Dipartimento di Ingegneria e Scienza dell'Informazione

KDI Lab 2019-20

Document data:	**Reference persons:**
13.12.2019	Valentina Sofia Pigato - 211461 Andrei Diaconu - 215026 Davide Cappellaro - 207691 Alessandro Conti - 215034 Seyyed Arya Hassanli - 212168 Andrea Matté - 215064 Andrea Iossa - 215174

Index

Revision History

Revision	Date	Author	Description of Changes

1	23.10.2019	Model group (Valentina, Davide, Andrei, Alessandro)	• Document created • Scenario description • Storytelling definition • Personas definition
2	13.11.2019	Model group	• Competency queries
3	17.11.2019	Alessandro	• First version of generalized queries
4	18.11.2019	Valentina	• Query patterns
5	24.11.2019	Alessandro, Andrei	• First version of the informal model
6	30.11.2019	Model group	• Finalized formal model • Formal model • Formal model evaluation
7	1.12.2019	Davide	• Lexical information (WodNet)
8	3.12.2019	Model group	• Model visualization
9	5.12.2019	Model group	• Review of model sections
10	9.12.2019	Arya	• Datasets Extraction • Datasets Cleaning
11	11.12.2019	Model group	• Top-level grounding
12	13.12.2019	Whole group	• Final review

1. Scenario description

Nowadays we are used to planning every aspect of our lives, and free-time is no exception. It is a very important resource and requires to be managed optimally to allow liberty to follow every one of our passion and hobby. Moreover, the absence of a single board from which search for the events in the nearby makes the organization even more difficult. It would be a good idea if in our free time we have the opportunity to access easily to a large number of facilities that provide cultural events which are interesting to us. Of course, you can spend your free time doing anything you want, but it is always fun to do something outside that also develop your mind and your culture. Going to a museum, to a cinema or a concert are activities that can be pleasant but at the same time helps you to learn new things and increase your knowledge.

Unfortunately, at the current state of the art, no application allows an accurate organization of our free time about the events to which we would like to attend. Furthermore, no application allows arranging activities related to the educational and cultural sphere and filtering them by interests or timetables. For example, existing applications provide only a simple list of events together with some basic information, but we cannot have them scheduled or organized according to the cultural categories we prefer.

For this specific reason, we are proposing a knowledge base that will support the creation of a platform that allows the user to discover and access a large number of events in Trentino. The activities are provided by educational and cultural facilities (cinemas, museums, theatres, ...) and can be filtered by one's interests, proximity and time. Our purpose is to develop a single platform that unifies data from different sources about Trentino facilities responsible for the organization and management of educational and cultural events. With that, the user will be able to easily find his preferred cultural activities, all grouped in one single place.

To support our project, we introduce now four personas, each one of them with its needs, concerns, and preferences for what regards free-time and its management. Giacomo Rossi is a 74 years old retired music teacher who loves music and develops this passion by attending a lot of music events. Giuseppe Bianchi is a 40 years old man who wants to give his children some educational experiences on the weekends. Sara Ferrari is a first-year university student that recently moved to Trento in order to study Economics and loves following meetings about contemporary economic and political topics. Luca Garda is an employer who loves both art and science. During his days off he loves to go with his friends to art exhibits and museums. He's not interested in the distance he has to travel to take a peek at new art pieces and to study something new.

1.1. Storytelling definition

User	Real-World Action by Persona	System action
Giacomo Rossi	Search for any classical music concert that will take place tonight at "Società Filarmonica di Trento".	Returns all the concerts about classical music that are scheduled for tonight. Moreover, they have to be held at "Società Filarmonica di Trento".
Giacomo Rossi	Find out whether in October there will be a music concert in Trentino where Sara Rossi is one of the performers.	Returns all the concerts that will be held in Trentino in October only if there is a performer whose name is Sara Rossi. Return both the date and the facility in which the events will take place.
Giacomo Rossi	Find out whether the next weekend there will be concerts about Puccini, Bach or Tchaikovsky and discover the related timetables to see if they are overlapping.	Returns the concerts where Puccini, Bach or Tchaikovsky music is played next weekend. The values must include also the timetables and must inform the user in the case in which these events are overlapping or note.
Giacomo Rossi	Search for a music event where "La Boheme" by Puccini will be performed and in which theatre in the whole region.	Returns to the user the music events where "La Boheme" by Puccini will be performed and list all the theatres in the whole region where it will take place. Returns also the timetable of every mentioned facility.
Sara Ferrari	Search for talks about political and economic topics in the city of Trento.	Returns all the educational or cultural events about talks which deal with political and economic issues that will take place in the city of Trento.
Sara Ferrari	Search for all educational events scheduled nearby within the next 4 days.	Returns all the educational events that will occur within the next 4 days, Moreover, the events must take place within a set range from Sara Ferrari's actual geographic position.

Sara Ferrari	Look for comedy and comic films that will take place the next 2 weekends in the closest 3 cinemas.	Returns all the events related to films whose genres are comedy or comic and that are scheduled in the closest 3 cinemas from Sara Ferrari's actual position. The results must include only screening in the next two Saturdays and Sundays.
Sara Ferrari	Find what films will be visioned in "Cinema Teatro Nuovo Roma" that night.	Returns all the films that will be projected that night at the cinema "Cinema Teatro Nuovo Roma".
Luca Garda	Search for art or science exhibitions in the whole region with available tickets.	Returns all the exhibits about science and art that will be held in the whole Trentino region. The list must show only the events with available tickets.
Luca Garda	Look for talks and conferences that will occur near his house and that are scheduled for the next three days.	Returns all the conferences and talks organized in the next three days. These must take place near Luca's house.
Luca Garda	Search science exhibitions and talks with a specific topic, that perform discounts for groups.	Returns all the events like science exhibitions or talks filtered according to the chosen topic. These events also have to have a discount for groups.
Luca Garda	Get notified whenever a new painting exhibit is organized in the province of Trento.	Sends a notification to the user whenever there is a new set up of an art gallery searching in the whole province of Trento
Giuseppe Bianchi	Search for events in Trento which allow animals.	Lists all events which allow animals, like outdoor events.
Giuseppe Bianchi	Search for movies for children during the weekend in Trento.	Lists all movies tagged as "for children" in all the cinemas in Trento.
Giuseppe Bianchi	Search for science-related events for children.	Lists all the events related to science in the whole region which are targeting children.
Giuseppe Bianchi	Search for parking near the Muse.	If available, returns all the parking near the museum.

1.2. Personas definition

Giacomo Rossi is a 74-year-old retired music teacher. He loves music since he was a little child: he can play a lot of different instruments, including the piano and the cello. He likes every kind of music, but he has a special preference for classical music and opera: his favourite artists are Bach and Puccini.

Now that he has retired, his day is not so busy. For this reason, he usually spends a lot of his time going to music concerts with his wife. He prefers concerts where they play Bach's and Puccini's music; on the other hand, his wife prefers those where Tchaikovsky's music is played. Furthermore, Giacomo is very proud of his daughter: her name is Sara and she has become a famous concert pianist. So very often he and his wife go to see her when she performs in a concert in Trentino.

The proposed application will be helpful to him to search for music concerts in Trentino. He would like the organizer of the concert to add more details to them. By doing so, it would make easier for him to discover, together with the date and time, which kind of music they play, of which artist and who are the performers. In this way, he could arrange with his wife if they have to go to a Bach's, Puccini's or Tchaikovsky's music concert or alternatively go to see a performance of their daughter.

Sara Ferrari is a 19 years old university student. She is from Verona, but she recently moved to Trento to study Economics at the University of Trento. She loves what she studies, moreover she is used to following meetings about contemporary political and economic topics. Meanwhile, during the weekend she sometimes entertains herself by going to the cinema and watching some drama.

Since it is her first year in Trento, she doesn't know anyone yet, and that makes it difficult for her to discover interesting activities that take place in the city. The problem is particularly relevant to her when she has to discover the kinds of meetings she is much used to follow and cannot do without.

She is interested in a platform that can be her main source of information about events taking place in Trento, more precisely about educational ones. It is of fundamental importance for her to be able to precisely find and follow meetings that are of political or economic pertinence as it is valuable for her to keep updated about actuality. Lastly, she would also like to occasionally search for particular categories of films taking place on the weekend, whose location is as close as possible to her. Implementing such a system would turn her first year at Trento way more easy and enjoyable from the first week.

Luca Garda is a 28 years old employer who has little to no time during the week to go to an event. Both his friends and he are very passionate about science and art and love to go to various exhibitions, as well as conferences and talks. The fact that all of them works in different places makes hard to organize a meeting. If we add to that the

lack of an organized list of exhibit the problem gets even more complicated.

With his friends, he likes to explore different topics one at a time. For example, he studies a particular art current before going to the corresponding painting gallery. Again, he could follow some talks on the same argument in a brief period to get a better understanding of the whole. Of course, this mindset requires a lot of organizations and advance. The time spent organizing augments even more by the fact that he plans to go to the speeches and the shows with its friends.

The existence of a list of all the events in the nearby would make Luca and its clique's life a lot easier. This application, which integrates over all the different sites they otherwise have to query manually, would be enough to make the organization faster and simpler. Moreover, the ability to search by topic and by date would require from the user little to no time to select the appropriate theme of the month.

Giuseppe Bianchi is a 40-year-old father. He has two very active and little children: Marco and Jacopo. He lives in Vicenza but during weekends always comes to Trentino with the whole family. Throughout the week, he and his wife Erika usually have no free time to spend with the little ones. Because of that, they would like to organize the weekends smartly.

Both him and Erika know very well the restaurants and a lot of relaxing places in Trentino. They used to travel with their Ford minivan, which is suitable for a family trip of one or two days, especially during the weekends. They also have two large dogs that they bring always with them.

Even if they know a lot about Trentino, they are not from the place: it is hard for them to discover events. Also, the fact that they have two little children and two dogs makes it harder for them to find activities for the whole family.

Having a list of all the facilities which organize events for families would certainly make Giuseppe's life easier. He is always looking for exhibits to which take his children so that they could learn something new. Moreover, if there are local festivals or events in parks he'd like to go with his wife, their children and the dogs.

