PABAKE: Linked Open Seas

Knowledge Engineering

Federico Battistella, Alessandro Pavesi

February 2022

Alma Mater Studiorum - Università di Bologna

Second Cycle Degree in Artificial Intelligence

1 Introduction

In this project we applied had to model a dataset containing observation of sea parameters read by sensors spread all over the Italian seas. Our approach has the purpose of built the best ontology and relative data to be used by whoever wants to retrieve information about the state of the seas. The use of ontology design pattern increases the readability and the similarity with the standard ontologies developed for similar problems.

The final ontology is published with the IRI:

$$https://w3id.org/stlab/ke/pabake\#$$

1.1 Tasks

We asked to work on ten tasks to build the best ontology we can do. These tasks includes lot of works on the data, additionally it's asked to configure tools to better understand the philosophical approach to build the ontology and moreover to integrate the ontology and the data derived into tools for data visualization to be published.

The tasks are:

1. Analysis of existing datasets using heterogeneous formats, to produce RDF knowledge graphs
2. Application of the eXtreme Design methodology (competency questions, ODP reuse, testing, etc.) to develop OWL ontologies for the knowledge graphs
3. Definition of mapping rules for transforming input data into semantic web knowledge graphs, according to the developed ontologies
4. Generation of URIs and publication of ontologies and knowledge graphs (with permanent URIs)
5. Application/use/configuration of tools for entity linking
6. Application/use/configuration of tools for ontology alignment
7. Publication of a SPARQL endpoint
8. Integration of LODView for knowledge graph browsing
9. Integration of LODE for producing human-readable documentation of the ontologies
10. Creation of a docker that will contain every thing.

The goal of this project is to analyze an input data source made up of CSV files containing sensor and observation data. Sensors of various sorts (such as wave gauges, thermometers, and barometers) are part of two connected networks of buoys and tide gauges that monitor the environmental and meteorological conditions of Italian waters. The objective is to decipher the data source's schema in order to model and build a knowledge graph. A modular ontology network for describing subject knowledge at various levels of granularity and flavors; and connected open dataset must be included in the knowledge graph.

The knowledge graph will be checked about the consistency with SSN/SOSA ontologies, and evaluated as a reorganization of the Italian Institute for Environmental Protection and Research's current KG (ISPRA).

1.2 Data

The original data sources provided to us include three dataset from the Linked Data by the Italian Institute for Environmental Protection and Research (ISPRA):

- RON Rete Ondametrica Nazionale
- RMN Rete Mareografica Nazionale
- RON and RMN observations

Given the huge amount of data present in the dataset, including values from different type of observation and different things to observe (like conductivity, or temperature or ph) the choice on what to model, made with the professor, fell of the *wave* data.

The data are given as csv files, the main files are:

- *observations.csv*: it provides the list of files to read, with information about the time and the place where the data are taken
- *observations_schemata.csv*: defines the observed parameters with unit of measure and description
- *sensors.csv*: includes the basic information about the sensors
- *station.csv*: includes the basic information about the stations
- *csv/wave folder*: contains all the csv files with the observation values, the main data are in here.

Analyzing the data we found the data are clean and correct for the major part, only some preprocessing was done. The small part of the data to be cleaned needs only some adjustment on how they are written, so we chose to apply a function to lower the capitalized characters. Moreover we check the dataset for error values or incomplete data, we found only some values that are missing but the small number of it makes us to chose to set the value at 0.0, for these values 0.0 is not allowed, so it's safe to assign it.

2 Discussion

To elaborate the project we followed the *eXtreme Design methodology* that is a framework for pattern based ontology design.

2.1 Concepts

The main concept are the observations from whom the final users will extract the information about the sea. The *observations* includes a lot of data provided by *sensors*, and also is connected to the *area of sea* it observes, with the *coordinates*, the *city* and more other parameters. The huge amount of data and connection between each data allow us to use different pattern, but also condition our approach to the modelling phase.

