Harmonisation of data to SmartOpenData model. Initial iteration

Deliverable D3.3 :: Public

Keywords: data harmonisation, ORM, RDFS

Linked Open Data for environment protection in Smart Regions
# Table of Contents

Table of Contents .................................................................................................................. 2  
List of Figures .......................................................................................................................... 5  
List of Tables .......................................................................................................................... 5  
Executive Summary .................................................................................................................. 7  
1 Introduction .......................................................................................................................... 7  
2 Implementation of the SmartOD Vocabularies ...................................................................... 9  
3 Development of the domain-specific vocabularies ............................................................... 12  
  3.1 Mapping rules for object-types ...................................................................................... 13  
  3.2 Mapping rules for value-types ...................................................................................... 13  
  3.3 Mapping rules for associations ....................................................................................... 14  
    3.3.1 Mapping rules for named roles of binary associations ............................................. 14  
    3.3.2 Mapping rules for objectified associations ............................................................... 14  
  3.4 ORM constraints .............................................................................................................. 16  
4 Definition of the mappings .................................................................................................... 16  
  4.1 Example dataset: parcels ............................................................................................... 17  
  4.2 Tutorial for the GUI of the RDF extension of OpenRefine .............................................. 18  
    4.2.1 Set the base URI ...................................................................................................... 19  
    4.2.2 Manage prefixes ..................................................................................................... 21  
    4.2.3 Type instances ........................................................................................................ 21  
    4.2.4 Add properties ........................................................................................................ 22  
    4.2.5 Customise resource nodes ...................................................................................... 22  
    4.2.6 Customise literal nodes ........................................................................................... 24  
5 Data transformation ............................................................................................................... 25  
  5.1 BatchRefine Example using cURL .................................................................................. 25  
6 Pilots Demonstration .............................................................................................................. 26  
  6.1 ARPA pilot ..................................................................................................................... 26  
  6.2 Spanish-Portuguese pilot ............................................................................................... 28  
  6.3 MAC pilot ....................................................................................................................... 29  
    6.3.1 ETIS & INSPIRE ...................................................................................................... 31  
    6.3.2 W3C namespaces, persistence & future management .............................................. 31  
    6.3.3 The ETIS Data Model ............................................................................................. 31  
    6.3.3.1 Namespaces used ............................................................................................... 33
6.3.3.2 Class etis:Destination ................................................................. 33
6.3.3.2.1 Object Properties ................................................................. 33
6.3.3.2.2 Datatype Properties .............................................................. 34
6.3.3.3 Class etis:PopulationInfo .......................................................... 35
6.3.3.3.1 Datatype Properties .............................................................. 35
6.3.3.4 Class etis:TouristData ............................................................... 35
6.3.3.4.1 Object Properties ................................................................. 36
6.3.3.4.2 Datatype Properties .............................................................. 36
6.3.3.5 Class etis:VisitorsByOrigin ....................................................... 36
6.3.3.5.1 Object Properties ................................................................. 37
6.3.3.6 Class etis:VisitorsByTransport .................................................. 37
6.3.3.6.1 Object Properties ................................................................. 37
6.3.3.6.2 Datatype Properties for Origin & Transport ......................... 37
6.3.3.7 Class etis:AverageSeasonalWeatherPattern .............................. 37
6.3.3.7.1 Datatype Properties .............................................................. 37
6.3.3.8 Class fc:TransportPoint ............................................................. 38
6.3.3.8.1 Object Properties ................................................................. 38
6.3.3.8.2 Datatype properties .............................................................. 38
6.3.3.9 Class etis:TransportLinks .......................................................... 38
6.3.3.9.1 Object Properties ................................................................. 38
6.3.3.9.2 Datatype Properties .............................................................. 38
6.3.3.10 Class etis:Carrier ................................................................. 39
6.3.3.10.1 Object Properties ............................................................... 39
6.3.3.10.2 Datatype properties ............................................................ 39
6.3.3.10.3 Object Properties ............................................................... 40
6.3.3.10.4 Datatype Properties ............................................................ 40
6.4 SAZP pilot ............................................................................... 40
6.4.1 Background ............................................................................. 40
6.4.2 Linked data .............................................................................. 41
6.4.3 Transformation tools (GML XML/CSW to RDF) and triple store installation .................................................. 41
6.4.4 Transformation setup & customization ....................................... 42
6.4.5 Transformation execution .......................................................... 42
6.4.6 Import of INSPIRE RDF datasets into triple store ........................ 42
6.4.7 Publish INSPIRE RDF datasets through GeoSPARQL interface ........................ 43
6.4.8 Publishing INSPIRE metadata via SPARQL interface ........................................ 43

7 Conclusions and Future Work .................................................................................. 44

  7.1 RDFS SmOD vocabularies .................................................................................... 44
  7.2 Extension of the SmOD vocabularies .................................................................... 45
  7.3 Definition of the mappings and transformations .................................................. 45

8 References .................................................................................................................. 46

Annex A: RDF reification for ORM objectified association ........................................... 47
Annex B: Spanish-Portuguese pilot: Dictionary & links to the TRAGSA database ....... 50
  Classes ......................................................................................................................... 50
  Properties ..................................................................................................................... 51
Annex C: Spanish-Portuguese pilot: ORM Documentation ............................................ 54
Annex D: Spanish-Portuguese pilot: user queries ......................................................... 57

Annex C: Translation of ShapeFile to WKT ................................................................. 59

Definitions ..................................................................................................................... 59
  GDAL/OGR .................................................................................................................. 59
  ShapeFile .................................................................................................................... 59
  PostgreSQL ................................................................................................................ 60
  PostGIS ...................................................................................................................... 60
  PgAdmin3 ................................................................................................................... 60
  WKT ............................................................................................................................ 60
  ShapeFile To WKT using Python GDAL ................................................................. 61
  ShapeFile To PostgreSQL / PostGIS ...................................................................... 62
  PostgreGIS to WKT .................................................................................................. 65
  Generating a table ..................................................................................................... 65
  Exporting table to *.WKT file .................................................................................. 67

Annex D: Translation of ShapeFile to WKT Open Land Use as Linked Data .............. 69
List of Figures

Figure 1: Workflow of the Initial Data Harmonisation ................................................................. 8
Figure 2: A Fragment of the ORM Diagram of the First Release of the Spanish-Portuguese
Pilot ........................................................................................................................................ 13
Figure 3: Objectified association “ForestryTileHasPlantSpecies” .............................................. 15
Figure 4: OpenRefine tutorial: TRAGSA dataset about parcels .................................................. 17
Figure 5: OpenRefine tutorial: intended RDF schema ................................................................. 17
Figure 6 OpenRefine tutorial: the GUI of the RDF extension ..................................................... 18
Figure 7 OpenRefine tutorial: the GUI of the RDF extension ..................................................... 19
Figure 8 OpenRefine tutorial: instance typing ........................................................................... 21
Figure 9 OpenRefine tutorial: customising nodes ..................................................................... 22
Figure 10 OpenRefine tutorial: preview values ......................................................................... 23
Figure 11 The proposed data model for ETIS ............................................................................ 32
Figure 12 Snap of the SAZP Parliament web Use Interface ....................................................... 43
Figure 13 Flow diagram for processing SPARQL queries in the TripleGeo - CSW middleware
[GEOKNOW - D2.7.1] ........................................................................................................ 44
Figure 14 Spanish-Portuguese pilot (TRAGSA) ORM: an example of an object type
verbalisation .......................................................................................................................... 56
Figure 15 Spanish-Portuguese pilot (TRAGSA) ORM: an example of a constraint verbalisation .................................................................................................................................. 56
Figure 17 Open Land Use data model ......................................................................................... 69
Figure 18 Transformation to RDF ............................................................................................. 70
Figure 18 Example of OLU object in RDF .................................................................................. 72

List of Tables

Table 1 SmOD Vocabularies ........................................................................................................ 10
Table 2 OpenRefine tutorial: URI construction ......................................................................... 20
Table 3 Namespaces used in Irish pilot ..................................................................................... 33
Table 4 olu:OLU_Object ........................................................................................................... 72
Table 5 lu:BackgroundMap ....................................................................................................... 72
<table>
<thead>
<tr>
<th>Contractual Date of Delivery to the EC:</th>
<th>November 2014</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Date of Delivery to the EC:</td>
<td>November 27th, 2014</td>
</tr>
<tr>
<td>Editor(s):</td>
<td>Tatiana Tarasova (SpazioDati)</td>
</tr>
<tr>
<td>Contributor(s):</td>
<td>Dumitru Roman (SINTEF), Phil Archer (W3C/ERCIM), Martin Tuchyna (SAŽP), Giovanni Tummarello (FBK, Sindice), John J O’Flaherty (MAC), Jesús Estrada Villegas (TRAGSA)</td>
</tr>
</tbody>
</table>

**DOCUMENT HISTORY**

<table>
<thead>
<tr>
<th>Version</th>
<th>Version date</th>
<th>Responsible</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>19/11/14</td>
<td>SpazioDati</td>
<td>Draft ready for review</td>
</tr>
<tr>
<td>0.02</td>
<td>21/11/14</td>
<td>SpazioDati</td>
<td>Updates based on input from the Irish and Slovakian pilots and TRAGSA</td>
</tr>
<tr>
<td>0.03</td>
<td>24/11/14</td>
<td>SpazioDati</td>
<td>Annex added for converting Shape files to WKT and proper updates are made to the report</td>
</tr>
<tr>
<td>1.0</td>
<td>27/11/2014</td>
<td>TRAGSA</td>
<td>Editorial review</td>
</tr>
</tbody>
</table>

The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Communities. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.

Copyright © 2014, SmartOpenData Consortium.
Executive Summary

In this deliverable we present D3.3 “Harmonisation of data to SmartOpenData model. Initial Iteration.” We describe a framework that can assist stakeholders in the processes of data harmonisation and publication.

In the initial version of data harmonisation we focus on harmonising tabular data available in one of the text-based formats, such as CSV, TSV, et c, and showcase our framework on the Italian, Spanish and Irish pilots of the framework. We conclude the report by introducing an alternative approach to data harmonisation devised by the Slovakian pilot. This approach complements our framework with the methodology and tools for transforming INSPIRE compliant metadata into RDF via XSL transformations.

1 Introduction

The SmartOpenData (SmOD) project is creating a Linked Open Data infrastructure for integrating and processing public data concerning biodiversity and environment protection in European protected areas. Task T3.3 “Initial Data Harmonisation” concerns the most basic element of the infrastructure, harmonisation of the data sources.

At the time of working on the current deliverable, the INSPIRE model was not available in the Resource Description Framework [RDF] (RDF). However, in the context of the SmOD project, RDF representation of the model is needed for the data transformation and publication processes. We carried out activities on implementing RDF representation of the SmOD model. The resulting SmOD vocabularies are available for all members of the project at https://smod-ontologies.spaziodati.eu/

username: smod
password: EnterSmartOpenData

In section 2 we discuss implementation details of the SmOD vocabularies.
In the rest of the document we present the data harmonisation workflow that consists of three processes: (1) development of the domain-specific vocabularies, (2) creation of mapping for the input data and (3) transformation of the input data into RDF. Figure 1 gives an overview of the workflow.

The SmOD model is a starting point for the pilots of the project. In most of the cases the model needs to be extended to cover particular domains of every pilot. We defined a methodology that can assist stakeholders in implementation of the domain-specific vocabularies in the RDF Schema language [RDFS]. We use the Object-Role Modelling (ORM) techniques to analyse and model the domain of interest. We translate the resulting ORM models into an RDFS vocabulary following a set of rules. The methodology was applied to extend the SmOD model with the terms of the Spanish pilot. We used data from this pilot to explain and exemplify the methodology in Section 3.

Creation of mappings concerns translating input data structures to the target RDF defined by the SmOD model. With the initial data harmonisation we provide support for mapping tabular data available in one of the text-based data formats (such as CSV, TSV, Excel, JSON, XML, RDF/XML, etc.). We rely on the OpenRefine\(^2\) tool with the RDF extension\(^3\) to define the mappings. An instance of OpenRefine was deployed for the project and is available at https://smod-refine.spaziodati.eu/\(^4\) A tutorial on how to use the graphical UI of the extension to create RDF mappings is presented in Section 4.

OpenRefine is primarily a personal desktop application and is meant to be used by a single user. While our intention is to use OpenRefine as a complementing in a batch transformation process. Within the scope of another EU FP7 project, Fusepool\(^5\), a batch

\[^2\]http://openrefine.org/
\[^3\]http://refine.deri.ie/
\[^4\]To access this instance of OpenRefine, use “smod”/“EnterSmartOpenData” credentials.
\[^5\]http://fusepoolp3.github.io/
version of OpenRefine is developed. APIs of the BatchRefine transformer\textsuperscript{6} enable embedding and driving of an OpenRefine engine programmatically. A public instance of BatchRefine is available at http://hetzy1.spaziodati.eu:7100/. We include a demonstration on how to use the BatchRefine transformer in Section 5.

Finally, preliminary results of data harmonisation are summarised in Section 6. We start by describing result of application of the presented data harmonisation workflow to the Italian and Spanish pilots. In addition to this demonstration, we present results on developing of an RDF model for the European Indicator System which is required for the Irish pilot. Pilots demonstration will be concluded with a short description of XSLT driven INSPIRE spatial data and metadata RDF harmonisation based on GeoKnow project\textsuperscript{7}. This exercise serves as an alternative approach towards the common objective to ensure transformation of input data and metadata into the linked open INSPIRE resources based on relevant available knowledge and experience. Methodology has been applied via INSPIRE protected sites data theme to harmonise data for the following SmOD pilot: the Slovakian pilot (led by SAŽP).

2 Implementation of the SmOD Vocabularies

D3.2 [SMODD32] presents the SmOD model at a conceptual using UML\textsuperscript{8}. The model reuses the INSPIRE data model, more specifically the following its elements:

- the Generic Concept Model\textsuperscript{9}
- the Geographic Names Theme\textsuperscript{10}
- the Protected Sites Theme\textsuperscript{11}
- the Land Use Theme\textsuperscript{12}

INSPIRE themes are available in GML\textsuperscript{13}, an XML based encoding standard for geographic information. In the context of the SmOD project, we are interested in the Resource Description Framework [RDF11] (RDF) representation of the themes, in order to use it in pilots in the data transformation and publication processes.