2. Queries Description

Persona	Competency question	Generalized query
Giacomo Rossi	Is there any concert whose musical genre is classic taking place tonight at “Società Filarmonica Trento”?	Search for musicEvents such that (musicEvent/genre = “Classical” and event/facility/name = “Società Filarmonica Trento”).
Giacomo Rossi	Which is the closest in time concert in Trentino where Sara Rossi is the performer?	Search for musicEvents such that (musicEvent/performer = “Sara Rossi”) and order them by event/dateTime/startDate.
Giacomo Rossi	Who is the musical group that is performing tonight at “Auditorium” in Trento?	Return the musicEvent/performer of the next event with (event/facility/name = “Trento Auditorium”).
Giacomo Rossi	List the timetables of Tchaikovsky’s music concert taking place next weekend nearby.	Return the event/facility/timetables for every facility which hosts a musicEvent such that (musicEvent/musicPlaylist/creator = “Tchaikovsky”) and that (event/dateTime/startDate in nextWeek)
Giacomo Rossi	At which time “Teatro Sanbapolis” in Trento will open tomorrow and what is its address?	Return facility/geoCoordinates and facility/timetables for facilities such that (name = “Teatro Sanbapolis”).
Sara Ferrari	Which of the next 20 scheduled events on the city of Trento are talks or meetings?	For the next 20 scheduled events with (event/geoCoordinates/addressLocality = “Trento”), return only those which are talkEvents.
Sara Ferrari	What is the price for “MUSE a tutto libro”?	Return event/price of events such that (event/name = “MUSE a tutto libro”).
Sara Ferrari	What are the films scheduled in	Return the list of screeningEvent

	Trento the next week that are not of comedy or horror genre?	such that (screeningEvent/movie/genre != (“Comedy” AND “Horror”)).
Sara Ferrari	Which cinemas between Rovereto and Trento has the cheapest ticket price for adults?	Return the list of facility which holds at least one screeningEvent ordered by event/price.
Sara Ferrari	What are the events nearby “Piazza di Fiera 4, Trento” scheduled in the next week?	Search for events such that (event/geoCoordinates near “Piazza di Fiera 4, Trento” and event/dateTime/startTime in nextWeek)
Luca Garda	What are the newly added events related to art or science in the area?	Search for the latest added visualArtEvents or scienceEvent such that (event/geoCoordinate is nearby)
Luca Garda	What are the events organized nearby that have discount for groups?	Search for events such that (event/geoCoordinates near me and event/discount/groups = true)
Luca Garda	Are there any art galleries about symbolism?	Return the list of visualArtEvents such that (visualArtEvent/artMovement = “Symbolism”)
Luca Garda	When will an art galleries of Depero take place in the region?	Search for every visualArtEvent such that at least one of the pieces has (visualArtEvent/visualArtwork/creator = “F. Depero”)
Luca Garda	Is the ticket available for the next organized TED event in Trento?	Return the talkEvents such that (talkEvent/isTicketAvailable = true)
Giuseppe Bianchi	What are the indoor events in the city of Trento where animals are allowed?	Select all the events such that (event/facility/animalsAllowed = true AND event/facility/isIndoor = true).

Giuseppe Bianchi	During next weekend are there any scheduled events for families in Valsugana?	Search for all the events such that (events/audience/families = true)
Giuseppe Bianchi	Does MUSE offer some activities for children?	Return all the events such that (event/facility/name = "Muse" AND event/audience/children = true).
Giuseppe Bianchi	Which are the outdoor activities in the Trento area scheduled for the next weekend?	Search for all the events such that (event/facility/isIndoor = false AND event/dateTime/startDate in nextWeekend)
Giuseppe Bianchi	Are there cinemas that offers family discounts?	Search for all the facility hosting at least one screeningEvent such that (event/discount/groups = true)

2.1. Query Patterns

In the above we reported only a fraction of the 120 competency queries we wrote together with the generalized queries. From the complete list we have extracted the most relevant patterns and manually generated the intersection between the attributes and the types needed to answer the questions, as shown in the following table.

ID	Types labels	Attributes labels
1	['Location']	city, address, region, geoCoordinates
2	['Event'], ['Art Event'], ['Concert'], ['Movie'], ['Science Exhibit'], ['Talk'], ['Theatre spectacle']	name, topic, description, price, duration, startDate, endDate, startTime, endTime
3	['Facility']	name, website, email, phone, timetables, hasParking, animalsAllowed, indoor
4	['Art Event']	creator, artMovement, visualArtwork
5	['Concert']	genre, performer, artist, songs, musicPlaylist, creator

6	[‘Talk’]	topic, guests, isTicketAvailable
7	[‘Movie’]	originalName, genre, salaNumber, launchDate
8	[‘Discount’]	groups
9	[‘Audience’]	families

Starting from those patterns, we designed the entities required, taking Schema.org as reference for enhanced accuracy. After working on the selection of the necessary and most relevant entities, we designed the relationships between them. Moreover, we associated to each entity a set of attributes derived from the extracted patterns and added any other necessary attribute in order to enrich and enhance our model and make it as complete as possible.

3. Dataset(s) extraction/generation and Dataset(s) description

By taking the “Cultural Events” as the keyword the process of searching for datasets started. After a quick lookup on the Internet, we could not find pre-existing datasets on the topic nor any other form of structured data. There are a few datasets about facilities but they are too generic and not related to events, therefore using the Google Places API was the best solution.

The only way to find information about events was looking directly on the specific websites and, for this reason, the only method of extraction used was web scraping.

This turned out to be quite a challenging task, because of the necessity to develop a different python script for each website. To facilitate this task we took advantage of a free web scraping tool called ParseHub. ParseHub provides the user with an easy to navigate interface that grants the possibility to scrape the information just by clicking on the wanted element on the website. For some websites, the weird structure of the HTML and the division of the information on multiple pages were too difficult to overcome using the standard tools. We resorted to XPath queries to select specific elements on the page, as this can be done directly within ParseHub. The good thing about ParseHub is that it creates projects that can be reapplied when there are new events on the websites.

At the end of this first process, we end up with different JSON files which cover the majority of the attributes needed. At this point was necessary to collect information about *facilities* and *creative works*.

To get data for both geocoordinates and facilities, we decided to use the Google APIs. The first API is Google Places, which provides a search function to retrieve possible points of interest, based on the location text of the event. The response to this request leads to a decision based on the number of results. Often there are no facilities (i.e. "places") on the event location, so the API results are empty. In this case, the request is reapplied to the Google GeoCoding API. This provides the geocoordinates for the event and, using our datasets, never failed to find results for a given city, address or

location name. When the Places API returns one or more results, we save the first one and redirect the request to the Places Details API. This is essential to get detailed information such as rating and opening hours.


```
"Museo Geologico delle Dolomiti di Predazzo": {
    "formatted_address": "Piazza SS. Filippo e Giacomo, 2, 38037 Predazzo TN",
    "formatted_phone_number": "0462 500366",
    "international_phone_number": "+39 0462 500366",
    "name": "Geological Museum of the Dolomites Predazzo",
    "opening_hours": {
        "open_now": true,
        "periods": [ ···
        ],
        "weekday_text": [
            "Monday: Closed",
            "Tuesday: 10:00 AM \u2013 12:30 PM, 4:00 \u2013 7:00 PM",
            "Wednesday: 10:00 AM \u2013 12:30 PM, 4:00 \u2013 7:00 PM",
            "Thursday: 10:00 AM \u2013 12:30 PM, 4:00 \u2013 7:00 PM",
            "Friday: 10:00 AM \u2013 12:30 PM, 4:00 \u2013 7:00 PM",
            "Saturday: 10:00 AM \u2013 12:30 PM, 4:00 \u2013 7:00 PM",
            "Sunday: 10:00 AM \u2013 12:30 PM, 4:00 \u2013 7:00 PM"
        ]
    },
    "rating": 4.4,
    "types": [
        "museum",
        "tourist_attraction",
        "point_of_interest",
        "establishment"
    ],
    "user_ratings_total": 237,
    "vicinity": "Piazza SS. Filippo e Giacomo, 2, Predazzo",
    "website": "http://www2.muse.it/museogeologico/"
}
```


The next step is to identify which results are to be considered as facilities. Most of them are simply routes or squares and we needed some criteria to separate facilities from mere geocoordinates. The simplest way was to filter results by their type, so that only those containing "point_of_interest" were considered facilities. This turned out to be a good filter as only 60 out of 250 locations have this type, probably because most of them are city names. Finally, extracting the information in python was pretty straightforward and we decided to use the searched text as URI for the geocoordinate's entities to simplify the linking process with the events output.

Creative works are the entities associated with every event related to mind products. For example, while "Joker" can have more screening on different dates and times, there is only one entity that describes the movie per se. For this reason, it was necessary to collect some information often, if not always, not provided on the websites like the *original name*, the *creator*, the *date of creation* and a URL that could identify this entity.

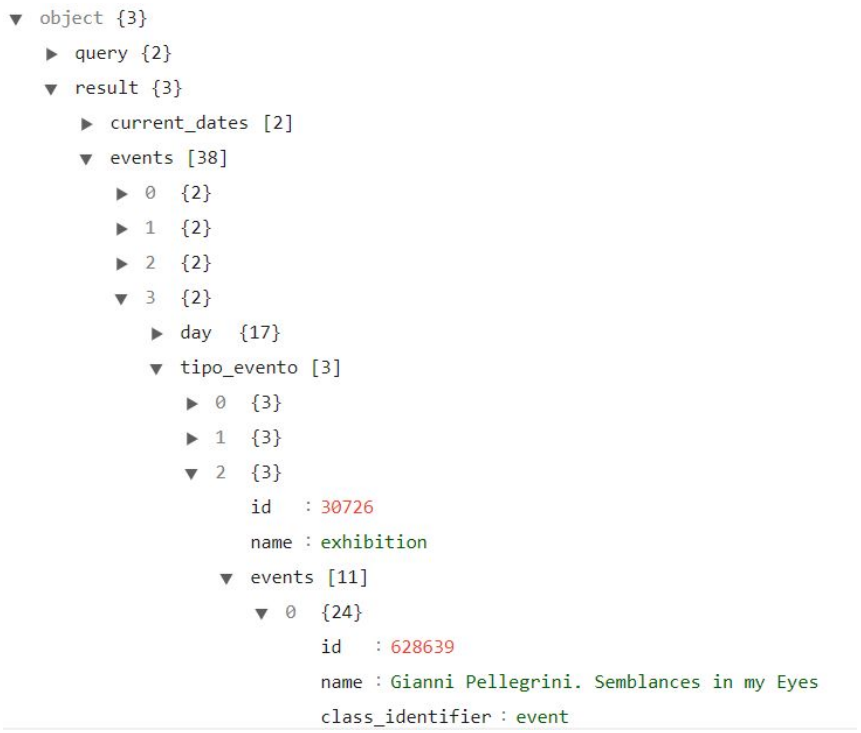
To gather these fields we tried to make use of Yago as suggested in class, but unfortunately the database has not been updated since 2017 and for this reason it was of no use. For example, the majority of our events related to creative works are movies, and those are surely too recent to be found on Yago. Luckily Google built an extensive knowledge graph to gather information from Wikipedia and a handful of other sources. When searching, for example, a movie on the browser, google show you what's called a *knowledge panel*, that gathers information about its creation, the actors that worked on it and other data, giving the user the possibility to navigate the graph moving between linked resources.

To access this information, Google provides a web API, called Google Knowledge Graph API, that lets the user access the graph with a simple GET request. This API is particularly useful because gives you the possibility to set some essential parameters like for example the language, that was a necessity since most of our data is scraped from italian websites. The response from the call is an object containing the list of every node of the graph identified by the query. From this list we were able to identify the resource we wanted thanks to the tags associated with it. For instance, with *Joker* as the input we consider only the entries with the tag "movie" to distinguish it from, for example, the comic character.

However, even Google's tool had its limitations. Some of the data held in the google knowledge graph comes from private sources and for this reason the company decided to limit the amount of information freely accessible with this API. We could not access any useful attributes of the creative work directly but luckily it was possible to retrieve the Wikipedia url for every resource. After obtaining the Url with the API it was necessary to scrape the web page to gather all the information needed.

