PESCaDO

FP7-248594

Personalized Environmental Service Configuration and Delivery Orchestration

PESCaDO Ontology Documentation

Start date of project: 1st January, 2010 Duration: 36 months

	PESCaDO Ontology Documentation
Project Acronym :	PESCaDO
Contract No :	FP7 - 248594
Due Date :	n/a
Reply To:	Marco Rospocher rospocher@fbk.eu
Actual date of delivery:	n/a

Document History

Version	Date	Reason of change
0.1	20.05.2011	Document Created (Rospocher)
0.4	27.05.2011	First draft of the whole document (Rospocher)
0.5	29.05.2011	Revision (Rospocher)
1.0	01.12.2011	Modularization of the ontology (Rospocher)
1.1	09.01.2012	Minor Adjustments (Rospocher)
2.0	11.07.2012	Started revision of documentation due to the new version (3.0) of the PESCaDO ontology (Rospocher)

Table of Contents

1 Introduction

In this document, we describe the in details the PESCaDO ontology (ver 3.0 – July 2012). A description on how the ontology has been built (ver. 1.0) is available in D4.2 "Basic Ontology Extension Techniques".

First, we describe the scope, intended users and uses, and requirements of the ontology. Second, we present some general information on the ontology. Then, we describe in details the entities that the ontology comprises, thus showing how a problem, data about it, and conclusions can be instantiated and related to each other.

We will conclude presenting some details on an ontology module called the PESCaDO logico-semantic relations module, which is orthogonal to the PESCaDO ontology (i.e. independent from the content expressed in the PESCaDO ontology), and which will be used to represent in the ontology logico-semantic relations between facts contained in it (e.g. this measure of carbon monoxide is causing that bad air quality index, this qualitative value is implied by that quantitative measure), useful for the generation of content to be shown to the user.

2 The PESCaDO Ontology Requirements Specification Document

The table below represent the PESCaDO Ontology Requirements Specification Document (ORSD), a document which specify why the ontology is being built, what its intended uses are, who the users are, and which requirements the ontology should fulfill (according to the guidelines provided in [Suárez-Figueroa et al., 2009]).

PESCaDO Ontology Requirements Specification Document	
1	Purpose
	The purpose of building the PESCaDO Ontology is to provide the PESCaDO platform with a knowledge base able to represent and store: • the requests submitted by the user; • the environmental-related data and content relevant for the given requests; • the content (e.g. warnings/suggestions and additional date) generated by the PESCaDO decision support service to be returned to the users according to the requests. The ontology is needed by the system in order to guarantee a motivated orchestration of heterogeneous environmental service nodes and to offer user-tailored decision support services and environmental information production.
2	Scope
	The PESCaDO ontology has to formally represent: • environmental content, such as meteorological conditions and phenomena, air quality, and pollen; • travel and road condition information; • environmental data (i.e. measurements); • environmental nodes (stations, web-sites, web-services) details; • user profile details, including human diseases, especially those caused or triggered by environmental condition; • geographical location data; • description of the user's inquiry, i.e. the kind of request (and related information, for instance the activity to be performed) the user asks for decision support about. • warnings and suggestions to be reported to the users. The level of granularity of the content is directly related to the competency questions (see the Ontology Requirements section of the template).
3	Implementation Language
	The ontology has to be implemented in OWL, the de-facto standard language for representing ontology in the Semantic Web.
4	Intended Users
	The user groups considered in the PESCaDO system have been described in Deliverable D8.4 – User Typology in PESCaDO. We briefly report them, with their main characteristics: User 1. *General public* (persons or organizations), interested in receiving information and

	decision support on environmental conditions potentially dangerous for the personal health; User 2. *Public administrators* (or professionals in general), interested in receiving information on air quality for professional reasons: legal, health care, etc User 3. *Environmental experts* and *meteorologists*, interested in (i) keeping up to date with respect to the evolving air quality situation, (ii) offering short-term air quality forecasts, (iii) notifying the interest groups when needed, and (iv) assessing the potential risks and benefits of both long- and short term actions and plans for the air quality. However, the PESCaDO users will not directly interact with the ontology, but they will submit their requests to the PESCaDO system that will exploit the ontology (and the data dynamically injected in it by the different services composing the system) to provide adequate decision support and user tailored content. Among the services using the content stored in the ontology (see Deliverable D8.6 “Set-up of an operational system platform”), there are: Service 1. The *User Interaction Service*, which allows the users to formulate their question; Service 2. The *User Profile Management Service*, which allows the users to edit its profile; Service 3. The *Problem Description Generation Service*, which create a problem description according to the input provided by the user in the user interface; Service 4. The *Related Aspects Computation Service*, which, given a problem description, determines which are the environmental data to be retrieved; Service 5. The *Answer Service*, which provides (via the *Data Retrieval Service* and Fusion *Service*) the environmental data and environmental nodes relevant for the user problem considered; Service 6. The *Decision Service*, which infers the conclusions and additional data relevant for a given user problem; Service 7. The *Content Selection Service*, which select the content to be reported to the user; Service 8. The *Data Retrieval Service*, which constitutes the front end of the environmental node search and discovery infrastructure.
5	Intended Uses
	As mentioned in the “Intended Users” slot of this template, the users will not directly interact with the ontology. Therefore, the intended uses of the ontology are mainly application-oriented. We identified the following ones: Uses 1. To retrieve the kind of user problems that are supported by the system. Uses 2. To store and retrieve the details of the profile of the users. Uses 3. To store all the details of the decision support problem that a user submit to the system. Uses 4. To determine the environmental data types (e.g. birch pollen, rain) which are relevant for the given problem submitted by the user to the system; Uses 5. To store and retrieve the environmental data (i.e. measurements) that are relevant for the specific problem submitted by the user; Uses 6. To store and retrieve the details about the environmental nodes;

	Uses 7. To store and retrieve the conclusions (e.g. suggestions/warnings, additional content) generated by the PESCaDO system according to the problem submitted by the user; Uses 8. To retrieve synonyms or alternative names of some environmental data type.
6	Ontology Requirements
	a. Non-Functional Requirements
	NFR1. The ontology should support representation of part of its content in the following language: English, Finnish, and Swedish. NFR2. Whenever possible, available standards should be adopted.
	b. Functional Requirements: Groups of Competency Questions
	Competency Questions Group 1: Types of environmental data CQ1. What are the main categories of environmental data? Weather, air quality, pollen, road condition, warning CQ2. What are the types of pollens? Alder pollen, birch pollen, grasses pollen, … CQ3. What are the weather environmental data? Temperature, wind speed, wind direction, humidity, … CQ4. What are the kinds of road conditions? Traffic road conditions and weather-related condition of road surface. CQ5. What are the air pollutant types? NO_2, CO, SO_2, … CQ6. What are the synonyms or alternative names of PM_{10}? “particulate matter 10”, “particulate matter < 10 µm”, “respirable particulate matter”, “breathable particulate matter”,… Competency Questions Group 2: Environmental data (measures) CQ7. To what environmental data type does an environmental data refers to? Rain, wind speed, NO_2, birch pollen, … CQ8. What is the nature of the environmental data considered? Observed data, forecasted data, historical data, variation data. CQ9. To what time period (from/to) does an environmental data refer? From date/time 2011-04-28T00:00:00+03:00, to date/time 2011-04-29T13:00:00+03:00. CQ10. What are the special time granularities considered for air quality data? 1h, 8h, 24h. CQ11. What is the (numerical) value of an environmental data? 30 C, 15m/s, 1020 hPa, … CQ12. What is the unit of measurement for numerical values of an environmental data of type wind speed? Meter per seconds (m/s) CQ13. What is the unit of measurement for numerical values of an environmental data of type wind direction? Degrees (^{o}) CQ14. What is the unit of measurement for numerical values of an environmental data of type temperature? Celsius degree (^{o}C) CQ15. What is the unit of measurement for numerical values of an environmental data of type humidity? Hectopascal (hPa)

CQ16. What is the unit of measurement for numerical values of an environmental data of type air pollutant? Micrograms per meter cubed (u/m^3)

CQ17. What is the unit of measurement for numerical values of an environmental data of type pollen? Grains per meter cubed (grains/m^3)

CQ18. What is the unit of measurement for numerical values of an environmental data of type humidity? Percent (%)

CQ19. What is the unit of measurement for numerical values of an environmental data of type rain? Millimeter per hour (mm/h)

CQ20. What is the unit of measurement for numerical values of an environmental data of type snow? Centimeter per hour (cm/h)

CQ21. What is the unit of measurement for numerical values of an environmental data of type sky condition (cloudiness)? Eights (/8)

CQ22. What is the index of an environmental data of type air quality? Good, satisfactory, fair, poor, very poor.

CQ23. What is the qualitative value (rating) of an environmental data of type air quality? Very low, low, moderate, high, very high.

CQ24. What is the qualitative value (rating) of an environmental data of type pollen? low, moderate, abundant.

CQ25. What is the qualitative value (rating) of an environmental data of type humidity? Wet, normal, dry.

CQ26. What is the qualitative value (rating) of an environmental data of type air pressure? Very low, low, moderate, high, very high.

