Documentation on project work of "Gr@phBrains" tourism domains ontology expansion

Subject: Artificial Intelligence

Teodors Lisovenko (t.lisovenko@studenti.uniba.it)

Delivered: TBA

Abstract: *The following documentation presents an ongoing textual summary of the performance of the subject's project work. It describes the work related to the "**Gr@phBrain**" project, particularly in its tourism domain. Documentation starts with an examination and description of the current domain's state of affairs which followingly would allow grounding the floor for its expansion. To be continued…*

Table of Contents

1 Introduction

Ontologies building (not mentioning the maintenance) is an iterative process, where each iteration is new toping onto its existing state. It may be referred to as the process of expansion. This project work is dealing with the domain of tourism or the knowledge base which tries to describe all related things that we-humans associate at the struck of vacation session. This project work regards itself as an expansion as it takes already existing ontology, which is provided by the *"Gr@phBrain"* project to enrich it with the new classes, properties, relationships between components, etc.to achieve more rich individuals or instances of the domain. This on another hand will provide consumers with *"Gr@phBrain"* service to ask more (in numerical and specifical aspects) questions and provide more rich answers and relations between the instances of the domain. But before anything (to appropriately expand) it is needed to describe the status quo of the domain and concurrently try to spot the places where one could contribute by supplementing it with some insufficiencies or possible additions.

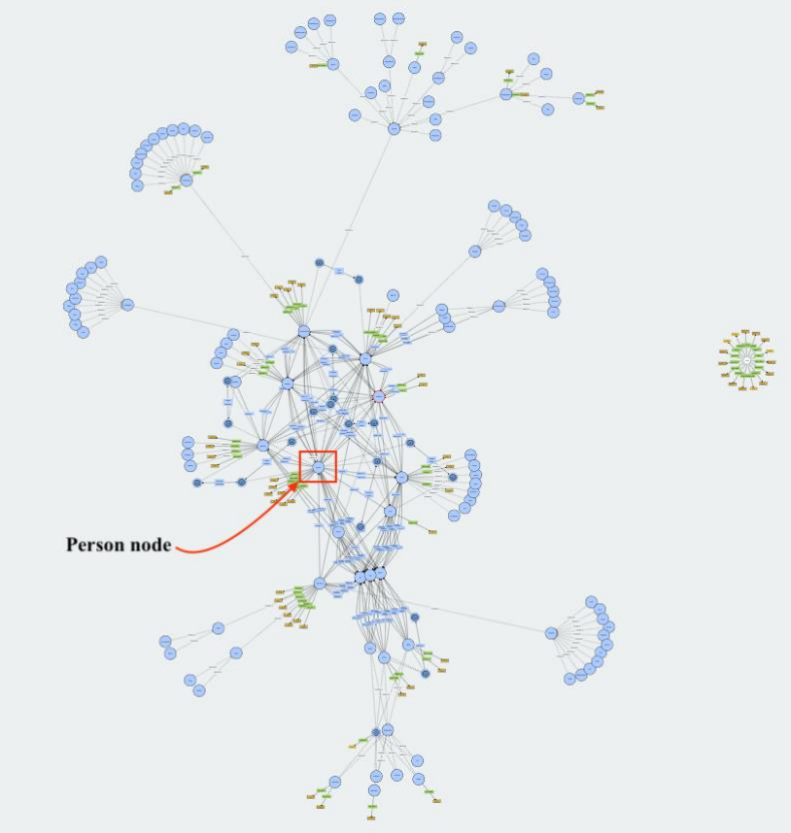


Figure 1. Tourism domain ontology's structure visualization with WebVOWL.

2 Tourism domains structural overview

Its exploration of the domain could be started by an abstract visualization of its whole structure. As “Gr@phBrain” provides an export of ontology to OWL format it is possible to use a well know tool for ontology building community of VOWL which is a standard with its specification for “visual notion for OWL ontologies” (it has also a web version “WebVOWL” which will be used further). The overview “from above” could be seen in 1. figure. The previous figure reviles that the ontology may be considered “tightly connected”, but there are few branches or “neighborhoods” that spawn outside the central network. The “WebVOWL” has one similarity to “Gr@phBrain” as it uses graphs for visualization and specifically “Neo4J”. At least visually it could be noted that the most connected node in the ontology graph is **“Person”** (which later is confirmed in its dedicated section) fashioning it the focal point in the network. The “WebVOWL” also provides plain facts about the current state of the domain (presented in 2. figure).

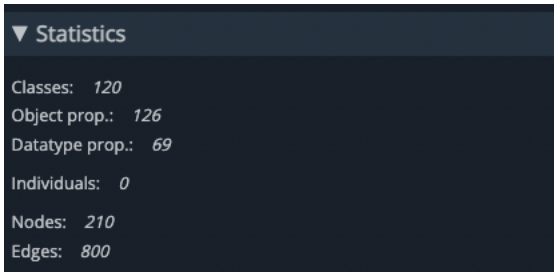


Figure 2. WebVOWL statistics of the tourism domain.

The 2. figure indicated zero individuals because only the ontologies schema was exported from “Gr@phBrain” which excludes the instances of the domain. The total number of classes may seem considerable at the beginning (as it goes beyond one hundred), but they could be sorted into 9 main classes, which leaves the rest 112 classes as their subclasses (3. figure).

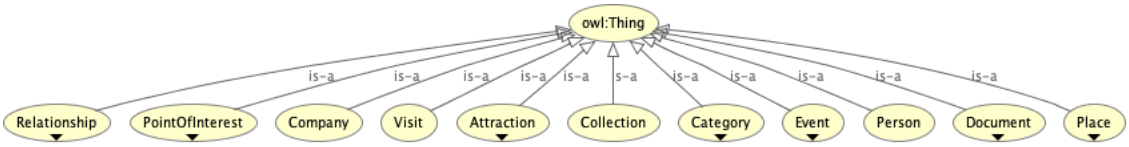


Figure 3. Top class hierarchy of tourism domain.

The 1. figure was decent to see ontology’s structure unplotted and undetailed “from above”, but to examine more closely one may use the “OWLViz” plugin in the “Protégé”

application. It reviles in 3. figure the 11 main top-level classes that may describe the instances of present tourism ontology.

3 Tourism domains XML description

Before “drill-downing” previously presented top-level classes it could be useful to inspect ontology XML schema. The proper schema could be seen in the 1. attachment as the following destructed XML snippets will be cleaned from some elements to improve the presentably and narrative.

The first level into which everything is wrapped is element of **“domain”**:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="domain">
        <!-- Contents -->
    <xs:attribute type="xs:string" name="name"/>
  </xs:element>
</xs:schema>
```

From upper schema it could indicated that the element that contains all of the sub components of ontology have only one string type attribute for its name. Furthering to the next sub section there are only two of them – **“entities”** and **“relationships”**:

```
<xs:element name="entities">
    <xs:element name="entity" maxOccurs="unbounded" minOccurs="0">
        <!-- Contents -->
      <xs:attribute type="xs:string" name="name" use="optional"/>
    </xs:element>
</xs:element>
<xs:element name="relationships">
    <xs:element name="relationship" maxOccurs="unbounded" minOccurs="0">
        <!-- Contents -->
      <xs:attribute type="xs:string" name="inverse" use="optional"/>
      <xs:attribute type="xs:string" name="name" use="optional"/>
    </xs:element>
</xs:element>
```

Both entities and relationship elements contain their consequent singular elements of **“entity”** and **“relationships”** and they are numerically unrestricted – they can be as many as needed or be absent (indicated in “maxOccurs” and “minOccurs” attributes). Moreover **“entity”** and **“relationship”** elements may have their optional names and additionally for

relationships also the attribute of “inverse,” tells that it is bidirectional. Continuing the **“entity”** branch it follows with another two elements of **“attributes”** and **“taxonomy”**. Each of them contains its consequent singular elements with may be absent, but of **“attributes”** unbonded:

```
<xs:element name="attributes">
      <xs:element name="attribute" maxOccurs="unbounded" minOccurs="0">
</xs:element>
<xs:element name="taxonomy" minOccurs="0">
      <xs:element name="value" maxOccurs="unbounded" minOccurs="0">
</xs:element>
```

“Attributes” are fourfold descriptive as they may describe datatype, name, target or indicate that this specific attribute is not nullable.

```
<xs:attribute type="xs:string" name="datatype" use="optional"/>
<xs:attribute type="xs:string" name="mandatory" use="optional"/>
<xs:attribute type="xs:string" name="name" use="optional"/>
<xs:attribute type="xs:string" name="target" use="optional"/>
```

As attribute can be describe such things, they eventually hold values for it:

```
<xs:element name="values" minOccurs="0">
      <xs:element name="value" maxOccurs="unbounded" minOccurs="0">
         <xs:extension base="xs:string">
           <xs:attribute type="xs:string" name="name" use="optional"/>
      </xs:element>
</xs:element>
```

Attribute may have multiple values and all of them are described in string they as previous xml snippet dictates. That would be all for the **“attributes”** of the **“entity”**, but there was also the **“taxonomy”** element and it has only one sub element of **“value”**:

```
<xs:element name="value" maxOccurs="unbounded" minOccurs="0">
      <xs:element name="attributes" minOccurs="0">
      <xs:element name="taxonomy" minOccurs="0">
   <xs:attribute type="xs:string" name="name" use="optional"/>
</xs:element>
```

The “Value” element can have an only optional name. Similarly, with **“attributes”** the taxonomy can have its **“attributes”** and, interestingly, another element of named

“taxonomy”. Of course, there is no conflict as the definition of the elements with the same name is located in different places and levels – it is generally not recommended to create confusion. Probably the first **“taxonomy”** element should be renamed from singular to plural as it serves just as a container.

XML element of **“attributes”** equally to one that was present for **“entity”** element can hold unlimited count of **“attribute”** elements. They in “string” format can describe the its name, datatype, target and if it can be nullable.

