Knowledge Engineering Project - Soil Heritage

Giulio Fortini - Alessandro Liscio

September 2021

1 Introduction

Knowledge graphs have emerged as a compelling abstraction for organizing world's structured knowledge, and a way to integrate information extracted from multiple data sources. A crucial step in formalizing knowledge as a linked dataset is the design of an ontology. An ontology is a formal specification of the conceptualization of a domain [1]. An ontology plays important role in information exchange and in capturing the background knowledge of a domain that could be used for reasoning and answering questions.

In this report we present the development process of an ontology and a Semantic Web Knowledge Graph starting from all the available data collected by the Italian Institute for Environmental Protection and Research (ISPRA) about the soil consumption in Italy.

Some background on the data that have been used will be given in Chapter 2. In Chapter 3 we will present our ontology and the design choices that have been taken. In Chapter 4 we briefly describe the technologies used to define mapping rules and build the knowledge graph. Necessary steps for the publication of the endpoint are described in Chapters 5 and 6. Finally, some SPARQL queries runnable on the final knowledge base are listed in chapter 7.

2 Background

The knowledge framework on soil consumption in Italy is available thanks to data from the soil consumption monitoring network, created by ISPRA with the collaboration of the Environmental Protection Agencies of the Regions and Autonomous Provinces. This framework allows ISPRA to reconstruct the trend of land consumption in Italy from the second post-war period to the present day.

This network guarantees the availability of high accuracy data from all over the country coming from more than 180,000 sampling points, all integrated with high-resolution geographical data.

The raw data source is a tabular collection of all the different locations, divided by administrative unit (National, Regions, Provinces and Cities), with a set of indicators specifying the amount of soil consumption at a given time together with other relevant information. It is in fact possible to extract from these data, not only the amount of soil consumed as a surface measure or a percentage of a specific region, but also additional information such as the amount of consumed soil in seismic risk areas, the population density or the distance from the coast. This availability of data allows to have a more detailed picture of the consumed soil in the country and provide a deep knowledge about risks and trends of human intervention on the soil.

This knowledge, however, is not easily accessible in the form of tabular data and it is for this reason that the raw data has been structured into a Linked Open Dataset in an effort by ISPRA to make more accessible the knowledge they collected using Semantic Web Technologies.

The scope of this project was to replicate this effort, building a proper ontology for the available data and publishing a SPARQL endpoint to interrogate the knowledge base.

3 Methodology

There is no single correct ontology-design methodology, since it is really dependent on the domain we are trying to describe and the answers we want to provide. For these reasons is always good to start from questions about the domain and the answers we expect to possibly get form the knowledge we have about that domain.

In the following paragraphs we are going to present the main steps we followed in designing the ontology. The ontology has been defined in OWL (Web Ontology Language) using a tool for ontology development called Protégé. This tool makes it easier to define classes and properties, giving a visual overview of the ontology and tools to keep it consistent (the reasoner) and to export everything as a OWL file.

We want an ontology that can give answers to our Competency Questions (CQ). And we want a development process without many constraints and easy to rethink and to evolve over time.

3.1 Ontology Design

Since designing ontologies is not an easy job, a good practice, already widespread in classical programming, is the reuse of patterns. For this reason the approach we used to develop our ontology is based on Ontology Design Patterns and took inspiration by eXtreme Programming in software engineering. This approach is called eXtreme Design with Ontology Design Patterns [2]. ODPs are modeling solutions to solve recurrent ontology design problems and together with an agile approach allow to develop ontologies in a fast and consistent way.