CQ27. What is the qualitative value (rating) of an environmental data of type rain? Dry weather (no rain), light rain, moderate rain, heavy rain.

CQ28. What is the qualitative value (rating) of an environmental data of type sky condition (cloudiness)? Clear sky, quite clear, partly cloudy, quite cloudy, cloudy, not determinable.

CQ29. What is the qualitative value (rating) of an environmental data of type temperature in summer? Extremely cold, remarkably cold, very cold, cold, chilly, mild, warm, hot, very hot, remarkably hot, extremely hot.

CQ30. What is the qualitative value (rating) of an environmental data of type temperature in winter? Thaw, weak frost, severe frost, extreme frost.

CQ31. What is the qualitative value (rating) of an environmental data of type wind speed? Calm, weak, moderate, gale, hard, storm, hurricane.

CQ32. To what numerical value range does weak wind correspond? From 1m/s to 4m/s.

CQ33. To what numerical value range does poor CO index correspond to? From 20000 u/m^3 to 30000 u/m^3.

CQ34. What is the type of aggregation an environmental data represent? Minimum, maximum, average.

CQ35. What is the probability of a forecasted data? E.g. 0.8 (or 80%)

CQ36. What is the uncertainty value attached to a forecasted data? E.g. 0.6 (or 60%)

CQ37. What is the time period a historical data refers to? A month (e.g April) or a whole year.

CQ38. What is the tendency of a variation data? Notably decreasing, decreasing, constant, increasing, notably increasing.

CQ39. What is the gradient of a variation data? Constant, sharp, steady

CQ40. What is the awareness level of on environmental data of type meteo-alarm warning? White, green, yellow, orange, red.

Competency Questions Group 3: Environmental nodes (stations/web-sites/web-services)

CQ41. What is the form (kind) of the environmental node? Stations, web-sites, web-services

CQ42. What is the confidence value associated to the environmental node? 0.9

CQ43. What is the type of environmental node? Background, traffic, industrial

CQ44. What are the sources of emission nearby the environmental node? Agriculture, commercial combustion, fossil fuels

CQ45. What is the land use nearby the environmental node? Agricultural, industrial, residential, …

CQ46. What is the type of area where the environmental node is located? Rural, suburban, urban.

CQ47. What are the environmental data provided by the environmental node? "from 2011-04-28T00:00:00+03:00 to 2011-04-29T13:00:00+03:00, data about temperature, quantitative value is 20 degrees Celsius, qualitative value is warm, forecasted data", …

CQ48. What is the name of the environmental node? Helsinki 1, AccuWeather, …

CQ49. What is the url of the web-site/web-service environmental node? www.accuweather.com

CQ50. What is the location of the environmental node? Geo-coordinates (60.245264, 24.960938, 100)

Competency Questions Group 4: Exceedances and Exceedance Types

CQ51. What is the environmental data causing the exceedance? From date/time 2011-04-28T00:00:00+03:00, to date/time 2011-04-29T13:00:00+03:00, data bout NO_2, value 450, time interval 1h.

CQ52. What is the type of the exceedance? European NO_2 Hourly limit value, European NO2 24h limit value, …

CQ53. What is the air pollutant to which the exceedance type applies? NO2, PM10, …

CQ54. What is the air pollutant time interval to which the exceedance type applies? 1h, 8h, 24h

CQ55. What is the lower bound for the exceedance type? 200 u/m^3, …

CQ56. How many times the lower bound of the exceedance type can be exceeded in a year? 25

CQ57. How many time in a year the exceedance type lower bound has been exceeded? 15

CQ58. What is the action an administrative person of the Helsinki Metropolitan Area should take when the exceedance considered in the short-term action plan is exceeded? Public bulletin, traffic limitation, …

Competency Questions Group 5: Geographic Data / Location

CQ59. What are the coordinates of the location? Latitude 60.245264, Longitude 24.960938, Altitude 100.

CQ60. What are the locations defining the region/route? (60.245264,24.960938,100), (60.26224, 23.120938, 120), …

CQ61. What is the first location of the route? (60.245264,24.960938,100)

CQ62. What is the name of the location? Helsinki

CQ63. What is the name Geonames URI corresponding to the location? Helsinki

CQ64. What are the environmental node providing data for the given geographical area? Helsinki 1, AccuWeather, en.ilmatieteenlaitos.fi, …

Competency Questions Group 6: Problem Description – Activities

CQ65. What are the activities the user can ask decision support about? Making some physical outdoor activity, travelling, attending some open air event, moving to live or going on holiday in a place.

CQ66. What is the travel modality the user as chosen? By car, by bicycle, by foot, …

CQ67. What is the physical outdoor activity the user want to make? Hiking, trekking, …

CQ68. What is the intensity level at which the user want to exercise? Intense, light, …

CQ69. What kind of open air event does the user want to attend? A concert, a sport competition, a public demonstration.

CQ70. What is the long term activity the user has planned to do? Going on holiday, moving to another place to live.

CQ71. What are the environmental data types which are relevant (i.e. measurement of these data type can affect the decision suggested by the system) for the activity of travelling? PM_{10}, road traffic condition, …

Competency Questions Group 7: Problem Description – Requests

CQ72. What is the type of request submitted? “Are there any health issue if I…”, “What is the situation of the air quality at the moment?”, …

CQ73. What is the type of activity associated to the request? Travelling, making some physical outdoor activity,…

CQ74. Which user as submitted the request? FionaFit, AllenAllergic, EricExpert,

CQ75. Which users are considered in the request? FionaFit and AllenAllergic, …

CQ76. Which is the time period (from/to) associated to the request? From date/time 2011-04-28T00:00:00+03:00, to date/time 2011-04-29T13:00:00+03:00.

CQ77. What is the geographic area associated to the request? The region bounded within the locations (60.245264,24.960938,100), (60.26224, 23.120938, 120), …

CQ78. What are the environmental data types which are relevant (i.e. measurement of these data type can affect the decision suggested by the system) for the request about health issues? Ozone, temperature, rain, …

CQ79. Does the request have any specific focus (i.e. any environmental data type to necessarily check), and if so, which one is it? Yes, NO_2.

Competency Questions Group 8: Problem Description – User Profile

CQ80. What are the main categories a user can belong to? End user (general public), Administrative User, Expert User

CQ81. What is the name of the user? Fiona Fit

CQ82. What is the age of the user? 28

CQ83. What is the preferred communication language for the user? English, Finnish, Swedish

CQ84. What is the area of expertise of the user (for administrative and expert users)? Air quality, meteorology, …

CQ85. What is the default location of the user? Helsinki (60.16952, 24.93545, 0)

CQ86. What is the gender of the user? Male/female

CQ87. Which disease is the user suffering of? Bronchial asthma, Acute topic conjunctivitis, …

CQ88. Which pollens is the user sensitive/allergic to? Birch pollen, alder pollen, …

CQ89. Which pollutant is the user sensitive to? Birch pollen, alder pollen, grasses pollen, …

CQ90. What is the intensity level at which the user usually does some physical activities? Light, moderate, …

CQ91. What are the environmental data types which are relevant (i.e. measurement of these data type can affect the decision suggested by the system) for a user sensitive to birch pollen? Temperature, wind speed, birch pollen, …

Competency Questions Group 9: Conclusions (Suggestion/Warnings)

CQ92. What is the message associated to the suggestion reported by the system? “Avoid exercising too much in the hours around midday”, …

CQ93. What is the message associated to the warning reported by the system? “Pollen concentration is very high, so you may experience shortness of breath”, …

CQ94. What are the warnings/suggestions returned by the system for the given request and geographic area? “Avoid exercising too much in the hours around midday”, “Pollen concentration is very high, so you may experience shortness of breath”, …

7	Pre-Glossary of Terms	
	a. Terms extracted from Competency Questions + Frequency	
	• environmental data 30 • environmental node 11 • measurement for numerical values 10 • qualitative value 9 • rating 9	• decision suggested 3 • reported 2 • pollen 2 • message 2 • index 2 • route 2

	• request 8 • exceedance type 5 • system 5 • environmental data type 5 • activity 4 • air pollutant 3 • pollen 4 • exceedance 3 • measurement 3 • data type 3 • affect 3 • exceeded 3 • temperature 3 • data 3 • user sensitive 3 • relevant 3 • humidity 3 • time period 3 • lower bound 3 • user want 3	• physical 2 • sky condition 2 • categories 2 • warning 2 • intensity level 2 • wind speed 2 • cloudiness 2 • air quality 2 • air 2 • rain 2 • variation data 2 • submitted 2 • administrative 2 • year 2 • activities 2 • forecasted data 2 • nearby 2 • numerical value range 2 • pollutant 2
	b. Terms extracted from Competency Questions Answers + Frequency	
	• rain 6 • date / time 6 • data 5 • birch pollen 5 • cold 4 • temperature 4 • road 4 • pollen 4 • hot 4 • traffic 4 • hour 3 • wind speed 3 • air quality 3 • light 3	• physical outdoor activity 2 • hpa 2 • grains 2 • grasses pollen2 • accuweather 2 • fionafit 2 • forecasted data 2 • celsius 2 • holiday 2 • weak 2 • dry 2 • cubed 2 • live 2 • weather 2

	• frost 3 • alder pollen 3 • helsinki 3 • cloudy 3 • meter 3 • public 3 • warm 2	• allenallergic 2 • limit value 2 • european 2 • travelling 2 • going 2 • place 2 • industrial 2
	c. Objects	
	• FionaFit, AllenAllergic, EricExpert,… • Helsinki • Meter, hpa, Celsius, … • Accuweather	

3 General information on the PESCaDO Ontology

In this section, we provide some details on the current version of the PESCaDO ontology (ver. 3.0 – September 2012). The ontology is available (together with its documentation) through the project web-site (http://www.pescado-project.eu/)[1].