```
<xs:element name="attribute" maxOccurs="unbounded" minOccurs="0">
  <xs:extension base="xs:string">
    <xs:attribute type="xs:string" name="datatype" use="optional"/>
    <xs:attribute type="xs:string" name="mandatory" use="optional"/>
    <xs:attribute type="xs:string" name="name" use="optional"/>
        <xs:attribute type="xs:string" name="target" use="optional"/>
</xs:element>
```

What is interesting in the case of the “**taxonomy”** element, which holds element **“value”** and its subsequent **“attributes”** and the duplication of the selfsame element of **“taxonomy”** – is that the **“taxonomy”** could be labeled as recursive (even if such term is unfitting for XML). It means that each subsequent **“taxonomy”** element mirrors the same substructure as the initial **“taxonomy”** element. Pseudocode for this case could be the following:

```
<!-- First level -->
<xs:element name="taxonomy">
  <xs:element name="value">
    <xs:element name="attributes">
    <!-- Second level -->
    <xs:element name="taxonomy">
      <xs:element name="value">
        <xs:element name="attributes">
        <!-- Third level -->
        <xs:element name="taxonomy">
          <xs:element name="value">
            <!-- Forth level and so on... -->
```

This concludes the **“entity”** element. The second branch was the **“relationships”** container element, that can hold unlimited sub elements of **“relationship”**. It can have their two sub elements that describe the relationship via element **“attributes”** and its involved actors via element of **“references”**:

```
<xs:element name="relationship" maxOccurs="unbounded" minOccurs="0">
   <xs:element name="references">
   <xs:element name="attributes">
   <xs:attribute type="xs:string" name="inverse" use="optional"/>
   <xs:attribute type="xs:string" name="name" use="optional"/>
</xs:element>
```

Element **“references”** holds **“reference”** element, which could be described in terms of object and subject – both of them is meant for entity names that form a relationship:

```
<xs:element name="references">
      <xs:element name="reference" maxOccurs="unbounded" minOccurs="0">
         <xs:extension base="xs:string">
            <xs:attribute type="xs:string" name="object" use="optional"/>
            <xs:attribute type="xs:string" name="subject" use="optional"/>
      </xs:element>
</xs:element>
```

As the reference elements are needed for set relationship connectivity between classes – relationship also must be described and for that is the element attributes (equally to previous **“attributes”** elements it holds its consequent singular attribute that describes name, datatype, etc.):