4. Dataset(s) cleaning, merging and analysis description

Each JSON file had a different structure that was not clean and tabular. Also, each dataset had some missing values which were needed. As an example, the structure of events in Cultura is shown below:



In addition, each website had used its specific format for date and time that some websites were human-written with no fixed format.

4.1. Cleaning procedure

4.1.1. Filling Missing Values

Related to the informal model each dataset had some missing values. Some of them could be filled by a more deep scraping. However, many missing values could just be found in the description of the event which was an unstructured text. Furthermore, some of them were fixed values, such as the "Location Name" in "Cinema Rovereto" dataset, which was a fixed location.

	name	price	description	website	duration	language	ticket	location	category	audience	date	time
Cinema Rovereto												
cineworld Trento												
Cultura												
Mart												

Muse												
trento Today												
visit Trentino												

The fields with Green background were filled using more scraping. The Yellow ones using extracting from the unstructured text or fixed values, and the White fields could not be filled at all. Therefore, as the "Suitable for", which is related to "Audience" property in the informal model, was empty in six out of seven datasets, the decision was to ignore that.

The scripts of this stage are available here.

4.1.2. Uniforming date and time

Each website uses its specific format for representing the date and time of the events. Some of them are well-formatted, like Cultura with DD-MM-YY format. On the contrary, for Muse, the date format was too complex. Following lines contains some examples of date values for Muse:

(2 e 3 novembre 2019), (3,10,17,24 nov 2019), (8 e 22 ott, 5 e 19 nov, 3 dic 2019), (20 marzo, 17 aprile, 15 maggio, 18 settembre, 16 ottobre e 13 novembre 2019)

According to these different formats, a python script is written for each dataset separately for uniforming the time and date. The scripts are available here.

4.1.3. Uniforming Category

According to the informal model, each event should be assigned to one of the event categories. For example, movies should be assigned to "Screening Events". Although this could be done by using the "Category" and "Sub Category" properties that were available on the datasets, these properties were uniform. Each website has different event types with different naming for them. Therefore, for each dataset, a dictionary is created to map the event category name which is given on the website to the desired set of categories. This is handled by python scripts already described.

```
dictionary = {
    "Cultural exhibitions and events,"            : "GeneralEvent",
    "Cultural exhibitions and events,Guided tour,": "GeneralEvent",
    "Dance,Opera and modern ballet,"              : "TheatreEvent",
    "Drama,"                                      : "TheatreEvent",
    "Meetings and conferences,"                   : "TalkEvent",
    "Meetings and conferences,Workshop,"          : "ScienceEvent",
    "Music,"                                      : "MusicEvent",
    "Music,Classical music concert,"              : "MusicEvent",
    "Music,Jazz concert,"                         : "MusicEvent",
    "exhibition,"                                 : "GeneralEvent",
    "exhibition,Art exhibition,"                  : "VisualArtsEvent",
    "exhibition,Photographic exhibition,"         : "VisualArtsEvent",
}
```

4.2. Merging procedure

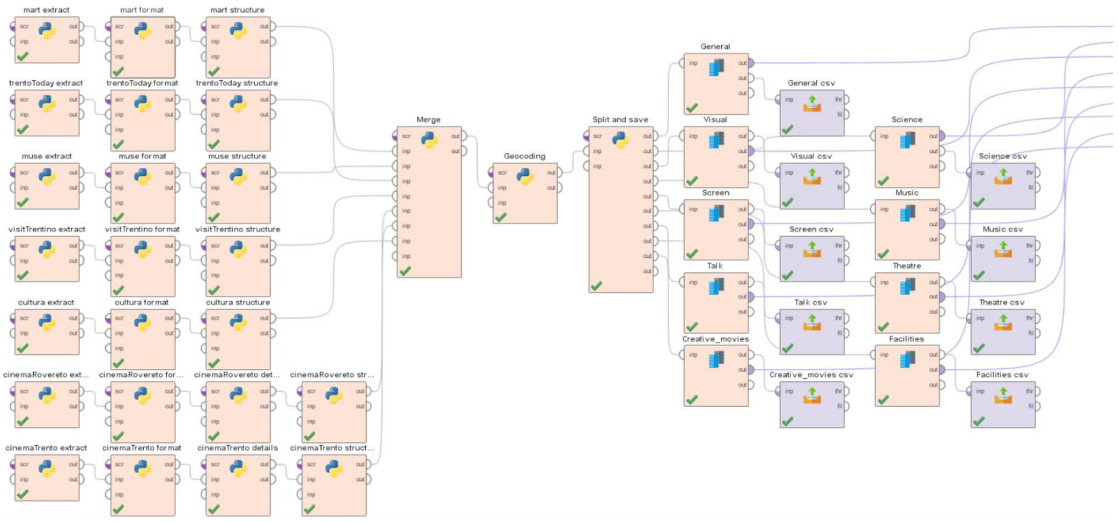
By appending the movie details to the movie events, the process of uniforming the structure is started. As mentioned before, each dataset had its JSON structure. Therefore, a tabular structure defined by considering the properties that were needed by the model. Datasets were converted to the defined tabular structure, each by a separate python script. The scripts are available on the repository.

Having all the sources as a uniform JSON structure allowed us to merge easily in python. We had to be careful with geo coordinates data in order to unify all the events of the same place. This was done using the searched location text. This method is not perfect, since a small difference in the location text would lead to different geocoordinates even if the google API returns the same results. To fix this, another merging step would be necessary, but we didn't have time to implement that too.

Merging all the creative works was also easy because we had only results from the movie ones. Having multiple creative work types would make the process a bit complicated, especially for the amount of outputs. The basic merging strategy was using the creative work's wikipedia url. Given that no wikipedia url implies no creative work data, all of the instances have a url. The merging took place in karma since we used the url as a component of the works URIs.

The final step is to split the dataset into subsets that are useful in the final integration step. The output is seven CSV files for seven different event categories, one CSV file for movies' detail information, and another CSV file for facilities' data.

4.3. Analysis



The final output includes separate datasets for each category. To have some analysis on data, General Events are considered.

GEN_name	GEN_price	GEN_descript...	GEN_website	.	.	.	GEN_geoURI	.	DATETIME_st...	DATETIME_endDate	DATETIME_startTime	DATETIME_endTime
Category	Category	Category	Category				Category		Category	Category	Category	Category
Mart Membersh...	free	Appuntamento ...	mart.trento.it/ag...	..	.	..	?	..	12-11-19	12-11-19	?	?
Mente e cervell...	free	Incontro della s...	mart.trento.it/ag...	..	.	..	http://www.sem...	..	14-11-19	14-11-19	20:30	22:00
Mart Blabla De...	Costo: € 5	Parlare tedesc...	mart.trento.it/ag...	..	.	..	http://www.sem...	..	15-11-19	15-11-19	18:00	19:30
Sarajevo - La st...	?	?	?	..	.	..	?	..	?	?	?	?
Isadora Duncan	Costo: € 2	Visita guidata a...	mart.trento.it/ag...	..	.	..	http://www.sem...	..	17-11-19	17-11-19	15:00	16:00
Trasformare il ...	free	Incontro della s...	mart.trento.it/ag...	..	.	..	http://www.sem...	..	21-11-19	21-11-19	20:30	22:00
Isadora Duncan	Costo: € 2	Visita guidata a...	mart.trento.it/ag...	..	.	..	http://www.sem...	..	24-11-19	24-11-19	15:00	16:00

As it was expected, there is no information available for “duration”, “language”, and “isTicketAvailable”. These columns are the collapsed ones in the picture above. The cost of the event is missing in 61% of entries and this percentage for the end time of the event is around 75. However, the start date and end date of events are available in more than 99% of entries.

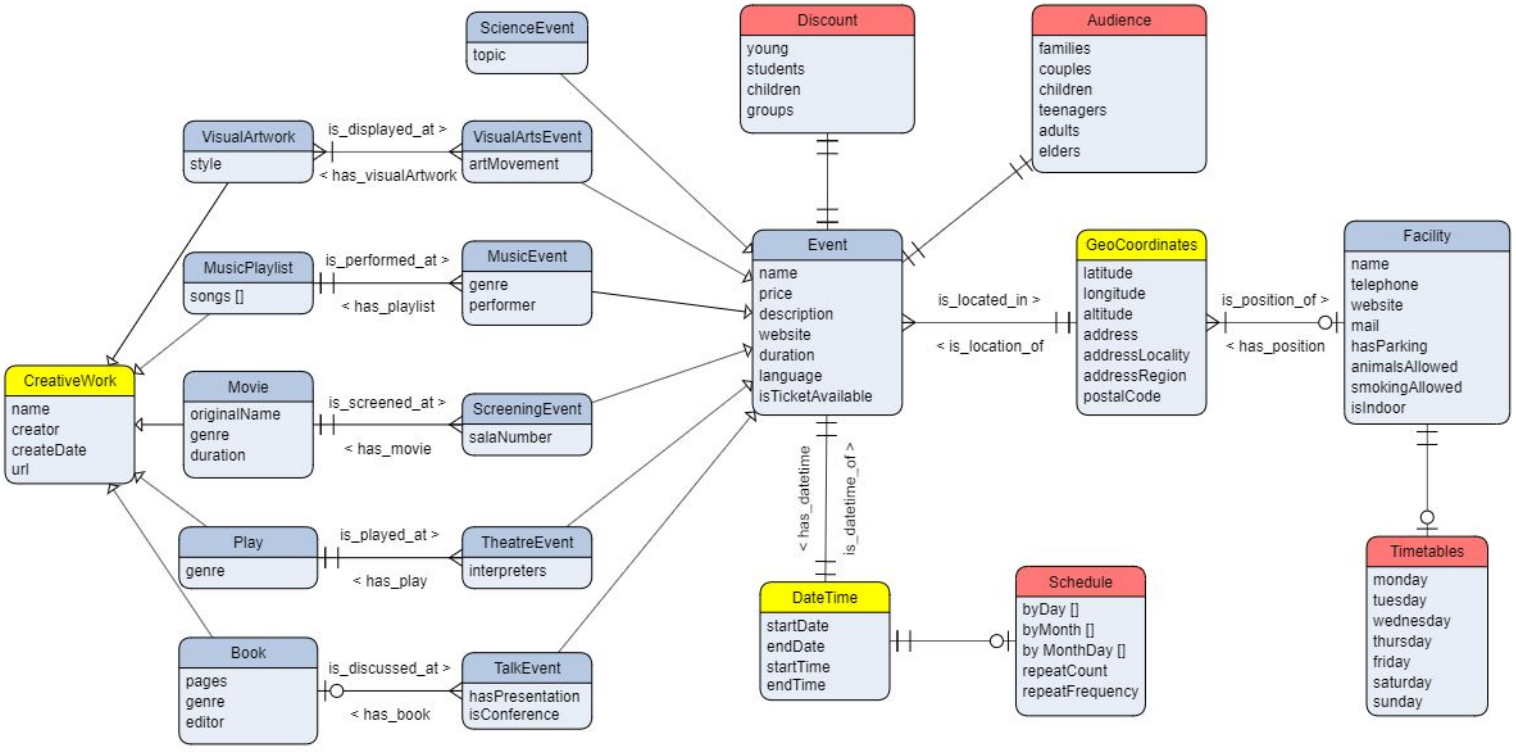
As described, there are totally nine output files that are available on the repository.