The ontology consists of the following nine modules,

- pescadoBase
- pescadoConclusions
- pescadoData
- pescadoDiseases
- pescadoExceedances
- pescadoGeo
- pescadoInstances
- pescadoNLP[2]
- pescadoNodes
- pescadoPDL

plus an additional file (pescadoAll.owl) which wraps all the modules together. The definition of these modules follows the grouping of the competency questions presented in the previous section. Each module will be described in details in Section 4.

Furthermore, the PESCaDO ontology reuses some available ontologies/schemas resulting from standardization efforts:

- geosparql (http://www.opengeospatial.org/standards/geosparql - version 1.0): this ontology corresponds to the OGC GeoSPARQL standard, to goal of which is to support representing and querying geospatial data on the Semantic Web. GeoSPARQL defines a vocabulary for representing geospatial data in RDF, and it defines an extension to the SPARQL query language for processing geospatial data.
- PROV-O: The PROV Ontology (http://www.w3.org/TR/prov-o/ - Working Draft 03 May 2012): this ontology expresses the PROV Data Model using the OWL2 Web Ontology Language (OWL2). It provides a set of classes, properties, and restrictions that can be used to represent and interchange provenance information generated in different systems and under different contexts.

These external modules have been specialized to create new classes and properties to model geographical and provenance information specific to the PESCaDO application domain. For example, classes Line, Polygon, and Point of the pescadoGeo module are defined as subclasses of the geosparql:Geometry class, thus inheriting the characteristics of this class (e.g., the possibility to define the actual coordinated delimiting the shape via the geosparql:asWKT serialization property[3]), while GeoArea, Region, Route, SpotLocation classes of the pescadoGeo module are defined as subclasses of the geosparql:Feature class. Similarly, EnvironmentalData class of the pescadoData module is defined as subclass of the (PROV-O) prov:Entity class,

[1]Most likely under the terms of the Creative Commons "Attribution Non-commercial Share Alike" license (version 3.0) - http://creativecommons.org/licenses/by-nc-sa/3.0/

[2] This additional module, not covered by the competency questions presented in Section 2, defines classes and properties needed for the generation of text from the ontology, and is described in details (together with the rationale behind its construction) in Deliverable D5.2.

[3] See the Geosparql User Guide for more details: http://ontolog.cim3.net/cgi-bin/wiki.pl?InteropProject/Geosparql_USER_GUIDE_2012

while EnvironmentalNode class of the pescadoNodes module is defined as subclass of the (PROV-O) prov:Agent class, thus enabling for instance to represent the provenance of an data used in the system.

The interconnections among the modules by means of ontology imports are shown in Figure 1.

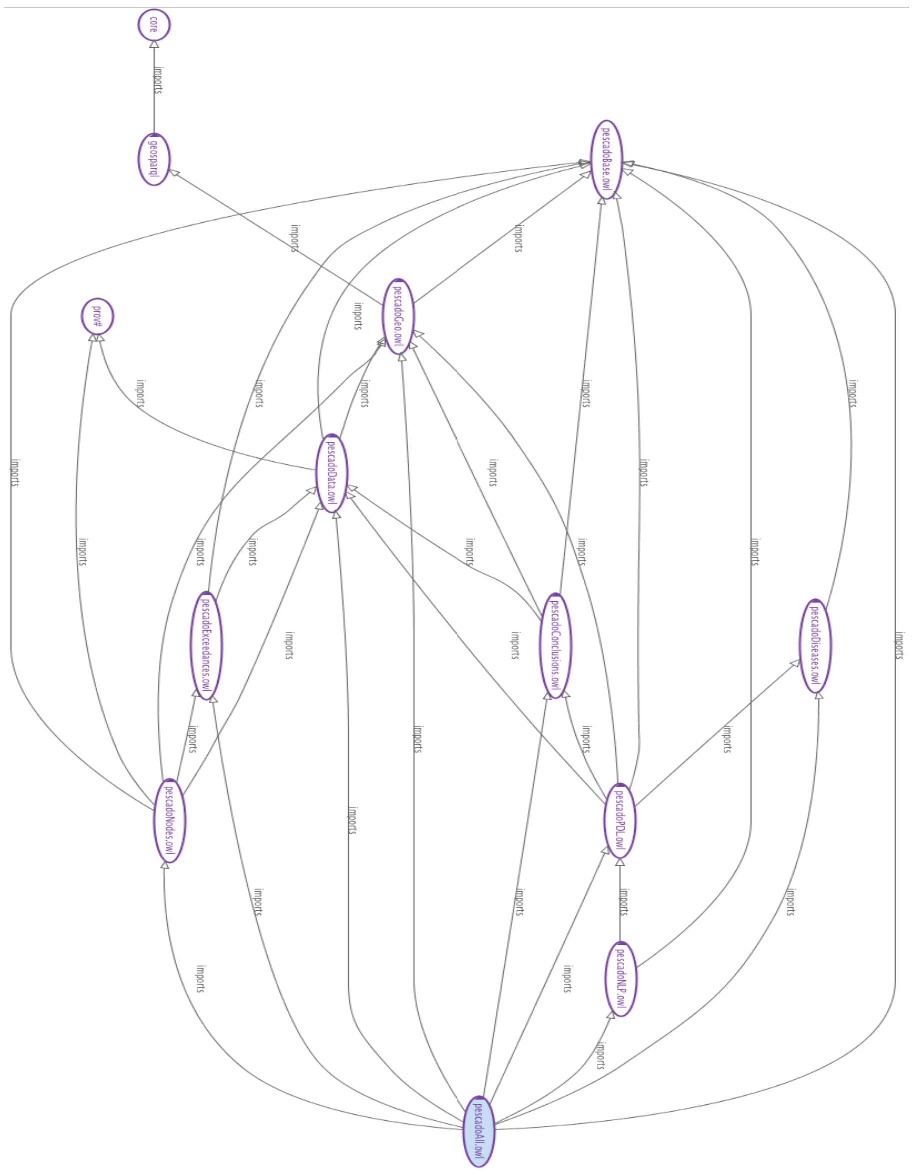
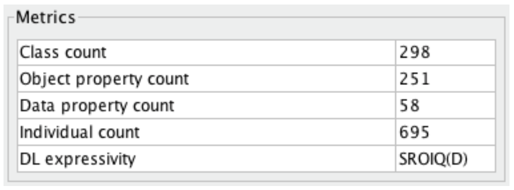


Figure 1 - Ontology import graph of the PESCaDO ontology modules

The table below reports some ontology metrics, as displayed by the *ontology metrics view* available in Protégé. The metrics refers to the whole un-instantiated ontology, i.e. without containing any data referring to a specific user request.



Metrics

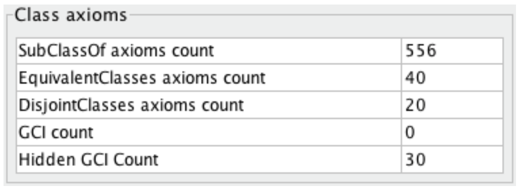
Class count	298
Object property count	251
Data property count	58
Individual count	695
DL expressivity	SROIQ(D)

The DL expressivity information tells in which Description Logics variant the ontology can be categorized. This information is computed by looking at the constructors and features of the language which are used in the ontology:

- S (or ALC): the base language (atomic negation, concepts intersection, universal restrictions, limited existential quantification), complex concept negation;
- R: complex role inclusions;
- O: nominals (i.e. enumerated classes or object value restriction)
- I: inverse properties;
- Q: qualified cardinality restrictions;
- (D): use of datatype properties.

We recall that reasoning (checking concept satisfiability and ABox consistency) on SROIQ(D) ontologies is a NExpTime-complete decision problem[4].

The ontology is quite rich in terms of class restrictions used to characterize the terminological knowledge to be represented, as shown by the class axioms counters below.