```
<xs:element name="attributes" minOccurs="0">
      <xs:element name="attribute" maxOccurs="unbounded" minOccurs="0">
         <xs:extension base="xs:string">
            <xs:attribute type="xs:string" name="datatype" use="optional"/>
            <xs:attribute type="xs:string" name="mandatory" use="optional"/>
            <xs:attribute type="xs:string" name="name" use="optional"/>
      </xs:element>
</xs:element>
```

4 Tourism domain’s class taxonomy

All all-further subsections will describe the domains taxonomy going through all 9 top-level classes. In each of them will be presented a top-level class definition, its attributes, and if exists also its following sub-taxonomy.

4.1 Person

The foremost class and element of tourism is the subject of which all its derived phenomenology (shops for tourists, touristic attractions, etc.) works around. In the current domain, it is represented as the class **"Person"**. It does not have inferior taxonomy (no subclasses). Graphicly **"Person"** could be described simplistically with two nodes (4. figure).

Figure 4. Taxonomy of class "Person".

Furthermore, it can be described with the following attributes upon the creation of **"Person"** instances (Table 1.):

Table 1. - Person class attribution.

Nr.	XML name	Mandatory	Data type	Enumeration	Described as another class of
1.	**name**	Yes	string	-	-
2.	**surname**	Yes	string	-	-
3.	**title**	No	string	-	-
4.	**knownAs**	No	string	-	-
5.	**nickname**	No	string	-	-
6.	**gender**	No	select	1. M 2. F 3. Other	-
9.	**bornIn**	No	entity	-	Place
10.	**diedIn**	No	entity	-	Place
11.	**birthDate**	No	date	-	-
12.	**deathDate**	No	date	-	-
13.	**nationality**	No	string	-	-
14.	**job**	No	string	-	-

4.1 Place

Tourism primary exists because of places that tourists are willing to see. Usually, the places are the ones which are located in “open or closed air spaces” (for example “lakes” or “museums”) The current taxonomy to encircle the range of touristic places are described in Figure 6.

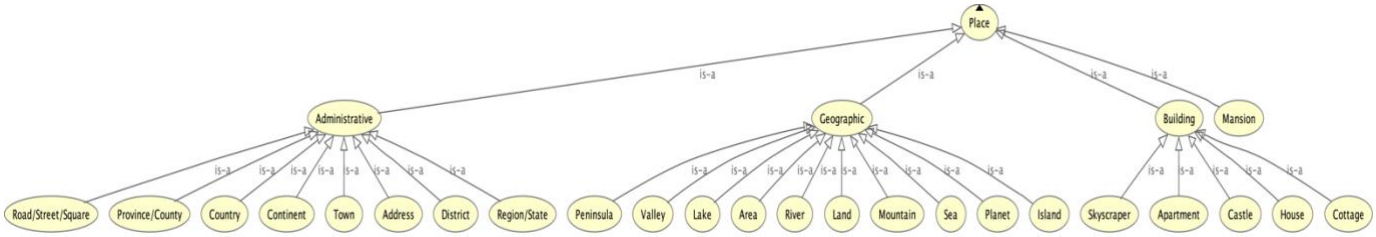


Figure 5. Taxonomy of class “Place”.

The **“Place”** class unlike the **“Person”** has more than 2 nodes as it provides two-level deep taxonomy. **“Place”** can be characterized as administrative, geographic, building, or mansion. For the first three, there is an extensive set of subclasses that allow describing a place more specifically as it can be building-like (e.g., castle) or administrative (e.g., district). The mansion is left out of any specialties and as such may be placed under the possibility for expansion or at least relocated underclass **“Buildings”**.

Nerveless **“Place”** has quite self-describing taxonomy for which additionally upon instance creation it has the attributes that allow setting the coordinates (latitude and longitude) and other seeable details from the following table:

Table 2. - Place class attributes

Nr.	XML name	Mandatory	Data type	Enumeration	Described as another class of
1.	**name**	Yes	string	-	-
2.	**language**	No	string	-	-
3.	**latitude**	No	real	-	-
4.	**longitude**	No	real	-	-
5.	**description**	No	string	-	-

4.1 Point of interest

A modern tendency of tourists is to have some sort of map with pinpoints of "interesting" and worth visiting places. For such a case in the domain exists the class **"PointOfInterest"**. It following taxonomy is divided between **"Service"** and **"PlaceToVisit"**. The **"PlaceToVisit"** has some sort of resemblance with the **"Place"** classes **"Geographic"** and **"Building"** subclasses. One interpretation could be that **"PointOfInterest"** is an extension to the **"Place"** (also from 3. Table one may see that **"Place"** includes under the **"PointOfInterest"** definition) as they both refer to the geographic dimension. But at the same time, there can be confusion as, for example, there can be a class for **"Castle"** when defining the instance of **"Place"** and castles usually is an object of tourism. "Place" is more general and also is reused in other class definitions like **"Person"**, but it does contain taxonomy which is more fitting for the **"PointOfInterest"** like **"Castle"** or **"Skyscraper"**. The **"PointOfInterest"** also has a third level of sub-taxonomy that give more specification for **"HealthService"**, **"Station"** and others that can be seen in 6. Figure.

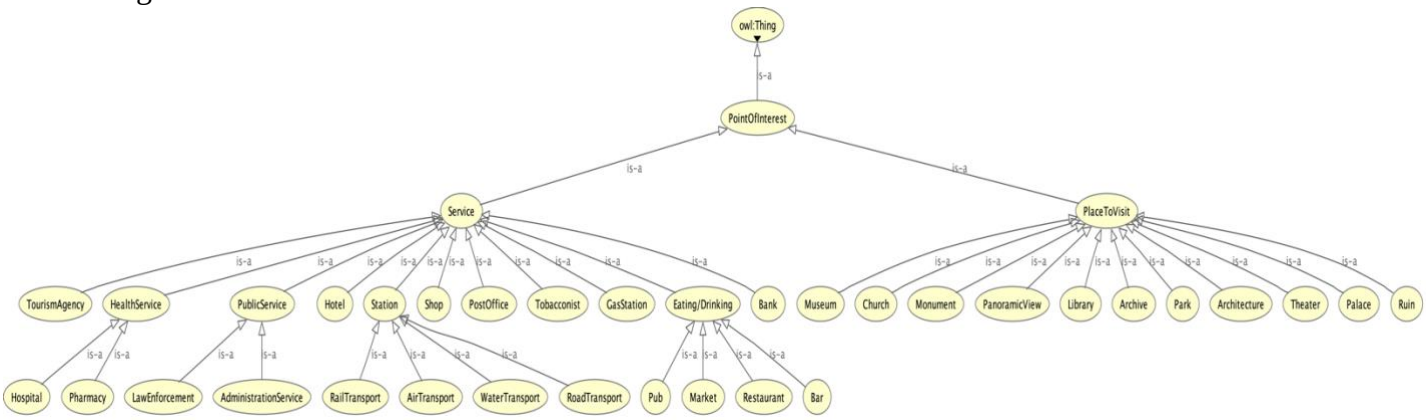


Figure 6. Taxonomy of class "PointOfInterest".

Table 3. – "PointOfInterest" class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
name	Yes	string	-	-
place	No	entity	-	Place
description	No	text	-	-

address	No	string	-	-
phone	No	string	-	-
estimatedCost	No	real	-	-

4.2 Visit

The last element that sets the tourism triangle of person and place is the action of actual tourism and in the domain for such is the **"Visit"** class. Figure 7. unravellers that there is no further taxonomy for the class.

Figure 7. Taxonomy of class "Visit".

The attributes of **"Visit"** can specify only the period of a stay and the budged (4. table). This modesty in. definition may be an opportunity to expand it in this paper's later section of the proposal.

Table 4. – Visit class attributes

Nr.	XML name	Mandatory	Data type	Enumeration	Described as another class of
1.	**startDate**	Yes	date	-	-
2.	**endDate**	No	date	-	-