5. Model design

Starting from the complete list of 120 competency queries together with the extracted patterns, we were able to define a good number of attributes. From them, we did identify three principal entities: facility, event and time. We were not satisfied with the number of attributes for the event entity, so we resort to looking for famous solutions. We looked at Schema.org for splitting the event into subclasses, but

unfortunately, the solution proposed was not of our pleasure: that's why we decided to split the events into subclassed defined by us.

The separation was meant to augment the degree of organization and the strictness of the relationships between objects. Also, by defining a finite set of subclasses we could specify the boundaries of our definition of "event". We do not consider every event in our solution, but rather only the cultural ones. After a series of iterations, we decided to consider six categories: science, visual art, music, screening, theatre and talk events.



What we did get from Schema.org was the idea of the "creative work" object. From that, we worked on its expansion so that it could be integrated with the different events we have. Also, the structure of DateTime was influenced by Schema.org, as well as by the "ical" format.

Core entities Our core entities are Event and Facility. Other than those, we can consider the specialized subclasses of CreativeWork and Event as core.

Event: it represents an activity at a particular time and in a particular place. It has six different children. In our model, different from the Schema.org one, we consider the same structure for both recurring and occasional events. It is linked to a GeoCoordinate and to a DateTime object. The attributes are:

- **name**: the name of the event (dt: string);
- **price**: the highest price of the event (consider only one price even when more are available; discounts are modelled with another attribute) (dt: float);

- **description**: define the content of the event; contains a lot of unstructured information (dt: string);
- **website**: if available, contains the url to the website of the event (dt: string);
- **duration**: the length of the event; when available represents how much time the event lasts (dt: string)
- **language**: the language of the event (dt: string);
- **isTicketAvailable**: when ticket is needed, it indicates whether or not it is still possible to buy one (dt: boolean).

More specific types of Event are the following:

- **ScienceEvent**: it represents any type of science event. The attributes are the ones taken from Event and:
  - **topic**: define the principal argument of the exhibit (dt: string).
- **VisualArtEvent**: represents any type of visual art event. It can be linked to one or more VisualArtwork. The attributes are the ones taken from Event and:
  - **artMovement**: define the movement of the exhibit (dt: string).
- **MusicEvent**: it represents any type of music event. It is linked to one MusicPlaylist. The attributes are the ones taken from Event and:
  - **genre**: represents the type of music it will be performed (dt: string);
  - **performer**: the person or group which plays (dt: string).
- **ScreeningEvent**: it represents any type of screening event. It is linked to a Movie. The attributes are the ones taken from Event and:
  - **salaNumber**: the identifier of the sala in which the movie will be displayed (dt: string).
- **TheatreEvent**: it represents any type of theatre event. It is linked to a Play. The attributes are the ones taken from Event and:
  - **interpreters**: the group which perform (dt: string).
- **TalkEvent**: it represents any type of talk event, such as talks or conferences. It can be linked to one book. The attributes are the ones taken from Event and:
  - **hasPresentation**: define if the event includes some kind of projected presentation (dt: boolean);
  - **isConference**: define if the event is a conference (so it takes the form of a debate in which everyone can talk) (dt: boolean).

Facility: it represents a physical facility in which one or more events take place. It is linked to one or more GeoCoordinate. Its attributes are:

- **name**: the name of the facility (dt: string);
- **telephone**: the phone number of the facility (dt: string);
- **website**: the url to the website of the facility (dt: string);
- **mail**: the mail address of the facility (dt: string);
- **hasParking**: whether or not the facility includes parking (dt: boolean);
- **animalsAllowed**: whether or not the facility allows animals (dt: boolean);
- **smokingAllowed**: whether or not it is allowed to smoke in the facility (dt: boolean);
- **isIndoor**: whether or not the facility include only a close space or not (dt: boolean).

Common entities Our common entities are taken from Schema.org, and are the CreativeWork, the DateTime and the GeoCoordinates.

CreativeWork: it represents any kind of product of the mind. In our model, the structure is split into different children to guarantee more robust constraints between events and works. The attributes are:

- **name**: the name of the product of the mind (dt: string);
- **creator**: the creator of the work (dt: string);
- **createDate**: the date of creation of the work (dt: dateTime);
- **url**: the url to the Wikipedia reference (dt: string).

More specific type of CreativeWork are the following:

- VisualArtwork: it represents any type of visual artwork. Its attributes are the ones taken from CreativeWork and:
  - **style**: the style of the artwork (dt: string).
- MusicPlaylist: it represents any type of music playlist. Its attributes are the ones taken from CreativeWork and:
  - **songs**: list of songs that compose the playlist (dt: string).
- Movie: it represents a movie. Its attributes are the ones taken from CreativeWork and:
  - **originalName**: the real name of the movie (not translated in italian) (dt: string);
  - **genre**: the genre of the movie (dt: string);
  - **duration**: the length of the movie in time (dt: string).
- Play: it represents a theatre play. Its attributes are the ones taken from CreativeWork and:
  - **genre**: the genre of the spectacle (dt: string).
- Book: it represents a book. Its attributes are the ones taken from CreativeWork and:
  - **pages**: the number of pages of the book (dt: integer);
  - **genre**: the genre of the book (dt: string);
  - **editor**: the editor which review the book (dt: string).

GeoCoordinates: it represents a point in space. It is linked to one or more event and to at most one facility. This intermediate entity between Event and Facility makes it possible to have events in locations which are not facilities (for example when we have a festival in Trento), as well as Facility with more than one GeoCoordinates. The attributes are:

- **latitude**: the latitude of the geocoordinates (dt: float);
- **longitude**: the longitude of the geocoordinates (dt: float);
- **altitude**: the altitude of the geocoordinates (dt: float);
- **address**: the address of the location (dt: string)
- **addressLocality**: the locality of the geocoordinates (e.g.: Trento) (dt: string);
- **addressRegion**: the region of the geocoordinates (e.g.: Trentino) (dt: string);
- **postalCode**: the postal code of the geocoordinates (dt: string).

DateTime: it represents the time in which an event occurs. The periodicity of an event is modelled as an external auxiliary entity, but is extracted from the core entity taken from Schema.org. The attributes are:

- **startDate**: the date in which the event start (dt: dateTime);
- **endDate**: the date in which the event ends (dt: dateTime);
- **startTime**: the time in which the event start (dt: dateTime);
- **endTime**: the time in which the event ends (dt: dateTime).

Auxiliary entities Our auxiliary entities are Discount, Audience, Schedule and Timetables. “Discount” includes a list of target prone to receive a reduction to events, while “Audience” represents the target of the event. “Schedule” includes all the attributes needed to model the recurrence of an event. “Timetables” represents the days and the hours of the week in which a facility is open.

6. Model formalization

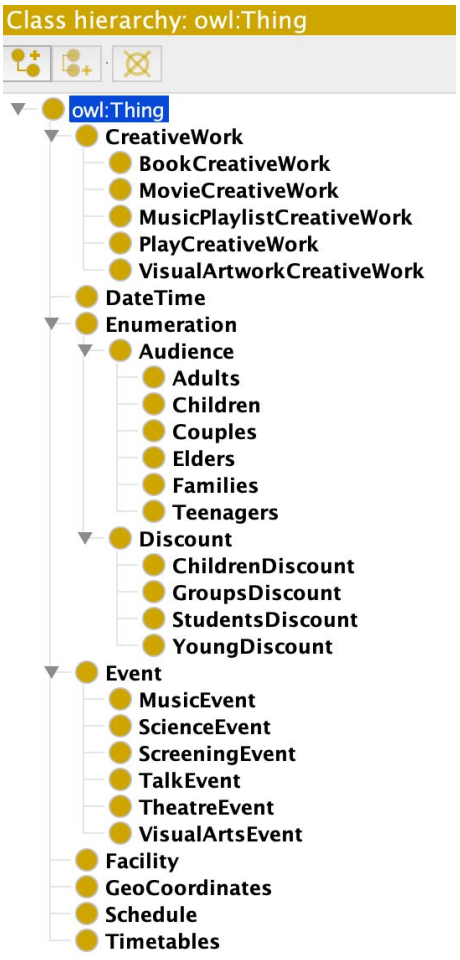
After designing the informal model, we proceeded with the creation of the formal version using Protégé. Starting from the EER diagram, we worked on converting every entity, attribute and relationship into classes, data properties and object properties.

Classes
All of the classes in the informal model were modelled to the formal. The structure of them was kept for almost everyone; the only exceptions are two of the four auxiliary entities.

As we have noticed during the informal phase, both the "audience" and the "discount" can take a finite set of values. Moreover, the two seems to be more of a collection than a single-valued attribute. For example, it's easy to imagine an event in which the discount is provided to children and students. The sample can be easily extended to cover also the case of "audience". That's the reason why we decided to model them as classes with values as children. In this way, it is possible to link different subclasses of both "audience" and "discount" to the same event, using an object property.

In the case of "audience", the subclasses are "young", "student", "child" and "group"; in the case of "discount", the values are "families", "couples", "children", "teenagers", "adults", "elders". The values written above are easily extractable from the informal model and are rewritten just for clarity.

For the above classes, the last piece to notice about the changes in the model are about their father. It can be seen from the picture below that both "audience" and "discount" are subclasses of Enumeration. The choice was done to simplify for future readers the understanding of the nature of these classes.



Data Properties

The number of data properties extracted from the informal model is huge. The reason behind its dimension is the fact that we need a lot of different information to answer probable queries. Most of the data properties maintain the name and the datatype defined in the informal model. The exceptions are the attributes that have the same name. To make the Data Properties more readable and maintainable we grouped them together in a hierarchy of properties relative to the same class.

It is easy to notice that the attribute "name" is used for Event, Facility and CreativeWork. For this particular reason, we have created three different data properties. For an event, we have the attribute "eventName", for a facility there is the "facilityName", while for a product of the mind, we have the "creativeWorkName".

For what concerns the datatypes, most of them are strings; this fact can be seen already in the informal model. The decision was made because our dataset was created starting from unstructured data retrieved from the web. We couldn't anticipate the complexity of the integration phase, so we resort to handle mostly unstructured attributes. For example, a discrete number of data properties can take values from a finite set, but we didn't have the resources to define every one of them.

Another issue was about the cases in which an event had more values for a single property. As an example, we retrieved events which had more than a single value for

the attribute "price". We couldn't handle every little exception, so we opted for generalization. In this specific instance, we decided to save only the highest price, and use the class "Discount" to model possible reductions. As another case, we can think of the attribute "language": 99% of the events should take only one value (and is always the same), but the exception may have two languages.

To conclude this section, we talk about the modelling of the data properties of timetables. Timetables have seven data properties, one for each day of the week. These data properties are represented as strings, more precisely as arrays of strings (we created a Datatype for this purpose). We chose to assign to each day of the week an array of strings containing the opening hours of a specific facility. The reason behind this choice can be linked back to the previous paragraphs: we deal with unstructured data.