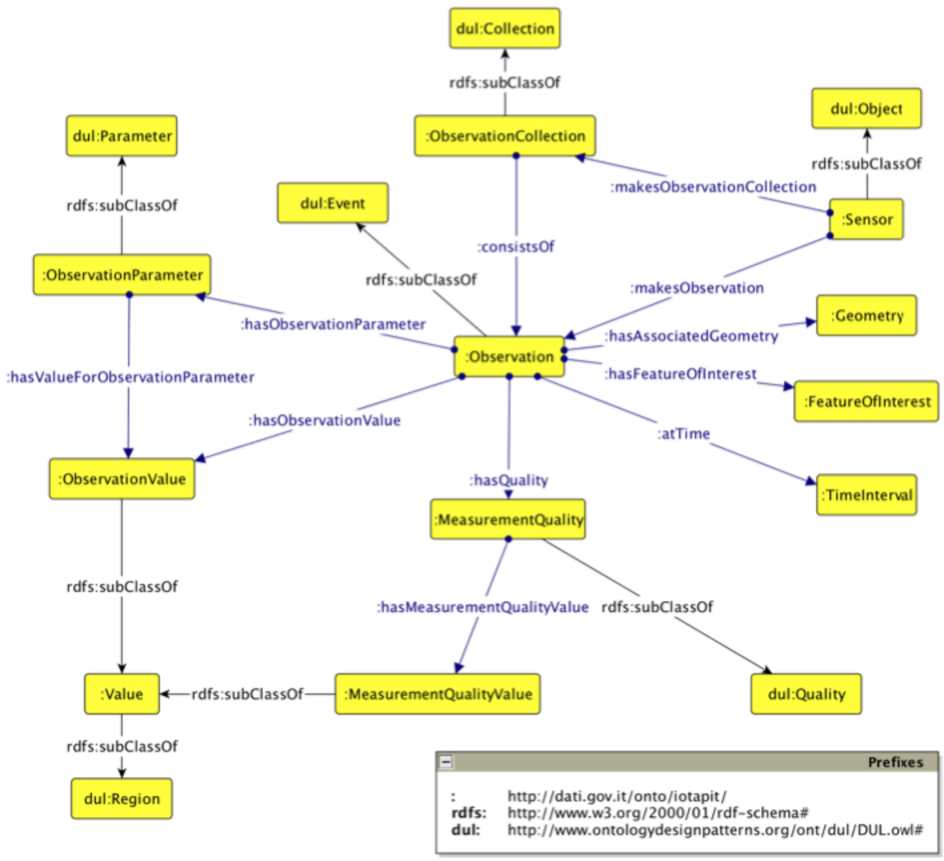
The main concepts used are:

- **Provider**: The provider of the station
- **Station**: A station represents a physical object that includes a set of Sensors.
- **Sensor**: It represents the physical object that takes the observation
- **ObservationCollection**: It represents a single observation that involves different parameters observation, made at the same time on the same station
- **Observation**: It is a concept that represents a specific observation in a defined state
- **ObservationValue**: The value of the sensor reading
- **ObservationParameter**: The parameter observed, it can be the direction of the wave, its height or its direction
- **City**: The city where the station is near
- **Sea**: The sea in which the station is

2.2 Ontology Design Pattern

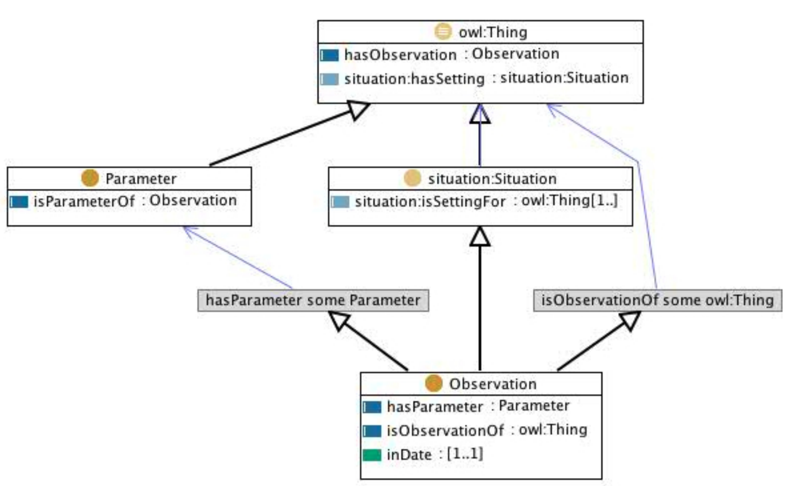
Ontology Design Patterns (ODPs) are a middle ground approach to ontology development. They may be thought of as a stripped-down form of design principles similar to those found in basic ontologies, but with less 'clutter.' They can be smartly modularized basic ontology pieces that serve as design snippets for effective modeling methods, in other words. They may also be considered as a method of bottom-up pattern discovery that is then repeated across the ontologies and made available to others as a 'best practices' design solution for some modeling component.