3.	**budged**	No	real	-	-

4.3 Event

Of course, one type of tourism may be hosted in a museum or palace, but the phenomenon that describes it as more than just "being in place" or visiting is the participation in attendance or participation in an **"Event"**. For such occasions, there is a class for it in the domain. Its sub-taxonomy can be viewed in 8. figure. It includes a variety of events like concerts, exhibitions, etc., but as the set is not exhaustive it may have some other types of events to add up in extension. Additionally, to it upon the creation of the **"Event"** instance, there are other details to set the events period, description, and other details from the 5. table.

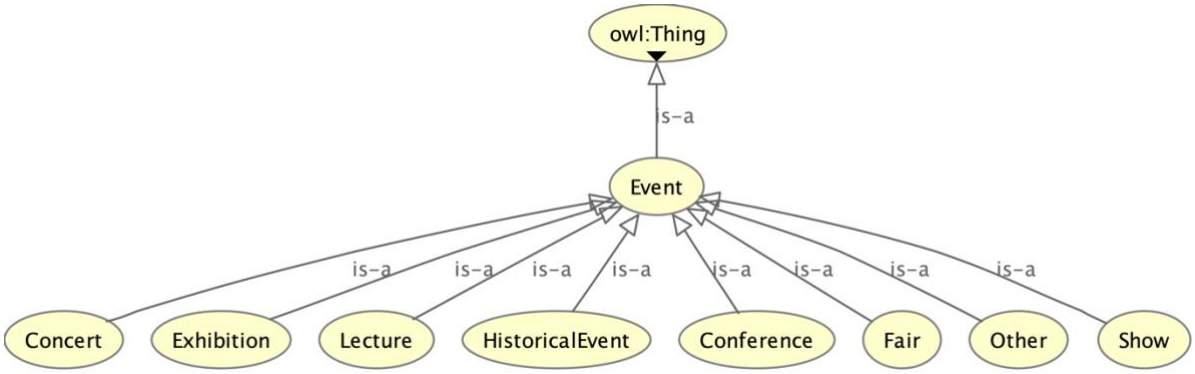


Figure 8. Taxonomy of class "Visit".

Table 5. – Event class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
name	Yes	string	-	-
acronym	No	string	-	-
description	No	string	-	-
startDate	No	date	-	-
endDate	No	date	-	-

4.4 Attraction

Another term that is important in tourism vocabulary is the notion of **"Attraction"**. If the "**PointOfInterest"** relates to "something of interest" in a geographical sense, then **"Attraction"** can be considered similarly but in sense of an object or thing. Figure 8. shows the state the of **"Attraction"** class sub-taxonomy. Currently, it consists just of **"Painting"**, **"WorkOfArt"** and **"Sculpture"**.

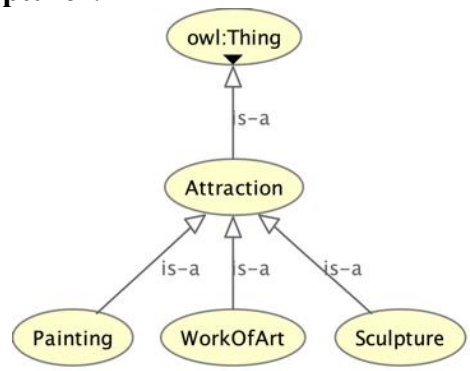


Figure 8. Taxonomy of class "Attraction".

Additionally, the domain offers to set a few attributes for the object of attraction like technique and material and other ones (6. table) for richer informational context.

Table 6. – Attraction class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
name	Yes	string	-	-
description	No	string	-	-
date	No	date	-	-
technique	No	string	-	-
material	No	string	-	-

4.5 Document

The touristic experience can be and usually is enhanced with supporting material that provides a piece of additional information about the place or the object in form of audio guides or leaflets to name a few. These informational resources can be considered as an indivisible part of the touristic activity and for such, there is a class **"Document"**. In 9. figure sub-taxonomy gives a possibility for creating documentations instance as **"Video"**, **"Audio"** and **"Printable"** furthering down with more specific classes. Similarly, to the rest of the classes, there is the possibility to set additional attributes seeable in 7. Table.

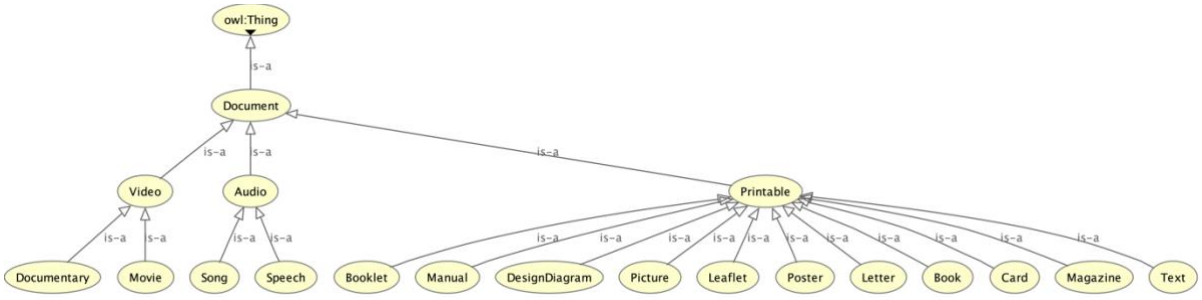


Figure 9. Taxonomy of class "Attraction".

Table 7. - Attraction class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
title	Yes	string	-	-

id	No	string	-	-
language	No	string	-	-
edition	No	string	-	-
copyright	No	date	-	-
format	No	string		
length	No	string		
date	No	date		
originalPrice	No	real		

4.6 Category

A supporting class in the tourism domain for classification and setting kinds, types, and varieties for the instances is **"Category".** It provides a possibility to set a category in terms of trends, concepts, periods, subjects (10. figure) or just stuff which combined with the ability of relationships allows the group.

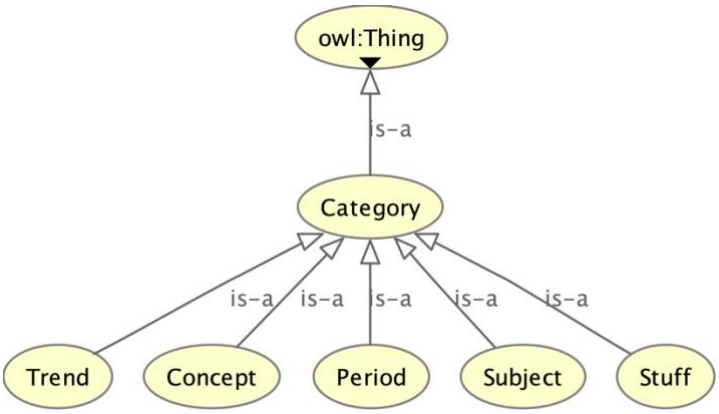


Figure 10. Taxonomy of class "Category".

A few extra details to add is seeable in the 8. table, which are name, id, and description of the category.

Table 8. - Category class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
name	Yes	string	-	-
id	Yes	string	-	-
description	No	date	-	-

4.7 Collection

Alongside the **"Category"** class exists **"Collection"** that offers a similar mechanism of grouping. It is another clustering class that gives the same possibility to group up some instances. More describing information is relationships onto which their collections hold and for that refer to the corresponding section discussing relationships. There are no subclasses for **"Collection"** (11. figure) and unlike the **"Category"** class the type of collection is specified in attributes of instance. As 9. table suggests there can be a collection of an enumeration type family, group, place, series, or other.

Figure 11. Taxonomy of class "Collection".

Table 9. - Event class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
name	Yes	string	-	-
acronym	No	string	-	-
type	No	select	1. Family 2. Group 3. Place 4. Series 5. Other	-

5 Tourism domains relationships

What makes a network or ontology is the relationships between instances on a more abstract level between classes. To know about these relationships, it is important to cover the currently defined relationship types. In total there are 18 relationships. Technically 9 relationships, but definition like e.g., **"developed"** is inversed variation of **"developedBy".** Definitions are described by their attributes (there can also be ones with the declaration, but without any attributes, which is the case for some of the tourism relationships). Attributes have 3 parameters – its name, datatype (can be string, date, etc.) and mandatory (nullable or not). All of them are listed below:

1. **“concerns”** (inversed as **“citedIn”**)
2. **“belongsTo”** (inversed as **“includes”)**
3. **“developed”** (inversed as **“developedBy”**) Attributes:
    - **“role”**, type string, not mandatory.