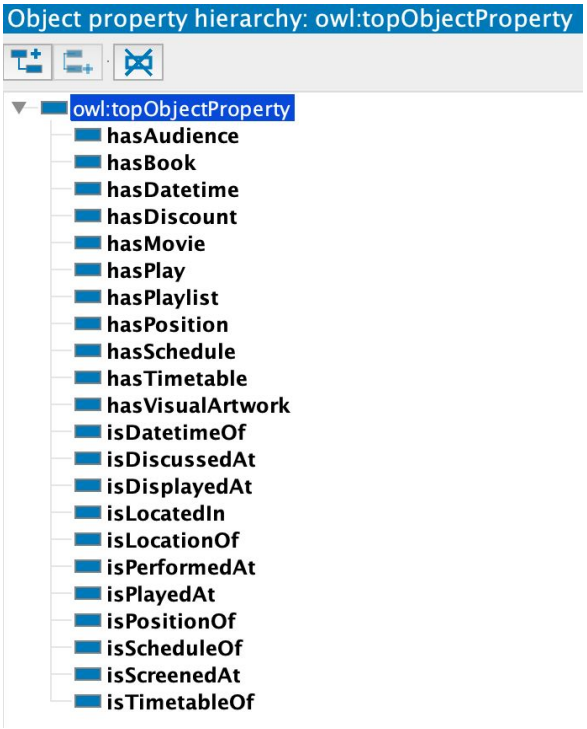
Object properties

For what concerns the object properties, they reflect perfectly the arrows inside the informal model. The only difference is the fact that the number is reduced. This change is not, in fact, a choice, but more of a translation issue. Indeed, in the formal

model, the "ISA" relationships are converted into subclasses instead of object attributes.

For each arrow in the EER diagram, we can define two object properties, one from entity A to entity B, and one in the opposite direction. The two properties are the first the reverse of the other.

In regards to the arity of the relationships, we have defined them inside the subclasses window. Using the keywords "MAX", "MIN", "EXACTLY" we defined relationships between classes in a manner similar to the one used for ER modelling. For example, looking at the informal model, we can see that an event has EXACTLY one geocoordinate and a geocoordinate is the position of MAX one facility. To conclude, from the other point of view, we have that a facility has MIN one geocoordinate.



Formal model evaluation

To evaluate the formal model we used the OOPS! (OntOlogy Pitfall Scanner!) tool in Linked Open Vocabulary. This tool can be useful to understand the level of completeness of the formal model and detect the presence of errors or anomalies in the ontology. OOPS! was run several times on our ontology and it was very helpful to us to find out faults in the designed model, indeed thanks to it we were able to find some mistakes like missing annotations, missing domains or ranges for properties and classes and so on. We kept correcting the ontology as soon as we got a final result consisting of these five warnings:

Results for P04: Creating unconnected ontology elements.	1 case | Minor
Results for P11: Missing domain or range in properties.	18 cases | Important
Results for P13: Inverse relationships not explicitly declared.	2 cases | Minor
Results for P22: Using different naming conventions in the ontology.	ontology* | Minor
Results for P41: No license declared.	ontology* | Important

Three of them are considered minor issues while the remaining two are critical issues.

The unconnected element that is mentioned is only the class “Enumeration”, indeed we modelled it is a class whose purpose in containing only some values as subclasses: for this reason we didn’t connect it to the rest of the ontology deliberately. This does not apply to the subclasses of “Enumeration” (“Audience” and “Discount”) which are not created in isolation and they are linked to other elements in the ontology.

We have also an error related to the missing domain or range in properties that refer to the Data Properties EventProperty, FacilityProperty and so on: these properties serve us just as a way to group the Object Properties that share the same domain and this to achieve more clarity when reading the ontology. So it was not necessary to model their domain and range since they are helpful only to group other Data Properties.

The inverse relationships which are not explicitly declared are the “hasAudience” and “hasDiscount”: these were not modelled deliberately since we were only interested in getting to know the target audience of an event and the discounts related to an event. For this application, we had no interest in finding which “Audience” and “Discount” are connected to an event.

Then, according to the use of different naming conventions, we searched everywhere inside the ontology but unfortunately, we couldn’t find them. It is not an important issue, however, having the same naming convention improves readability and is useful to maintain uniform all the names.

Among all, the missing licence is considered indeed a piece of important missing information. Despite the presence of the dc:licence annotation in our ontology, this problem persists.

6.1. Lexical information upload description

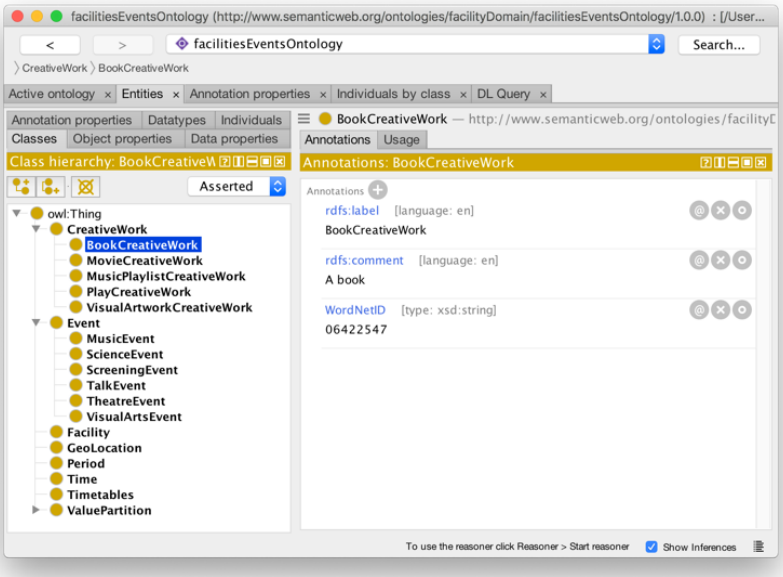
A highly reusable model implies the integration of meta-information that allow others to avoid misunderstandings due to word ambiguity or wrong contextualization. WordNet is a popular resource that helped us to link precious semantic and lexical information to our ontology elements. To accomplish this we took advantage of the web tool that is possible to find at the Princeton University website. The most appropriate and fitting definition found in the WordNet database was empirically selected, then its unique ID has been copied into our ontology as the "WordNetID" annotation using a string format.

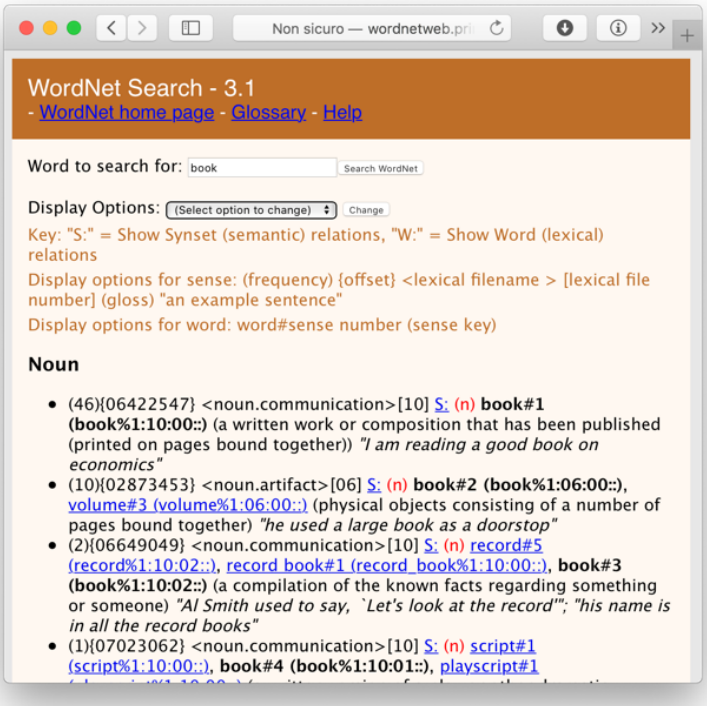
As one can imagine, some of the ontology elements were easy to find in WordNet and had a very satisfying description, though other elements weren't available or weren't consistent with our concept.

Here following are presented some significant cases that can give an overall idea of this task.

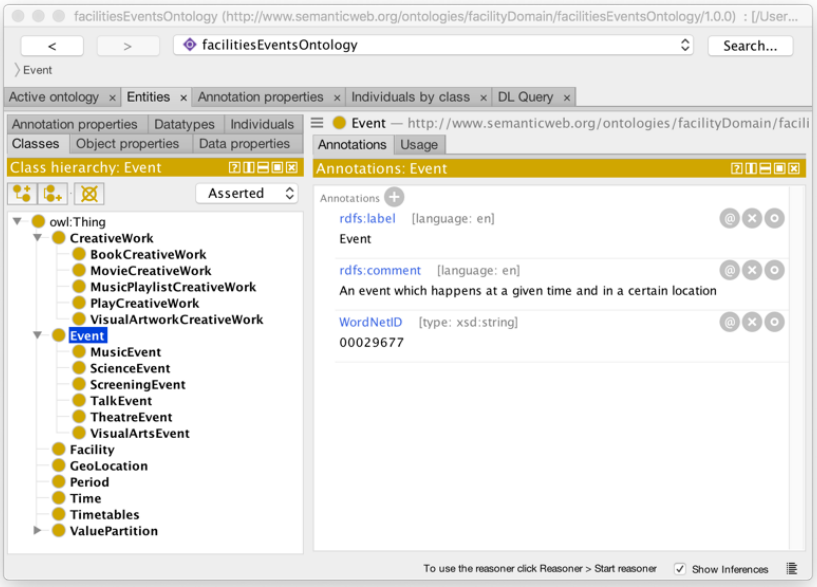
Entities

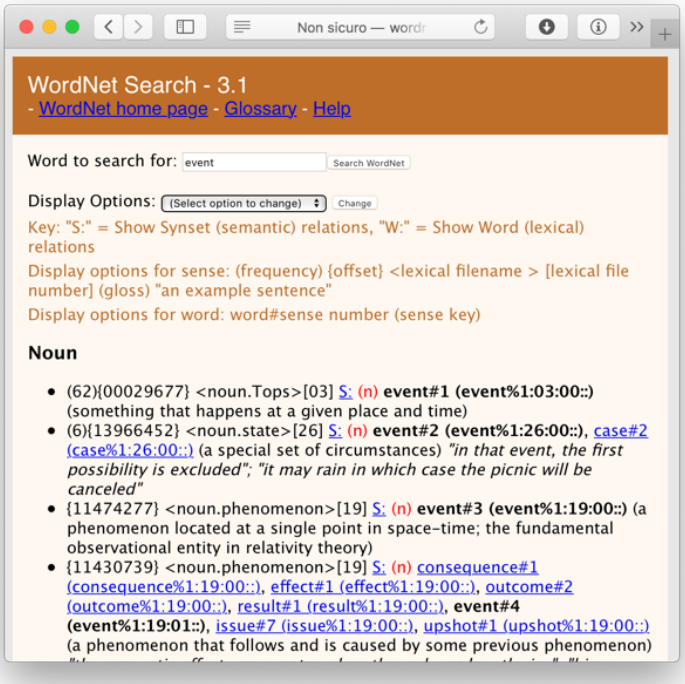
- **Creative Work:** with these two words we intended the most generic kind of creative work including books, movies, music playlists, plays, and visual artworks. This concept has no representation in the WordNet database, hence a WordNetID annotation has not been embedded. On the other side, all its subclasses have a very precise and coherent WordNet representation. Obviously for the searching task into WordNet, has been removed the "CreativeWork" from the entity label, as is possible to understand from the images hereafter.