ObservationCollection The Ontology Design Pattern that mostly shapes our ontology is the *Observation Collection* patter:

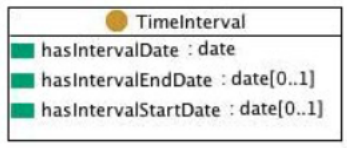


It includes various concept, the basic is Observation that is connected to it's value and the other parameters like the one shaping the feature of interest and the time, as well as the measurement. The observation collection concept is necessary to keep track of different parameter observations that are made by a single sensor in a single station. It has a huge importance on clarify the structure behind the observation retrieval, this pattern defined all those part involved in a simply manner.

The Observation Collection is useful but leak of some specifics that for us are important, so we decide to include also the *Observation* pattern that adds the Situation to the class. It is similar, and for small part override, the observation collection patter but it is useful for model the city and the sea correlation with the observation.



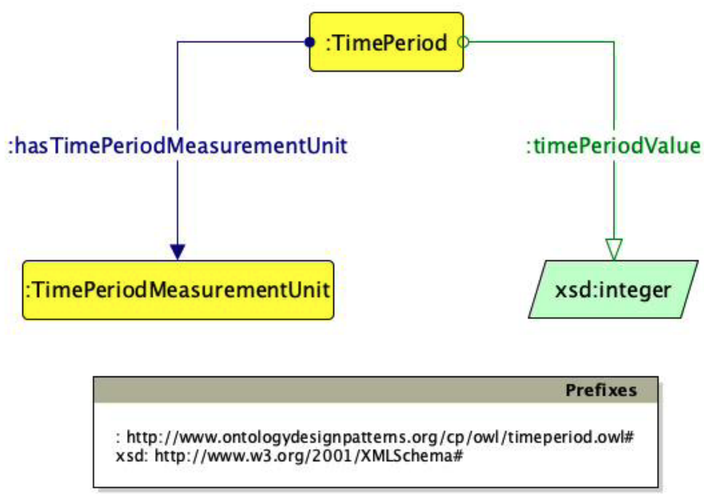
TimeInterval Directly connected with the ObservationCollection pattern the *TimeInterval* pattern:



It is used to model an interval of time in which something happens, in our case we use this specification as the father class of another time related concept: *TimeInstant*.

We choose to use a subclass of TimeInterval for model our case in which the observation is made in a single point of time, without a duration because it is only a read of sensors.

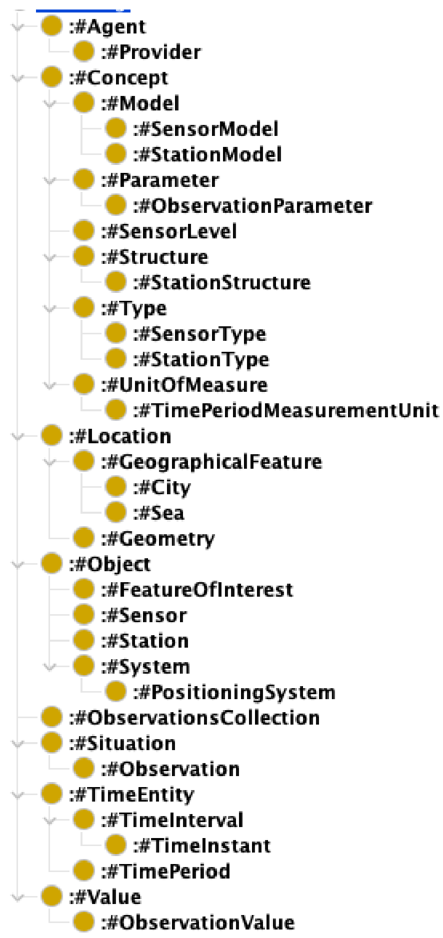
TimePeriod Laying in the same category of patter we decided to use also the *TimePeriod* pattern:



The difference from the previous is that in this case the period of time is not defined by a start and and end date, including the hours and minutes, but it is described with words that describes period of time. For example "one month" is a measure of time that hasn't the start/end defined by a date, but it also define a period of time.