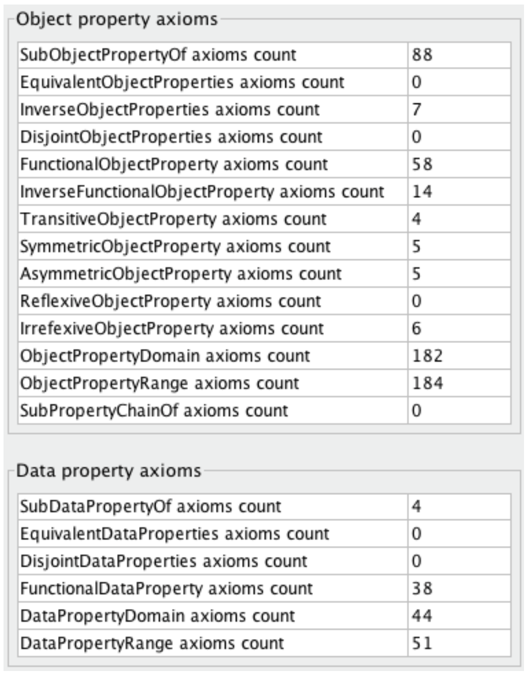


Class axioms

SubClassOf axioms count	556
EquivalentClasses axioms count	40
DisjointClasses axioms count	20
GCI count	0
Hidden GCI Count	30

The PESCaDO ontology contains also quite a number of object properties and datatype properties, used to make assertions on the individuals described in the ontology. In particular, datatype properties play an important role in the PESCaDO ontology, as much of the knowledge to be formalize deals with expressing values (e.g. the concentration level of a pollutant), time periods (from when date/time to when date/time a measure refers to), As shown in the two tables below, many characteristic and features have been used to accurately formalize objects properties and datatype properties in the ontology.

[4]See http://www.cs.man.ac.uk/~ezolin/dl/ for detailed references on this result.



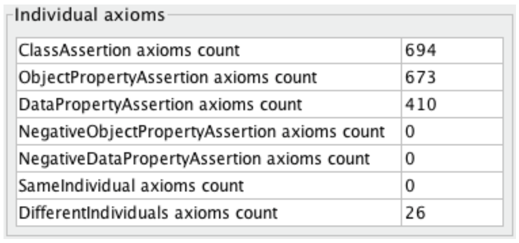
Object property axioms

SubObjectPropertyOf axioms count	88
EquivalentObjectProperties axioms count	0
InverseObjectProperties axioms count	7
DisjointObjectProperties axioms count	0
FunctionalObjectProperty axioms count	58
InverseFunctionalObjectProperty axioms count	14
TransitiveObjectProperty axioms count	4
SymmetricObjectProperty axioms count	5
AsymmetricObjectProperty axioms count	5
ReflexiveObjectProperty axioms count	0
IrrefexiveObjectProperty axioms count	6
ObjectPropertyDomain axioms count	182
ObjectPropertyRange axioms count	184
SubPropertyChainOf axioms count	0

Data property axioms

SubDataPropertyOf axioms count	4
EquivalentDataProperties axioms count	0
DisjointDataProperties axioms count	0
FunctionalDataProperty axioms count	38
DataPropertyDomain axioms count	44
DataPropertyRange axioms count	51

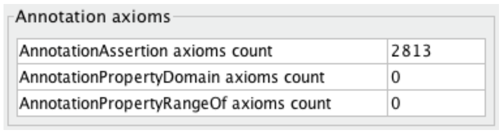
The PESCaDO ontology also contains a relevant number of individuals and assertions on them. These individuals are used to represent e.g. the units of measurement used by the data, the types of environmental data, and attributes needed to characterize some ontological classes.



Individual axioms

ClassAssertion axioms count	694
ObjectPropertyAssertion axioms count	673
DataPropertyAssertion axioms count	410
NegativeObjectPropertyAssertion axioms count	0
NegativeDataPropertyAssertion axioms count	0
SameIndividual axioms count	0
DifferentIndividuals axioms count	26

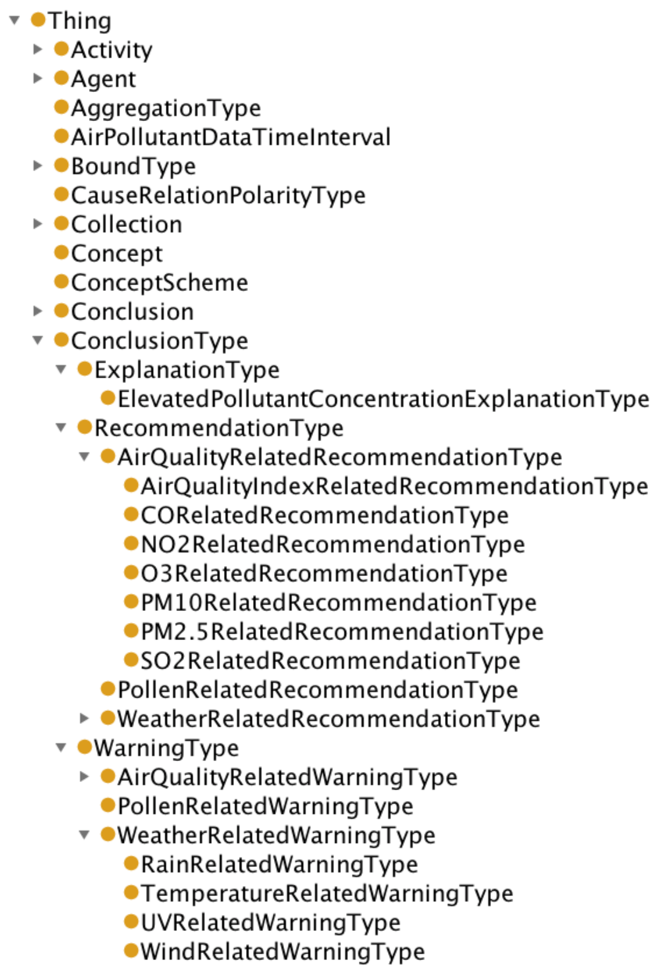
Finally, some annotations properties and assertions have also been defined, in order to represent human friendly labels for the ontology entities (to be used when the system has to display some element to the end user), comments explaining some modeling choices or definitions, and synonyms (in some cases also used to encode some multilingual information in the ontology) defined to support query expansion and environmental node search by the PESCaDO system.



Annotation axioms

AnnotationAssertion axioms count	2813
AnnotationPropertyDomain axioms count	0
AnnotationPropertyRangeOf axioms count	0

An exhaustive description of the entities defined in the PESCaDO ontology is provided in Section 4. An excerpt of the taxonomy behind the PESCaDO ontology is illustrated Figure 2.



```
▼ ●Thing
    ▶ ●Activity
    ▶ ●Agent
      ●AggregationType
      ●AirPollutantDataTimeInterval
    ▶ ●BoundType
      ●CauseRelationPolarityType
    ▶ ●Collection
      ●Concept
      ●ConceptScheme
    ▶ ●Conclusion
    ▼ ●ConclusionType
        ▼ ●ExplanationType
              ●ElevatedPollutantConcentrationExplanationType
        ▼ ●RecommendationType
            ▼ ●AirQualityRelatedRecommendationType
                  ●AirQualityIndexRelatedRecommendationType
                  ●CORelatedRecommendationType
                  ●NO2RelatedRecommendationType
                  ●O3RelatedRecommendationType
                  ●PM10RelatedRecommendationType
                  ●PM2.5RelatedRecommendationType
                  ●SO2RelatedRecommendationType
              ●PollenRelatedRecommendationType
            ▶ ●WeatherRelatedRecommendationType
        ▼ ●WarningType
            ▶ ●AirQualityRelatedWarningType
              ●PollenRelatedWarningType
            ▼ ●WeatherRelatedWarningType
                  ●RainRelatedWarningType
                  ●TemperatureRelatedWarningType
                  ●UVRelatedWarningType
                  ●WindRelatedWarningType
```

Figure 2 - Excerpt of the PESCaDO ontology taxonomy

3.1 Changes with respect to PESCaDO ontology v2.0

Several changes have been made to the ontology with respect to version 2.0, to formalize the additional content needed for the additional scenario that will be considered in the PESCaDO final demonstrator:

- added the new request type "DetailedAirQualityReport", which is supposed to be the request submitted for the additional scenario;
- added the relevant aspect mappings to the request type "DetailedAirQualityReport" (coherently with the “checkAirQualityLimits” request;
- added a "shown" annotation property to all the request: this could be used by the UI to grey out in the interface the request of the PDL currently not supported by the demonstrator;
- added submissionDateTime on request: in addition to the from/to date/time, we need also this information for the new demonstrator, for handling the past/future hours;
- added ExceedanceAggregationData class with some object/datatype properties to handle the "aggregation" (per station) of exceedances of the same type (hasExceedanceExceedanceType);
- added the Task class with some object/datatype properties to handle the submission of batch/scheduled requests to the system;

- added the Explanation, ExplanationType, and SolutionType classes to represent the cause (and solution, i.e. other things a user may want to check) for possible high concentration of pollutants. Explanation will be instantiated "linked" to the max concentration aggregated value of the air pollutants;
- added mikeManager profile, to be used to demonstrate the system in the new scenario.

Furthermore, this new version of the ontology imports two third-party modules, geosparql and PROV-O, thus reusing some available ontologies/schemas resulting from standardization efforts.

4 The PESCaDO Ontology

4.1 Module: Problem Description Language (pescadoPDL.owl)

Class **Problem**

An individual of this class represent a specific problem submitted by the user. It is linked to one of more individuals of type Request by the assertions on properties "hasProblemRequest", "hasProblemFirstRequest", and "hasProblemLastRequest", which indicate any, the first and the last (respectively) request associated to the problem.