    - **“order”**, type integer, not mandatory.
4. **“isA”** (inversed as **“kindOf”**)
5. **“makes”** (inversed as **“madeBy”**) Attributes:
    - **“role”**, type integer, mandatory.
6. **“owned”** (inversed as **“ownedBy”**) Attributes:
    - **“quantity”**, type integer, not mandatory.
    - **“startDate”**, type date, not mandatory.
    - **“endDate”**, type date, not mandatory.
    - **“public”**, type boolean, not mandatory.
7. **“partOf”** (inversed as **“hasPart”**) Attributes:
    - **“startDate”**, type date, mandatory.
    - **“endDate”**, type date, not mandatory.
8. **“relevantFor”** (inversed as **“pertains”**) Attributes:
    - **“role”**, type string, not mandatory.
9. **“wasIn”** (inversed as **“hosted”**). Attributes:
    - **“reason”**, type string, not mandatory.
    - **“address”**, type string, not mandatory.
    - **“startDate”**, type date, not mandatory.
    - **“endDate”**, type date, not mandatory.

With these at hand, every present relationship is defined in the current state of the domain. To illustrate better for each top class there will be a graph and a formal listing detailing the subject-object of the relationship.

5.1 Person

Tourism concentrates around its subjects and naturally, it is the focal class that has relationships with all other top-level classes (12. figure). Except with only **“Visit”** the **“Person”** have multiple relationships with other classes under different definitions like with **“Attraction”** – it may be under **“owned”** or **“developed”.**

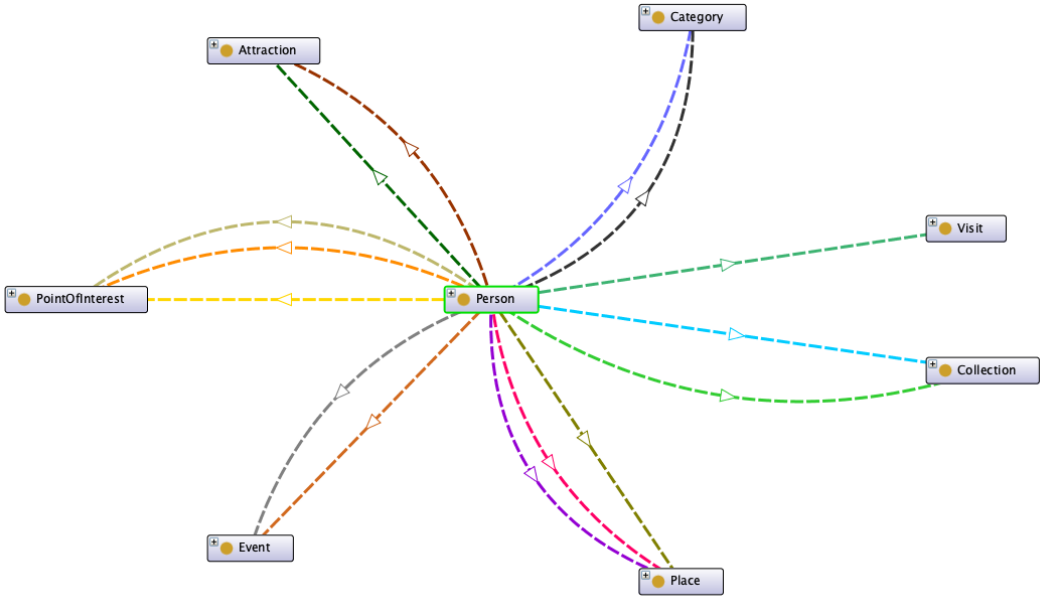


Figure 12. Relationships of class “Person”.

These classes are presented in 12. figure is involved in the following relationships listed below:

1. from **“Person”** to **“Document”** by **“citedIn”** (inverse as **“concerns”**).
2. from **“Person”** to **“Category”** by **“belongsTo”** (inverse as **“includes”**).
3. from **“Person”** to **“Category”** by **“pertains”** (inverse as **“releventFor”**).
4. from **“Person”** to **“Attraction”** by **“developed”** (inverse as **“developedBy”**).
5. from **“Person”** to **“Attraction”** by **“owned”** (inverse as **“ownedBy”**).
6. from **“Person”** to **“Event”** by **“wasIn”** (inverse as **“hosted”**).
7. from **“Person”** to **“Event”** by **“makes”** (inverse as **“madeBy”**).
8. from **“Person”** to **“PointOfInterest”** by **“owned”** (inverse as **“ownedBy”**).
9. from **“Person”** to **“PointOfInterest”** by **“releventOf”** (inverse as **“pertains”**).
10. from **“Person”** to **“PointOfInterest”** by **“wasIn”** (inverse as **“hosted”**).
11. from **“Person”** to **“Collection”** by **“partOf”** (inverse as **“hasPart”**).
12. from **“Person”** to **“Collection”** by **“wasIn”** (inverse as **“hosted”**).

13. from **“Person”** to **“Place”** by **“wasIn”** (inverse as **“hosted”**).
14. from **“Person”** to **“Visit”** by **“makes”** (inverse as **“madeBy”**).
15. from **“Person”** to **“Period”** by **“wasIn”** (inverse as **“hosted”**).

5.2 Place

Class **“Place”** has relationships with most of the top-level classes as 12. figure shows that.

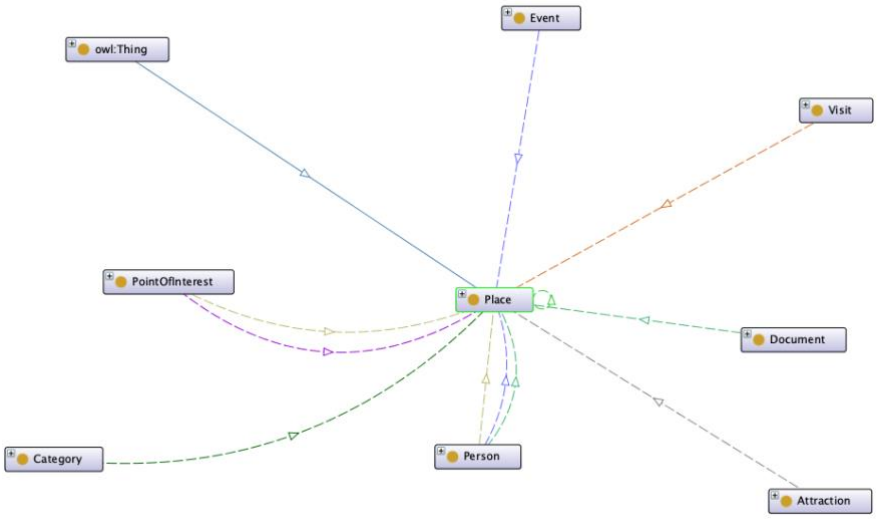


Figure 12. Relationship of class “Place”.

The formal listing of all of them could be seen below. It should be noted that **“Place”** also has a recursive relationship with itself.

1. from **“Place”** to **“Collection”** by **“belongsTo”** (inverse as **“includes”**).
2. from **“Place”** to **“Collection”** by **“wasIn”** (inverse as **“hosted”**).
3. from **“Place”** to **“PointOfInterest”** by **“isA”** (inverse as **“kindOf”**).
4. from **“Place”** to **“PointOfInterest”** by **“hosted”** (inverse as **“wasIn”**).
5. from **“Place”** to **“Category”** by **“pertains”** (inverse as **“releventFor”**).
6. from **“Place”** to **“Category”** by **“releventOf”** (inverse as **“pertains”**).
7. from **“Place”** to **“Category”** by **“belongsTo”** (inverse as **“includes”**).
8. from **“Place”** to **“Document”** by **“citedIn”** (inverse as **“concerns”**).
9. from **“Place”** to **“Attraction”** by **“hosted”** (inverse as **“wasIn”**).
10. from **“Place”** to **“Event”** by **“hosted”** (inverse as **“wasIn”**).
11. from **“Place”** to **“Person”** by **“hosted”** (inverse as **“wasIn”**).
12. from **“Place”** to **“Place”** by **“hosted”** (inverse as **“wasIn”**).

13. from **“Place”** to **“Visit”** by **“hosted”** (inverse as **“wasIn”**).

5.3 Point of interest

The class of **“PointOfInterest”** is just 3 relationships and they are visualized with a dashed line (the continuous line is for superclass-subclass). They are **“Place”**, **“Category”** and **“Collection”** (13. figure).

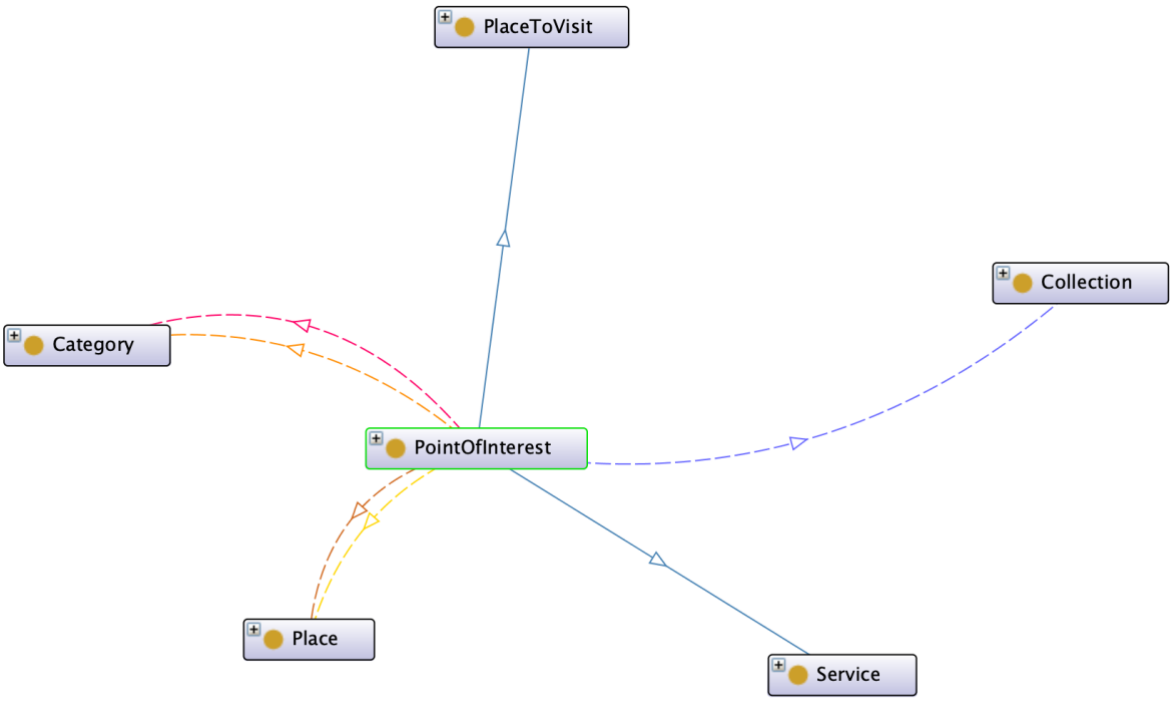


Figure 13. Relationship of class “PointOfInterest”.

The formal listing of all of them could be seen below. It should be noted that **“PointOfInterest”** also has a recursive relationship with itself.

1. from **“PointOfInterest”** to **“Category”** by **“belongsTo”** (inverse as **“includes”**).
2. from **“PointOfInterest”** to **“Category”** by **“pertains”** (inverse as **“relevantFor”**).
3. from **“PointOfInterest”** to **“Category”** by **“relevantFor”** (inverse as **“kindOf”**).
4. from **“PointOfInterest”** to **“Place”** by **“kindOf”** (inverse as **“isA”**).