In task T3.3 we carried out activities on implementing RDF representation of the SmOD model. Below we present major implementation details.

Vocabulary Description Language

RDF provides a generic, abstract data model for describing resources using subject, predicate, and objects triples. However, it does not provide any domain-specific terms for describing classes of things in the world and how they are related. This function is served by vocabularies expressed in one of the vocabulary description languages [HB11].

\textsuperscript{6} https://github.com/fusepoolP3/p3-batchrefine
\textsuperscript{7} https://web.imis.athena-innovation.gr/redmine/projects/geoknow_public/wiki/Inspire2RDF
\textsuperscript{8} www.uml.org
\textsuperscript{9} http://inspire.ec.europa.eu/documents/Data_Specifications/D2.5_v3.4.pdf
\textsuperscript{13} http://www.opengeospatial.org/standards/gml
To implement the SmOD vocabularies, we chose the RDF Vocabulary Description Language [RDFS] (also known as the RDF Schema (RDFS language)). Firstly, in a Linked Data context it is always sufficient to express vocabularies in RDFS. Secondly, RDFS is sufficient to fulfil the purpose of the SmOD vocabularies, such as to provide a common shared terminology for publishing in a structured form data from various resources in rural and protected areas.

**Publication**

Neologism\(^1\), a vocabulary publishing platform that supports RDFS, was used to implement the SmOD vocabularies. An instance of Neologism with the SmOD vocabularies is available for the project's partners at [https://smod-ontologies.spaziodati.eu/](https://smod-ontologies.spaziodati.eu/). Neologism provides HTML representation and a diagram view of the vocabularies, as well as Turtle and RDF/XML representations of the vocabularies.

**Namespaces**

As the base namespaces we used the ones suggested in [SMODD32]. Table 1 gives a summary of the SmOD vocabularies.

<table>
<thead>
<tr>
<th>vocabulary</th>
<th>vocabulary URI</th>
<th>prefix</th>
<th>base namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Vocabulary</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 1 SmOD Vocabularies**

**Implementation**

The process of implementation of the SmOD vocabularies was straightforward, as D3.2 [SMODD32] gives clear guidelines on how to define every class and property of the SmOD model in RDF, which vocabularies to re-use and what INSPIRE code lists to use. Here we will not go into implementation details, as D3.2 already does this. We will present generic implementation rules.

We used the basic primitives of RDFS, such as rdfs:Class, rdf:Property, rdf:type, rdfs:range and rdfs:domain\(^1\). The following classes were defined:

- ps:ProtectedSite
- lu:ExistingLandUseObject
- gn:SpellingOfName

\(^1\) [http://neologism.deri.ie/](http://neologism.deri.ie/)

\(^1\) Detailed description of all RDFS primitives can be found in [http://www.w3.org/Talks/1998/0417-WWW7-RDF/slide6-0.htm](http://www.w3.org/Talks/1998/0417-WWW7-RDF/slide6-0.htm) and [http://dublincore.org/workshops/dc6/pp/swick-rdf.ppt#268,6,RDF Model Primitives](http://dublincore.org/workshops/dc6/pp/swick-rdf.ppt#268,6,RDF Model Primitives)
• gn:GeographicalName
• gcm:SpatialObject
• gcm:ReferencableSpatialObject
• gcm:Identifier
• gcm:ObjectWithIdentifier

Human-readable description of the classes were taken from the corresponding INSPIRE specifications.

The ranges of the properties were defined as follows:
• if a property provides values from an existing INSPIRE code list, its range is skos:Concept
• if a property provides text values, its range is rdfs:Literal

Annex B contains a listing of all the classes and properties together with the terms description and domain and range definitions.

JRC publication of RDF

When D3.2 was delivered, the Joint Research Centre (JRC) was in the process of defining RDF of a set of the INSPIRE themes. At the time of writing JRC already released in RDF the INSPIRE Registry, including code lists and code values\(^\text{16}\). Together with this, documentation about the work around RDF and Persistent IDs for INSPIRE was published \([\text{JRC14}]\). This work contains RDF vocabularies\(^\text{17}\) developed in the framework of this study, which includes the following INSPIRE application schemas and themes:

• BU: Buildings
• AM: Area Management Restriction Regulation Zones and Reporting units
• EF: Environmental Monitoring Facilities
• GCM: Generic Conceptual Model
• GN: Geographical Names
• HY: Hydrography
• LC: Land Cover
• SU: Statistical Units
• TN: Transport Networks

The released vocabularies are under editing and reviewing process. They represent test examples of a potential methodology for transforming INSPIRE application schemas in UML into RDF vocabularies. However, the project should consider these test vocabularies, to see whether any amendments are needed to the SmOD vocabularies.


\(^{17}\) A link to download vocabularies \url{https://ies-syn.jrc.ec.europa.eu/attachments/download/483/ARE3NA_D_TD.03_AnnexB_Vocabularies.zip} There four folders with the vocabularies based on the expert who did the job (CP, LVDB, SW).
3 Development of the domain-specific vocabularies

In this initial step regarding Data Harmonisation, the tasks have been related only with TRAGSA datasets. This partner is one of the data providers of Spanish-Portuguese pilot. The datasets from Direçao Geral do Territorio, Portugal, will be added in following steps. During the analysis of the case studies, in particular the TRAGSA data, as part of Spanish-Portuguese pilot, it became evident that significant effort was needed in order to properly understand the data and select relevant subsets for the data publication process. A strong understanding of data was needed in order to generate proper vocabularies and RDFize the data. It turned out that the schemas of the database that TRAGSA has chosen for RDFization have been developed and incrementally extended over long periods of time, and different experts made changes to the schemas. In addition there was very little documentation for the design of the schemas. That meant that data analysis and modelling were needed prior to the creation of relevant vocabularies and RDFization of data.

Object-Role Modelling (ORM) [HM08] was chosen as the modelling and analysis technique for the domain schemas. Object-role models are based on elementary facts, and expressed in diagrams that can be verbalised into natural language. This feature of ORM was essential for communication between the technical partners working on the data publication process and the experts at TRAGSA that had the domain knowledge. Several rounds of domain analysis and data modelling activities took place in collaboration with one of the pilot Spanish-Portuguese pilot providers (TRAGSA) and other relevant partners, in which the relevant data was modelled and fetched from the original databases. This process resulted in a set of documented ORM models that served as input for the creation of the RDFS vocabulary used in the data transformation/publication process. The modelling approach and exercise carried out by in collaboration with the pilot provider proved to be crucial in the understanding of the data publication process for the Spanish pilot.

Below we will illustrate our methodology on the example of the Spanish-Portuguese pilot.

The scope of the conversion is every ORM construct of the first release of the Spanish-Portuguese pilot. As a running example we consider a fragment of the ORM, which is depicted in Fig. 2\textsuperscript{18}.

\textsuperscript{18} The diagram is accessible online at \url{http://smod-fp7.github.io/tragsa/orm/ChemicalCharacteristics.png}
3.1 Mapping rules for object-types

Object-types are abstractions of objects. They are represented as soft rectangles in the figure above: Parcel, Soil and Permeability.

Each object-type is mapped to an rdfs:Class. For example, the object-type Parcel is mapped to the class tragsa:Parcel. The table below lists all object-type to rdfs:Class mappings.

<table>
<thead>
<tr>
<th>ORM object-type</th>
<th>rdfs:Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parcel</td>
<td>tragsa:Parcel</td>
</tr>
<tr>
<td>Animal Species</td>
<td>tragsa:AnimalSpecies</td>
</tr>
<tr>
<td>Protected Site</td>
<td>ps:ProtectedSite</td>
</tr>
<tr>
<td>Forestry Tile</td>
<td>tragsa:ForestryTile</td>
</tr>
<tr>
<td>Habitat</td>
<td>tragsa:Habitat</td>
</tr>
<tr>
<td>Animal Species</td>
<td>tragsa:AnimalSpecies</td>
</tr>
<tr>
<td>Soil</td>
<td>tragsa:Soil</td>
</tr>
<tr>
<td>Permeability</td>
<td>tragsa:Permeability</td>
</tr>
<tr>
<td>Plant Species</td>
<td>tragsa:PlantSpecies</td>
</tr>
</tbody>
</table>

Note, that the object-type Protected Site has been mapped to ps:ProtectedSite, which is defined in the Protected Sites vocabulary.

3.2 Mapping rules for value-types

Value-types are abstractions of values that are used to characterized objects. They are represented as dashed soft rectangles in Fig.2: Rain Fall Level, Weather and Permeability Rate.

Each value-type is captured as rdfs:Literal in the TRAGSA vocabulary.
3.3 Mapping rules for associations

**Associations** are types of relationships between objects. They are represented as connected rectangles that have links to object-types. Each rectangle defines a role of an object-type in the association. Roles may be given names, e.g., *Parcel* plays role “has” in the association with *Soil*.

In the first release of the Spanish-Portuguese pilot (TRAGSA) ORM we have **binary** and **objectified associations**.

### 3.3.1 Mapping rules for named roles of binary associations

**Binary associations** are associations with two roles, such as the association between *Parcel* and *Soil*. Each named role in a binary association is mapped to an RDFS property, *rdf:Property*. For example, the role “has” that *Parcel* plays in the association with *Soil* is mapped to *tragsa:hasSoil*. The table below lists all named roles mappings.

<table>
<thead>
<tr>
<th>ORM binary association's role</th>
<th>rdf:Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>(Parcel)</em> intersects <em>(Protected Site)</em></td>
<td><em>tragsa:intersects</em></td>
</tr>
<tr>
<td><em>(Parcel)</em> is located in <em>(Forestry Tile)</em></td>
<td><em>tragsa:isLocatedIn</em></td>
</tr>
<tr>
<td><em>(Parcel)</em> has <em>(Habitat)</em></td>
<td><em>tragsa:hasHabitat</em></td>
</tr>
<tr>
<td><em>(Habitat)</em> supports <em>(Animal Species)</em></td>
<td><em>tragsa:supports</em></td>
</tr>
<tr>
<td><em>(Animal Species)</em> has <em>(Conservation Status)</em></td>
<td><em>tragsa:hasConservationStatus</em></td>
</tr>
<tr>
<td><em>(Parcel)</em> has <em>(Rain Fall Level)</em></td>
<td><em>tragsa:hasRainFallLevel</em></td>
</tr>
<tr>
<td><em>(Parcel)</em> has <em>(Weather)</em></td>
<td><em>tragsa:hasWeather</em></td>
</tr>
<tr>
<td><em>(Parcel)</em> has <em>(Soil)</em></td>
<td><em>tragsa:hasSoil</em></td>
</tr>
<tr>
<td><em>(Parcel)</em> has <em>(Permeability)</em></td>
<td><em>tragsa:hasPermeability</em></td>
</tr>
</tbody>
</table>

Note 1: In ORM all the roles “has” are considered to be distinct. In RDFS we also created distinct properties with different names, e.g., *tragsa:hasWeather* and *tragsa:hasSoil*.

Note 2: The domain and range of properties were defined by the object-types that participate in the corresponding association. For example, the domain of *tragsa:hasSoil* is the class *tragsa:Parcel*. The range is the class *tragsa:Soil*.

Note 3: When an association connects an object-type with a value-type, such as *(Permeability)* has *(Permeability Rate)*, the range of the corresponding property is *rdfs:Literal*:

*tragsa:hasPermeabilityRate rdfs:range rdfs:Literal*.

### 3.3.2 Mapping rules for objectified associations

**Objectified associations** are associations that become objects and may play roles in other associations. They are depicted as soft rectangles with the name in quotes.

The only objectified association that is present in the first release of the Spanish-Portuguese pilot ORM is the relationship between *Forestry Tile* and *Plant Species* with the name “ForestryTileHasPlantSpecies”. Figure 3 illustrates this association.
The purpose of objectifying this association was to be able to specify *Representative Level* of *Plant Species* in *Forestry Tile*. In other words, objectification allows us to add properties on associations.

To encode objectified associations in RDF, we can adapt classical RDF reification approach within the triple model\(^{19}\) or reification via subgraphs that extends the triple model\(^{20}\). Examples of both approaches and analysis of their pros and cons are given in Annex A.

Reification via subgraphs would be our choice for TRAGSA data in case we would need to implement modifications to “ForestryTileHasPlantSpecies” at a later stage (e.g., addition of the fourth Representative Level). However, we don’t foresee such changes. Considering this and an obvious disadvantage of the verbosity of reification, we implemented an alternative solution for TRAGSA data. We defined different properties to encode associations with different representative levels:

- `tragsa:hasPrimarySpecies`
- `tragsa:hasSecondarySpecies`
- `tragsa:hasTertiarySpecies`

Below is an example of how to encode the following fact in RDF:

```
• *Forestry Tile “157348-MFE25” has Plant Species “Quercus pyrenaica” that has Representative Level “1”.*

:157348-MFE25 tragsa:hasPrimaryPlantSpecies _:primary .
_:primary tragsa:plantSpecies :Quercus_pyrenaica .
_:primary tragsa:density "70" .
```

Note, we use a blank node to be able to attach relevant information on `tragsa:hasPrimaryPlantSpecies`:

\(^{19}\) [http://www.w3.org/TR/2004/REC-rdf-primer-20040210/#reification](http://www.w3.org/TR/2004/REC-rdf-primer-20040210/#reification)

\(^{20}\) [https://dvcs.w3.org/hg/rdf/raw-file/default/rdf-mt/index.html#reification](https://dvcs.w3.org/hg/rdf/raw-file/default/rdf-mt/index.html#reification)
• tragsa:plantSpecies – the species
• tragsa:density – the density of the species in the forestry tile

### 3.4 ORM constraints

The first release of the Spanish-Portuguese pilot ORM model includes the following constraints:

- **internal uniqueness constraints** (depicted as bars over roles)
  
  For example, in Fig.1 a uniqueness constraint is imposed on the association between Soil and Permeability. Which means that each soil has at most one permeability.

- **value constraints** (lists of possible values in braces)
  
  For example, \{“High”, “Medium”, “Low”\} is a value constraint on the value-type Rain Fall Level.

The uniqueness constraint and enumerations are not supported by RDFS.