Class **Activity**

This class has four sub-classes (AttendingOpenAirEvent, LongTermStaying, PhysicalOutdoorActivity, Travelling) that represent the possible activities a user can specify in the problem description. To this class (and its subclasses), some additional restriction of type "has value" are imposed, in order to represent the kind of environmental data which are relevant for the given activity. These restrictions are of the form "hasRelevantAspect value indiv_XYZ", where indiv_XYZ is an individual of class RelevantAspect.

Class **AttendingOpenAirEvent**

This class has to be instantiated together with one assertion (hasAttendingOpenAirEventOpenAirEventType) stating the kind of open air event the user wants to attend. The possible values of open air event are described in class (OpenAirEventType).

Class **LongTermStaying**

This class has to be instantiated together with one assertion (hasLongTermStayingLongTermStayingType) stating the kind of long term staying activity the user wants to perform. The possible values of long term staying activity are described in class (LongTermStayingType). The class has two subclasses which specialize the class according the above assertion made.

- Class GoingOnHolidayLongTermStaying
- Class LivingLongTermStaying

Class **PhysicalOutdoorActivity**

This class has to be instantiated together with two assertions: one (hasPhysicalOutdoorActivityPhysicalExcerciseIntensityLevel) stating the intensity level at which the user wants to perform a certain activity. The possible values of intensity level are described in class (PhysicalExerciseIntensityLevel). The other assertion (hasPhysicalOutdoorActivityPhysicalOutdoorActivityType) is about the type of physical outdoor activity the user wants to perform. The possible values of physical outdoor activity type are described in class (PhysicalOutdoorActivityType). The class PhysicalOutdoorActivity has some subclasses which specialize the class according the assertion about the physical exercise intensity level.

- Class IntensePhysicalOutdoorActivity

- Class LightPhysicalOutdoorActivity
- Class ModeratePhysicalOutdoorActivity
- Class NonePhysicalOutdoorActivity

Class **Travelling**

This class has to be instantiated together with one assertion (hasTravellingTravelModality) stating the kind of travel means the user wants to use. The possible values of travel means are described in class (TravelModality). The class has some subclasses that specialize the class according the above assertion made.

- Class BikeOrFeetTravelling
- Class BikeTravelling
- Class FeetTravelling
- Class CarTravelling
- Class PublicTransportTravelling

Class **Request**

This class has several sub-classes that represent the possible types of requests the user can specify in the problem description. To this class (and its subclasses), some additional restriction of type "has value" are imposed, in order to represent the kind of environmental data which are relevant for the given request. These restrictions are of the form "hasRelevantAspect value indiv_XYZ", where indiv_XYZ is an individual of class RelevantAspect.

The specific request submitted by a user will be an instance of one of the possible subclasses of the class Request, in particular those at the "leaves" level of the hierarchy rooted at "Request", that is: SuggestAdministrativePlan, CheckAirQualityLimits, CheckBlackIceCondition, CompareAirQualityInMultipleRegions, ReportAirQualityForecast, AnyHealthIssue, AnyRestrictionForPrivateTransport, WarningDueToEnvironmentalConditions.

The individual of type Request can be linked to other individual of the KB via the following properties:

- hasRequestActivity: the target is an individual of class Activity; it is used to indicate the activity associated to the request (if any).
- hasRequestFocus: the target is an individual of class EnvironmentalDataType; it is used to indicate the possible focus of the request (if any).
- hasRequestGeoAre: the target is an individual of class GeoArea; it is used to indicate the geographic area relevant for the request.
- hasRequestNextRequest and hasRequestPreviosRequest: the target are individuals of type Requests (used if there are multiple ordered requests attached to the same problem);
- hasRequestPrimaryUser: the target is an individual of class User; it is used to indicate the user submitting the request.
- hasRequestUser: the target is an individual of class User; it is used to indicate additional users involved in the request (if any).

The individual of type Request can have the following datatype property assertions:

- hasFromDateTime (every request should have exactly one of these assertions): it indicates the starting date / time considered in the request;
- hasToDateTime (every request should have exactly one of these assertions): it indicates the ending date / time considered in the request;
- hasSubmissionDateTime (every request should have exactly one of these assertions): it indicates the date / time the request is submitted by the user.

Further restrictions are in place on the subclasses of request, for example to state if a request is allowed only for specific type of user, or if the request does not come with an activity attached.

Class **User**

This class is used to identify users of the system. All the assertions on an individual of type User allow representing the full profile of a user. To this class (and its subclasses), some additional restriction of type “has value” are imposed, in order to represent the kind of environmental data which are relevant for the given user. These restrictions are of the form “hasRelevantAspect value indiv_XYZ”, where indiv_XYZ is an individual of class RelevantAspect.

Assertions that can be made on users individuals are:

- hasUserPreferredLanguage: the target is an individual of class Language; it is used to indicate the language chosen by the user for the content output.
- hasUserGender: the target is an individual of class Gender; it is used to indicate the gender of the user.
- hasUserPreferedTransportation: the target is an individual of class TravelModality; it is used to indicate the user preferred travel modality.
- hasUserDefaultPhysicalExerciseIntensityLevel: the target is an individual of class PhysicalExerciseIntensityLevel; it is used to indicate the intensity at which the user usually performs physical outdoor activities.
- hasUserDefaultLocation: the target is an individual of class Location; it is used to indicate the default location the user wants to be considered for the requests.
- hasUserAreaOfExpertise: the target is an individual of class AreaOfExpertise; it is used to indicate the area of expertise of the user (mainly for Administrative and Expert users).
- isUserSensitiveTo: the target is an individual of class AirPollutantDataType or class PollenDatatType, or the individual UVindex (of class WeatherDataType). It is used to indicate whether the user is sensitive/allergic to some air pollutant, or pollen, or the UVindex.
- userSuffersOf: the target is an individual of class Disease; it is used to indicate whether the user suffers of any disease.

The individual of type User can have the following datatype property assertions:

- hasUserName: it indicates the name of the user;
- hasUserAge: it indicates the age of the user;
- isUserPregnant: it is used to indicate whether a female is user is also pregnant.

There are three main subclasses of User, according to the PESCaDO user typology:

- AdministrativeUser
- EndUser
- Expert

Each of this class is further specified according the values and assertions made on them: e.g. class AdultUser is the class of those individuals of class EndUser that have more than 17 years (according to the value of assertion hasUserAge).

Class **RelevantAspect**

The individuals of this class represent relevant aspects for the subclasses of User, Request, Activity classes. Each individual of this class has exactly one assertion of type hasRelevantAspectEnvironmentalDataType with target an individual of class EnvironmentalDataType.

Class **Task** [new in version 3.0]

The individuals of this class are used to formalize batch/scheduled tasks to be submitted to the system. Somo object properties assertions can be made on individuals of this class:

- hasTaskProblem: links to a problem instance encoding the user request to be scheduled;
- hasTaskFrequency: determines the frequency at which to schedule the problem (daily/hourly).

Some datatype properties assertions can also be made

- hasTaskDailyAtTime: the time at which to schedule a daily task;
- hasTaskFutureHours: the number of hours in the future (to be used to compute the "hasToDateTime" assertion on request);
- hasTaskPastHours: the number of hours in the past (to be used to compute the "hasFromDateTime" assertion on request);
- hasTaskOnlyIfExceedances: a Boolean to represent whether the report should be communicated to the user only if exceedances are mentioned.

Additional classes of the module:

Gender, LongTermStayingType, OpenAirEventType, PhysicalExerciseIntensityLevel, PhysicalOutdoorActivityType, TravelModality, Language, AreaOfExpertise, TaskFrequencyType

4.2 Module: Diseases (pescadoDiseases.owl)

Class **Disease**

The individuals of this class represent possible diseases the user suffers of. The individuals are organized in sub-classes, according to the ICD-10 WHO classification (this part of the ontology has been partially reused from the ICD-10 ontology available at https://dkm.fbk.eu/index.php/ICD-10_Ontology)

<u>4.3 Module: Geographical data (pescadoGeo.owl)</u>

Class **GeoArea** [revised in version 3.0]

This class describes the possible types of geographical areas that can be used by the system. <u>It is defined as a subclass of the geosparql:Feature class</u>. Assertions that can be made on individuals of this class are:

- hasGeoAreaType (every GeoArea should have exactly one of these assertions): the target is an individual of class GeoAreaType; it is used to indicate the type geographic area considered. Three subclasses of GeoArea are defined according to the assertion made: Region, Route, SpotLocation.
- hasGeoAreaSubGeoArea, hasGeoAreaFirstSubGeoArea, hasGeoAreaLastSubGeoArea: the target of these properties are individuals of class GeoArea. They are used in the case a Region or a Route are segmented in sub-areas according to some segmentation criteria. In case of ordered list of sub-areas, hasGeoAreaFirstSubGeoArea and hasGeoAreaLastSubGeoArea are used to indicate the first and last GeoArea in the list.
- hasGeoAreaNextGeoArea, hasGeoAreaPreviousGeoArea: the target of these properties are individuals of class GeoArea. They are used to indicate the next and previous GeoArea, when the current individual is one of the elements of a list of individuals of type GeaArea.
- hasGeoAreaConclusion: the target of this property are individuals of class Conclusion. It is used to indicate the conclusions generated by the Decision Service for the current area considered.
- hasGeoAreaEnvironmentalNode: the target of this property are individuals of class EnvironmentalNode. It is used to indicate the environmental nodes providing environmental data for the current area considered.
- hasGeoAreaEnvironmentalData: the target of this property are individuals of class EnvironmentalData. It is used to indicate (aggregated) environmental data representative of the current area considered. (*not used at the moment*)
- hasGeoAreaWholePeriodAggregatedEnvironmentalData: the target of this property are individuals of class EnvironmentalData. It is used to indicate aggregated environmental data representative of the whole period for the current area considered.
- geosparql:hasGeometry: the target of this property is the subclass of geosparql:Geometry class.

