID-43516 with **1 instance** and **387 links** to other etypes.
  2. Calendar_GID-80596 with **43 instances** and approximately **3K links** to other etypes.
  3. Line_GID-46379 with **43 instances** and approximately **7K links** to other etypes.
  4. Stop_GID-5446 with **1112 instances** and approximately **160K links** to other etypes.
  5. Trip_GID-1617 with **3127 instances** and approximately **85K links** to other etypes.
  6. Schedule_GID-34818 with **77923 instances** and approximately **389K links** to other etypes.
- 5 object properties
- 21 data properties

9.2 Coverage

Furthermore, the coverage of both etypes and object/data properties was considered as how much of the original data successfully filled the knowledge layer of the KG, with accordance to the CQs posed at the beginning. In this case, the coverage for the etypes was 1, taking into account that the datasets covered $6/6$ of the etypes present. This is of course an important result, as each entity is crucial for answering the CQs effectively (except *TransportationAgency_GID-43516* in this case) and having an imperfect coverage in this aspect would be disastrous.

In the case of coverage for the properties, the original datasets lacked the information for 2 data properties, namely *has_covering_GID-22933_Type-5446* and *has_seats_GID-15151_Type-5446* (both regarding *Stop_GID-5446*). This means that the coverage obtained is $24/26 = 0.92$. This is still a good result in practical terms, however, it's important to keep in mind that the last two CQs would have been impossible to answer if not by the fact that example data was generated.

9.3 KG exploitation through SparQL

For verifying that the produced KG is capable of answering the CQs, it was queried for each QC using SparQL through the tool GraphDB. By doing this, all CQs were able to be answered. Below are some examples of the queries performed and the results given by the tool (a full list of all the queries is available in the repository of the project).

```
###########################
#CQ4
###########################
# Rodrigo works at the restaurant during the weekend and needs to find the best time for reaching his workplace at 11 am.
PREFIX rdf: <http://knowdive.disi.unitn.it/etype#>

select distinct ?name_start ?name_end ?departure ?arrival ?linenumber ?linename ?weekdays
where {
    ?line a rdf:Line_GID-46379;
        rdf:has_name_GID-34017_Type-132 ?linenumber;
        rdf:has_long_name_GID-34017_Type-46379 ?linename;
        rdf:has_trips_GID-1501_Type-46379 ?trip.

    ?trip
        rdf:has_calendar_GID-80596_Type-1617 ?calendar;
        rdf:has_schedule_GID-34769 ?schedule_start;
        rdf:has_schedule_GID-34769 ?schedule_end.

    ?calendar rdf:has_weekdays_GID-80597_Type-80596 ?weekdays.

    ?schedule_start
        rdf:has_stops_GID-5446_Type-34818 ?stops_start;
        rdf:has_departure_time_GID-80846_Type-34818 ?departure.

    ?schedule_end
        rdf:has_stops_GID-5446_Type-34818 ?stops_end;
        rdf:has_arrival_time_GID-80845_Type-34818 ?arrival.

    ?stops_start rdf:has_name_GID-34017_Type-132 ?name_start.
    ?stops_end   rdf:has_name_GID-34017_Type-132 ?name_end.

    FILTER (!regex (?weekdays, "00$"))
    FILTER (?arrival < "11:00:00")
    FILTER (?departure < ?arrival)

    FILTER (CONTAINS(?name_start, "Piazza Dante"))
    FILTER (CONTAINS(?name_end, "Mattarello Gotarda"))

}
ORDER BY DESC (?arrival) LIMIT 10
```

Figure 14: Example of SparQL query for answering CQ #4.

	name_start	name_end	departure	arrival	linenumber	linename	weekdays
1	"Piazza Dante "Stazione F s"	"Mattarello Gotarda"	"10:37:00"	"10:58:00"	"8"	"Centochiavi Piazza Dante Mattarello"	"1111110"
2	"Piazza Dante "Stazione F s"	"Mattarello Gotarda"	"10:37:00"	"10:58:00"	"Rosmini "Cimitero" / Mus e"	"Centochiavi Piazza Dante Mattarello"	"1111110"
3	"Piazza Dante "Stazione F s"	"Mattarello Gotarda"	"10:07:00"	"10:28:00"	"8"	"Centochiavi Piazza Dante Mattarello"	"1111110"
4	"Piazza Dante "Stazione F s"	"Mattarello Gotarda"	"10:07:00"	"10:28:00"	"Rosmini "Cimitero" / Mus e"	"Centochiavi Piazza Dante Mattarello"	"1111110"
5	"Piazza Dante "Stazione F s"	"Mattarello Gotarda"	"09:50:00"	"10:11:00"	"8"	"Centochiavi Piazza Dante Mattarello"	"0000001"