### 4 Definition of the mappings

OpenRefine is a free, open source powerful tool that helps people to work with messy data using a variety of data cleaning techniques, such as faceting, clustering, editing cells, etc. There is a large community of OpenRefine users\(^2\). In this section we will not cover OpenRefine’s features for manipulating the data, there are many tutorials available\(^2\). Instead, the purpose of the tutorial presented in this section is to guide users in using RDF extension of OpenRefine\(^2\) that adds a graphical UI for exporting the data in RDF.

OpenRefine is a desktop application that can be downloaded, installed\(^4\) and run on your own computer. It runs as a Web server and should automatically open your Web browser to the correct address. Note, that the RDF extension should be downloaded and installed separately\(^5\). An instance of OpenRefine with the RDF extension is available at https://smod-refine.spaziodati.eu/ for the members of the project.

---


\(^2\) For example, video tutorials are available at [http://openrefine.org/](http://openrefine.org/), otherwise, offers a great introduction to OpenRefine. A list of tutorials is given in [https://github.com/OpenRefine/OpenRefine/wiki/External-Resources](https://github.com/OpenRefine/OpenRefine/wiki/External-Resources)

\(^2\) [http://refine.deri.ie/](http://refine.deri.ie/)

\(^4\) [https://github.com/OpenRefine/OpenRefine/wiki/Installation-Instructions](https://github.com/OpenRefine/OpenRefine/wiki/Installation-Instructions)

\(^5\) [http://refine.deri.ie/](http://refine.deri.ie/)
4.1 Example dataset: parcels

As a running example, we will consider a dataset about parcels\textsuperscript{26} from the Spanish pilot. Figure 4 shows first few rows and columns of the dataset.

![Figure 4: OpenRefine tutorial: TRAGSA dataset about parcels](image)

The intended RDF schema of the example dataset is illustrated in Figure 5. This schema is essentially a part of the TRAGSA vocabulary\textsuperscript{27}.

![Figure 5: OpenRefine tutorial: intended RDF schema](image)

In order to translate the example dataset into the intended RDF, we followed the following rules:

1. **Table-to-Class**: each table represents a class in the target RDF;
2. **Row-to-Resource**: each row of a table is a resource, an individual which class is represented by the table;
3. **Column to Property**: each column is a property in the target RDF;
4. **Cell-to-Literal-Value**: each cell with a literal value is rdfs:Literal;
5. **Cell-to-Resource**: each cell with a foreign key constraint is an RDF resource.

\textsuperscript{26} Parcel is a continuous area of land within an agricultural unit with the same agricultural use.

\textsuperscript{27} https://smod-ontologies.spaziiodati.eu/tragsa
Note: While in the example dataset idPhytoCli and idRainFall are foreign keys, we will treat them using the 4th rule. The reason for that is that in the intended RDF the corresponding properties, tragsa:hasWeather and tragsa:hasRainfallLevel, have rdfs:Literal as the domain, and not a resource.

### 4.2 Tutorial for the GUI of the RDF extension of OpenRefine

OpenRefine projects are created by loading data using one of the options available on the “Create Project” tab\(^\text{28}\). For Spanish-Portuguese pilot, using TRAGSA data, we created projects by uploading files (see Fig. 6). For mappings, it is not necessary to upload the whole dataset; a sample of the dataset is enough. We used a sample of the dataset with the first 50000 rows\(^\text{29}\). The project of the running example can be accessed at [https://smod-refine.spaziodati.eu/project?project=1730464948039](https://smod-refine.spaziodati.eu/project?project=1730464948039)

![OpenRefine tutorial: the GUI of the RDF extension](image)

“Edit RDF Skeleton...” command, which can be found under the “RDF” menu, opens a GUI of the RDF extension in a separate window. Figure 7 illustrates the GUI of the example project with TRAGSA parcels.

---

\(^{28}\) The tab is located in the workspace directory of OpenRefine, e.g., [https://smod-refine.spaziodati.eu/](https://smod-refine.spaziodati.eu/)

\(^{29}\) The whole dataset consists of 510522 rows.
The GUI allows one to design a skeleton of the desired RDF. A skeleton is displayed as a node-link diagram. A user defines what resources and literals to include in the RDF graph, what relations to set between them and what URIs to use for resources. The skeleton is applied to every row in the dataset generating a subgraph based on cell's content. The final RDF dataset is the result of merging all the generated subgraphs.

Typically, building an RDF skeleton involves the following steps:

1. set the base URI
2. manage prefixes
3. type instances
4. add properties
5. customise resource nodes
6. customise literal nodes

Below we illustrate the steps using the example of TRAGSA parcels.

### 4.2.1 Set the base URI

The base URI defines the default namespace against which all other URIs will be resolved. It can be set by clicking “edit” near “Base URI” (at the top of the screenshot in Fig. 7). To define
the base URI, we followed the URI pattern discussed in [SMODD32] in Section 5 “URI Construction”:

http://{domain}[/collection*]//[so]/[/{class}]/[inspireLocalId]/[/{inspireVersionId}]

<table>
<thead>
<tr>
<th>#</th>
<th>URI part</th>
<th>meaning</th>
<th>TRAGSA value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{domain}</td>
<td>the internet domain named</td>
<td>data.smartopendata.eu</td>
</tr>
<tr>
<td>2</td>
<td>{collection}</td>
<td>any number of segments that provide hints to humans as to the dataset to which the identified object belongs</td>
<td>SpanishPilot/Tragsa</td>
</tr>
<tr>
<td>3</td>
<td>/[so]</td>
<td>the literal string /so hinting that this URI identifies a spatial object</td>
<td>so/</td>
</tr>
<tr>
<td>4</td>
<td>/[class]</td>
<td>an optional segment describing the class of spatial objects, e.g., ProtectedSite</td>
<td>depends on the data being mapped</td>
</tr>
<tr>
<td>5</td>
<td>{inspireLocalId}</td>
<td>the local identifier for the site that SHOULD be included and SHOULD match the value of the gcm:localID property of gcm:Identifier</td>
<td>depends on the data being mapped</td>
</tr>
<tr>
<td>6</td>
<td>/[inspireVersionId]</td>
<td>the optional segment giving the version ID</td>
<td>not present in the TRAGSA data</td>
</tr>
</tbody>
</table>

Table 2 OpenRefine tutorial: URI construction

The first three parts of the URI in the “TRAGSA value” column constitute the base URI of the Spanish-Portuguese pilot data:

http://data.smartopendata.eu/SpanishPilot/Tragsa/so/

Other parts depends on the data being mapped and will be generated automatically.
4.2.2 Manage prefixes

The RDF extension can import RDF vocabularies available on the Web. This can be done via “add prefixes” and “manage prefixes” buttons. Built-in vocabularies include:

<table>
<thead>
<tr>
<th>prefix</th>
<th>URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>rdfs</td>
<td><a href="http://www.w3.org/2000/01/rdf-schema#">http://www.w3.org/2000/01/rdf-schema#</a></td>
</tr>
<tr>
<td>foaf</td>
<td><a href="http://xmlns.com/foaf/0.1/">http://xmlns.com/foaf/0.1/</a></td>
</tr>
<tr>
<td>xsd</td>
<td><a href="http://www.w3.org/2001/XMLSchema#">http://www.w3.org/2001/XMLSchema#</a></td>
</tr>
<tr>
<td>owl</td>
<td><a href="http://www.w3.org/2002/07/owl#">http://www.w3.org/2002/07/owl#</a></td>
</tr>
<tr>
<td>rdf</td>
<td><a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#">http://www.w3.org/1999/02/22-rdf-syntax-ns#</a></td>
</tr>
</tbody>
</table>

In addition to the default vocabularies, we imported the GCM vocabulary of the SmartOpenData model (see Section 2) and the TRAGSA vocabulary (see Section 3).

<table>
<thead>
<tr>
<th>prefix</th>
<th>URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcm</td>
<td><a href="http://smod-ontologies.spaziodati.eu/gcm#">http://smod-ontologies.spaziodati.eu/gcm#</a></td>
</tr>
<tr>
<td>tragsa</td>
<td><a href="http://smod-ontologies.spaziodati.eu/tragsa#">http://smod-ontologies.spaziodati.eu/tragsa#</a></td>
</tr>
</tbody>
</table>

4.2.3 Type instances

Every row in our dataset corresponds to an instance of tragsa:Parcel. We explicitly defined this using “add rdf:type”, see screenshot in Fig. 8. Autocomplete support for imported ontologies is provided.

Figure 8 OpenRefine tutorial: instance typing
4.2.4 Add properties

We used “add property” button to add properties to parcel. As per the intended RDF (see Fig. 5), we added the following properties:

<table>
<thead>
<tr>
<th>#</th>
<th>property</th>
<th>rdfs:range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>gcm:identifier</td>
<td>gcm:Identifier</td>
</tr>
<tr>
<td>2</td>
<td>tragsa:isLocatedIn</td>
<td>tragsa:ForestryTile</td>
</tr>
<tr>
<td>3</td>
<td>tragsa:hasHabitat</td>
<td>tragsa:Habitat</td>
</tr>
<tr>
<td>4</td>
<td>tragsa:hasRainfallLevel</td>
<td>rdfs:Literal</td>
</tr>
<tr>
<td>5</td>
<td>tragsa:hasWeather</td>
<td>rdfs:Literal</td>
</tr>
</tbody>
</table>

Properties #1-3 link the root node (tragsa:Parcel) to another resource. Properties #4-5 link the root node to literals. Below we will explain how to customise resource and literal nodes.

4.2.5 Customise resource nodes

Figure 9 shows a node dialog of customising the parcel URI, the root node.

![RDF Node](image)

Figure 9 OpenRefine tutorial: customising nodes

Nodes values are either constants or expressions based on cells' content. OpenRefine has an expression language called GREL\(^{31}\) which can be used to define custom values based on cells' content from the imported vocabularies. For this, type the full (e.g., https://smod-ontologies.spaziiodati.eu/gcm#identifier) or prefixed name of the property (e.g., gcm:identifier). To define properties in the default namespace specified by the base URI, type “:” followed by the name of the property (e.g., :identifier).

---

\(^{30}\) Properties can be added from the imported vocabularies. For this, type the full (e.g., https://smod-ontologies.spaziiodati.eu/gcm#identifier) or prefixed name of the property (e.g., gcm:identifier). To define properties in the default namespace specified by the base URI, type “:” followed by the name of the property (e.g., :identifier).
content. It is important to note that expressions that produce errors or evaluate to empty strings are ignored.

To build the parcel URI we:

- used content from the column idParcel
- chose “as a URI” option to
- used custom expression

“preview/edit” opens a window (see Fig. 10) in which one can use GREL to define the custom expression and see the first 10 rows of values.

![Preview URI values](image)

The GREL expression we wrote for the parcel URI is:

```
"Parcel/" + value
```

In the preview one can see:

- row – the row number
- value – the original value of idParcel (e.g., to “470137939” of the first row)
- “Parcel/” + value - value after applying GREL expression to the original value
- resolved against the base URI – value that will appear in the final RDF; for example, the final URI that will be generated for the first row is:

```
http://data.smartopendata.eu/SpanishPilot/Traagselo/Parcel/470137939
```

---

In Table 1 we presented the parts of the INSPIRE URIs and claimed that \[/{class}\] and \{inspireLocalId\} depend on the data being mapped. Essentially, our GREL expression filled these parts in:

- “Parcel/” \[\[/{class}\]\]
- value \[\{inspireLocalId\}\]

Below we summarise information about customisation of the resource nodes of the RDF skeleton of the running example:

<table>
<thead>
<tr>
<th>class</th>
<th>used column</th>
<th>cell's content</th>
<th>GREL expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>tragsa:Parcel (root node)</td>
<td>idParcel</td>
<td>as a URI</td>
<td>&quot;Parcel/&quot; + value</td>
</tr>
<tr>
<td>gcm:Identifier</td>
<td>idParcel</td>
<td>as a URI</td>
<td>&quot;Identifier/&quot; + value</td>
</tr>
<tr>
<td>tragsa:ForestryTile</td>
<td>idForestry</td>
<td>as a URI</td>
<td>&quot;ForestryTile/&quot; + value</td>
</tr>
<tr>
<td>tragsa:Habitat</td>
<td>idHabitat</td>
<td>as a URI</td>
<td>&quot;Habitat/&quot; + value</td>
</tr>
</tbody>
</table>

### 4.2.6 Customise literal nodes

To customise literals, one uses the same node dialog presented in Fig. 9. Below we summarise information on customisation of the literal nodes:

<table>
<thead>
<tr>
<th>values of</th>
<th>used column</th>
<th>cell's content</th>
<th>GREL expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>tragsa:hasWeather</td>
<td>idPhytoCli</td>
<td>as text</td>
<td>If (value == &quot;10&quot;, &quot;VI (IV) 2 Wet genuine nemoromediterranean&quot;, if (value == &quot;14&quot;, &quot;VI(V) cool-warm genuine Nemoral&quot;, if (value == &quot;15&quot;, &quot;VI fresh genuine Nemoral&quot;, None)) )</td>
</tr>
<tr>
<td>tragsa:hasRainfallLevel</td>
<td>idRainFall</td>
<td>as text</td>
<td>if (or(value == &quot;1&quot;, value == &quot;2&quot;), &quot;Low&quot;, if (or(or(value == &quot;3&quot;, value == &quot;4&quot;), value == &quot;5&quot;), &quot;Medium&quot;, if (or(value == &quot;6&quot;, value == &quot;7&quot;), &quot;High&quot;, None)))</td>
</tr>
</tbody>
</table>

The GREL expressions we defined mean to output text values of the “weather” and “rainfall level” instead of their ids.

Note, that literals can either be plain (as in our running example) or typed using xsd\(^{32}\) or custom datatypes.

\(^{32}\) [http://www.w3.org/2001/XMLSchema#](http://www.w3.org/2001/XMLSchema#)
5 Data transformation

OpenRefine allows users to export data in various formats (e.g., HTML table and Excel). The RDF extension adds the possibility to output data in RDF in one of the serialization formats: RDF/XML or Turtle. This can be done using the “Export” menu. While this option is helpful to verify the resulting RDF and can be used to perform one-time transformation, we are interested in using OpenRefine to perform data transformation in a batch mode.

In this section we discuss the usage of the BatchRefine APIs for embedding and driving the OpenRefine engine programmatically. BatchRefine is developed by another EU FP7 project, Fusepool P3, as a batch version of OpenRefine which exposes standard REST APIs for a stateless transformation service.