2.3 Ontology

The ontology includes all the concept we described before but more are defined to give the correct structure based on the patterns and our logical interpretation of the data.



We set a lot of axioms to define the sub class property, or axioms to restrict the number of relations between two classes. Moreover we set domains and ranges for each Object Property and Data Property

2.4 Competency Questions

We utilized competence questions to better discover potential Ontology Create Patterns (ODPs) in order to design our result graphs, following the principles of eXtreme Design methodology. Our competence questions are as follows:

1. How many stations are deployed in the X seas?/deployed in the X city?
2. How many/What seas are monitored?
3. How many/What models of station are actually deployed?
4. How many/What materials are used for build the stations?
5. How many/What places are monitored?
6. What is the value of “parameter” for the sea X in a certain period Z? for the city Y? for different parameter
7. What is the parameter X for the city Y at time Z? And in the interval of time from Z1 until Z2?
8. What are the observable property for the city X?
9. What are the sensors for the city X? for the sea X?

10. How many stations there are for the network X?
11. How many sensors have the station X?

The ontology is so huge that the number of information that can be retrieved massive, from the information about the sea or the city to a particular characteristic of a single station like the material with which is built.

SPARQL We will show a small part of the competency questions that can be defined, using the SPARQL program:

Here the query to extract the values of a specific parameter (Hm0) for a defined city (Alghero, ISTAT code 90003):

```
1)
Select ?value
where {
    ?obs pabake:hasParameter <https://w3id.org/stlab/ke/pabake/ld/observation_parameter/
        pabake:hasObservationValue ?value;
        pabake:isMadeBy ?sensor.
    ?sensor pabake:hasStation ?station.
    ?station pabake:isInProximityOf <https://w3id.org/stlab/ke/pabake/ld/city/090003> .
}
```

It can be modified to extract all the observed parameters or for extract data for different cities.

Here the query to extract the material with which the stations are made:

```
2)
Select DISTINCT(?material)
where {
    ?station pabake:hasStationModel ?stationModel.
    ?stationModel pabake:hasStationStructure ?material
}
```

From here we can extract the station information about the positioning system, the diameter, the manufacturer and other parameters.

Here the query to extract the cities inclueded in this project:

```
3)
Select DISTINCT(?city)
where {
    ?station pabake:isInCity ?city.
}
```

2.5 Mapping Rules

To build the instances based on the ontology we have to use a framework to map the data to the respective concept, called *PyRML*. This framework provides a natural way of map data from csv files into triples in RDF type. PyRML is a python package that allows to apply a template to the data, so it's easy to transform data into a knowledge graphs. The template we made are:

- *observation.ttl*: the observation template is used to create the ObservationCollection classes and the other class TimeInstant
- *observationsDir.ttl*: this template model the subclass observation, here with the information about the *Direction of the wave*
- *observationsTp.ttl*: here the template model the information about the *Spectral peak wave period*
- *observationsTm.ttl*: here the template is used for the information about the *Average wave period*
- *observationsHm0.ttl*: used for the information about the *Significant wave height*
- *project.ttl*: contains all the other classes, including the stations, the sensors, the city or the sea

2.6 Ontology Alignment

The ontology alignment is the process consisting in the determination of correspondences between concepts in ontologies.The process has been performed on three fondational ontologies: Dolce, Ssn, Sosa.

The first ontology Dolce is a foundational ontology, due to its very general applicability we have found a lot of correspondencies between Pabake and it, however we can state that most of the Dolce concepts can generalize Pabake's, which is very specific to for its domain. There are some concepts that seems to coincide perfectly, for example "Time Interval" express the same concept in both the ontologies, on the other hand, the very generalizing concept "Endurant" can only enclose on of the most general concepts of the Pabake ontology as "Object". We can then sum up that DOLCE ontology can nearly enclose the whole Pabake ontology in itself because of the flexibility and non-deep specific domain design of it. Although there are some concepts that remain difficult to express or to enclose in Dolce, for example the "Observation Collection". The Sosa ontology seems to be the ontology with the most correspondences with Pabake, the domain of both is very similar and quite specific so we can find a lot of equivalent concepts between the two ontologies.