The individual of type GeoArea can have the following datatype property assertions:

- hasGeoAreaName: it indicates the name of the geographic area considered (if any);

Class **Line** [new in version 3.0]

This class describes one of the possible types of geometrical shapes used by the system to represent geographical areas, the Line. <u>It is defined as a subclass of the geosparql:Geometry class</u>. The individual of this class should have an assertion of type geosparql:asWKT providing the serialization in WKT format of the geometrical shape. E.g.

LINESTRING(long1 lat1, long2 lat2, …)

Class **Point** [new in version 3.0]

This class describes one of the possible types of geometrical shapes used by the system to represent geographical areas, the Point. It is defined as a subclass of the geosparql:Geometry class. The individual of this class should have an assertion of type geosparql:asWKT providing the serialization in WKT format of the geometrical shape. E.g.

POINT(longitude latitude)

Class **Polygon** [new in version 3.0]

This class describes one of the possible types of geometrical shapes used by the system to represent geographical areas, the Polygon. It is defined as a subclass of the geosparql:Geometry class. The individual of this class should have an assertion of type geosparql:asWKT providing the serialization in WKT format of the geometrical shape. E.g.

(polygon with no hole) POLYGON((long1 lat1, long2 lat2, … , long1 lat1))

(polygon with hole) POLYGON((long1 lat1, long2 lat2, … , long1 lat1), (longA latA, longB latB, …, longA latA))

Two subclasses according to these characterization are defined, PolygonWithHole and PolygonWithoutHole.

4.4 Module: Environmental Data (pescadoData.owl)

Class **EnvironmentalDataType**

The individuals of this class represent all the possible types of environmental data considered in the system. The individuals are categorized in subclasses, the top level ones being: AirQualityDataType, PollenDataType, RoadConditionDataType, WarningDataType, WeatherDataType.

Class **EnvironmentalData** [revised in version 3.0]

The individuals of this class represent environmental data (of a certain environmental data type). It is defined as a subclass of the prov:Entity class, as it is an entity on which provenance information can be attached to. They are meant to indicate observed, forecasted, historical, statistical data environmental data. Assertions that can be made on individuals of this class are:

- hasEnvironmentalDataAggregationType: the target of this property are individuals of class AggregationType. It is used to indicate whether the data indicates a kind (and which kind) of aggregated data (e.g. minimum, maximum, average).
- hasEnvironmentalDataEnvironmentalDataType (every EnvironmentalData should have exactly one of these assertions): the target of this property are individuals of class EnvironmentalDataType. It is used to indicate to which kind of environmental data type the data refers to.
- hasEnvironmentalDataNature (every EnvironmentalData should have exactly one of these assertions): the target of this property are individuals of class EnvironmentalDataNature. It is used to indicate the nature (observed, forecasted, historical, variation) of the environmental data.
- hasEnvironmentalDataRating (hasEnvironmentalDataLowerRating, hasEnvironmental-DataUpperRating): the target of this property are individuals of class Rating. It is used to indicate the whether the environmental data considered has a qualitative value attached to it. Properties hasEnvironmentalDataLowerRating, hasEnvironmentalData-UpperRating can be used to express if the data refers to a range of qualitative values (e.g. from good to excellent).
- hasEnvironmentalDataValue (hasEnvironmentalDataLowerValue, hasEnvironmental-DataUpperValue): the target of this property are individuals of class Rating. It is used to indicate the whether the environmental data considered has a numerical value attached to it. Properties hasEnvironmentalDataLowerValue, hasEnvironmentalDataUpperValue can be used to express if the data refers to a range of numerical values (e.g. from 0.1 to 40.0).
- hasAirQualityDataIndex (hasAirQualityDataLowerIndex, hasAirQualityDataUpper-Index): the target of this property are individuals of class Index. The property applies to individuals having also the assertion "hasEnvironmentalDataEnvironmentalDataType" X, where X is an individual of class AirQualityDataType. It is used to indicate a qualitative value representing the index (qualitative value) of a pollutant or the air quality. Properties hasAirQualityDataLowerIndex, hasAirQualityDataUpperIndex can be used to express if the data refers to a range of indexes (e.g. from good to fair).
- hasAirPollutantTimeInterval: the target of this property are individuals of class AirPollutantTimeInterval. The property applies to individuals having also the assertion "hasEnvironmentalDataEnvironmentalDataType" X, where X is an individual of class AirPollutantDataType. It is used to indicate the granularity period of the standard air pollutant measurements (1h, 8h, 24h).

- hasEnvironmentalDataWarningType: the target of this property are individuals of class MeteoAlarmWarningType. The property applies to individuals having also the assertion “hasEnvironmentalDataEnvironmentalDataType” X, where X is an individual of class WarningDataType. It is used to indicate the type of meteo-alarm warning associated to the WarningData;

The individual of type EnvironmentalData can have the following datatype property assertions:

- hasFromDateTime (every EnvironmentalData should have exactly one of these assertions): it indicates the starting date / time to which the data refers to;
- hasToDateTime (every EnvironmentalData should have exactly one of these assertions): it indicates the ending date / time to which the data refers to.

Being subclass of the prov:Entity class, the following assertion can also be made on its individual:

- prov:wasDerivedFrom: for instance to state from which raw data a fused data has been obtained from (and thus from which entity);
- prov:wasGeneratedBy: for instance to state which prov:Activity has generated a given data;
- prov:wasAttributedTo: to state the prov:Agent which is the provider of the data.

Class **Value**

This class is used to represent specific numerical values, together with the corresponding unit of measurement. Assertions that can be made on individuals of this class are:

- hasUnit: the target of this property are individuals of class Unit. It is used to indicate the unit of measurement associated to the value individual.

The individual of type WarningData can have the following datatype property assertions:

- hasValueValue: it indicates the actual numerical value associated to the individual.

There are several subclasses, specific for the various environmental data considered, each one having a certain type of numerical value and unit of measurement: AirPollutantValue, HumidityValue, PollenValue, PressureValue, RainValue, SkyConditionValue, SnowValue, TemperatureValue, UVIndexValue, WindDirectionValue, WindSpeedValue,

Class **Rating**

This class is used to represent the qualitative values that can be associated to environmental data. Assertions that can be made on individuals of this class are:

- hasRatingRatingValue: the target of this property are individuals of class RatingValue. It is used to indicate the specific qualitative value (e.g. good, veryHigh, clear, etc).

The individual of type WarningData can have the following datatype property assertions:

- hasRatingWeight: it indicates a weight (as a numeric value) associated to the qualitative value.

Class **RatingValue**

There are several subclasses, specific for the various environmental data considered, each one representing certain qualitative value scales: HumidityRatingValue, MeteoAlarmOfficialWarningAwarenessLevel, PollenRatingValue, PollutantRatingValue, PressureRatingValue, RainRatingValue, RoadTrafficConditionRatingValue, RoadWeatherConditionRatingValue, SkyConditionRatingValue, SnowRatingValue, TemperatureRatingValue, SummerTemperatureRatingValue, WinterTemperatureRatingValue, UVIndexRatingValue, WindDirectionRatingValue, WindSpeedRatingValue, FinlandWindSpeedRatingValue, InternationalWindSpeedRatingValue.

Assertions that can be made on individuals of these classes are:

- hasLowerBoundType, hasLowerBoundType: the target of these properties are individuals of class BoundType. It is used to indicate the specific type of bound corresponding to extremes of the numerical interval corresponding to the rating value (e.g. greater or equal than, lesser than).

The individual of type WarningData can have the following datatype property assertions:

- hasLowerBoundValue, hasLowerBoundValue: they indicate the extremes of the numerical interval corresponding to the rating value;
- hasOrdinalNumer: it indicates the relative position of the qualitative value wrt the scale of the qualitative values defined in the class. (e.g. dryHumidity individual has ordinal number 0 because it is the lower level of the scale of qualitative values for HumidityRatingValue class).

Class **Index**

This class is used to represent the qualitative index values that can be associated to air quality data. Assertions that can be made on individuals of this class are:

- hasIndexIndexValue: the target of this property are individuals of class IndexValue. It is used to indicate the specific qualitative index value (e.g. good, satisfactory, poor, etc).