BatchRefine can be deployed and built in two ways: with Docker and from sources. There are also two public endpoints with BatchRefine available:

- [http://hetzy1.spaziodati.eu:7200](http://hetzy1.spaziodati.eu:7200) to perform synchronous transformations
- [http://hetzy1.spaziodati.eu:7200](http://hetzy1.spaziodati.eu:7200) to perform asynchronous transformations

In its simplest form BatchRefine is accessed via a POST request with:

**Headers**
Content-Type: text/csv;charset=utf-8 (or another charset)
Accept: text/csv | text/turtle | application/rdf+xml

**Parameters**
refinejson=URI, where the URI is the location of an OpenRefine JSON transform or mapping rule.

**Content**
The input data as text.

5.1 BatchRefine Example using cURL

Below is the POST request using cURL to perform transformation of the complete TRAGSA dataset with parcels, pd_0600_smod_cartography.csv. We run the request on a personal machine. Currently, BatchRefine assumes that the input file is available locally, while the
JSON file with OpenRefine mappings must be available online. We served the required mapping file at [http://d3s.disi.unitn.it/~mega/tragsa-transform.json](http://d3s.disi.unitn.it/~mega/tragsa-transform.json).

```
curl
  -X POST
  -H "Content-Type:text/csv"
  -H "Accept:text/turtle"
  --data-binary @pd_0600_smod_cartography.csv
```

The JSON file with the mappings was extracted using the GUI of OpenRefine: press “Extract” on the “Undo/Redo” tab and copy the mappings from the “Extract Operation History” window.

## 6 Pilots Demonstration

In this section we present preliminary results on harmonising data from the project's pilots.

### 6.1 ARPA pilot

For the Italian pilot we worked together with ARPA to transform data about protected areas in Italy to the SmOD vocabularies (discussed Section 2). The data was downloaded from the portal [http://www.datopen.it/](http://www.datopen.it/), an Italian portal of Open Data, in a CSV format. The dataset contains Italian sites of community importance and special areas of conservation that became part of the Natura2000 network. We created an OpenRefine project with this dataset [https://smod-refine.spaziodati.eu/project?project=2142201009574](https://smod-refine.spaziodati.eu/project?project=2142201009574) and defined mappings using the RDF extension of OpenRefine. The mappings can be extracted from the project.

The RDF of the ARPA pilot is available at [https://s3-eu-west-1.amazonaws.com/smod-repo/ARPA-protected-areas.rdf](https://s3-eu-west-1.amazonaws.com/smod-repo/ARPA-protected-areas.rdf). Below is a listing of the RDF representation of the protected site “Rupi di Catalfano e Capo Zafferano” in Sicily:

```plaintext
@prefix gn: <http://smod-ontologies.spaziodati.eu/gn#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix geo: <http://www.opengis.net/ont/geosparql#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix gcm: <http://smod-ontologies.spaziodati.eu/gcm#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
```

37 To download the data, go to [http://www.datopen.it/it/opendata/Siti_di_Importanza_Comunitaria_SIC__3071](http://www.datopen.it/it/opendata/Siti_di_Importanza_Comunitaria_SIC__3071), tab “Scarica”

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix vcard: <http://www.w3.org/2006/vcard/ns#> .
@prefix ps: <http://smod-ontologies.spaziodati.eu/ps#> .

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/ITA020019> a ps:ProtectedSite .

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/Geometry/ITA020019> a geo:Geometry ;
    geo:asWKT "POINT (38.1040110252 13.5227466722)"^^geo:wktLiteral .

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/ITA020019> geo:hasGeometry

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/siteName/ITA020019> a gn:GeographicalName ;
    gn:nameStatus <http://inspire.ec.europa.eu/codelist/NameStatusValue/official> ;
    gn:sourceOfName "Ministero dell'ambiente e della tutela del territorio e del mare" .

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/SpellingOfName/ITA020019> a gn:SpellingOfName ;
    gn:text "Rupi di Catalfano e Capo Zafferano"@it , "Latin" .

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/identifier/ITA020019> a gcm:Identifier ;
    gcm:localId "ITA020019" .

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/Identifier/ITA020019> ps:legalFoundationDocument

Note, that the value of ps:legalFoundationDocument is the FTP URI. Another option would be to represent legal foundation document as literal:

<http://data.smartopendata.eu/ItalianPilot/Arpa/so/ProtectedSite/Natura2000GR/ITA020019> ps:legalFoundationDocument
However, this will not be compatible with the definition ps:legalFoundationDocument in the Protected Sites vocabulary, which has foaf:Document as a range. Hence, to adapt the second solution, either the range of the property must be changed to rdfs:Literal or another suitable property must be used. For example, one can use dcterms:source from the Dublin Core Terms vocabulary\(^{39}\) that defines a related resource from which the described resource is derived but does not restrict it to be a resource.

Another note to add is about the range of ps:legalFoundationDate. While in D3.2 [SMODD32] it was already argued that xsd:date is more appropriate than xsd:dateTime as a value of the ps:legalFoundationDate, we found that even the day is not present in the ARPA dataset, but the year and the month of foundation. While we generated a valid xsd:date for the dataset (e.g., "2012-01-01"^^xsd:date in the listing above), the original value was “201210”, which is the year “2010” and the month “10” of foundation. Another solution could be to modify the range of ps:legalFoundationDate to be xsd:gYearMonth.

### 6.2 Spanish-Portuguese pilot

Work on the Spanish-Portuguese pilot was carried out in collaboration with the domain experts from TRAGSA. What differs this effort from the ARPA pilot is that we had to extend the SmOD vocabularies with terms of the TRAGSA data. For this, we used the methodology described in Section 3. The resulting vocabulary is published at [https://smod-ontologies.spaziodati.eu/tragsa#](https://smod-ontologies.spaziodati.eu/tragsa#).

The original data is in the Microsoft Access database. We exported the following tables from the database in the CSV format:

- aux_00020100_forestryMap – data on forestry tiles
- aux_00020400_InventoryTerrestrialAnimalSpecies – data on animal species
- aux_00020300_habitatsDirective – data on habitats
- aux_00030800_lithology – data on soil
- aux_00031000_lithostratigraphy_permeability – data on soil permeability
- aux_00030900_pluviometry – data on rainfall level
- aux_00030600_PhytoClima - data on weather conditions
- pd_0600_smod_cartography – main table with parcels and links to other tables

Basically, every table represents entities of certain classes, except for the 6\(^{th}\) and 7\(^{th}\), which will provide literal values to the corresponding properties of parcels. Refer to Fig. 5 to understand relationship between different classes.

For every table we created an OpenRefine project:

- forestry map - [https://smod-refine.spaziodati.eu/project?project=1482201380897](https://smod-refine.spaziodati.eu/project?project=1482201380897)
- animal species - [https://smod-refine.spaziodati.eu/project?project=1755797335258](https://smod-refine.spaziodati.eu/project?project=1755797335258)

\(^{39}\) [http://dublincore.org/documents/2012/06/14/dcni-terms/](http://dublincore.org/documents/2012/06/14/dcni-terms/)
• habitats - https://smod-refine.spaziodati.eu/project?project=1709540331358
• soil - https://smod-refine.spaziodati.eu/project?project=2155351922577
• soil permeability - https://smod-refine.spaziodati.eu/project?project=1421981817600
• parcels - https://smod-refine.spaziodati.eu/project?project=1730464948039

Note, that for parcels we created a sample of the dataset with the first 50000 rows. However, we used the whole dataset to transform the data using the BatchRefine API. We didn’t make separate projects for weather conditions and rainfall level, as we used their values in the mappings of parcels.

RDF representation of the TRAGSA data can be downloaded from the following links:
• forestry map - https://s3-eu-west-1.amazonaws.com/smod-repo/VERSION1-tragsa-forestryMap.rdf
• animal species - https://s3-eu-west-1.amazonaws.com/smod-repo/VERSION1-tragsa-animalSpecies.rdf
• habitats - https://s3-eu-west-1.amazonaws.com/smod-repo/VERSION1-tragsa-habitatsDirective.rdf
• soil - https://s3-eu-west-1.amazonaws.com/smod-repo/VERSION1-tragsa-lithology.rdf
• soil permeability - https://s3-eu-west-1.amazonaws.com/smod-repo/VERSION1-tragsa-lithostratigraphy_permeability.rdf
• parcels - https://smod-refine.spaziodati.eu/project?project=1730464948039

6.3 MAC pilot

The Irish pilot has different characteristics to the others in that at present there is no data. Rather, there is a system for collecting data that is being rolled out across Europe – the European Tourism Indicator System, ETIS 40. This was developed by the Commission as a follow-up to the Communication “Europe, the world’s No 1 tourist destination – a new political framework for tourism in Europe”41 for Sustainable Management at Destination Level. This is a comprehensive system, simple to use, flexible and suitable for all tourism destinations42. The European Indicator System aims to contribute to improving the sustainable management of destinations by providing tourism stakeholders with an easy and useful toolkit. It helps stakeholders to measure and monitor their sustainability management processes, and enable them to share and benchmark their progress and performance in the future.

Quoting from the ETIS implementation guide:

"It is designed as a locally owned and led process for monitoring, managing, and enhancing the sustainability of a tourism destination. It has been developed as a result of lessons learned from previously existing Indicator System initiatives and fine-tuned as a result of feedback collected from field testing, in a number of different destinations in Europe."

The system largely comprises an Excel Spreadsheet and a PDF guide – the ETIS Toolkit. It is designed to allow managers of different tourist destinations to gather data using the familiar Excel tool and then use visualisations from that to inform management decisions. Any comparison between destinations would be done manually. However, one aim of the Irish pilot is to enable the managers of the Burren Geopark to compare indicators with other Geoparks. In order to help facilitate this, SmartOpenData is working to make the data collected in the Excel sheets available as Linked Data by providing an ETIS web service and app.

The ETIS web service will enable the Burren GeoPark initially (during the WP5 pilots and all other GeoParks subsequently) to use the mobile App to:

(i) Set up their destination with suitable indicators and targets (by its Local Destination Co-ordinator and Stakeholder Working Group).
(ii) Provide online data collection by each stakeholder group (including Destination management, Enterprise, Resident and Visitor Surveys) – this will include automatic updating from appropriate online source databases.
(iii) Review progress and results achieved to date at their destination by Monitoring Results and Charting Destination, Enterprises, Residents, Visitors Impressions, Spending and Time – this will include automatic geographic visualisation by linking to appropriate Geospatial data sources. This will enable the Stakeholder Working Group and visualisation by the various stakeholders to provide an ongoing community “crowdsourcing verification” that the results and data being entered matches the perceptions of the various stakeholders.
(iv) Provide benchmarking with other destinations (e.g. other GeoParks) through each of these views and access to their linked open datasets.

It is anticipated that as each destination’s use of the ETIS matures and the linked open indicator data collected becomes more extensive, the web service will enable open comparisons of the destination’s progress against international benchmarks. This will give greater context to the achievements and give destination stakeholders motivation to take further actions to improve results. It will also encourage knowledge sharing between destinations. The intention is not to create competition between destinations, but to recognise that the results generated through the process are core to the decision making plans for each destination.

So, taking the ETIS toolkit as input, we want to derive a data model that can be represented in RDF. As far as possible, this will reuse the SmOD core data model which itself is based on INSPIRE. In line with best practice, existing classes and properties will be re-used wherever possible. Unfortunately, few suitable classes and properties are already defined.
6.3.1 ETIS & INSPIRE

The ETIS toolkit has been designed separately from the INSPIRE standards and makes no reference to it. However, an ETIS Destination is, or can be thought of as, a spatial object. By doing so, we provide hooks between INSPIRE compliant data and tourism data. Since an aim of ETIS is to aid sustainable management of tourist resources, this is clearly a useful feature. However; the separate design leads to some unfortunate mismatches in the model. For example, ETIS provides a list of habitat types as potential values for a Destination's dominant habitat type. These do not match the ones used in INSPIRE although we can hope to make some semantic connections.

6.3.2 W3C namespaces, persistence & future management

The intention is to publish the schema and associated SKOS Concept schemes on w3.org. The persistence policy at W3C and the neutrality of the organisation, maximises the usefulness and reliability of the effort. However, it is important to note that this does NOT make the schema a W3C standard, nor does it imply endorsement of any kind by W3C or its members. Nevertheless, the hope is that the work will be useful beyond the SmartOpenData pilots. W3C's Community Group infrastructure provides the means (and licensing conditions) under which the schema can be improved and extended by others. Such a Community Group can be formed, without cost, at any time by a group of at least 5 interested individuals or organisations.

6.3.3 The ETIS Data Model

What follows is a best effort to render the ETIS Toolkit as a data model and associated RDF schema. The model is still under construction and it is near-certain that it will need to be adapted in the light of experience, but the key aspects are presented here. Figure 14 shows the model as a UML diagram.

---

43 [http://www.w3.org/Consortium/Persistence](http://www.w3.org/Consortium/Persistence)
44 [http://www.w3.org/community/](http://www.w3.org/community/)
Figure 11  The proposed data model for ETIS
6.3.3.1 Namespaces used

<table>
<thead>
<tr>
<th>Namespace</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>etis</td>
<td><a href="http://www.w3.org/2014/11/etis#">http://www.w3.org/2014/11/etis#</a></td>
</tr>
<tr>
<td>gcm</td>
<td><a href="http://inspire.jrc.ec.europa.eu/schemas/gcm/3.0/">http://inspire.jrc.ec.europa.eu/schemas/gcm/3.0/</a></td>
</tr>
<tr>
<td>ps</td>
<td><a href="http://inspire.jrc.ec.europa.eu/schemas/ps/3.0/">http://inspire.jrc.ec.europa.eu/schemas/ps/3.0/</a></td>
</tr>
<tr>
<td>fc</td>
<td><a href="http://inspire.ec.europa.eu/featureconcept/">http://inspire.ec.europa.eu/featureconcept/</a></td>
</tr>
<tr>
<td>geonames</td>
<td><a href="http://www.geonames.org/ontology#">http://www.geonames.org/ontology#</a></td>
</tr>
<tr>
<td>org</td>
<td><a href="http://www.w3.org/ns/org#">http://www.w3.org/ns/org#</a></td>
</tr>
<tr>
<td>skos</td>
<td><a href="http://www.w3.org/2004/02/skos/core#">http://www.w3.org/2004/02/skos/core#</a></td>
</tr>
<tr>
<td>dcterms</td>
<td><a href="http://purl.org/dc/terms/">http://purl.org/dc/terms/</a></td>
</tr>
<tr>
<td>foaf</td>
<td><a href="http://xmlns.com/foaf/0.1/">http://xmlns.com/foaf/0.1/</a></td>
</tr>
<tr>
<td>geo</td>
<td><a href="http://www.opengis.net/ont/geosparql#">http://www.opengis.net/ont/geosparql#</a></td>
</tr>
<tr>
<td>fc</td>
<td><a href="http://inspire.ec.europa.eu/featureconcept/">http://inspire.ec.europa.eu/featureconcept/</a></td>
</tr>
<tr>
<td>schema</td>
<td><a href="http://schema.org/">http://schema.org/</a></td>
</tr>
<tr>
<td>transit</td>
<td><a href="http://vocab.org/transit/terms/">http://vocab.org/transit/terms/</a></td>
</tr>
</tbody>
</table>

Table 3 Namespaces used in Irish pilot

6.3.3.2 Class etis:Destination

This is defined as a sub class of gcm:spatialObject. It might also be appropriate to assert that it’s a Protected Site or any other INSPIRE type. This is the key to interoperability with INSPIRE data since we can use that system’s structures to provide geometries, identifiers, names etc.