- **Event:** with this entity name we intended the event in a social sense, as an occurrence created by people for other people like a concert or an art gallery, not a volcano eruption for example, which is always something that happens in a certain set of time and space coordinates. This is partly covered by the WordNet definitions but has a very broad meaning. Although this meaning does not "fit" perfectly, we decided to keep it anyway because it is not misleading for people who might use our ontology. Another important reason that led us to this choice is the absolute lack of entries of the Event subclasses.





Data Properties

The same issues were encountered for the data properties labels, in addition to the lack of concepts represented in WordNet here were missing some more articulated concepts like “byDay”, “byMonth”, “byMonthDay”.

Furthermore, a unique WordNet concept has been assigned to multiple data properties due to its feasibility. This happened especially for “bookGenre”, “movieGenre”, “musicalGenre”, “playGenre” which do not find a perfect WordNet representation but have been using the same one for all of them: the definition of “Genre”.

Finally, a slightly different problem encountered when dealing with properties like "pages". Because of its nature, this concept could not be represented. The word "page" in the WordNet search gives some results. Our property, however, does not express the concept of a page but an amount of those, intended as a number. Hence we preferred not to link any WordNet IDs because it might have led to some misunderstandings, given that the most suitable would have been "amount".

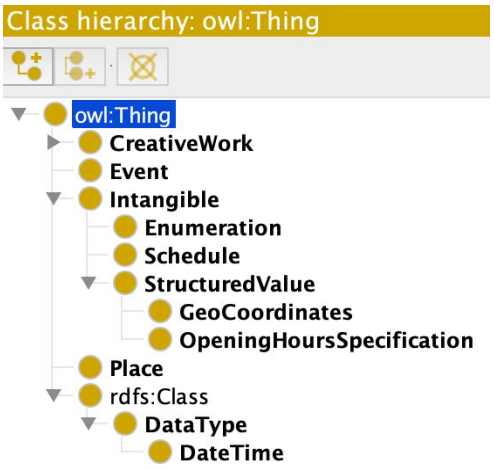
7. Top-level Grounding

In this chapter, we performed a top-level grounding to verify the structure of our model as compared to the CSK ontology obtained from schema.org. The purpose of this task is to adjust our model to a higher abstraction of the world.

Firstly, is important to mention that the knowledge we choose as reference for the grounding, schema.org, is a very broad ontology with a strong focus on usability, and with a relatively small vocabulary. As a consequence of this, we easily found candidates for mapping our classes, but some of those were too broad, of different meanings or different implementations. In the section below we described different considerations about our mapping.

The OWL file provided by schema.org is an incomplete representation of their full ontology, thus many properties have missing information, like domain and range. Fortunately, the lack of information did not hinder our grounding which was limited to the classes. Another general difference with our ontology is that we specified the data types as datatypes, whilst for schema.org everything is a class and there are no data properties.

To build the top-level grounding, we took the Schema.org ontology, knowing that in their structure the relationship between classes is a guideline, differently from our ontology in which we have strict rules. We then extracted a subset of classes from it. The classes extracted are the representation of some of the same concepts that we used to model our domain. The classes that we decided to maintain from Schema.org's ontology are the following:

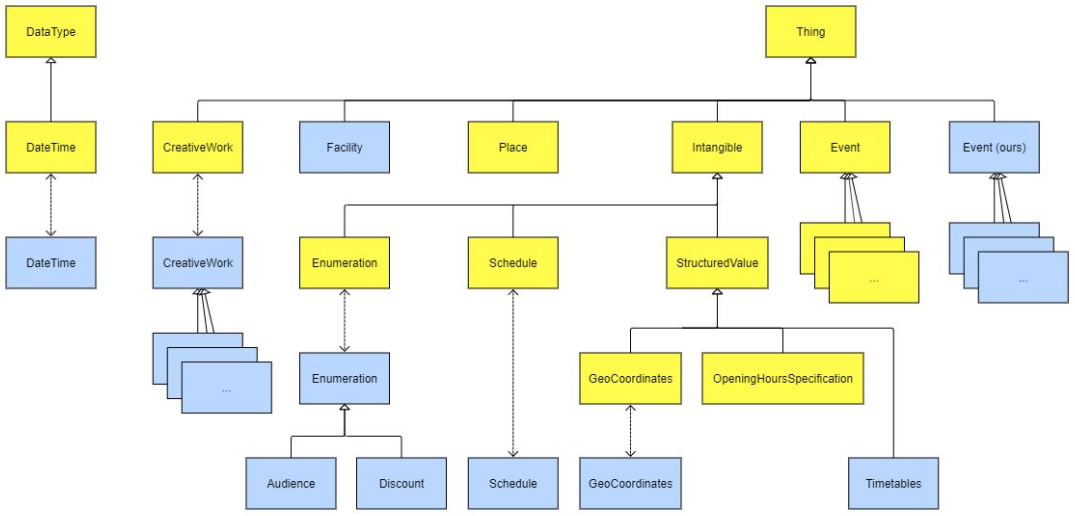


A good mapping between our ontology and schema.org's exists, although there are some points of incongruence both in the structure and in the classes.

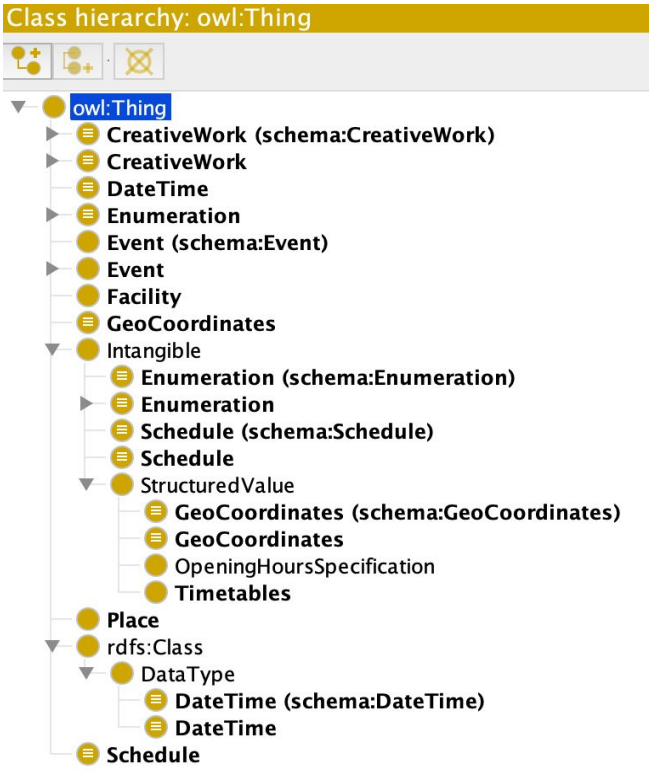
For example, the first thing we can notice is that classes in schema.org's ontology are differently structured: Enumeration and Schedule are not simply a subclass of Thing, as it is in our ontology, they are placed as subclasses of an entity instead, which is called Intangible.

Moreover, it has been made a deeper classification with GeoCoordinates and OpeningHoursSpecification, since they are not simply subclasses of Intangible but they are grouped into a more specific class which is StructuredValues in schema.org. All the other classes belong to the same hierarchy as it is in our ontology: CreativeWork, Event, Place are all subclasses of Thing. The only exception is DateTime which in Schema's ontology is considered a subclass of a more generic DataType As is possible to see in the image below.

To formalize the top grounding between the two ontologies we created this mapping using yEd: with the yellow shapes we represented entities from schema.org, on the other hand, with the blue shapes we represented entities of our ontology.



Then, to build the top-level grounding ontology we imported this sort of filtered Schema.org ontology in our formal model. To do this process the Import Ontology feature in Protégé has been used. Once we imported these classes, we proceeded with classifying whether these classes can be considered equivalent, sibling and so of their corresponding in our ontology. The work that has been done can be explained starting from this picture about the final top-level grounding between our ontology and Schema.org's one.



Event

Schema.org's Event in seemed to us a bit divergent from our Event, that's why we decided to keep them both and on the same hierarchical level: all of them are in fact subclasses of Thing, they are siblings.

Facility

As we did for Event, even for Facility we decided to do a similar operation. In Schema.org there was not a direct mapping with Facility, but we found "Place" to be a good synonym for our context. In this case, Place has a very broad meaning, and we could have included Facility as a subclass of Place in some extent, though we decided to keep them separate because the very broad meaning could be misleading.

CreativeWork

Here, we created the class using the corresponding one in schema.org as a strict reference, therefore we deemed the mapping between these two classes as an equivalence using the function "Equivalent To" in Protégé.

GeoCoordinates

GeoCoordinates is another class that we defined using as reference the same-named one in schema.org, so we defined them as equivalent too. In this way, our GeoCoordinates becomes also a subclass of StructuredValue, an entity that belongs to schema.org.

DateTime

Also for this class, the mapping was immediate since defined it based on the one from schema.org. The mapping between the two classes is obviously and equivalence one, and as a consequence of this, our DateTime has been automatically inferred to be a subclass of Datatype too.

Schedule

Originally called Period, we renamed this class to Schedule after we found a direct mapping to the Period class in schema.org. We defined the "Equivalent To" rule between these two classes and, as a consequence, our class has been classified as a subclass of Intangible too.

Enumeration

Our Enumeration class, which is the father of Discount and Audience, is the same as Schema's Enumeration. By defining them equivalent, as a result, our class became a subclass of Intangible.