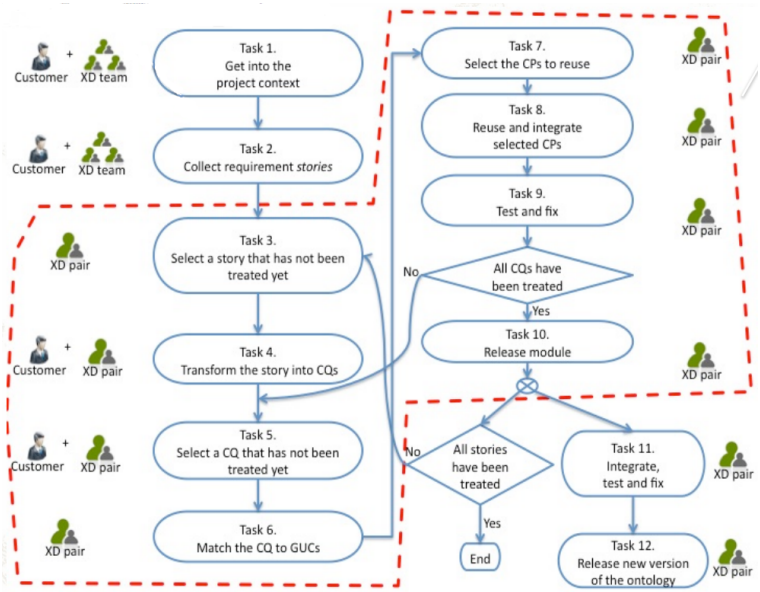

Figure 1: eXtreme Design Cycle

3.1.1 Competency Questions

After a first analysis of the data and the definition of some text requirements, we proceeded to define the Competency Questions. As said before, these are crucial for a correct design and consequent assessment of the ontology. These questions will be useful to test the knowledge graph and see if it responds to the initial requirements, providing the expected information when translated into a SPARQL query.

The following is a not exhaustive list of competency questions for our ontology:

- How much soil has been used in percentage on the Italian soil in 2015?
- Which are the first 10 cities for soil consumed in percentage in 2012?
- Which is the first region for soil consumed in ha in 2012?
- Which province increased the most its soil consumption between 2012 and 2015, in percentage?
- Which are the first 10 provinces for soil consumed on high seismic risk areas?

3.1.2 Ontology Design Patterns

Once the CQs have been defined, a correct definition of classes and relationships is needed to produce an ontology able to satisfy them. Following the iterative

approach proposed in the eXtreme Design methodology, different requirements for the ontology emerged at every step, requiring solutions that would answer the questions while keeping the ontology simple and consistent.

Ontology Design Patterns are reusable solutions to different common design problems, and they allow a faster and more correct implementation of ontologies. For these reasons, when studying possible implementations of classes and relationships, is good to look up at at a list on patterns that may solve that particular design issue.

For our particular case, three specific patterns have been used:

- N-ary relation: for connecting all the variables involved in a particular measurement in a specific class, called Indicator. This is the class representing a triple of measurement-time-location.
- Collection Pattern: by defining a collection we create a container, in our case is the class IndicatorsCollection. A membership relation connect all the indicators for a particular time and place to a unifying collection.
- Classification Pattern: the Metric class is a classifier for Indicators.