6.3.3.2.1 Object Properties

geonames:parentCountry

The geonames vocabulary has a useful property for asserting that a Destination is within a country. It could also be done with geo:ehContains but the geonames one looks closer in intent. The EU Publications Office MDR provides an authoritative, multilingual list of countries as a SKOS concept scheme at http://publications.europa.eu/mdr/authority/country/

Note that use of this property implies that a Destination is a geonames:Feature

etis:dominantHabitat

ETIS provides an enumerated list of values that will be rendered here as a SKOS Concept Scheme. Sadly there is no direct linkage with INSPIRE as this would have been particularly useful for finding all relevant data concerning given habitat types, whether from the tourism perspective or others. Where possible, these will be linked (with broader/narrower relationships) with the INSPIRE habitats and biotopes list.

etis:biodiversityLevel
This will point to the relevant SKOS Concept Scheme giving the level of biodiversity (the range of different plant and animal species). Values are simply low, medium and high.

**geo:ehContains**

The ETIS model for a Destination provides a place holder for the 5 top attractions within the Destination. These attractions are also to be modelled as Destinations so that the geo:ehContains property can be used to link a Destination to the attractions within it.

The underlying assumption here is that that data about each of the attractions will be sufficient to compute the top 5 attractions dynamically, thus providing more flexibility than hard coding the order of their popularity into the data.

**etis:hasWeatherPattern**

Links a Destination to its Average Seasonal Weather pattern class.

**etis:hasTransportLinks**

Links a destination to a TransportLinks class.

**etis:hasAccommodation**

Links a Destination to information about local accommodation, including hotels, camp sites etc.

**etis:hasPopulationInfo**

Links a Destination to information about the local and tourist populations.

**etis:hasTouristData**

Links a destination to information about tourists.

### 6.3.3.2 Datatype Properties

**dcterms:description**

The general (free text) description of the destination

**etis:noKmFromCapital**

This could be calculated dynamically if the centroid of the capital were known but since Destinations and capitals tend to stay in their relative positions, a simple record of the result of calculation seems appropriate.
etis:tourismOverview

Free text description of the main types of tourism and most popular activities.

etis:geographicalDescription

Free text giving the approximate size of the Destination (in km²) and principle physical characteristics (rivers, hills, bays etc).

etis:nearbyAirport

A plain text description of nearby local and international airport(s) serving the Destination. Processable data about local airports is provided via the Transport Links etc.

6.3.3.3 Class etis:PopulationInfo

INSPIRE's Population Distribution – Demography theme provides a very detailed model for recording population statistics. The PD theme does not require that the area identified is an INSPIRE Spatial Object and, since the model is detailed, is likely to be more complex than is necessary or available for ETIS data. Therefore simple datatype properties are used to record the population numbers.

6.3.3.3.1 Datatype Properties

etis:permanentPopulation

The number of people living in the Destination

etis:populationDensity

This could be calculated from other information available but even when that is possible, it's easier to have it recorded as a number. This should be recorded as population per km²

etis:peakSeasonResidentTouristRatio

The ratio of permanent residents to tourists at peak season

etis:percentageEmployedInTourism

The percentage of the population employed in tourism-related jobs.

6.3.3.4 Class etis:TouristData

This is a class for representing information about tourists visiting a Destination. The dcterms:temporal property can be used to indicate the period for which the data applies, such as a year.
6.3.3.4.1 Object Properties

**etis:visitorOrigin**

Links a Tourist Data class to information about tourists from a specific location, such as a country.

**etis:visitorTransport**

Links a Tourists class to information about tourists arriving at the Destination via a specific form of transport.

6.3.3.4.2 Datatype Properties

**etis:totalAnnualVisitors**

The annual total number of visitors.

**etis:annualDayVisitors**

The total number of people visiting the destination for a day without an overnight stay.

**etis:annualDomesticVisitors**

The total number of annual domestic visitors (I think this is under-defined!)

**etis:avDailySpend**

The tourist average daily spend. This value should be a typed literal (to give currency information).

**etis:avLengthOfStay**

The average length of stay at the destination in days.

**dcterms:temporal**

If the data applies to a single year or other period, use this property to indicate which.

6.3.3.5 Class **etis:VisitorsByOrigin**

This is a class representing the number of visitors from a specific location, usually a country.
6.3.3.5.1 Object Properties

etis:countryOfOrigin

Links instances of Visitors By Origin to their origin, usually a country, expressed as a SKOS Concept.

6.3.3.6 Class etis:VisitorsByTransport

A class representing the number of visitors arriving at the destination using a specific mode of transport.

6.3.3.6.1 Object Properties

etis:transportUsed

Links instances of Visitors By Transport to their specific transport, expressed as a SKOS Concept.

6.3.3.6.2 Datatype Properties for Origin & Transport

etis:noVisitorsByType

The number of visitors associated with a specific origin, mode of transport etc.

6.3.3.7 Class etis:AverageSeasonalWeatherPattern

The etis:AverageSeasonalWeatherPattern class has 8 datatype properties, all of which take typed literal values, expected to be integers, and are self-explanatory.

6.3.3.7.1 Datatype Properties

etis:hoursSunshineSummer
etis:hoursSunshineWinter
etis:mmPrecipitationSummer
etis:mmPrecipitationWinter
etis:dayTempCelciusSummer
etis:nightTempCelciusSummer
etis:dayTempCelciusWinter

etis:nightTempCelciusWinter

6.3.3.8 Class fc:TransportPoint

This is the INSPIRE feature concept of “A point spatial object - which is not a node - that represents the position of an element of a transport network.” Examples include bus and railway stations, airports etc.

6.3.3.8.1 Object Properties

foaf:homepage

The URL of the airport's homepage

6.3.3.8.2 Datatype properties

schema:iataCode

The common code for the airport, e.g. LHR, DUB etc. Note that schema.org also has icaoCode if needed

foaf:name

The name of the airport

6.3.3.9 Class etis:TransportLinks

A class to represent the transport links that serve a Destination.

6.3.3.9.1 Object Properties

etis:hasOperator

Links a TransportLinks class to a class representing an operator of a local service.

6.3.3.9.2 Datatype Properties

etis:transportDescription

Free text giving general information about local transport and/or private transport that is available. This fulfils the role in the ETIS toolkit of “Additional private arrangements for
transporting visitors to the destination” and can also handle any transport descriptions that
don’t fit the more regular format of etis:Operator and etis:transportServiceType.

6.3.3.10 Class etis:Carrier

Sub class of schema:Organization and foaf:Organization (and therefore org:Organization).

An airline, cruise, ferry, railway, bus, tram operator etc. that operates a service to the
destination.

The ETIS toolkit makes a distinction between airlines and cruise operators, and other more
local forms of transport. If this is useful in the model then we might want to use the transit
vocabulary which is based on the Google's GTFS, see http://vocab.org/transit/terms/.html.
That has concepts of routes, stations, station stops etc. It can be added if the use cases
demand but my feeling is that the simpler case might be sufficient.

6.3.3.10.1 Object Properties

etis:transportServiceType

Links an etis:Carrier to a SKOS Concept that describes the type of service offered, such as
train, metro, tram etc.

foaf:homepage

The URL of the carrier’s homepage

6.3.3.10.2 Datatype properties

skos:prefLabel

The name of the organisation offering Accommodation is given using skos:prefLabel in
conformance with the ORG Ontology.

dcterm:definition

A free text description of the services offered e.g. “2 flights daily from Munich”, “Bus route
34 runs from the station and drops you right outside the museum”

schema:iataCode

If the Carrier is an airline, use this property to provide its IATA code, e.g. BA, EI etc.

etis:Accommodation
A class representing any accommodation within or near the Destination, such as a hotel, camp site etc. It is a sub class of org:Organization.

6.3.3.10.3  Object Properties

etis:accommodationType

Links an instance of the Accommodation class to its type (3 star hotel, Guest House etc.) which is expressed as a SKOS Concept.

6.3.3.10.4  Datatype Properties

etis:approximateCapacity

The number of beds in the hotel, pitches at the camp site etc.

skos:prefLabel

The name of the organisation offering Accommodation is given using skos:prefLabel in conformance with the ORG Ontology.

6.4  SAZP pilot

As stated in the executive summary, alternative approach to execute the transformation of input INSPIRE protected sites from Slovakia including INSPIRE compliant metadata into the RDF encoding via XSLT transformation took place within the Environmental data reuse pilot.

6.4.1 Background

SAZP/SEA generally keeps geodata in database solutions PostgreSQL/PostGIS or ArcSDE/Oracle. Some datasets are stored also in ESRI file geodatabase or ESRI shape files. Publishing INSPIRE datasets (e.g. INSPIRE Protected sites) is backed by Geoserver solution, capable of reading many types of geo-datasources and transforming source data into GML 3.2.1 INSPIRE application schemas on-the-fly. Such INSPIRE datasets is served through Geoserver OGC WFS (INSPIRE Download service) interface. INSPIRE metadata is published through OGC CSW (INSPIRE Discovery service) interface implemented by PyCSW open-source solution.

46 http://inspire.geop.sazp.sk/ows?service=wfs&version=2.0.0&request=GetCapabilities
47 http://csw.sazp.sk/?service=CSW&version=2.0.2&request=GetCapabilities
6.4.2 Linked data

Publishing of Linked data have been based on methodology and tools described in document [GEOKNOW-D2.7.1][48], which is centred about idea of transformation INSPIRE GML datasets into respective RDF representation via XSLT transformation, suitable for Linked data applications.

**Process of publishing INSPIRE linked data is comprised by following steps:**

- Transformation tools (GML, XML/CSW to RDF) and triple store installation
- Transformation setup & customization
- Transformation of source INSPIRE GML datasets into RDF representation
- Import of INSPIRE RDF datasets into triple store
- Publish INSPIRE RDF datasets through GeoSPARQL interface
- Publishing INSPIRE metadata via SPARQL interface

**Software tools used:**

- TripleGeo – transformation tool supporting INSPIRE GML to RDF transformation
- TripleGeo-CSW – middleware tool for translating SPARQL requests into OGC CSW requests presenting responses in RDF
- Parliament – triple store and SPARQL endpoint implementation

6.4.3 Transformation tools (GML XML/CSW to RDF) and triple store installation

Target OS platform: Ubuntu server 14.04 LTS (64bit)

**TripleGeo**

This is an Open-source standalone GML-RDF transformation tool implemented in JAVA. Latest version of the tool was deployed directly from official git repository [49].

**Parliament**

Open-source triple store server supporting GeoSPARQL implemented in JAVA/C++. Version 2.7.4 was downloaded [50] and deployed according to instructions in chapter „2.4 Alternate Server Configurations“ of user guide [PARLIAMENT_UG][51] as Tomcat servlet application by deploying prebuilt war archive. Parliament comes with web GUI interface which can be used to perform administrative tasks (e.g. Index creation, Data import etc.), by default this interface is unsecured, thus security provisions have been made to authorize user access to sensitive parts of web interface.

**TripleGeo-CSW**

[https://github.com/GeoKnow/TripleGeo][49]
[http://semwebcentral.org/frs/download.php/508/ParliamentQuickStart-v2.7.4-ubuntu-gcc-64.zip][50]
Open-source Python implementation of a CSW-to-RDF middleware web application. Latest version was downloaded directly from official github repository\(^{52}\). Application was deployed as WSGI Python application under Apache HTTP server. WSGI deployment was not working properly out of box. WSGI application wrapper file had to be created to support Apache mod_wsgi deployment.

### 6.4.4 Transformation setup & customization

INSPIRE GML to RDF transformation is carried by TripleGeo tool, which internally make use of XSLT transformation for this purpose. In fact TripleGeo tool is just frontend running XSLT transformation using default Java XSLT processor. (In addition, TripleGeo tool can handle more data sources e.g. *.SHP thus more JAVA code is needed, but in above mentioned implementation this was not the case). That means any XSLT stylesheets can be used bundled with the tool and preferred XSLT processor. No matter how XSLT's are run, there is only need to setup parameters in relevant XSLT stylesheet, which are well described in Appendix of [GEOKNOW -D2.7.1].

Apart from parameters setup, extra customization (debugging) have been made:

1. Fix MULTIPOLYGON encoding in GML2WKT.xsl\(^{53}\)
2. Fix WKT literal type namespace in Inspire_main.xsl\(^{54}\)
3. Add support for WFS response encoding (wfs:member) in Inspire_main.xsl
4. Update of INSPIREPS2RDF stylesheet for transforming datasets compliant with INSPIRE Data Themes I: PROTECTED SITES\(^5\) from GML into RDF file\(^{55}\)

### 6.4.5 Transformation execution

Transformation was executed directly running XSLT stylesheets bundled with TripleGeo using `xsltproc` XSLT processor (usually bundled with libxslt library). XSLT transformation operated on INSPIRE GML dataset previously downloaded from respective WFS service (obtained by WFS GetFeature request e.g. by using `curl`). The output RDF was ready for import into triple store. Drawback of this procedure is its offline nature which means RDF must be re-created and re-imported upon source data (data behind WFS) change to stay updated with source. On other hand all mentioned steps can be easily scripted - automated and then run on regular basis (e.g. by cron), which is satisfactory in many scenarios.