Figure 15: Result of the SparQL query for CQ #4.

```
###########################
#CQ5
###########################
# Giulia is going to a birthday party next Saturday in the city centre (Trento). The party tends to end at midnight, so she is wondering when is the last bus to return back home.
PREFIX rdf: <http://knowdive.disi.unitn.it/etype#>
PREFIX omgeo: <http://www.ontotext.com/owlim/geo#>

select distinct ?stopname ?departure ?linenumber ?linename
where {
      ?line a rdf:Line_GID-46379;
            rdf:has_name_GID-34017_Type-132 ?linenumber;
            rdf:has_long_name_GID-34017_Type-46379 ?linename;
            rdf:has_trips_GID-1501_Type-46379 ?trip.

      ?trip rdf:has_calendar_GID-80596_Type-1617 ?calendar;
            rdf:has_schedule_GID-34769 ?schedule_start;
            rdf:has_schedule_GID-34769 ?schedule_end.

      ?calendar rdf:has_weekdays_GID-80597_Type-80596 ?weekdays;
            FILTER regex (?weekdays, "1[01]$").

      ?schedule_start rdf:has_stops_GID-5446_Type-34818 ?stops_start;
            rdf:has_departure_time_GID-80846_Type-34818 ?departure.

      ?schedule_end
            rdf:has_stops_GID-5446_Type-34818 ?stops_end;
            rdf:has_arrival_time_GID-80845_Type-34818 ?arrival.

      ?stops_start rdf:has_name_GID-34017_Type-132 ?stopname;
            rdf:has_latitude_GID-46264_Type-132 ?lat;
            rdf:has_longitude_GID-46270_Type-132 ?long.

      ?stops_end rdf:has_name_GID-34017_Type-132 ?destiny.

      FILTER (?departure < ?arrival)

      ### Any stop at less than 500m from Piazza Dante train station
      FILTER (omgeo:distance(46.07209811, 11.11955396, ?lat, ?long) < 0.5)
      FILTER (CONTAINS(?destiny, "Gardolo 4 Nov. \"Piscina"))
}
ORDER BY DESC (?departure) LIMIT 5
```

Figure 16: Example of SparQL query for answering CQ #5.

	stopname	departure	linenumber	linename
1	"Romagnosi Vannetti"	"23:41:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"
2	"Sanzio "Castello"	"23:39:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"
3	"Romagnosi Vannetti"	"23:11:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"
4	"Sanzio "Castello"	"23:09:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"
5	"Romagnosi Vannetti"	"22:41:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"

Figure 17: Result of the SparQL query for CQ #5.

```
###########################
#CQ10
###########################
# Vahan must go from Povo to his dorm (12:30 pm) and wants to try to take a train. However, he heard that there's some maintenance on the rails and he wants to check which is the next available train.
PREFIX rdf: <http://knowdive.disi.unitn.it/etype#>
select distinct ?stopname ?departure ?linenumber ?linename
where {
        ?line a rdf:Line_GID-46379;
            rdf:has_name_GID-34017_Type-132 ?linenumber;
            rdf:has_long_name_GID-34017_Type-46379 ?linename;
            rdf:has_trips_GID-1501_Type-46379 ?trip;
            rdf:has_type_GID-25142_Type-46379 ?type;
            FILTER(?type = "2").

        ?trip rdf:has_schedule_GID-34769 ?schedule;
            rdf:has_direction_GID-46668_Type-1617 ?direction;
            rdf:has_calendar_GID-80596_Type-1617 ?calendar;
            FILTER(?direction = "1").

        ?calendar rdf:has_weekdays_GID-80597_Type-80596 ?weekdays;
            FILTER regex(?weekdays, "^11111").

        ?schedule rdf:has_stops_GID-5446_Type-34818 ?stops;
            rdf:has_departure_time_GID-80846_Type-34818 ?departure;
            FILTER(?departure >= "12:30:00").

        ?stops rdf:has_name_GID-34017_Type-132 ?stopname;
            FILTER CONTAINS(?stopname, "Povo - Mesiano").
}
ORDER BY ASC (?departure) LIMIT 5
```

Figure 18: Example of SparQL query for answering CQ #10.

	stopname	departure	linenumber	linename
1	"Povo - Mesiano"	"13:19:00"	"R16"	"Trento Bassano del Grappa"
2	"Povo - Mesiano"	"14:19:00"	"R16"	"Trento Bassano del Grappa"
3	"Povo - Mesiano"	"14:44:00"	"R16"	"Trento Bassano del Grappa"
4	"Povo - Mesiano"	"15:19:00"	"R16"	"Trento Bassano del Grappa"
5	"Povo - Mesiano"	"15:44:00"	"R16"	"Trento Bassano del Grappa"