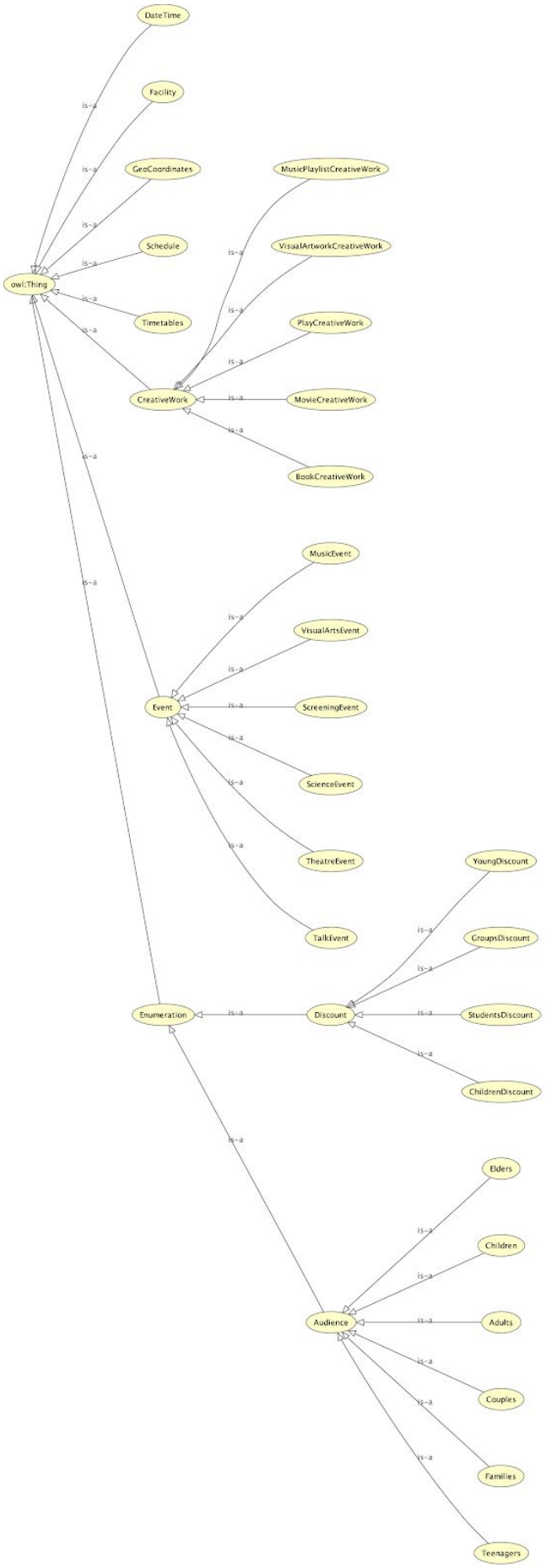
Timetables

The nearest meaning for our Timetables in schema.org is OpeningHoursSpecification, which is a subclass of StructuredValue, though it is not suitable for our ontology so we decided to keep them separate. As we did for other classes we decided to keep them at the same hierarchical level as siblings.

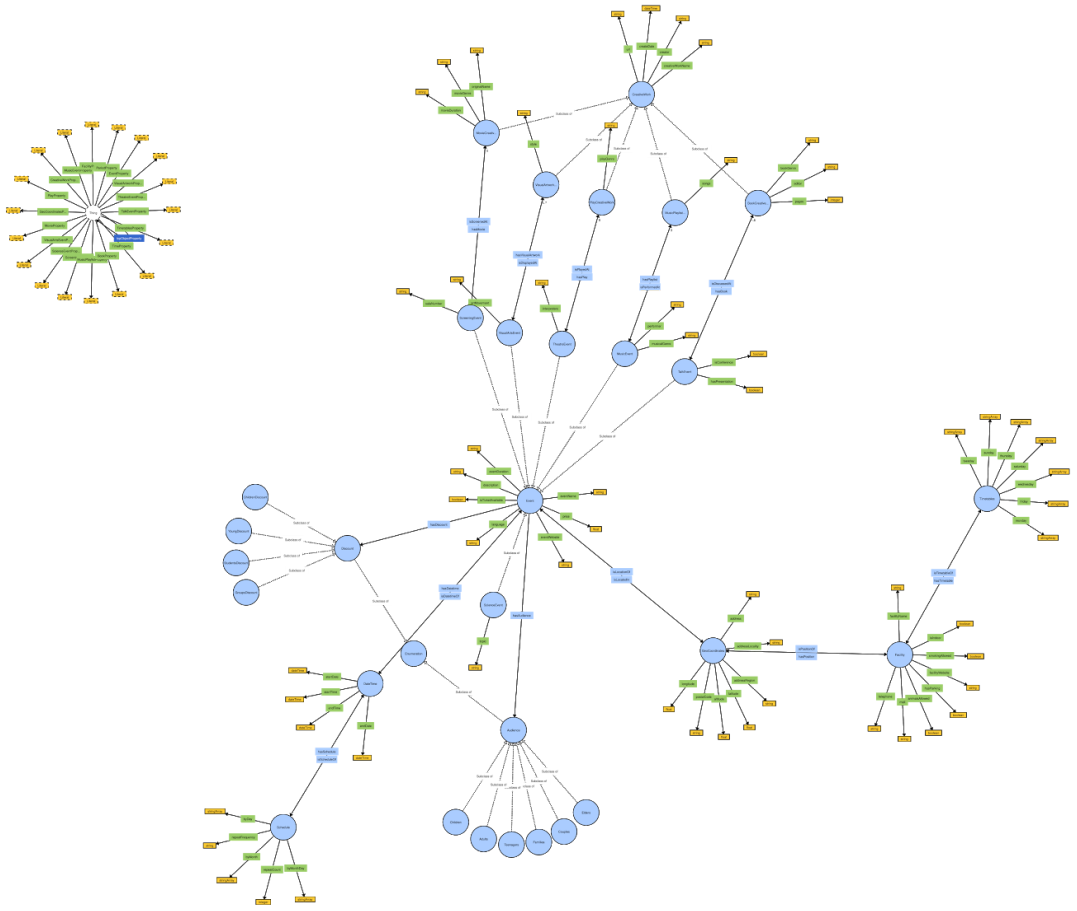
8. Model visualization

A final step that concerns the formal model is its visualization. To provide a graphical representation of it we used two different tools that present two different perspectives.

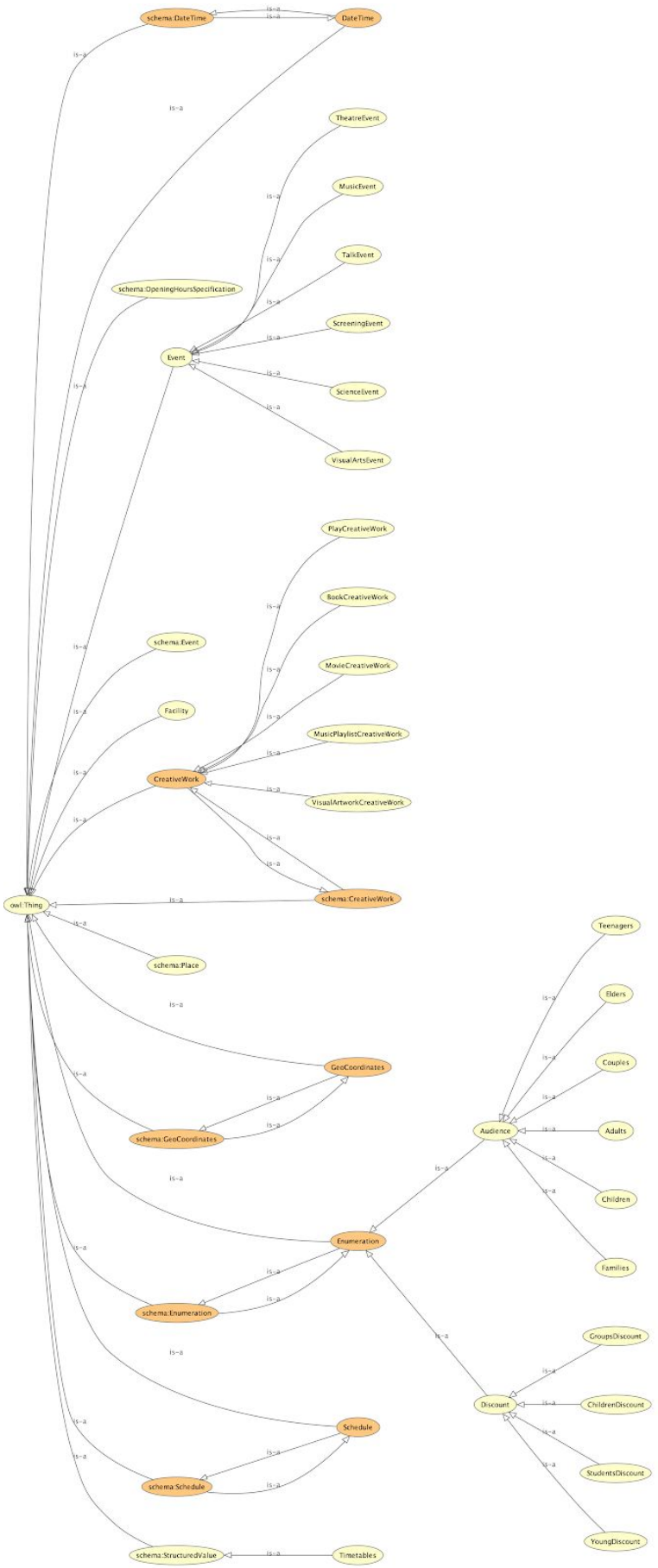
The first image was created using the OWLViz tool in Protégé and offers a representation of the whole hierarchy of the final output ontology focusing only on a complete view of the classes and the is-a relationships that connect the subclass to its classes.


DateTime
Facility
GeoCoordinates
Schedule
owl:Thing
Timetables
MusicPlaylistCreativeWork
VisualArtworkCreativeWork
PlayCreativeWork
CreativeWork
MovieCreativeWork
BookCreativeWork
MusicEvent
VisualArtsEvent
Event
ScreeningEvent
ScienceEvent
TheatreEvent
TalkEvent
YoungDiscount
GroupsDiscount
Enumeration
Discount
StudentsDiscount
ChildrenDiscount
Elders
Children
Audience
Adults
Couples
Families
Teenagers
is-a

The second image was created with the WebVOWL tool and offers a more complete visualization of the full ontology. It does not only represent the class hierarchy as the previous one but also the object properties, data properties and even the datatypes, differently coloured to better appreciate how each element of the ontology is related to all the others.



We added also a third picture created with OWLViz about the Top Level Grounding between our ontology and Schema.org's.


schema:DateTime
DateTime
is-a
TheatreEvent
MusicEvent
TalkEvent
schema:OpeningHoursSpecification
ScreeningEvent
Event
ScienceEvent
VisualArtsEvent
PlayCreativeWork
BookCreativeWork
schema:Event
MovieCreativeWork
Facility
MusicPlaylistCreativeWork
CreativeWork
VisualArtworkCreativeWork
schema:CreativeWork
owl:Thing
schema:Place
GeoCoordinates
schema:GeoCoordinates
Teenagers
Elders
Couples
Audience
Adults
Children
Families
Enumeration
schema:Enumeration
GroupsDiscount
Discount
ChildrenDiscount
StudentsDiscount
YoungDiscount
Schedule
schema:Schedule
schema:StructuredValue
Timetables