### 6.4.6 Import of INSPIRE RDF datasets into triple store

In case of Parliament triple store there is no other simple possibility to import RDF data but to upload RDF file through form in GUI web interface provided. This of course can be scripted too, but not in elegant way comparing the use of REST interface. Upon successful import automatic creation of spatial indexes is started. Note: It’s recommended to check it,

\(^{52}\) [https://github.com/GeoKnow/TripleGeo-CSW](https://github.com/GeoKnow/TripleGeo-CSW)

\(^{53}\) [http://smartopendata.sazp.sk/upload/GML2WKT.xsl](http://smartopendata.sazp.sk/upload/GML2WKT.xsl)

\(^{54}\) [http://smartopendata.sazp.sk/upload/Inspire_main.xsl](http://smartopendata.sazp.sk/upload/Inspire_main.xsl)

\(^{55}\) [http://smartopendata.sazp.sk/upload/InspirePS2RDF.xsl](http://smartopendata.sazp.sk/upload/InspirePS2RDF.xsl)
to find out if Parliament recognized geometries in imported data – e.g. problem with WKT literal encoding or unsupported geometry type.

### 6.4.7 Publish INSPIRE RDF datasets through GeoSPARQL interface

After import of RDF data into Parliament, these become available through (Geo)SPARQL endpoint automatically. Parliament has no user-access management so if needed, this must be implemented either via Tomcat or Apache authorization and security mechanisms e.g. HTTP authorization etc. In this case „everything is public“ option was allowed to fulfil open linked data principles.

Transformed INSPIRE Protected sites data from Slovakia were published and are queryable via following endpoints:

- **Parliament web Use Interface:** [http://data.sazp.sk/parliament](http://data.sazp.sk/parliament)
- **Parliament SPARQL endpoint:** [http://data.sazp.sk/parliament/sparql](http://data.sazp.sk/parliament/sparql)

![Figure 12 Snap of the SAZP Parliament web Use Interface](image)

### 6.4.8 Publishing INSPIRE metadata via SPARQL interface

INSPIRE compliant metadata published as Catalogue Services for the Web (CSW) provides significant information resource describing the spatial data and services published under the INSPIRE framework. In order to allow possibility to make them discoverable also via virtual SPARQL interface as RDF triples, TripleGeo - CSW Middleware has been deployed.

TripleGeo-CSW tool is used for publishing INSPIRE metadata as Linked data. This works as on-line translation & transformation tool, which accepts SPARQL query, translates it into OGC CSW request, sends request to configured OGC CSW servers and transform response from OGC CSW to RDF representation. All these steps are performed on-the-fly.

TripleGeo-CSW did not work properly with PyCSW CSW implementation so few tweaks had to be done:

- Fix encoding of `<ogc:PropertyIsLike>` element, missing required „escapeChar“ added
• Fix encoding of `<ogc:And>` element in case where SPARQL request contains only one search criteria expression

In addition TripleGeo-CSW cannot handle SPARQL request containing non ASCII characters, this is however more complicated problem (comparing two mentioned above) which need to be fixed in codebase. This can have impact on usage of TripleGeo-CSW in context of languages with specific characters. That part has not been done yet and will need further investigation.

![Flow diagram for processing SPARQL queries in the TripleGeo-CSW middleware](http://92.52.20.94:8082/ogcwxs/html/)

**Figure 13 Flow diagram for processing SPARQL queries in the TripleGeo-CSW middleware [GEOKNOW-D2.7.1]**

Based on the pilot requirements two SPARQL endpoints have been deployed allowing queries against:

- Slovakian INSPIRE national metadata CSW endpoint: [http://data.sazp.sk/triplegeo-csw/sparql](http://data.sazp.sk/triplegeo-csw/sparql)
- List of OGC services discovered on Google: [http://tokenbros.com:8082/sparql](http://tokenbros.com:8082/sparql)

## 7 Conclusions and Future Work

### 7.1 RDFS SmOD vocabularies

The results of the initial data harmonisation include implementation of the RDFS vocabularies of the Initial SmartOpenData model. At the time of working no RDF representation of the INSPIRE models underlying the SmOD model was available. However, recently JRC released case studies on transforming the INSPIRE models to RDF. Revision of this work is needed. Depending on the outcome of the revision adjustments to the SmOD vocabularies might be needed.

Further, we presented the initial data harmonisation workflow that consists of the following three processes:

56 [http://92.52.20.94:8082/ogcwxs/html/](http://92.52.20.94:8082/ogcwxs/html/)
(i) extension of the SmOD vocabularies with the domain-specific terms and development of the RDFS vocabularies that form the schema of the target RDF
(ii) mapping of the local data into terms of the SmOD vocabularies
(iii) transformation of the local data into RDF using the mappings

7.2 Extension of the SmOD vocabularies

We described the methodology for developing the domain-specific vocabularies which is based on ORM-to-RDFS translation. We defined rules to translate the ORM constructs to the RDFS classes and properties. The methodology was applied to extend the Initial SmOD Data Model with the TRAGSA specific terms https://smod-ontologies.spaziodati.eu/tragsa#. We plan to re-use the presented methodology to implement vocabularies to cover domains of other pilots.

In the current version of data harmonisation we considered only the ORM constructs used in the first release of the Spanish-Portuguese pilot ORM: object-types, value-types and binary associations. There are different ways of translating objectified associations to RDFS. For the objectified association of TRAGSA, “ForestryTileHasPlantSpecies”, we opted for distinct RDFS properties. Alternative solutions were presented in Annex A. In general, for future releases of the Spanish-Portuguese pilot and other SmOD pilots, every new objectified association should be considered separately, and the best approach between reification and distinct properties should be chosen.

The uniqueness constrains and enumerations are not supported by RDFS. While for the Spanish-Portuguese pilot they were not needed, this can be an issue for other pilots.

7.3 Definition of the mappings and transformations

In the current deliverable we covered mapping and transformation of tabular data available in one of the text-based data formats (such as CSV, TSV, Excel, JSON, XML, RDF/XML, etc.). We presented a tutorial on how to use the GUI of the RDF extension of OpenRefine to map data into the target RDF. We demonstrated usage of BatchRefine to transform the input data into RDF using the mappings defined in OpenRefine.

We plan to add support for harmonising data available in other formats, such as INSPIRE compliant XML. For this, we plan to increase the re-use of the work done by the GeoKnow project\(^57\). If requested by the SmOD pilots we will also cover harmonisation of relational data using the D2RQ platform\(^58\).

\(^57\) https://web.imis.athena-innovation.gr/redmine/projects/geoknow_public/wiki/Inspire2RDF
\(^58\) http://d2rq.org/
8 References


[RDFS] “RDF Schema 1.1” W3C Recommendation http://www.w3.org/TR/rdf-schema/


Annex A: RDF reification for ORM objectified association

Figure 3 illustrates an objectified association “ForestryTileHasPlantSpecies” between Forestry Tile and Plant Species. Let’s consider the following facts that involve this association:

- **Forestry Tile “157348-MFE25” has Plant Species “Quercus pyrenaica” that has Representative Level “1”**.
- **Forestry Tile “157348-MFE25” has Plant Species “Pinus pinaster” that has Representative Level “2”**.
- **Forestry Tile “157348-MFE25” has Plant Species “Quercus robur” that has Representative Level “3”**.

Below we give examples of how to encode these facts in RDF using classical RDF reification and reification via subgraphs. We will use the following RDF properties:

- :hasPlantSpecies to specify that Forestry Tile has a Plant Species
- :hasRepresentativeLevel to specifies representative level of the Plant Species

**Classical RDF reification**

Classical RDF reification assumes that extra information is added on triples within the triple model, i.e., subject-predicate-object. The following listing is the example of how to encode in RDF the first fact:

```
:157348-MFE25 :hasPlantSpecies :Quercus_pyrenaica .
```

Reification:

```
_:x rdf:type rdf:Statement .
_:x rdf:subject :ForestryTile1 .
_:x rdf:predicate tragsa:hasPlantSpecies .
_:x rdf:object :Quercus_pyrenaica .
_:x tragsa:hasRepresentativeLevel "1" .
```

Similarly, second and third facts are encoded.

**Advantages**

- can speak about individual triples in multiple reifications
- can reify reifications

**Disadvantages**

1. difficult to speak about groups of triples together

2. expensive to find reification for a triple
3. unintuitive model and representation
4. semantically not suitable for N-are relations, as “The RDF reification vocabulary is designed to talk about statements – individuals that are instances of rdf:Statement... In n-ary relations, however, additional arguments in the relation do not usually characterize the statement but rather provide additional information about the relation instance itself.”
5. verbose

Reification via subgraphs

While in the classical RDF reification extra information can be added to one triple at a time, reification via subgraphs allows for adding information on a set of triples, which is identified by a URI (aka a named graph).

Let's created three named graphs:

<http://example.org/primary-plant-species>
:157348-MFE25 :hasPlantSpecies :Quercus_pyrenaica .

<http://example.org/secondary-plant-species>
:157348-MFE25 :hasPlantSpecies :Pinus_pinaster .

<http://example.org/tertiary-plant-species>
:157348-MFE25 :hasPlantSpecies :Quercus_robur .

Next, we need to specify that the first graph contains primary plant species, the second one - secondary and the third one - tertiary. We add the following statements (in another reification graph):

<http://example.org/reification>
<http://example.org/primary-plant-species> :hasRepresentativeLevel "1" .
<http://example.org/secondary-plant-species> :hasRepresentativeLevel "2" .
<http://example.org/tertiary-plant-species> :hasRepresentativeLevel "3" .

Advantages

- easy to speak about groups of triples -> less verbose than the classical reification
- more intuitive model and representation
- cheaper to retrieve reifications for a triple, and vice-verse (assuming appropriate indexes)
- easier to do per-row/line-delimited/streaming processing

---

60 http://www.w3.org/TR/swbp-n-aryRelations/#RDFReification
61 https://dvcs.w3.org/hg/rdf/raw-file/default/rdf-mt/index.html#reification
- can extend quads to quins, etc., as necessary

**Disadvantages**
- implies non-standard model (named-graphs are exceptions)
- only non-standard exchange syntaxes are available (not true for quads)
- may be difficult to handle intersecting “graphs” (need to duplicates triple or extend tuples or define a set theory for graphs)
Annex B: **Spanish-Portuguese pilot: Dictionary & links to the TRAGSA database**

**Classes**

**tragsa:AnimalSpecies**
*URI:* [http://smod-ontologies.spaziodati.eu/tragsa#AnimalSpecies](http://smod-ontologies.spaziodati.eu/tragsa#AnimalSpecies)
*Database table/field:* Aux_000204000_InventoryTerrestrialAnimalSpecies / code
*Description:* Terrestrial animal species whose presence has been detected in a particular place, recording their abundance and conservation status.
*Description (ES):* Especie animal terrestre cuya presencia ha sido detectada en un lugar determinado, recogiendo su abundancia y estado de conservación.

**tragsa:ForestryTile**
*URI:* [http://smod-ontologies.spaziodati.eu/tragsa#ForestryTile](http://smod-ontologies.spaziodati.eu/tragsa#ForestryTile)
*Database table/field:* Aux_000201000_forestryMap / idForestry
*Description:* This class represents a unit of a forest map, which is defined as an area of homogeneous forest structure distinguishable from other units.
*Description (ES):* Unidad cartográfica del Mapa Forestal definida por constituir un recinto de estructura forestal homogénea distinguible del resto de unidades. El Mapa Forestal describe la situación de las masas forestales según sus usos de suelo, describiendo la ecología y estructura de las masas forestales.

**tragsa:Habitat**
*URI:* [http://smod-ontologies.spaziodati.eu/tragsa#Habitat](http://smod-ontologies.spaziodati.eu/tragsa#Habitat)
*Database table/field:* Aux_000203000_habitatsDirective / idHabitat
*Description:* This class represents habitats, environment that is occupied by a biological population of particular specie of animal, plant or other type of organism. It is the space that meets the conditions for the specie to live and reproduce. These habitats are collected and documented in Annex I of the Habitats Directive 92/43/ECC.
*Description (ES):* Hábitat es el ambiente que ocupa una población biológica. Es el espacio que reúne las condiciones adecuadas para que la especie pueda vivir y reproducirse. Un hábitat queda descrito por los rasgos que lo definen ecológicamente, distinguiéndolo de otros en los que las mismas especies no podrían desarrollarse. Estos Hábitat están recogidos y documentados en el Anexo I de la Directiva Hábitat 92/43/CEE

**tragsa:Parcel**
*URI:* [http://smod-ontologies.spaziodati.eu/tragsa#Parcel](http://smod-ontologies.spaziodati.eu/tragsa#Parcel)
*Database table/field:* Pd_0600_smod_cartography / idParcel
*Description:* This class represents parcels. A parcel is a continuous area of land within an agricultural unit, with the same agricultural use.

Description (ES): Superficie continua de terreno, dentro de una unidad agrícola, con un mismo uso agrícola. La parcela es susceptible de dividirse en recintos menores en función de las características físicas del terreno. Estas unidades menores podrían constituir unas “Unidades de Gestión”. Unidad de Gestión: Superficie continua de terreno, dentro de una parcela agrícola, con un mismo uso agrícola y unas características físico-químicas similares. Unidad espacial menor.

tragsa:Permeability
URI: http://smod-ontologies.spaziodati.eu/tragsa#Permeability
Database table/field: Aux_00031000_lithostratigraphy_permmeability / idLithoPer
Description: This class represents concept of permeability, the ability of a material (soil type) to be traversed by a fluid without altering its structure. A material is permeable if you miss a considerable amount of liquid, and impermeable if the amount is nil or negligible.

tragsa:PlantSpecies
URI: http://smod-ontologies.spaziodati.eu/tragsa#PlantSpecies
Database table/field: Aux_00020100_forestryMap / [Specie_1/Specie_2/Specie_3]
Description: Forest tree species listed in the Forest Map and defining the ecology and structure of a forest. The Forest Map describes the situation of forests according to their land use, describing the ecology and structure of forest stands.

tragsa:Soil
URI: http://smod-ontologies.spaziodati.eu/tragsa#Soil
Database table/field: Aux_00030800_lithology / idLitholo
Description: This class represents soil, the surface portion of the earth’s crust that remains biologically active and influences the development of different plant species.