3.1.3 Ontology Schema

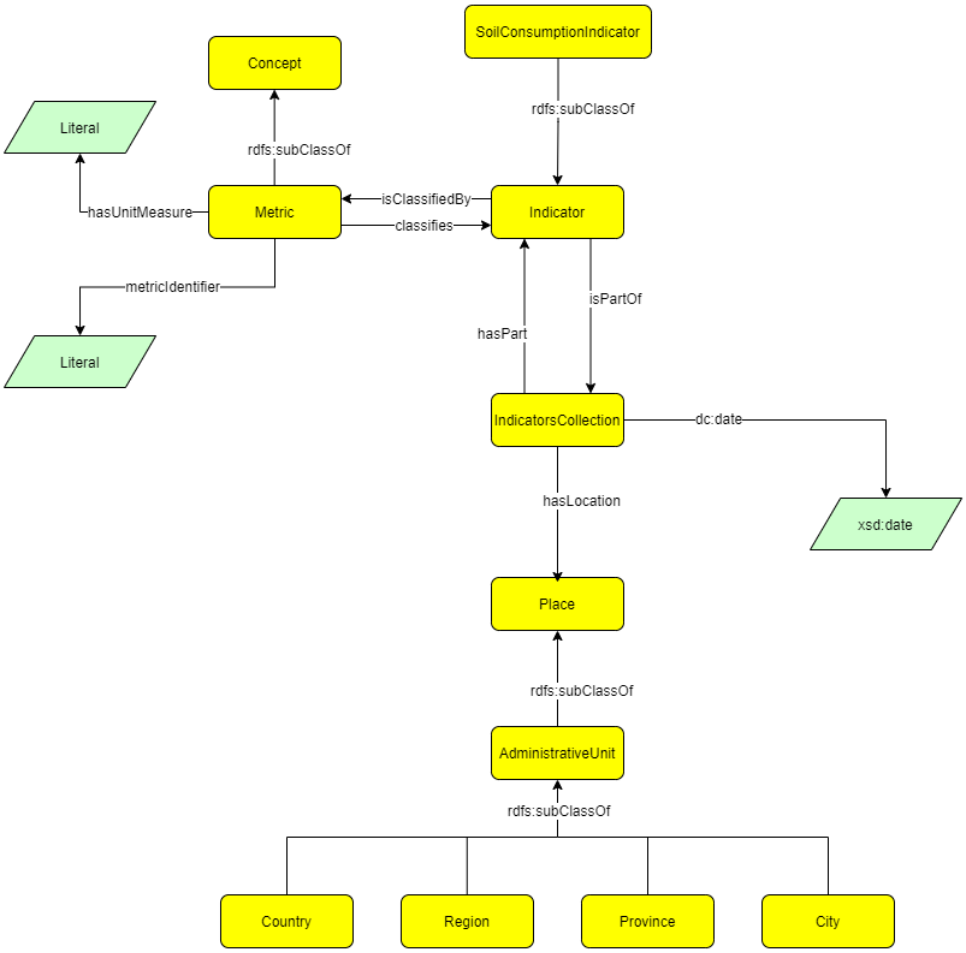

Figure 2: Schema of the ontology

4 Knowledge Graph Implementation

Once the ontology have been developed in the Protégé environment and its consistency verified using a reasoner, it has been exported as an owl file. This file has subsequently been serialized in our Python environment, thanks to the RDFLib package, to define the namespace and the individuals that populate the knowledge graph.

RDFLib is a Python library for working with RDF. This library contains parsers/serializers for almost all of the known RDF serializations, like Turtle, both in-memory and persistent Graph back-ends for storing RDF information and numerous convenience functions for declaring graph namespaces and running SPARQL queries.

Mapping rules have been defined to store triples correctly, in conformity with the defined ontology, inside the knowledge graph. Data had to be extracted from their tabular storage and transformed in triples using our new and already existing ontologies. Thanks to RDFLib this step is straightforward and the resulting graph is easily exported in a Turtle format for later publication.

5 Ontology and Data Alignment

Since the purpose of building knowledge graphs is generally to publish and make the knowledge available on the web, it is critical to make our knowledge graph inter operable with the knowledge already present on the Semantic Web.

Aligning classes, relationships and individuals gives, both to human and machines, the possibility to better comprehend the meaning of concepts described in the graph thanks to equivalent definitions coming from other ontologies. In owl this kind of equivalency is indicated with the owl:sameAs relationship.

Tools exist to find all the possible matches among entities of different graphs. LIMES (LInk discovery framework for MEtric Spaces) is such a tool and we used it to discover all the possible matches with other knowledge bases on the web. It is sufficient to define a source and a target endpoint to interrogate, a restriction on the entities we want to match and a metric with a threshold for acceptance. We aligned our graph with entities coming from DBpedia and ISPRA places.

6 Ontology Publication

Once the knowledge base has been serialized into the turtle file by the means of the python environment, several services have been implemented in order to enhance the user experience while interacting with the knowledge base.

First of all, the graph contained in the turtle file is serialized to a Virtuoso SPARQL endpoint, which is made available as a service in docker. This allows for the user user to query the knowledge base with the ease of a web application.

As far as visualization is concerned, several services have been made available:

- LODE: a Tomcat server application used to create HTML documentation for Web Ontology Language (OWL) ontologies;
- LodeView: a Java Web application which, used in conjunction with a SPARQL endpoint, allows for the publication of RDF data according to all defined standards for Linked Open Data;
- LodeLive: a tool based solely on SPARQL endpoints that can be used for connecting RDF browser capabilities with the effectiveness of data graph representation;
- WebVOWL: a web application for the interactive visualization of ontologies, either by providing the ontology IRI or by manually uploading the ontology file.

All of the services have been implemented using docker in order to make the services available in any operating system without running into inconsistencies.

7 SPARQL Queries

8 Conclusions

An OWL ontology for the soil consumption in Italy has been developed following the best practices. An agile approach, the reuse of patterns and the adoption of a reasoner allowed to produce a consistent and versatile ontology.

Using Python and its RDF libraries a knowledge base has been populated with all the individuals available in the raw dataset and mapped into RDF triples following the ontology schema.

The alignment makes the knowledge base inter operable with other ontologies on the web and the publication of it as a SPARQL endpoint makes it easy to interrogate and extract knowledge from data.

References

[1] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[2] Valentina Presutti, Enrico Daga, Aldo Gangemi, and Eva Blomqvist. extreme design with content ontology design patterns. In *WOP*, 2009.