The individual of type WarningData can have the following datatype property assertions:

- hasIndexWeight: it indicates a weight (as a numeric value) associated to the index value.

Class **IndexValue**

There are several subclasses, specific for the various air quality data considered, each one representing the qualitative index scales: AirPollutantIndexValue, COIndexValue, NO2IndexValue, O3IndexValue, PM10IndexValue, PM2.5IndexValue, SO2IndexValue, TRSIndexValue, AirQualityIndexValue.

Assertions that can be made on individuals of these classes are:

- hasLowerBoundType, hasLowerBoundType: the target of these properties are individuals of class BoundType. It is used to indicate the specific type of bound corresponding to extremes of the numerical interval corresponding to the index value (e.g. greater or equal than, lesser than).

The individual of type WarningData can have the following datatype property assertions:

- hasLowerBoundValue, hasLowerBoundValue: they indicate the extremes of the numerical interval corresponding to the index value;
- hasOrdinalNumer: it indicates the relative position of the index value wrt the scale of the index values defined in the class. (e.g. veryPoorCOIndex individual has ordinal number 4 because it is the fifth element of the scale of index values for COIndexValue).

Class **Unit**

The individuals of this class are units of measurements used for the numerical values of the environmental data. The list of these units of measurements is mostly taken from the SWEET ontology (http:// sweet.jpl.nasa.gov).

Additional classes defined in this module:

BoundType, LowerBoundType, UpperBoundType, Gradient, HistoricalDataPeriod, MeteoAlarmWarningType, PollenAllergenicity, Tendency, TimeInterval, AirPollutantDataTimeInterval

4.5 Module: Environmental Nodes (pescadoNodes.owl)

Class **EnvironmentalNode** [revised in version 3.0]

The individuals of this class represent the environmental nodes considered in the system (e.g. web-site, web-service, stations). It is defined as a subclass of the prov:Agent class, as it is a source which provides data (and thus represent the provider of the information). Assertions that can be made on individuals of these classes are:

- hasEnvironmentalNodeDataAboutEnvironmentalDataType: the target of this property are individuals of class EnvironmentalDataType. It is used to indicate the environmental date type the environmental node provides data about (e.g. rain, CO, windSpeed, etc). (*not used at the moment*)
- hasEnvironmentalNodeEnvironmentalData: the target of this property are individuals of class EnvironmentalData. It is used to indicate the environmental data that this environmental node provides for the given user request.
- hasEnvironmentalNodeForm (every EnvironmentalNode should have exactly one of these assertions): the target of this property are individuals of class EnvironmentalNodeForm. It is used to indicate if the environmental node is a station, a web-site, or a web-service.
- hasEnvironmentalNodeEnvironmentalNodeAreaType: the target of this property are individuals of class EnvironmentalNodeAreaType. It is used to indicate the type of area in which the environmental node, in particular a station, is placed (e.g. rural, urban).
- hasEnvironmentalNodeEnvironmentalNodeLandUseType: the target of this property are individuals of class EnvironmentalNodeLandUseType. It is used to indicate the intended use of the land where the environmental node, in particular a station, is placed (e.g. agricultural, industrial).
- hasEnvironmentalNodeEnvironmentalNodeSourceOfEmissionType: the target of this property are individuals of class EnvironmentalNodeSourceOfEmissionType. It is used to indicate the type of sources of pollutants in the area where the environmental node, in particular a station, is placed (e.g. fossil fuels combustion, industrial combustion).
- hasEnvironmentalNodeEnvironmentalNodeType: the target of this property are individuals of class EnvironmentalNodeType. It is used to indicate the setting of area where the environmental node, in particular a station, is placed (e.g. background, traffic).
- hasEnvironmentalNodeExceedanceData: the target of this property are individuals of class ExceedanceData. It is used to indicate the exceedances of limit values, threshold, etc. detected at that environmental node.
- hasEnvironmentalNodeStatisticalExceedanceData: the target of this property are individuals of class StatisticalExceedanceData. It is used to indicate the statistical data on the exceedances of limit values, threshold, etc. detected at that environmental node.
- hasEnvironmentalNodeLocation: the target of this property are individuals of class Location. It is used to indicate the location where the environmental node is placed.
- hasEnvironmentalNodeWholePeriodAggregatedEnvironmentalData: the target of this property are individuals of class EnvironmentalData (usually Aggregated EnvironmentalData). It is used to indicate the environmental data that are representative of the whole area for the given request.

The individual of type EnvironmentalNodecan have the following datatype property assertions:

- hasEnvironmentalNodeConfidenceValue: it indicates the confidence value (from 0.0 to 1.0) of the environmental node.
- hasEnvironmentalNodeDistanceFromNearestStreetCrossing_meter: it indicates the distance (in meters) of the environmental node from the nearest street crossing (especially for stations).
- hasEnvironmentalNodeName: it indicates the name (if any) of the environmental node.
- hasEnvironmentalNodeTrafficVolumeWithin50m_vehiclePerDay: it indicates the traffic volume (in vehicles per day) in the range of 50 meters from the environmental node (especially for stations).
- hasEnvironmentalNodeURL: it indicates the URL the environmental node (especially for web-sites and web-services).

Additional classes:

EnvironmentalNodeAreaType, EnvironmentalNodeForm, EnvironmentalNodeLandUseType, EnvironmentalNodeSourceOfEmissionType, EnvironmentalNodeType

4.6 Module: Exceedance Data (pescadoExceedances.owl)

Class **ExceedanceData**

This class is used to represent an exceedance data for what concerns air pollutant data.

Assertions that can be made on individuals of these classes are:

- hasExceedanceDataEnvironmentalData: the target of this property are individuals of class EnvironmentalData. It is used to indicate the environmental data that are causing the exceedance.
- hasExceedanceDataExceedanceType: the target of this property are individuals of class ExceedanceType. It is used to indicate the type of exceedance that occurred.

Class **ExceedanceType**

This class is used to represent the possible types of exceedances that are considered in the system. Assertions that can be made on individuals of these classes are:

- hasExceedanceTypeAirPollutant: the target of this property are individuals of class AirPollutantDataType. It is used to indicate the air pollutant this type of exceedance is about.
- hasExceedanceTypeAirPollutantDataTimeInterval: the target of this property are individuals of class AirPollutantDataTimeInterval. It is used to indicate the type of time granularity (1h, 8h, 24h) of the air pollutant date the exceedance type refers to.
- hasExceedanceTypeHSYShortTermActionPlanActionType: the target of this property are individuals of class HSYShortTermActionPlanActionType. It is used to indicate the HSY short term action plan that should be undertaken if an exceedance of this exceedance type occurs.
- hasExceedanceTypeHistoricalDataPeriod: the target of this property are individuals of class HistoricalDataPeriod. It is used to indicate if the exceedance type refers to average value of the air pollutant concentration of a year.
- hasExceedanceTypeNumberOfExceedancePerYear: the target of this property are individuals of class NumberOfExceedancePerYear. It is used to indicate how many times this type of exceedance this can be exceeded in a year.
- hasLowerBoundType, hasLowerBoundType: the target of these properties are individuals of class BoundType. It is used to indicate the specific type of bound corresponding to threshold associated to the exceedance type (e.g. greater or equal than, lesser than).

The individual of type EnvironmentalNodecan have the following datatype property assertions:

- hasExceedanceTypeComputationProcedure: it indicates the computation procedure that the system should use to calculate if an exceedance of this type has occurred.
- hasExceedanceTypeForecastToContinue: it indicates whether in order to trigger an exceedance of this type, a forecast that the condition will continue is also required.
- hasExceedanceTypeMinimumNumberOfStations: it indicates whether in order to trigger an exceedance of this type, there is a minimum number of stations (>1) in which the condition associated to the exceedance should occur.
- hasExceedanceTypeNumberOfConsecutiveDays: it indicates whether in order to trigger an exceedance of this type, the condition associated to the exceedance should occur for at least a certain number of consecutive days.

- hasExceedanceTypeNumberOfConsecutiveHours: it indicates whether in order to trigger an exceedance of this type, the condition associated to the exceedance should occur for at least a certain number of consecutive hours.
- hasLowerBoundValue, hasLowerBoundValue: they are used to indicate the value of the threshold associated to the exceedance type (e.g. greater or equal than, lesser than).

Several sub-classes of ExceedanceData have been identified, according to the organization that issues the exceedance threshold limits: AirPollutantEUInformationThreshold, AirPollutantEULimitValue, HumanHealthAirPollutantEULimitValue, AirPollutantEUTargetValue, HumanHealthAirPollutantEUTargetValue, AirPollutantEUWarningThreshold, AirPollutantWHOGuideline, HelsinkiShortTermActionPlan.

Class **StatisticalExceedanceData**

This class is used to represent statistics about exceedances, like e.g. the number of exceedance in a give period of time.