5. from **“PointOfInterest”** to **“Place”** by **“wasIn”** (inverse as **“hosted”**).
6. from **“PointOfInterest”** to **“Person”** by **“ownedBy”** (inverse as **“owned”**).
7. from **“PointOfInterest”** to **“Person”** by **“pertains”** (inverse as **“relevantFor”**).
8. from **“PointOfInterest”** to **“Person”** by **“hosted”** (inverse as **“wasIn”**).

9. from **“PointOfInterest”** to **“Collection”** by **“hosted”** (inverse as **“wasIn”**).
10. from **“PointOfInterest”** to **“Collection”** by **“belongsTo”** (inverse as **“includes”**).
11. from **“PointOfInterest”** to **“Attraction”** by **“hosted”** (inverse as **“wasIn”**).

5.4 Visit

Class **“Visit” is** associated with events, places, documents, and subjects of tourism (13. figure). Additionally, **“Visit”** is the class that can be included in **“Collection”.**

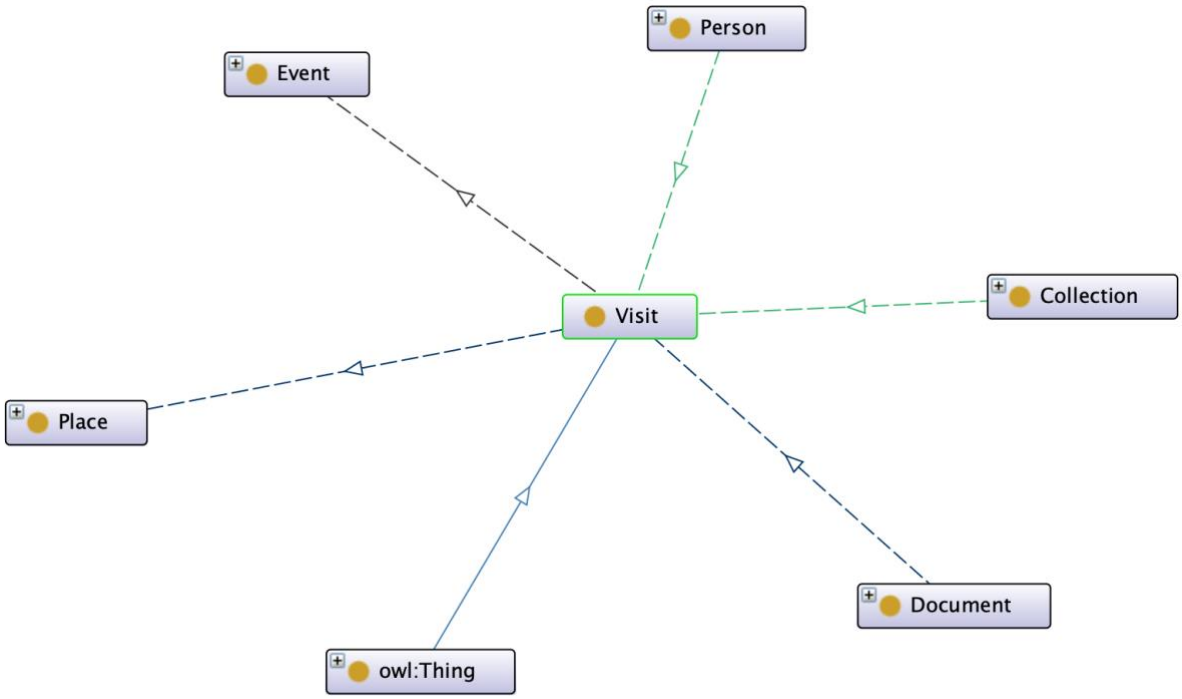


Figure 13. Relationship of class “Place”.

These classes are presented in 13. figure is involved in the following relationships listed below:

1. from **“Visit”** to **“Document”** by **“citedIn”** (inverse as **“concerns”**).
2. from **“Visit”** to **“Collection”** by **“madeBy”** (inverse as **“makes”**).
3. from **“Visit”** to **“Person”** by **“madeBy”** (inverse as **“makes”**).
4. from **“Event”** to **“Visit”** by **“hosted”** (inverse as **“wasIn”**).
5. from **“Event”** to **“Place”** by **“hosted”** (inverse as **“wasIn”**).

5.5 Event

Class **“Event”** is associated with most of the top-level classes as shown in 14. figure.

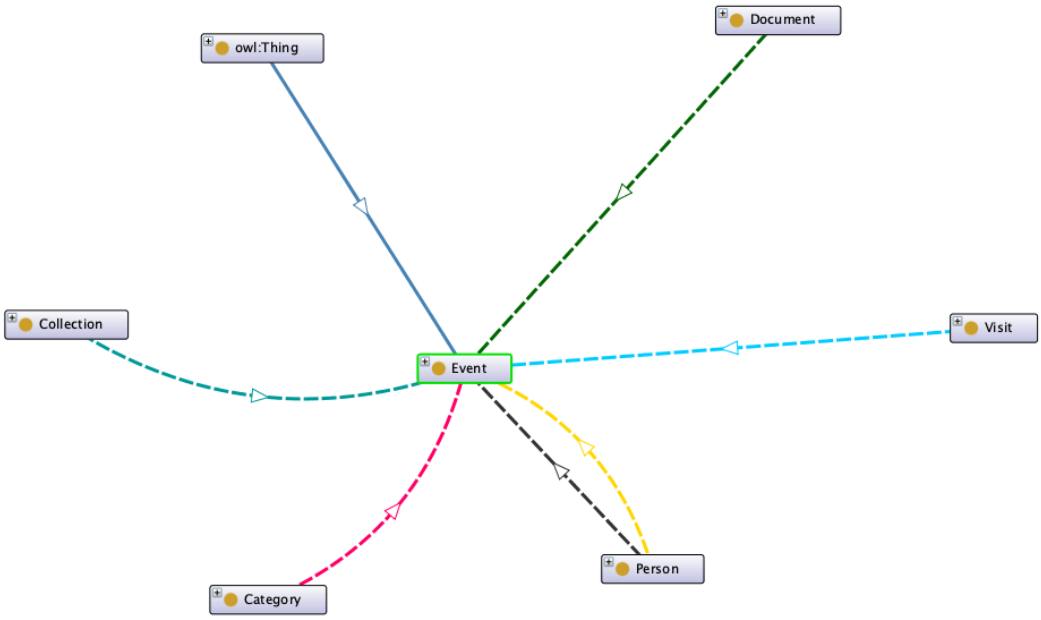


Figure 14. Relationship of class “Event”.

These classes are presented in 14. figure is involved in the following relationships listed below:

1. from **“Event”** to **“Collection”** by **“concerns”** (inverse as **“citedIn”**).
2. from **“Event”** to **“Collection”** by **“belongsTo”** (inverse as **“includes”**).
3. from **“Event”** to **“Collection”** by **“makes”** (inverse as **“madeBy”**).
4. from **“Event”** to **“Person”** by **“relevantFor”** (inverse as **“pertains”**).
5. from **“Event”** to **“Person”** by **“hosted”** (inverse as **“wasIn”**).
6. from **“Event”** to **“Document”** by **“concerns”** (inverse as **“citedIn”**).
7. from **“Event”** to **“Category”** by **“pertains”** (inverse as **“relevantFor”**).
8. from **“Event”** to **“Place”** by **“wasIn”** (inverse as **“hosted”**).
9. from **“Event”** to **“Visit”** by **“hosted”** (inverse as **“wasIn”**).

5.6 Attraction

Class **“Attraction”** has only 3 relationships – one that involves it into **“Collection”** and other ones that places it in specified locations via classes **“PointOfInterest”** and **“Place”** (15. figure)

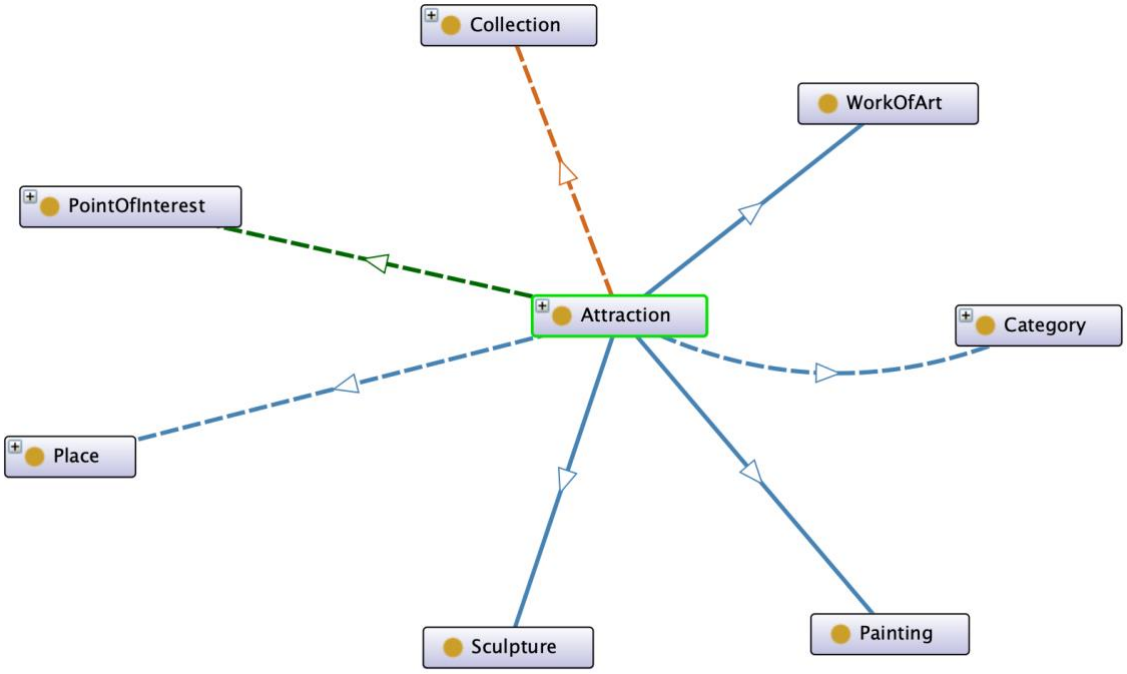


Figure 15. Relationship of class “Attraction”.

These classes are presented in 14. figure is involved in the following relationships listed below:

1. from **“Attraction”** to **“Category”** by **“relevantFor”** (inverse as **“pertains”**).
2. from **“Attraction”** to **“Category”** by **“pertains”** (inverse as **“relevantFor”**).
3. from **“Attraction”** to **“Person”** by **“developedBy”** (inverse as **“developed”**).
4. from **“Attraction”** to **“Person”** by **“ownedBy”** (inverse as **“owned”**).
5. from **“Attraction”** to **“Collection”** by **“belongsTo”** (inverse as **“includes”**).
6. from **“Attraction”** to **“Place”** by **“wasIn”** (inverse as **“hosted”**).
7. from **“Attraction”** to **“PointOfInterest”** by **“wasIn”** (inverse as **“hosted”**).

5.7 Document

Class **“Document”** is associated with events, places, visits, and subjects of tourism (16. figure). All of the relations are made by the **“citedIn/concerns”** definition.

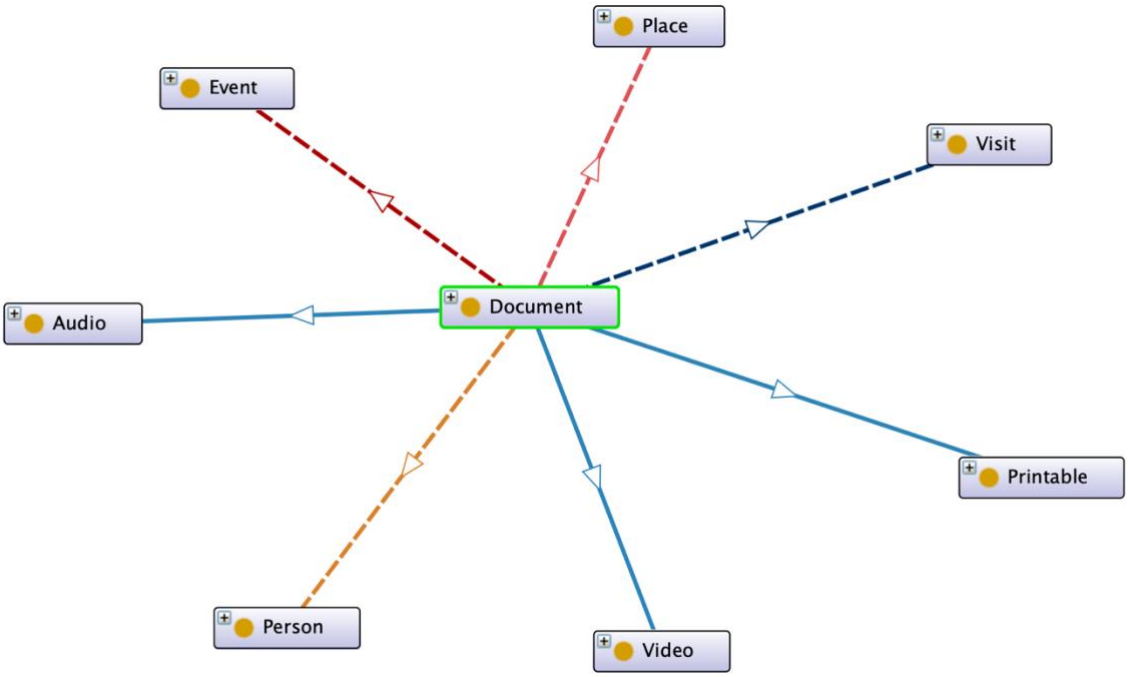


Figure 16. Relationship of class “Document”.

These classes are presented in 16. figure is involved in the following relationships listed below:

1. from **“Document”** to **“Event”** by **“citedIn”** (inverse as **“concerns”**).
2. from **“Document”** to **“Visit”** by **“citedIn”** (inverse as **“concerns”**).
3. from **“Document”** to **“Place”** by **“citedIn”** (inverse as **“concerns”**).
4. from **“Document”** to **“Person”** by **“citedIn”** (inverse as **“concerns”**).

5.8 Category

“Category” is one of two classes that are made for grouping and that is why it has relationships with every other top-level class and even with the **“Collection”** as one can be described with some sort of type. This is visualized in 17. figure.

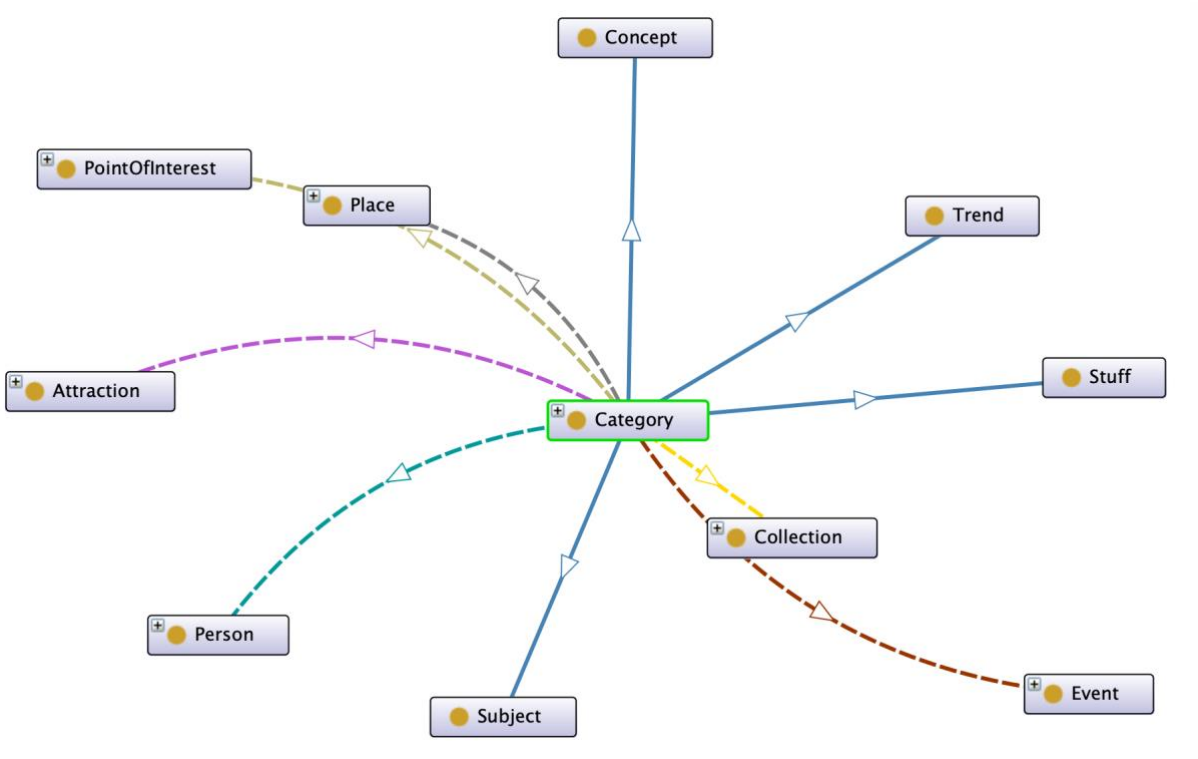


Figure 17. Relationship of class “Category”.

These classes are presented in 17. figure is involved in the following relationships listed below:

1. from **“Category”** to **“Attraction”** by **“relevantFor”** (inverse as **“pertains”**).
2. from **“Category”** to **“Attraction”** by **“pertains”** (inverse as **“relevantFor”**).
3. from **“Category”** to **“Person”** by **“relevantFor”** (inverse as **“pertains”**).
4. from **“Category”** to **“Person”** by **“includes”** (inverse as **“belongsTo”**).
5. from **“Category”** to **“Person”** by **“pertains”** (inverse as **“relevantFor”**).
6. from **“Category”** to **“Place”** by **“includes”** (inverse as **“belongsTo”**).
7. from **“Category”** to **“Place”** by **“pertains”** (inverse as **“relevantFor”**).