Properties

tragsa:hasConservationStatus
URI: http://smod-ontologies.spaziodati.eu/tragsa#hasConservationStatus
Domain: tragsa:AnimalSpecies
Range: rdfs:Literal
Values: http://www.iucnredlist.org/
Database table/field: Aux_00020400_InventoryTerrestrialAnimalSpecies / idEndangered
Description: This property provides values of the Red List of threatened species as defined in the International Union for Conservation of Nature and Natural Resources (IUCN).
Description (ES): Categorías declaradas en la Lista Roja de la UICN (Unión Internacional para la Conservación de la Naturaleza).

tragsa:density
URI: http://smod-ontologies.spaziodati.eu/tragsa#density
Domain: rdfs:Resource
Range: xsd:integer
Values: [1..10] - degree of presence of a species in forest tile (%)
Database table/field: Aux_00020100_forestryMap / [Occupation_1/Occupation_2/Occupation_3]
Description: This property specifies presence of the Plant Species (in %) in the Forest Tile.
Description (ES): Grado de presencia (%) de las especies descritas en una misma tesela del Mapa Forestal. En el Mapa Forestal pueden describirse un máximo de tres especies arbóreas por tesela.

tragsa:plantSpecies
URI: https://smod-ontologies.spaziodati.eu/tragsa#plantSpecies
Domain: rdfs:Resource
Range: tragsa:PlantSpecies
Database table/field: Aux_00020100_forestryMap / [Specie_1/Specie_2/Specie_3]
Description: This property specifies the plant species of one of the relationships tragsa:hasPrimaryPlantSpecies, tragsa:hasSecondaryPlantSpecies or tragsa:hasTertiaryPlantSpecies.

tragsa:hasPrimaryPlantSpecies
URI: https://smod-ontologies.spaziodati.eu/tragsa#hasPrimaryPlantSpecies
Domain: tragsa:ForestryTile
Range: rdfs:Resource
Database table/field: Aux_000201000_forestryMap / Specie_1
Description: This property specifies relationship between a forest tile and a plant species covering a given area at representative level 1.
Description (ES): Especies forestales descritas en el Mapa Forestal presentes en la parcela.

tragsa:hasSecondaryPlantSpecies
URI: https://smod-ontologies.spaziodati.eu/tragsa#hasSecondaryPlantSpecies
Domain: tragsa:ForestryTile
Range: rdfs:Resource
Database table/field: Aux_000201000_forestryMap / Specie_2
Description: This property specifies relationship between a forest tile and a plant species covering a given area at representative level 2.
Description (ES): Especies forestales descritas en el Mapa Forestal presentes en la parcela.

tragsa:hasTertiaryPlantSpecies
URI: https://smod-ontologies.spaziodati.eu/tragsa#hasTertiaryPlantSpecies
Domain: tragsa:ForestryTile
D3.3 Harmonisation of data to SmartOpenData model

**Range:** rdfs:Rresource

**Database table/field:** Aux_000201000_forestryMap / Specie_3

**Description:** This property specifies relationship between a forest tile and a plant species covering a given area at representative level 3.

**Description (ES):** Especies forestales descritas en el Mapa Forestal presentes en la parcela.

**tragsa:hasHabitat**

**URI:** [http://smod-ontologies.spaziodati.eu/tragsa#hasHabitat](http://smod-ontologies.spaziodati.eu/tragsa#hasHabitat)

**Domain:** tragsa:Parcel

**Range:** tragsa:Habitat

**Database table/field:** Pd_0600_smod_cartography / idHabitat

**Description:** This property specifies the habitat of the parcel.

**Description (ES):** Código de los hábitat presentes en la parcela.

**tragsa:hasPermeabilityRate**

**URI:** [http://smod-ontologies.spaziodati.eu/tragsa#hasPermeabilityRate](http://smod-ontologies.spaziodati.eu/tragsa#hasPermeabilityRate)

**Domain:** tragsa:Parcel

**Range:** rdfs:Literal

**Values:** Aux_00031000_lithostratigraphy_permeability / PERME

**Database table/field:** Pd_0600_smod_cartography / idLithoPer (FK) → Aux_00031000_lithostratigraphy_permeability

**Description:** This property specifies permeability of a parcel's soil.

**Description (ES):** Capacidad que tiene un material (tipo de suelo) de ser atravesado por un líquido sin alterar su estructura.

**tragsa:hasRainFallLevel**

**URI:** [http://smod-ontologies.spaziodati.eu/tragsa#hasRainFallLevel](http://smod-ontologies.spaziodati.eu/tragsa#hasRainFallLevel)

**Database table/field:** Pd_0600_smod_cartography /idRainFall (FK) → Aux_00030900_pluviometry

**Domain:** tragsa:Parcel

**Range:** rdfs:Literal

**Values:** Aux_00030900_pluviometry / DefinitionClustered

**Description:** This property specifies the rainfall level in the Parcel.

**Description (ES):** Nivel de precipitación calculado para la parcela a partir de datos climáticos regionales.

**tragsa:hasSoil**

**URI:** [http://smod-ontologies.spaziodati.eu/tragsa#hasSoil](http://smod-ontologies.spaziodati.eu/tragsa#hasSoil)

**Database table/field:** Pd_0600_smod_cartography / idLitholo (FK) → Aux_000308000_lithology

**Domain:** tragsa:Parcel

**Range:** tragsa:Soil

**Values:** Aux_000308000_lithology / Definition

**Description:** This property specifies the soil of the parcel.

**Description (ES):** Toda parcela tiene uno o varios tipos de suelo que condiciona el desarrollo de diferentes especies vegetales.
tragsa:hasWeather
URI: http://smod-ontologies.spaziodati.eu/tragsa#hasWeather
Domain: tragsa:Parcel
Range: rdfs:Literal
Values:
Database table/field: Pd_0600_smod_cartography / idPhytoCli (FK) → Aux_00030600_PhytoClima / description
Description: This property specifies the climate conditions of the parcel which were calculated from regional data.
Description (ES): Toda parcela tiene unos datos climáticos calculados a partir de datos regionales.

tragsa:isLocatedIn
URI: http://smod-ontologies.spaziodati.eu/tragsa#isLocatedIn
Domain: tragsa:Parcel
Range: tragsa:ForestryTile
Database table/field: Pd_0600_smod_cartography / idForestry
Description: This property specifies the location of the parcel in one or more forestry tiles.
Description (ES): Una parcela está localizada en una o varias teselas del Mapa Forestal.

Annex C: Spanish-Portuguese pilot: ORM Documentation

Natural ORM Architect (NORMA)\textsuperscript{63} was used to model TRAGSA data in ORM. NORMA supports automated generation of verbal description of the models. Verbalisation of the Spanish-Portuguese pilot (TRAGSA) ORM models is available in the form of browsable html pages at http://smod-fp7.github.io/tragsa/orm/ObjectTypeList.html. Below is the list of all object-types:

- Animal Species
- Animal Species_name
- Conservation Status
- Density
- Forestry Tile
- Forestry Tile_id
- ForestryTileHasPlantSpecies
- Habitat
- Habitat_code
- HabitatSupportsAnimalSpecies
- Parcel
- Parcel_id

\textsuperscript{63} http://www.ormfoundation.org/files/
- ParcelHasHabitat
- ParcelHasSoil
- ParcelIntersectsProtectedSite
- ParcelIsLocatedInForestryTile
- Permeability
- Permeability Rate
- Permeability_id
- Plant Species
- Plant Species_name
- Protected Site
- Protected Site_name
- Rain Fall Level
- Representative Level
- Soil
- Soil_name
- Weather

Every object-type is described in a separate page, for example, Fig. 13 displays a screenshot showing verbalisation of “Animal Species”.
In addition to verbalisation of object-types, we generated a constraint validation report. The report is available online at [http://smod-fp7.github.io/tragsa/orm/ConstraintValidationReport.html](http://smod-fp7.github.io/tragsa/orm/ConstraintValidationReport.html). Figure 14 displays an example of a constraint, an internal uniqueness constraint on the association “hasConservationStatus”.

Figure 15 Spanish-Portuguese pilot (TRAGSA) ORM: an example of a constraint verbalisation

Each Animal Species has at most one Conservation Status.
Annex D: Spanish-Portuguese pilot: user queries

“Biodiversity and Protected Sites” Use Case

User: General User  
Question: Are there threatened animal species which habitat overlaps my parcel?

“Threatened” is defined by conservation status of animal species as per the IUCN Red List of Threatened Species\(^\text{64}\), e.g., “CR” is the code for “Critically Endangered” species.

This information can be retrieved from the following relationships:
- Parcel has Habitat
- Habitat supports Animal Species

“Agroforestry Management” Use Case

User: Agroforestry Company  
Question1: Has a given parcel specific conditions to grow a specific tree species?

For example, being interested in growing “Chestnut” and knowing that it needs:
- Mild weather
- High Rain level
- Siliceous fields
- Never limestone

find out if a given parcel has these conditions.

This information can be retrieved from the following relationships:
- Parcel has Rain Fall Level.
- Parcel has Weather.
- Parcel has Soil.

Question2: Does a given parcel overlap a Forestry Tile which main specie is a specific given specie?

Relevant information can be retrieved from:
- Parcel is located in Forestry Tile.
- Forestry Tile has Plant Species.
- Forestry Tile has Plant Species that has Representative Level.

“Funds assignment” Use Case

User: Farmer

\(^{64}\)http://en.wikipedia.org/wiki/Conservation_status
**Question**: Has a given parcel specific conditions to grow a certain tree species, oriented towards capturing CO2?

For example, “Pinus pinaster” is supposed to be good specie for CO2 retention (48tm per year). In the following years, European subsidies to agriculture will be also related with “Green Economy”. So, if a farmer plants some pine trees, this parcel will receive more subsidies from the Commission.

“Pinus pinaster” needs:
- High permeability
- Low density (It needs high exposure to solar light)
- Silicious fields
- Never limestone

This information can be retrieved from:
- Parcel has Soil.
- Soil has Permeability.
- Forestry Tile has Plant Species.
- Plant Species has Density.
Annex C: Translation of ShapeFile to WKT

Definitions

GDAL/OGR

Is a translator library for raster and vector geospatial data formats that is released under an X/MIT style Open Source license by the Open Source Geospatial Foundation. As a library, it presents a single raster abstract data model and vector abstract data model to the calling application for all supported formats. It also comes with a variety of useful commandline utilities for data translation and processing. GDAL library is used in almost all free and opensourc GIS, appliactions also in a lot of proprietary software. Also ETL (Extract Transform Load) tools are using GDAL, for example GeoKettle, Talend ETL. To make GDAL more available in different development environments GDAL has language bindings - Perl, Python, VB6 Bindings, Java, C# / .Net, Ruby and R.

ShapeFile

It is a proprietary spatial data format developed by ESRI Company. It is the de facto standard for geographic information exchange between Geographic Information Systems (GIS).

It is a vector data storing format. It stores the localization of the geographic elements and its attributes. It does not store topologic information.

Although it is commonly referred as shapefile, it is generated by a collection of files:

- a) .shp – shape format; the feature geometry itself.
- b) .shx - shape index format; a positional index of the feature geometry to allow seeking forwards and backwards quickly.
- c) .dbf - attribute format; columnar attributes for each shape, in dBase IV format.
- d) .prj - projection format; the coordinate system and projection information, a plain text file describing the projection using well-known text format.
- .sbn y .sbx - a spatial index of the features.
- .fbl y .fbx - a spatial index of the features that are read-only.
- .ain y .aih - an attribute index of the active fields in a table.
- .shp.xml - geospatial metadata in XML format, such as ISO 19115 or other XML schema.
PostgreSQL

It is an object-relational database management system (ORDBMS), developed under BSD license\(^{65}\).

PostGIS

It is a module that adds geographic capacity to the object-relational database PostgreSQL, turning it into a spatial database for its use in GIS. This module is published under GNU License\(^{66}\).

PgAdmin3

It is a multiplatform graphical interface which manages the PostgreSQL/PostGIS databases. It eases the administration and management of database using SQL instructions or with the help of a graphical environment.

WKT

Well Known Text – It is a codification or syntax in ASCII format designed to describe spatial data in a vector form. This format is able to describe the following objects:

1. Points
2. Multipoints
3. Lines
4. Multilines
5. Polygons
6. Multipolygons
7. Geometry collections
8. Points in 3 and 4 dimensions

It is promoted from the Open Geospatial Consortium (OGC). It has a simple to use syntax. It has been widespread as well-used format for the geographic information exchange. WKT syntax consists of a description of the vertices which compose the geometry. For this form of geometry description to make sense, this data has to go with an indication of the spatial reference or cartographic protection used in the vector. WKT is the foundation of other better known formats such as KML and many of the spatial databases. PostgreSQL uses this codification through its extension PostGIS.

---

\(^{65}\) BSD: Berkeley Software Distribution – It is a permissive open source license. It has fewer restrictions than others like GPL, being very close to the public domain. This license allows the use of open source code in non-free software.

\(^{66}\) GNU: General Public License – It is the most widespread open source license. It intends to protect the user rights by the Use-Share-Study-Modify schema, against the source code appropriation attempts.
ShapeFile To WKT using Python GDAL

Example how to access ShapeFile or any other GDAL supported vector file and process geometry in WKT format. Such approach can be realised also in other languages where GDAL language bindings are available.

Installing Python GDAL Language bindings

- Install python environment. In Linux environment installed by default, Windows users must download and install.
- GDAL bindings installation
  - Linux users can use two methods
    - Ubuntu/Debian users: apt-get install python-gdal
    - pip install python-gdal
  - Windows users can download right installation package from here and install: http://www.lfd.uci.edu/~gohlke/pythonlibs/#gdal

Code snippet to access, read and print data

Python code example. As test data set is used ShapeFile 'clc2012.shp' and the file is located in the same folder where following python code example.

```python
# -*- coding: utf-8 -*-
import os
from osgeo import ogr #be sure you have python-gdal bindings installed

#Here we declare driver we want to use, this time 'ESRI ShapeFile' but can be 'DGN','DXF' or any other gdal/ogr supported format.
driver = ogr.GetDriverByName('ESRI Shapefile')

#Here we open ShapeFile (in this case shapefile is located in the same folder where script.
inDataSet = driver.Open(r'clc2012.shp') #replace with path to your shp file

#Reading the Layer
inLayer = inDataSet.GetLayer()
```
#Iterating over each feature in layer.
for feature in inLayer:
	ry:
    #Here we print each feature geometry represented as WKT
    print feature.GetGeometryRef().ExportToWkt()
    #Here we print value of attribute field 'CODE'
    print feature.GetField("CODE")
except:
    print "Something wrong"

The example is showing a basic way how to read and access vector data using GDAL library.
In this case we simply print out the values in terminal, but that can be replaced with:

• store data in CSV file or any other structured format;
• processed using python XML libraries and stored in XML format;
• processed using python rdflib, that allows to store data already in RDF/XML, N3, NTriples, Turtle, TriX, RDFa and Microdata.