Figure 19: Result of the SparQL query for CQ #10.

```
###########################
#CQ12
###########################
# Diego was shopping in the city center and it started raining, so now he’s looking to go back home. He would prefer waiting on a bus stop that allows to cover from the rain.
PREFIX rdf: <http://knowdive.disi.unitn.it/etype#>
PREFIX omgeo: <http://www.ontotext.com/owlim/geo#>

select distinct ?stopname ?departure ?linenumber ?linename
where {
      ?line a rdf:Line_GID-46379;
            rdf:has_name_GID-34017_Type-132 ?linenumber;
            rdf:has_long_name_GID-34017_Type-46379 ?linename;
            rdf:has_trips_GID-1501_Type-46379 ?trip.

      ?trip rdf:has_calendar_GID-80596_Type-1617 ?calendar;
            rdf:has_schedule_GID-34769 ?schedule_start;
            rdf:has_schedule_GID-34769 ?schedule_end.

      ?calendar rdf:has_weekdays_GID-80597_Type-80596 ?weekdays;
            FILTER regex (?weekdays, "1[01]$").

      ?schedule_start rdf:has_stops_GID-5446_Type-34818 ?stops_start;
            rdf:has_departure_time_GID-80846_Type-34818 ?departure.

      ?schedule_end
            rdf:has_stops_GID-5446_Type-34818 ?stops_end;
            rdf:has_arrival_time_GID-80845_Type-34818 ?arrival.

      ?stops_start rdf:has_name_GID-34017_Type-132 ?stopname;
            rdf:has_covering_GID-22933_Type-5446 ?covering;
            rdf:has_latitude_GID-46264_Type-132 ?lat;
            rdf:has_longitude_GID-46270_Type-132 ?long.

      ?stops_end rdf:has_name_GID-34017_Type-132 ?destiny.

      FILTER (?departure >= "13:00:00")
      FILTER (?departure < ?arrival)

      ### Any stop at less than 500m from Piazza Dante train station
      FILTER (omgeo:distance(46.07209811, 11.11955396, ?lat, ?long) < 0.5)
      FILTER (CONTAINS(?destiny, "Verona \"Questura"))

     FILTER (?covering = "1")
}
ORDER BY ASC (?departure) LIMIT 20
```

Figure 20: Example of SparQL query for answering CQ #12.

	stopname	departure	linenumber	linename
1	"Segantini Centa"	"13:04:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"
2	"Segantini Dogana"	"13:04:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"
3	"Piazza Dante "Stazione Fs"	"13:05:00"	"3"	"Cortesano Gardolo P.Dante Villazzano 3"
4	"Piazza Dante "Stazione Fs"	"13:06:00"	"13"	"P.Dante Rosmini S.Rocco Povo Polo Soc."
5	"Piazza Dante "Stazione Fs"	"13:06:00"	"3 Nov. "Ponte Cavalleggeri"	"P.Dante Rosmini S.Rocco Povo Polo Soc."
6	"Segantini Centa"	"13:06:00"	"8"	"Centochiavi Piazza Dante Mattarello"

Figure 21: Result of the SparQL query for CQ #12.

In the specific case of CQ #3 (comparison of time duration between taking two different lines), the approach proposed is to perform several queries instead of just one. Each query would give as a result the time duration of taking a single line, then it’s assumed that further simple operations can be performed to compare these times. In the given document a single query is provided, where the lines that need to be changed for performing the other queries is commented out.

10 Conclusions & Open Issues

During the development of the project several challenges came up which were partly overcome. The main issues were related to the tools dealing with the Knowledge Resources.

One of these was found in the Protege development tool during the Informal and Formal Modeling phases. In these stages several ontologies need to be merged, and for this reason, they were imported separately under one ontology. After making the appropriate changes the final ontology was not being properly saved and some information was lost. These required reconsideration of the development of the intermediate results by recreating the ontologies without importing others, i.e. manually creating a merged file and modifying that.

The second issue was related to the artificial generation of annotations in the ".owl" format files. This was leading to the non-visibility of domains of the data and object properties. As in the previous situation, this issue was solved by the manual removal of the annotations.

Another issue was found in the KarmaLinker data linking tool. In some cases the tool was not allowing to generate the connections from the data set variables. This has been solved by adding initial entities and only after that creating outgoing links to the data variables accordingly.

In brief, although, multiple obstacles were met during the project development phases, which were partly related to the used tools, solutions have been found for each of them. These steps allowed creating a reusable Knowledge Graph which in its turn may be used for meeting the requirement of the initial purpose and answering CQs by queries.