8. from **“Category”** to **“PointOfInterest”** by **“pertains”** (inverse as **“relevantFor”**).
9. from **“Category”** to **“PointOfInterest”** by **“includes”** (inverse as **“belongsTo”**).
10. from **“Category”** to **“PointOfInterest”** by **“relevantFor”** (inverse as **“pertains”**).

11. from **“Category”** to **“Event”** by **“pertains”** (inverse as **“relevantFor”**).
12. from **“Category”** to **“Event”** by **“relevantFor”** (inverse as **“pertains”**).
13. from **“Category”** to **“Place”** by **“relevantFor”** (inverse as **“pertains”**).
14. from **“Category”** to **“Collection”** by **“pertains”** (inverse as **“relevantFor”**).

5.9 Collection

“Collection” or the other grouping class that similarly to **“Category”** deals with all other top-level classes to combine them into collections.

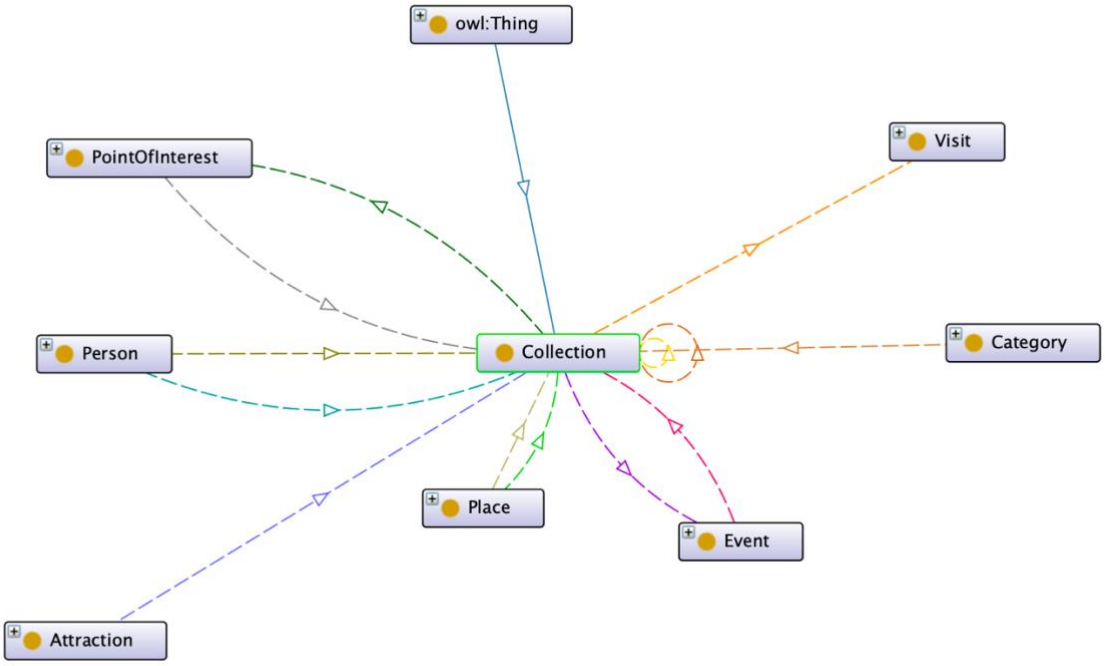


Figure 18. Relationship of class “Collection”.

These classes are presented in 18. figure is involved in the following relationships listed below:

1. from **“Collection”** to **“Attraction”** by **“includes”** (inverse as **“belongsTo”**).
2. from **“Collection”** to **“Event”** by **“includes”** (inverse as **“belongsTo”**).
3. from **“Collection”** to **“Event”** by **“madeBy”** (inverse as **“makes”**).
4. from **“Collection”** to **“Event”** by **“relevantFor”** (inverse as **“pertains”**).
5. from **“Collection”** to **“PointOfInterest”** by **“includes”** (inverse as **“belongsTo”**).
6. from **“Collection”** to **“PointOfInterest”** by **“wasIn”** (inverse as **“hosted”**).

7. from **“Collection”** to **“PointOfInterest”** by **“includes”** (inverse as **“belongsTo”**).
8. from **“Collection”** to **“Collection”** by **“includes”** (inverse as **“belongsTo”**).
9. from **“Collection”** to **“Collection”** by **“hosted”** (inverse as **“wasIn”**).
10. from **“Collection”** to **“Place”** by **“includes”** (inverse as **“belongsTo”**).
11. from **“Collection”** to **“Place”** by **“wasIn”** (inverse as **“hosted”**).
12. from **“Collection”** to **“Person”** by **“hasPart”** (inverse as **“partOf”**).
13. from **“Collection”** to **“Person”** by **“wasIn”** (inverse as **“hosted”**).
14. from **“Collection”** to **“Visit”** by **“madeBy”** (inverse as **“makes”**).

5.10 Full graph

To represent all relationships in one unified manner the 10 tables are created to observe the current state of class interrelations.

Table 10 – All defined relationships among the top-level classes									
Object/ Subject	**Attraction**	**Category**	**Collection**	**Document**	**Event**	**Person**	**Place**	**Ponts of interest**	**Visit**
Attraction	/	pertains/ relevantFor	includes/ belongsTo	/	/	/	hosted/ wasIn	hosted/ wasIn	/
Category	pertains/ relevantFor	/	pertains/ relevantFor	/	pertains/ relevantFor	pertains/ relevantFor	pertains/ relevantFor	pertains/ relevantFor - pertains/ relevantFor	/
Collection	/	/	includes/ belongsTo - hosted/ wasIn	/	madeBy/ makes	/	/	hosted/ wasIn	madeBy/makes

Document	/	/	/	/	/	/	/	/	/
Event	/	pertains/ relevantFor - pertains/ relevantFor	include/ belongsTo	citedIn/ concerns	/	/	/	/	/
Person	developedBy/ developed - ownedBy/ owned	includes/ belongsTo - pertains/ relevantFor	hasPart/ partOf - hosted/ wasIn	citedIn/ concerns	madeBy/ makes - wasIn/ hosted - hosted/ wasIn	/	hosted/ wasIn	ownedBy/ owned - hosted/ wasIn - pertains/ relevantFor	madeBy/makes
Place	/	includes/ belongsTo - pertains/ relevantFor	includes/ belongsTo - hosted/ wasIn	citedIn/ concerns	/	/	hosted/ wasIn	kindOf/ isA - hosted/ wasIn	/

Ponts of interest	ownedBy/ owned	includes/ belongsTo - pertains/ relevantFor	includes/ belongsTo	/	/	/	/	/	/
Visit	/	/	/	citedIn/ concerns	hosted/ wasIn	/	hosted/ wasIn	/	/

6 Related domains

One of the *"Gr@phBrain"* innovations that are acknowledged by its developers is that ontologies, which are hosted by the system, are not isolated from others. As they say, "[users] may build and maintain several ontologies by specifying for each the hierarchies of classes and relationships to be considered"[1]. They indicate that such an approach is intending that "some classes and relationships may appear in different ontologies, possibly with different attributes, to reflect different perspectives on them"[1]. Essentially these results are "cross-fertilization among, and knowledge reuse across, different domains: individuals used by different ontologies act as bridges among those ontologies, allowing the users of a domain to obtain additional information coming from other domains"[1]. This is also the case of tourism domain, which share few classes with other domain namely "food". Of course, food is a vital part of any touristic experience and via some commonly shared classes with the tourism domain, they establish in size a greater ontology that they could build independently. Therefore, to have the full picture, before purposing the expansion, it is needed to describe also the food domain, which will be done the in the following section.

7 Food domain's class taxonomy

In total food, a domain consists of top-level classes (19. figure). Some of them are already covered in the section describing the tourism domain classes. These are classes of **"Person"**, **"Place"**, and **"PointOfInterest"**. Although these classes (except **"Place"** and **"Person"** that are identical used in the tourism domain) are different in the attributes used in their definitions. The class of **"User"** comes from the domain **"General"**, which provides the base of common classes for other domains to reuse.