Assertions that can be made on individuals of these classes are:

- hasStatisticalExceedanceDataExceedanceType: the target of this property are individuals of class ExceedanceType. It is used to indicate the type of exceedance that is considered by this statistical data.
- hasStatisticalExceedanceDataLastExceedanceData: the target of this property are individuals of class ExceedanceData. It is used to indicate the last exceedance occurred within the period considered.
- hasStatisticalExceedanceDataNumberOfExceedances: the target of this property are individuals of class NumberOfExceedances. It is used to indicate the number of exceedances occurred within the period considered.
- hasStatisticalExceedanceDataNumberOfExceedancesInExceedance: the target of this property are individuals of class NumberOfExceedances. It is used to indicate the number of exceedances past the maximum number allowed by the exceedance type, occurred within the period considered.

The individual of type StatisticalExceedanceData can have the following datatype property assertions:

- hasFromDateTime (every StatisticalExceedanceData should have exactly one of these assertions): it indicates the starting date / time to which the data refers to;
- hasToDateTime (every StatisticalExceedanceData should have exactly one of these assertions): it indicates the ending date / time to which the data refers to.

Class **ExceedanceAggregationData** [new in version 3.0]

This class is used to aggregate exceedance data of the same exceedance type over a period of time.

The individual of type ExceedanceAggregationData can have the following object property assertions:

- hasStatisticalExceedanceDataExceedanceType: the target of this property are individuals of class ExceedanceType. It is used to indicate the type of exceedance that is considered by this aggregated data.

- hasExceedanceAggregatedDataExceedanceData: the target of this property are individuals of class ExceedanceData. It is used to indicate an exceedance data occurred within the period considered.
- hasExceedanceAggregatedDataFirstExceedanceData: the target of this property are individuals of class ExceedanceData. It is used to indicate the first exceedance data (chronologically) occurred within the period considered.

The individual of type ExceedanceAggregationData can have the following datatype property assertions:

- hasExceedanceAggregationDataNumber: to indicate the number of exceedance data in the period;
- hasFromDateTime (every ExceedanceAggregationData should have exactly one of these assertions): it indicates the starting date / time to which the data refers to;
- hasToDateTime (every ExceedanceAggregationData should have exactly one of these assertions): it indicates the ending date / time to which the data refers to

Additional classes:

NumberOfExceedances, HSYShortTermActionPlanActionType

4.7 Module: Conclusions (pescadoConclusions.owl)

Class **Conclusion**

This class is used to represent the possible conclusions (warnings, conclusions) that are generated by the system.

The individual of type Conclusion can have the following datatype property assertions:

- hasConclusionWeight (every Conclusion should have exactly one of these assertions): it is used to represent how strong/important is the conclusion generated by the system.

The class has to main sub-classes: Suggestion and Warning.

Class **Reccomendation**

This class is used to represent the recommendations that are generated by the system.

Assertions that can be made on individuals of these classes are:

- hasRecommendationRecommendationType (every Recommendation should have exactly one of these assertions): the target of this property are individuals of class RecommendationType. It is used to indicate the type of recommendation to be reported.

Class **Warning**

This class is used to represent the warnings that are generated by the system.

Assertions that can be made on individuals of these classes are:

- hasWarningWarningType (every Warning should have exactly one of these assertions): the target of this property are individuals of class WarningType. It is used to indicate the type of warning to be reported.

Class **R**ecommendation**Type**

This class is used to represent the type of recommendations that can be generated by the system. The message to be reported to the user is stored (with language tag) in an rdf:message annotation. The type of recommendations is organized in sub-classes, according to the environmental data type that triggers the recommendation.

Class **WarningType**

This class is used to represent the type of warnings that can be generated by the system. The message to be reported to the user is stored (with language tag) in an rdf:message annotation. The type of warnings is organized in sub-classes, according to the environmental data type that triggers the warning.

Class **Explanation** [new in version 3.0]

This class (subclass of Conclusion) is used to represent possible explanations of high pollutants concentration values, that are generated by the system. Explanation consist of two aspects:

- “cause” (modeled in explanationType class): a possible reason for a high value pollutant concentration;

- “solution” (modeled in explanationType class): suggestions of other things to check to confirm the cause proposed by the system;

Assertions that can be made on individuals of these classes are:

- hasExplanationExplanationType (every Explanation should have exactly one of these assertions): the target of this property are individuals of class ExplanationType. It is used to indicate the type of warning to be reported.

Class ExplanationType

This class is used to represent the type of explanations that can be generated by the system. The message to be reported to the user is stored (with language tag) in an rdf:message annotation. The type of warnings is organized in sub-classes, according to the environmental data type that triggers the warning.

Assertions that can be made on individuals of these classes are:

- hasExplanationTypeSolutionType: the target of this property are individuals of class SolutionType. It is used to indicate the type of solution/suggestion to be reported.

Class SolutionType

This class is used to represent the type of solutions/suggestions that can be generated by the system, if a cause for high pollutant concentration if given. The message to be reported to the user is stored (with language tag) in an rdf:message annotation. The type of warnings is organized in sub-classes, according to the environmental data type that triggers the warning.

4.8 Module: Logico-semantic relation (pescadoNLP.owl)

Class **LogicoSemanticRelation**

This class is used to represent logico-semantic relations between the individuals stored in the PESCaDO ontology. It does not refer to specific PESCaDO content

Assertions that can be made on individuals of these classes are:

- hasLogicoSemanticRelationArgument1 (for all relations except ListRelation): the target of this property are individuals of the ontology. It is used to indicate the first argument of the relation considered.
- hasLogicoSemanticRelationArgument2 (for all relations except ListRelation): the target of this property are individuals of the ontology. It is used to indicate the second argument of the relation considered.
- hasLogicoSemanticRelationArgument3 (only for ImplicationRelation and ViolationOfExpectationRelation): the target of this property are individuals of the ontology. It is used to indicate the third argument of the relation considered.
- hasLogicoSemanticRelationSubType: the target of this property are individuals of class LogicoSemanticRelationSubType. It is used to indicate the particular subtype of the relation.
- isLogicoSemanticRelationAssociatedToRequest (every LogicoSemanticRelationshould have exactly one of these assertions): the target of this property are individuals of class Request. It is used to indicate to what request this particular relation is associated.

The class has several sub-classes according the type of relation considered: CauseRelation, ContrastRelation, ElaborationRelation, ImplicationRelation, ListRelation, ViolationOfExpectationRelation.

Class **CauseRelation**

This class is used to represent the cause logico-semantic relations.

Assertions that can be made on individuals of these classes are:

- hasCauseRelationCauseRelationPolarityType: the target of this property are individuals of class CauseRelationPolarityType. It is used to indicate the particular polarity of the cause relation (e.g. positive, negative or neutral).

An additional restriction is imposed to the class to ensure that the LogicoSemanticRelationSubType asserted via the hasLogicoSemanticRelationSubType is a member of the class CauseRelationSubType.

Class **ContrastRelation**

This class is used to represent the contrast logico-semantic relations. An additional restriction is imposed to the class to ensure that the LogicoSemanticRelationSubType asserted via the hasLogicoSemanticRelationSubType is a member of the class ContrastRelationSubType.

Class **ElaborationRelation**

This class is used to represent the elaboration logico-semantic relations. An additional restriction is imposed to the class to ensure that the LogicoSemanticRelationSubType asserted via the hasLogicoSemanticRelationSubType is a member of the class ElaborationRelationSubType.

Class **ImplicationRelation**

This class is used to represent the implication logico-semantic relations. An additional restriction is imposed to the class to ensure that the LogicoSemanticRelationSubType asserted via the hasLogicoSemanticRelationSubType is a member of the class ImplicationRelationSubType.

Class **ListRelation**

This class is used to represent the list logico-semantic relations. An additional restriction is imposed to the class to ensure that the LogicoSemanticRelationSubType asserted via the hasLogicoSemanticRelationSubType is a member of the class ListRelationSubType.

Assertions that can be made on individuals of these classes are:

- hasListRelationFirstListRelationElement, hasListRelationListRelationElement: the target of these properties are individuals of class ListRelationElement. They are used to indicate the first and any ListRelationElement in the list currently considered.

Class **ViolationOfExpectationRelation**

This class is used to represent the violation of expectation logico-semantic relations.

Class **ListRelationElement**

This class is used to represent the elements of list logico-semantic relations.

Assertions that can be made on individuals of these classes are:

- hasListRelationElementAttribute: the target of this property is any individual of the ontology. It is used to indicate the actual individual of the ontology which is part of the given list.
- hasListRelationElementNextListRelationElement: the target of this property is an individual of class ListRelationElement. It is used to indicate the ListRelationElement following the current individual in the list.

Additional Classes:

LogicoSemanticRelationSubType, CauseRelationSubType, ContrastRelationSubType, ElaborationRelationSubType, ImplicationRelationSubType, ListRelationSubType, CauseRelationPolarityType

4.9 Module: Base (pescadoBase.owl)

This module defines some basic annotation properties, as well as the toDateTime and fromDateTime datatype properties, used in several modules

References

[Suárez-Figueroa et al., 2009] Suárez-Figueroa, MC. and Gómez-Pérez, A. and Villazón-Terrazas, B. How to write and use the Ontology Requirements Specification Document. In: The 8th International Conference on Ontologies, DataBases, and Applications of Semantics, ODBASE 2009, 03/11/2009 - 04/11/2009, Vilamoura, Algarve-Portugal.