Such script usage can be effective for batch processes and routine processes, especially in cases when the source data structure is stable.

**ShapeFile To PostgreSQL / PostGIS**

**Installing PostgreSQL + PostGIS**

The setup Package (9.3.5 version) is available here:


It is a one click installation. It includes the additional packages pgAdmin and PostGIS.

1. Install it and Create the superuser: admin / admin
2. Port: 5432

After finishing the installation, launch the Stack Builder for downloading and installing the additional packages

• If this module crashes, download PostGIS [http://postgis.net/windows_downloads](http://postgis.net/windows_downloads)
  ○ Install it creating the example spatial database (postgis_21_sample)
• This installation adds the “PostGIS ShapeFile Import/Export Manager” tool, which eases the ShapeFile files importation.
Creating a spatial database

Create a new database using the pgAdmin III tool. It could be called \textit{Test\_Database}

- Define the proprietary, “tableSpace”, etc. if necessary.
- Set the character dataset if necessary.
- To create a spatial database set Template = postgis\_21\_sample

\textbf{Note}: If a spatial database (postgis\_21\_sample) has been created in the PostGIS module installation, it can be used to import the ShapeFile.

Importing ShapeFile to PostGIS

\textbf{Using pgAdmin III}

Open pgAdmin III and connect to spatial database \textit{Test\_Database}

Open the importation tool
PostGIS Shapefile Import/Export Manager tool
Access the spatial database through pgAdmin III

Importing Shape
Verify the import options
1. SRID: Indicate the coordinate system according to EPSG codification
2. Verify character codification if necessary.

---

**EPSG: European Petroleum Survey Group** - It is a scientific organization composed by specialists who work in geodesy, topography and cartography applied to the oil extraction. This organization collected and spread the geodesic parameters group EPSG, a database widely used which contains ellipsoids, datum, coordinate systems, cartographic projections, etc.
**PostgreGIS to WKT**

**Generating a table**

There are two options to perform this transformation:

1. **ST_AsText**:
   - It converts a geometry to WKT without including SRID metadata
   - It is the OGC standard
2. **ST_AsEWKT**:
   - It converts a geometry to WKT with SRID metadata
   - Nonstandard

Use

```sql
SELECT gid, field1, field2, fieldN, Shape_leng, Shape_area, ST_AsText(geom) FROM tabla;
SELECT gid, field1, field2, fieldN, Shape_leng, Shape_area, ST_AsEWKT (geom) FROM tabla;
```

Example

```sql
SELECT gid, parcel_id, Shape_leng, Shape_area, ST_AsText(geom) FROM shape_test;
```

SELECT gid, parcel_id, Shape_leng, Shape_area, ST_AsEWKT (geom) FROM shape_test;
Result

Shape_Test_asText.wkt

"gid";"parcel_id";"shape_leng";"shape_area";"st_astext"

1;"Parcel_01";168.849670704;1609.08897462;"MULTIPOLYGON(((118804.7917
4690992.4745,118809.779 4690982.2311,118805.6914 4690979.3626,118809.2785
4690973.0233,118765.4146 4690972.7196,118755.2857 4691012.203,118766.4008
4691019.7706,118804.7917 4690992.4745)))"

2;"Parcel_02";676.756986097;14145.6060838;"MULTIPOLYGON(((118809.2785
4690973.0233,118812.7073 4690966.9637,118804.2195 4690956.883,118796.6661
4690950.5986,118780.0653 4690937.9711,118764.4357 4690926.374,118750.8237
4690915.2835,118734.8938 4690904.0296,118719.4974 4690891.1126,118696.188
4690873.0795,118683.2206 4690863.644,118670.3061 4690854.2899,118657.0693
4690844.9999,118646.0269 4690838.0652,118634.2525 4690832.2066,118626.3762
4690828.6204,118620.2961 4690826.0093,118601.872 4690904.6724,118616.8156
4690910.2839,118615.2307 4690916.4581,118632.857 4690928.5533,118648.931
4690892.859,118688.2875 4690912.627,118675.763 4690958.0607,118755.2857
4691012.203,118765.4146 4690972.7196,118809.2785 4690973.0233)))"

Shape_Test_as EWKT.wkt

"gid";"parcel_id";"shape_leng";"shape_area";"st_asewkt"

1;"Parcel_01";168.849670704;1609.08897462;"SRID=25830;MULTIPOLYGON(((118804.7917
4690992.4745,118809.779 4690982.2311,118805.6914 4690979.3626,118809.2785
4690973.0233,118765.4146 4690972.7196,118755.2857 4691012.203,118766.4008
4691019.7706,118804.7917 4690992.4745)))"

2;"Parcel_02";676.756986097;14145.6060838;"SRID=25830;MULTIPOLYGON(((118809.2785
4690973.0233,118812.7073 4690966.9637,118804.2195 4690956.883,118796.6661
4690950.5986,118780.0653 4690937.9711,118764.4357 4690926.374,118750.8237
4690915.2835,118734.8938 4690904.0296,118719.4974 4690891.1126,118696.188
4690873.0795,118683.2206 4690863.644,118670.3061 4690854.2899,118657.0693
4690844.9999,118646.0269 4690838.0652,118634.2525 4690832.2066,118626.3762
4690828.6204,118620.2961 4690826.0093,118601.872 4690904.6724,118616.8156
4690910.2839,118615.2307 4690916.4581,118632.857 4690928.5533,118648.931
4690892.859,118688.2875 4690912.627,118675.763 4690958.0607,118755.2857
4691012.203,118765.4146 4690972.7196,118809.2785 4690973.0233))"
Exporting table to *.WKT file

4690873.0795,118683.2206 4690863.644,118670.3061 4690854.2899,118657.0693
4690844.9999,118646.0269 690838.0652,118634.2525 4690832.2066,118626.3672
4690828.6204,118620.2961 4690826.0093,118601.872 4690904.6724,118616.8156
4690910.2839,118615.2307 4690916.4581,118632.857 4690928.5533,118648.931
4690926.4627,118675.763 4690958.0607,118755.2857
4691012.203,118765.3827 4690972.7196,118809.2785 4690973.0233)))"
Annex D: *Translation of ShapeFile to WKT Open Land Use as Linked Data*

The Open Land Use (OLU) principles are based on collecting and interconnection of free spatial data sets related to themes land use and land cover (as they are defined in the INSPIRE document “Definition of Annex Themes and Scope”).

The target of this activity is to develop an application (or service) providing as much detail and accurate information on land use and land cover as is possible (in local as well as in global level). It means that the whole world is covered by a global data set (such as GlobCover), but for selected areas we are able to provide much more detail information or data. For example in the Europe there is CORINE land cover, urbanized areas are covered by Urban Atlas data (that is more detailed than CORINE land cover) and in the Czech Republic we are able to use cadastral data which provides an information on land use and land cover on the level of parcels.

The original data model (Figure 1) is based on INSPIRE land use specification. This model joins two basic data model of above-mentioned specification – existing land use and planned land use. Therefore it is not fully compliant with the specification, but it can be decomposed into two independent models being compliant existing land use and planning land use data models.

OLU data model connects planning and existing land use data. Both models are transformable to each other and it is also possible to migrate data from these model to or from other data sets that are in harmony with INSPIRE specification. The main reason for above-mentioned differences is determine by different usage of data and data models. OLU will be used for any land use (and land cover) data, Land Use Database Schema serves just to spatial planning data as a special part of land use data.

```
Open Land Use
+Inspired: Identifier
+geometry: GM_MultiSurface
+hlucLandUse: HILUCSVvalue
+regulationNature: RegulationNatureValue [0..1]
«lifeCycleInfo, voidable»
   +beginLifespanVersion: DataTime
   +endLifespanVersion: DataTime [0..1]
«voidable»
   +hlucPresence: HILUCSPresent [0..*]
+specificLandUse: LandUseClassificationValue[1..*]
+specificPresence: SpecificPresence [0..*]
+observationDate: Date
+processStepGeneral: ProcessStepGeneraValue
+backgroundMap: BackgroundMapValue
+dimensioningIndication: DimensioningIndicationValue [0..1]
+validFrom: Date [0..1]
+validTo: Date [0..1]
«OpenLandUse Specific attributes»
+note: Citation [0..1]
+landCoverClassValue: CorineValue
```

Note OLU model is combining existing and planning land use data models. All descriptions of attributes are in the INSPIRE Land Use specification.

```
<codeList>
  CorineValue
tags
  asDictionary = true
  extensibility = any
  xsdEncodingRule = iso19136_2007_INSPIRE_Extensions
```

Figure 16 Open Land Use data model
Because there are many various input data using different classification systems, mapping and harmonization rules between common OLU model using HILUCS (Hierarchical Inspire Land. Use Classification System) and original nomenclatures of particular data sets have had to be developed. It is necessary to mention that data model of integrated data sets are much more shorter than HILUCS and therefore there are use just first or second level of HILUCS. The reclassification uses existing methodologies as well as empiric experience, but in many cases the category “6_6_NotKnownUse”.

The requirement to publish OLU data as Linked data (in the RDF format) was realized with use the process (Figure 2) connecting original data (usually in SHP format, which has to be transformed to GML) and XSLT styles (describing mapping between particular data models and classifications and OLU model). Both inputs are processed by XSLT processor (we use Saxon in our experiments), which generates RDF file(s) with particular objects.

Figure 17 Transformation to RDF

The mapping rules between classification systems and HILUCS are stored in a simple ontological system, which is more flexible than a table.

The OLU RDF model deals with these namespaces (the URI of olu: namespaces will be defined later, but probably it be a part of plan4all or plan4business domain):

- `geo`: GeoSPARQL - http://www.opengis.net/ont/geosparql#
- `olu`: Open Land Use
The current version of the OLU RDF data model is composed of the land use objects `olu:OLU_Object` which has following properties\(^{68}\) (Table 1).

<table>
<thead>
<tr>
<th>Property</th>
<th>Explanation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gcm:inspireId</code></td>
<td>Identifier of each data object.</td>
<td><code>rdfs:Literal</code></td>
</tr>
<tr>
<td><code>geo:hasGeometry</code></td>
<td>Geometry (list of coordinates)</td>
<td><code>rdfs:Literal</code></td>
</tr>
<tr>
<td><code>lu:hilucsLandUse</code></td>
<td>Main land use type according to the HILUCS classification</td>
<td>Link to INSPIRE registry</td>
</tr>
<tr>
<td><code>lu:regulationNature</code></td>
<td>Legal nature of the land use regulation from a national perspective.</td>
<td>'Enumeration: bindingForDevelopers</td>
</tr>
<tr>
<td><code>gcm:beginLifespanVersion</code></td>
<td>Date and time at which this version of the existing land use data set was inserted or changed in the provided set of data.</td>
<td><code>xsd:date</code></td>
</tr>
<tr>
<td><code>gcm:endLifespanVersion</code></td>
<td>Date and time at which this version of the existing land use data set was superseded or retired in the provided set of data.</td>
<td><code>xsd:date</code></td>
</tr>
<tr>
<td><code>lu:specificLandUse</code></td>
<td>Code, description or URI of the original land use classification.</td>
<td><code>rdfs:Literal</code></td>
</tr>
<tr>
<td><code>lu:observationDate</code></td>
<td>The observation date associated to a description.</td>
<td><code>xsd:date</code></td>
</tr>
<tr>
<td><code>lu:processStepGeneral</code></td>
<td>Ennumeration: (adoption</td>
<td>elaboration</td>
</tr>
<tr>
<td><code>lu:dimensioningIndication</code></td>
<td>Specifications about the dimensioning of the urban developments.</td>
<td><code>rdfs:Literal</code></td>
</tr>
<tr>
<td><code>validFrom</code></td>
<td>The time when the existing land use data set started to exist in the real world</td>
<td><code>xsd:date</code></td>
</tr>
<tr>
<td><code>validTo</code></td>
<td>The time from which this existing land use data set no longer exists in the real world.</td>
<td><code>xsd:date</code></td>
</tr>
<tr>
<td><code>olu:note</code></td>
<td>Any remark, note or comment,</td>
<td><code>rdfs:comment</code></td>
</tr>
</tbody>
</table>

\(^{68}\) Voidable attributes `hilucsPresence` and `specificPresence` were removed from the model, because majority of real land use data does not deal with multiple land use type connected to one area.
### Table 4 olu:OLU_Object

The basic olu:OLU_Object is (or could be) connected with background map (lu:BackgroundMap; Table 2) by the object relation lu:backgroundMap.

<table>
<thead>
<tr>
<th>Property</th>
<th>Explanation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>lu:backgroundMapDate</td>
<td>Date of creation of the background map.</td>
<td>xsd:date</td>
</tr>
<tr>
<td>lu:backgroundMapReference</td>
<td>Identification of the background map that has been used for constructing this zoning element.</td>
<td>rdfs:Literal</td>
</tr>
<tr>
<td>lu:backgroundMapURI</td>
<td>URI of the background map</td>
<td>URI</td>
</tr>
</tbody>
</table>

The mapping between the original data model and RDF version is quite simple. The connection between particular attributes are derive-able from the names of attributes and relations in each model. It is necessary to mention that RDF model does not express multiplicity of properties.

The sample RDF file (Figure 4) show that the data object contains only mandatory information (Id, geometry and HILUCS codes).

---

**Figure 18 Example of OLU object in RDF**

_version 1.0_ Page 72 of 73 © SmartOpenData Consortium 2014_
There are a lot of tasks to be solved in the following phases of the OLU activities:

(iv) Integration of official INSPIRE Linked data resources
(v) Revision and optimization of data model and mapping rules
(vi) Transformation from file-based approach to a robust database solution
(vii) Massive linking to external resources
(viii) Data quality information for common users