9. Integration process description

9.1. Karma integration

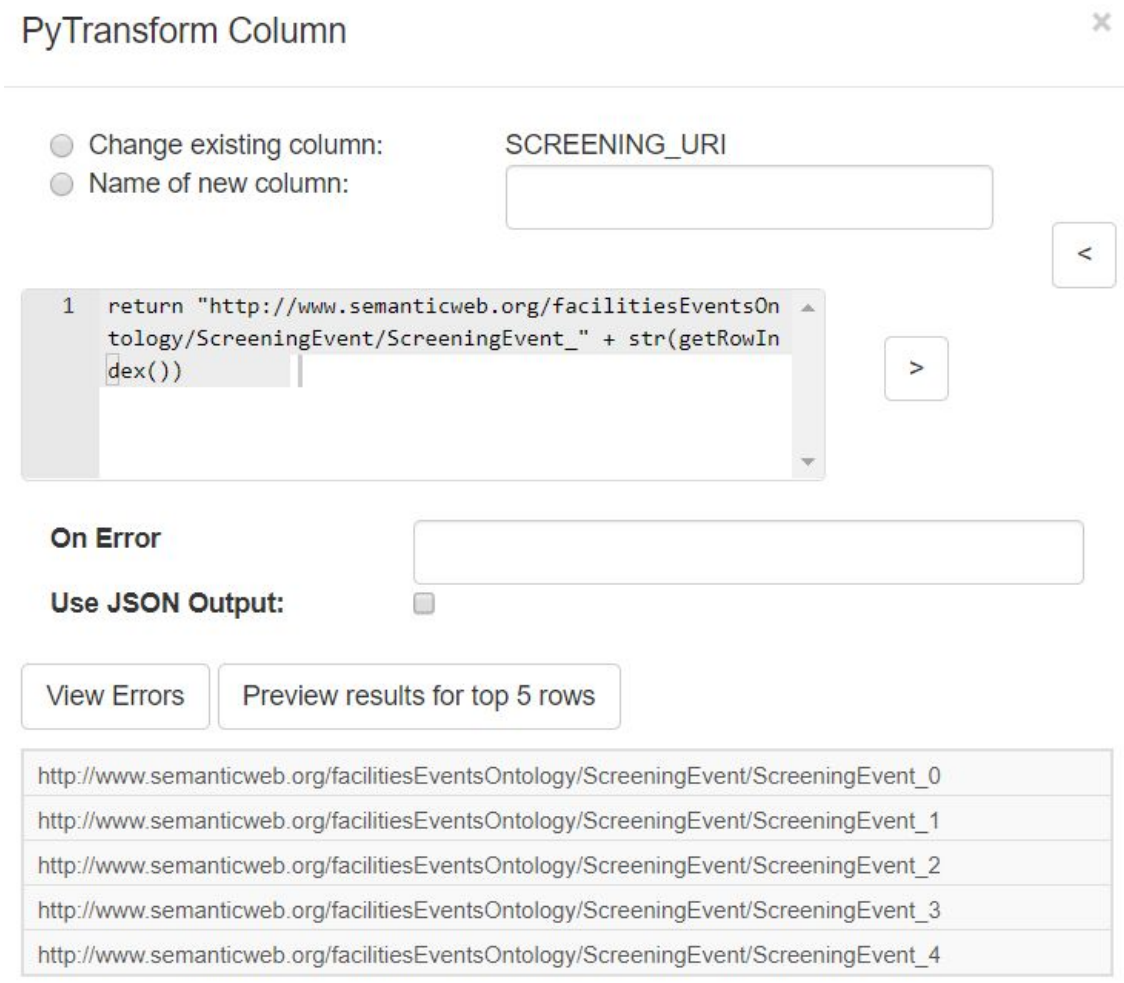
The integration using Karma was a bit tricky. Initially we had a single json file as output because we thought that it would be easier to link the different entities. We immediately switched to a csv file for each event category, a file for creative works and one for facilities. Given all the cleaning and formatting process we did with python scripts, the process was straightforward but still slow.

Since we had multiple event types and a total of 9 files, we created a R2RML template for linking the columns present in all event types (datetimes, schedule, uri of geo coordinates and uri of creative work).

The creative work URI corresponds to the location text originally searched on google. Since all the data we have about creative works comes from wikipedia, each one of them has a wikipedia url, that we escaped using python's 'urllib.quote' function.

Geocoordinates URI is instead generated using the location text originally parsed from the web. Originally we thought that this would prevent different places with the same approximate address from being unified as same geo coordinates. For example, there might be events where the resulting location is only the city Trento, even if the location text is more specific. We later realized that this approach implies that same facilities but with differently parsed names result in different geocoordinate instances. For example, "Cinema Modena, Trento" and "Multiplex Modena, Trento" give the same geo coordinates as result, but are considered different since the searched name is different.

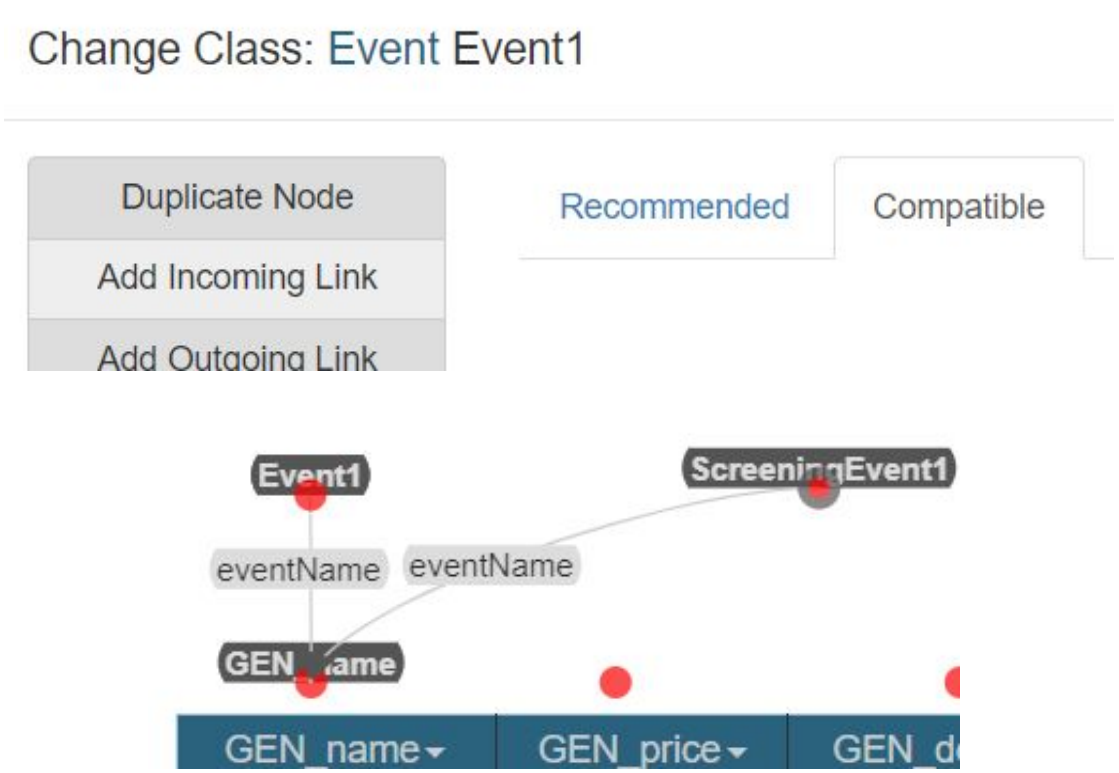
For all other URIs we used a pytransform that gets the current row because we didn't have any valid key fields. We still had to tell apart different event types, so inside the URI we set a prefix depending on the event type and the entity type.



9.2. Issues found

Most of the issues generated from the process are due to unexpected property names of the model. For instance, there was 'Event.hasSchedule' between Event and DateTime, even though there is an entity named 'Schedule' right next to 'DateTime'. Those issues were fixed quickly by the modeling group, but it took quite some time to update the karma models.

Another problem was dealing with general event columns. Event name, description, price, etc.. had all the same prefix ‘GEN_’. We tried to link them to the general event class in out template. When applying the template to a specific event type, such as ScreeningEvent, we expected to be able to change the class from Event to ScreeningEvent, but Karma didn’t recognise the inheritance of those classes. This made the process slower as even the common properties had to be remapped for each different event class. The suggestions didn’t help either, since the columns all had the same name, while the class changed from file to file.

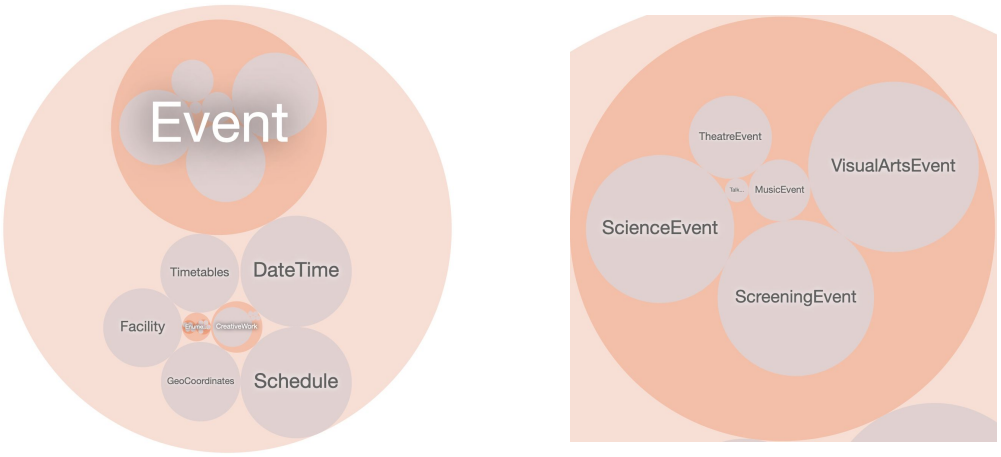


The last problem we encountered was at the first visualization in GraphDB. Apparently, creating links in Karma using 'Add incoming link' from A to B is not the same as creating them with 'Add outcoming link' from B to A. The resulting KG is still working and the links appear as intended, but the class relationships diagram of GraphDB is missing the respective outcoming/incoming links. We just had to rebuild our Karma models to fix that.

10. K-Graph Description

Once terminated the karma process we were able to import every RDF file into graphDB obtaining useful statistics about our final product and the ability to navigate the graph, with also the possibility to test our query.

From the "class hierarchy" tab we can get information about how many nodes we have for each class, giving us a better idea about what type of events are more common.



From the image we can get a general idea of the distribution of each type of event. Looking in detail, we can see that we collected more than one thousand unique event, and that the vast majority of them are "general event" (the darker part on the right). This means that for the most part we were unable to identify a category to assign at the event, either for the lack of information on the source website or the impossibility to automatically compute it via python. In addition we can see that for what concern the categorized events, this are more or less evenly divided between Science, Visual and Screen events.

11. Final considerations and open issues

The initial focus of the project was on facilities, but we quickly shifted towards events. We lost track of the potential of the facilities data and did not exploit it to its limit. One possible improvement would be to integrate all the data from the points of interest of Trentino and improve them using the Google Places API. This would lead to a more complete dataset about facilities, with the data required for good queries. If we discovered earlier the potential of the API this might have been the obvious fundamentals of our project.

Another refinement is needed for the geo coordinates results. There are some locations that have slightly different spelling and are therefore separated in two geo coordinates. A better solution would be to use the resulting address, but it needs a lot of attention on the facility name. Two different facilities could have the same coordinates and address but, according to our model, there cannot be two facilities per geo coordinates.

A last improvement would be adding a type property to the facility class. This would make it easier to search for specific facilities, such as cinemas. Initially we didn't add this field because we thought that all queries would focus on the events. This is a huge and unnecessary restriction, since the Google Places API already returns a list of 'types'. As of now, it is simply ignored, but it could be very useful for filtering the

locations of specific events. For example, it would open the possibility for the query: ‘search all music events that occur in a theatre’.

We think that the project could be applied on a larger scale, but not using manual web scraping for the events. Getting a lot of data for facilities is easier than getting data from thousands of event related websites.