There are a lot of concepts that we can consider equivalent:

- ObservableProperty - ObservationParameter
- time:instant - TimeInstant
- Observation - Observation

- time:interval - TimeInterval
- Sensor - Sensor
- QuantityValue - Value

These are some of the most representative equivalent classes in the alignment of the two ontologies, we also found few object properties common to both the ontologies which have the same meaning and the same function, and in most cases, also the same name. One example is the object property makesObservation from Pabake which is equal to detects in Sosa. The Sosa ontology is very suited for the representation of the datas considered in this paper, due to the similarity of the contexts represented in the ontologies we managed to find a lot of equaivalences, and much less subclasses than Dolce. Although not all the concepts are common to both the ontologies, for example Sosa contains the "Actuator" concept which has not been represented in Pabake because it wasn't present in the data.

2.7 Entity Linking

To check the correspondences of concept and classes of our knowledge graph and other external knowledge graphs like DOLCE o SSN/SOSA we used LIMES Axel-Cyrille Ngonga Ngomo Mohamed Ahmed Sherif Kleanthi Georgala Mofeed Hassan Kevin Dreßler Klaus Lyko Daniel Obraczka, 2021. LIMES is a link discovery framework for the Web of Data.

The program exports a file in which are listed the concepts similar between two knowledge graphs using the Jaccard Index. We maintain the concepts that have at least a 0.5 Jaccard Index.

DOLCE is a conceptual ontology so there are few correspondences with our ontology, but still the use of patterns helps us to connect DOLCE with PABAKE:

DOLCE	PABAKE
time-interval	TimeInterval
feature	FeatureOfInterest
feature	GeographicalFeature

Moreover we have SSN/SOSA that is more like ours, in fact the similarity includes almost all the concept of that ontologies:

DOLCE	PABAKE
Interval	TimeInterval
Instant	TimeInstant
FeatureOfInterest	FeatureOfInterest
ObservableProperty	ObservableParameter

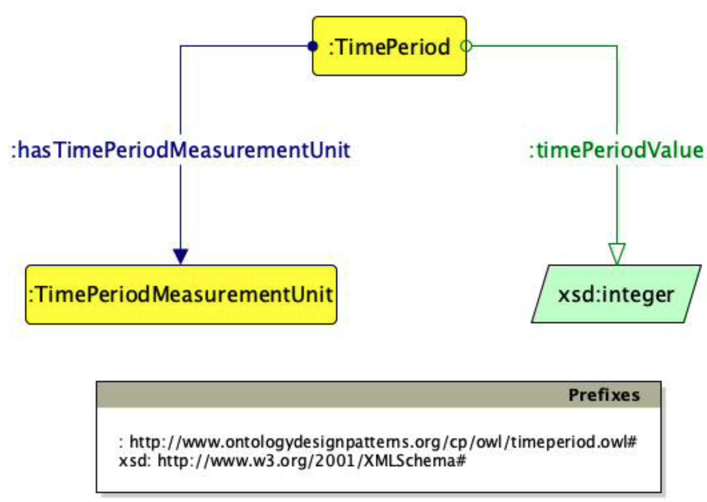
2.8 Docker

Finally the entire project is delivered as a Docker container in which there is the integration of our data with *LODE* and *LODView* for knowledge graph browsing and for producing human-readable documentation of the ontology. The *Docker* contains also the a SPARQL endpoint used by the other services and useful to query the knowledge graphs, it is included on the *Virtuoso* service.

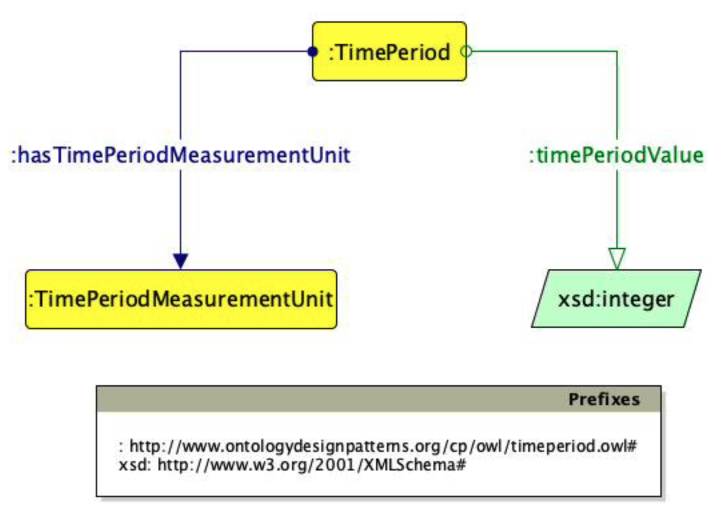
The exposed port for each service are:

- *Virtuoso*: http://localhost:8891/sparql
- *LODE*: http://localhost:9090/lode
- *LODView*: http://localhost:8181/lodview

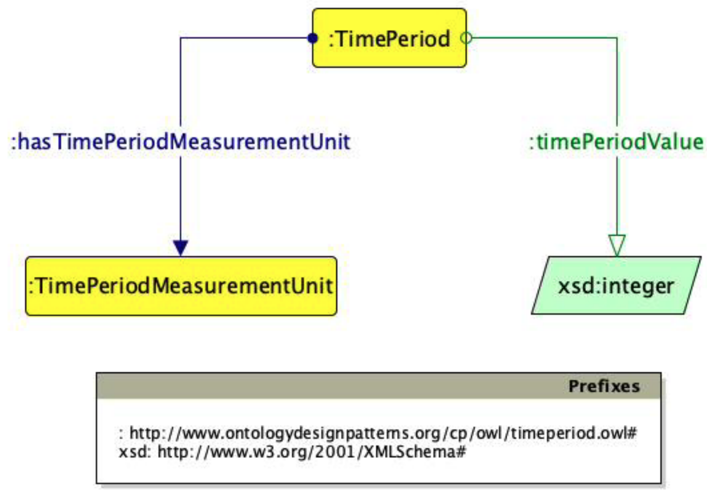
The services needs to be called in the right manner to fully works, for example the Virtuoso service shows a page in which it is possible to write and execute queries.



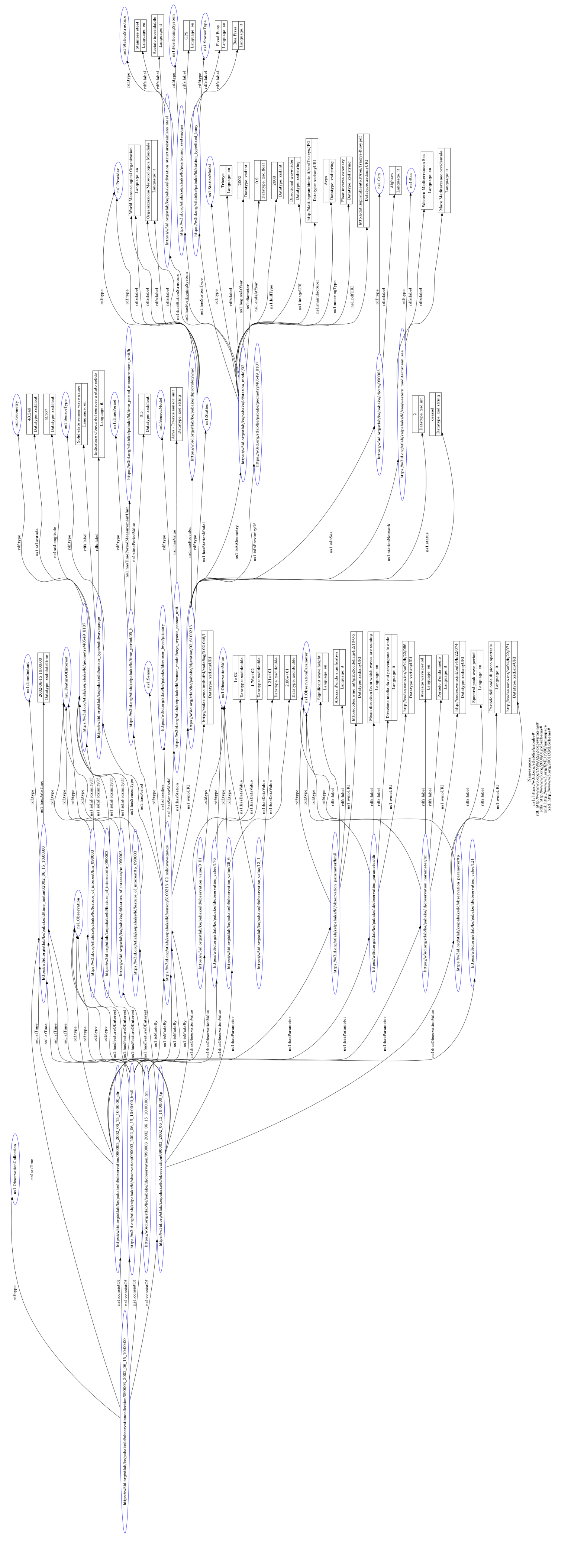
LODE provide a visualization service of the knowledge graphs, so the instances of our ontologies:



LODE provide a visualization service of the ontology, so it needs to be called with the URL of the ontology:


:TimePeriod
:hasTimePeriodMeasurementUnit
:timePeriodValue
:TimePeriodMeasurementUnit
xsd:integer
Prefixes
: http://www.ontologydesignpatterns.org/cp/owl/timeperiod.owl#
xsd: http://www.w3.org/2001/XMLSchema#

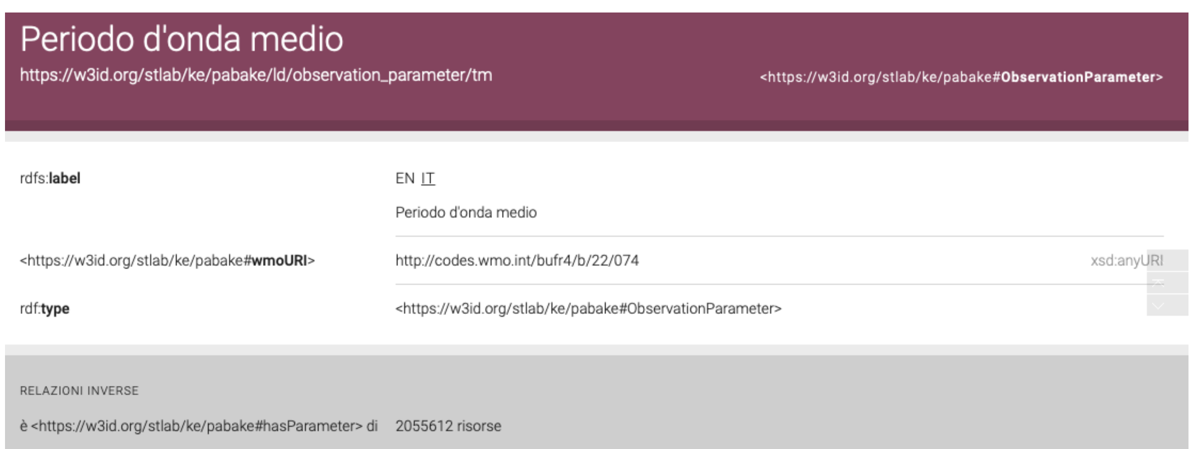
An example of final data elaborated with our ontology:



Using the services we can explore the knowledge graph.

For example we can explore the entire Ontology using LODE:

Or we can explore an instance of the ontology using LODView:



3 Conclusion

Our project includes to understand the schema underlying the data source for modelling and constructing a knowledge graph. With the purpose of create a modular ontology to represents the domain knowledge exposed by the linked open dataset provided. Our approach uses different Ontology Design Patterns to exploit the data organization defined by sector experts. The amount of data created is huge, it is possible to explore it with the docker published on GitHub.

To conclude we are satisfied with the organization of the concepts as well as the creation of a project that includes all the relevant analysis, the services implementation and extensions required.

References

A., Gangemi (n.d.). “Ontology Design Patterns for Semantic Web Content”. In: ().

Axel-Cyrille Ngonga Ngomo Mohamed Ahmed Sherif Kleanthi Georgala Mofeed Hassan Kevin Dreßler Klaus Lyko Daniel Obraczka, Tommaso Soru. (2021). “LIMES - A Framework for Link Discovery on the Semantic Web”. In.

D., Merkel (n.d.). “Docker: lightweight linux containers for consistent development and deployment.” In: ().

D. V. Camarda S. Mazzini, A. Antonuccio (2014). “LodView”. In.

E., Presutti V. Daga E. Gangemi A. Blomqvist (2009). “eXtreme Design with Content Ontology Design Patterns.” In.

M.A., Musen (2015). “The Protégé project: A look back and a look forward.” In.

Peroni S. Shotton D.M., Vitali F. (2012). “Making Ontology Documentation with LODE.” In.

Z., Auer S. Bizer C. Kobilarov G. Lehmann J Cyganiak R. Ives (2007). “DBpedia: A Nucleus for a Web of Open Data.” In.