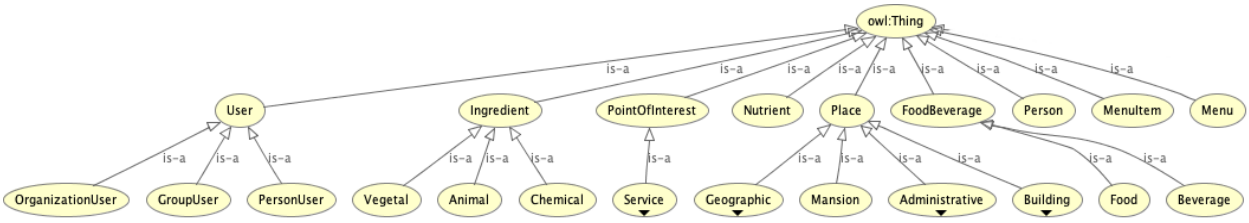


Figure 19. Food domains top level classes.

To avoid repeated description of **"Place"** and **"Person"** class definitions please refer to their corresponding sections described previously.

7.1 Point of interest

Although the name is commonly shared with one that is used in the tourism domain the taxonomy and attributes are different. In 20. figure the taxonomy the of **"PointOfInterest"** class reuses only the **"Service"** subclass. This is different from tourism **"PointOfInterest"** which in its taxonomy uses also the **"Place"** subclass.

Figure 20. Taxonomy of class "PointOfInterest".

The 11. table shows attributes of **"PointOfInterest"** - some of them like "address", "phone", and "place" are absent in the definition that was seen previously in the tourism domain. However previously unseen "noOfSeats" attribute is added.

Table 11. – "PointOfInterest" class attributes

XML name	Mandatory	Data type	Enumeration	Described as another class of
name	Yes	string	-	-
description	No	text	-	-
estimatedCost	No	real	-	-
noOfSeats	No	integer	-	-

7.2 User

The class of **"User"** comes from the general domain, which as previously mentioned, provided a common set of classes for other ontologies for their buildup. The taxonomy of **"User"** consists of 3 classes – **"organizationalUser"**, **"GroupUser"** and **"PersonUser"** (21. figure).

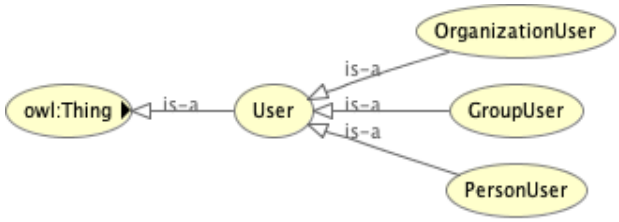


Figure 21. Taxonomy of class "User".

Table 12. – "User" class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
username	Yes	string	-	-
user	No	string	-	-
description	No	real	-	-
registrationDate	Yes	date	-	-
eraseDate	No	date		

7.3 Nutrient

Continuing with food domain-specific classes first and furthermost there is a class of **"Nutrients"** that can be used for describing the dish with its ingredients. However, there is no further sub-taxonomy for the class. **"Nutrient"** main involvement is through the relationships that are applied with the **"Ingredient"** class. Attributes of **"Nutrient"** are just 2 – name and type. The latter is an enumeration of nutrients.

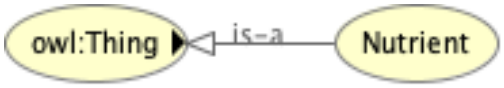


Figure 22. Taxonomy of class "Nutrient".

Table 13. – "Nutrient" class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**

name	Yes	string	-	-
type	No	select	1. Carbohydrate 2. Lipid 3. MineralSalt 4. Protein 5. Sugar 6. Vitamin 7. Other	-

7.4 Ingredient

The focal and food building class of **"Ingredient"** has an obvious purpose. Its current taxonomy allows the ingredient to be "animal", "chemical" or "vegetal" (23. figure). The only attribute that is given for ingredients is the name (14. table).

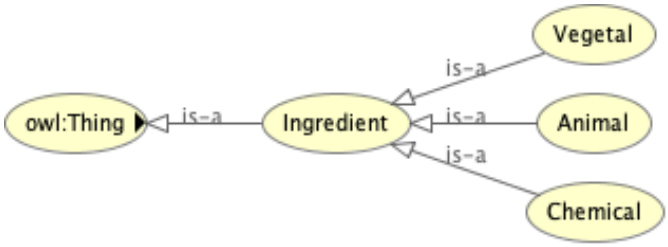


Figure 23. Taxonomy of class "Ingredient".

Table 14. – "Ingredient" class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
name	Yes	string	-	-

7.5 Food Beverage

Undoubtedly the main class of the food ontology is the **"FoodBeverage",** which is split into **"Beverage"** and **"Food"** subclasses (24. figure). The only attributes for in-class attribute disposal are the name and description (15. table).

Figure 24. Taxonomy of class "FoodBeverage".

Table 15. – "FoodBeverage" class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
name	Yes	string	-	-
description	No	string	-	-

7.6 Menu and menu item

The two last classes in the food domain are **"Menu"** and **"MenuItem"**. Both of them do not have subclasses (25. and 26. figure), but they as will be seen in the relationships section are connected to each another.

Figure 25. Taxonomy of class "Menu".

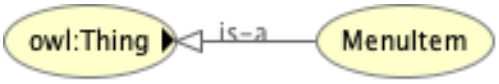


Figure 26. Taxonomy of class "MenuItem".

The **"Menu"** class definition of attributes is identical to one that was the seen in section about **"Nutrient"** – name and the type (17. table).

Table 17. – "Menu" class attributes				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**

name	Yes	string	-	-
type	No	select	1. Lunch 2. Dinner 3. Kids 4. Other	-

The class **"MenuItem"** is richer in attribute definition as there a is name, description, price, and type that categorizes if the dish belongs to the first course, second or it is desert (18. table).

Table 18. – "MenuItem" class attributes

XML name	Mandatory	Data type	Enumeration	Described as another class of
name	Yes	string	-	-
description	No	string	-	-
price	No	real	-	-
type	No	select	1. Appetizer 2. FirstCourse 3. SecondCourse 4. SideDish 5. Dessert 6. Beverage 7. Other	-

8 Food domains relationships

Technically food domain relationships do not differ so much from the tourism domain, just different names, and connections. There are 9 relationship definitions and they are following:

1. **"associatesWith"** (inversed as **"associatesWith"**)
2. **"disliked"** (inversed as **"dislikedBy")**
    a. **"reason"**, type select, not mandatory.

3. **“instanceOf”** (inversed as **“hasInstance”**)
4. **“interestedIn”** (inversed as **“interests”**)
5. **“isA”** (inversed as **“kindOf”**)
6. **“partOf”** (inversed as **“hasPart”**)
7. **“purposes”** (inversed as **“purposedBy”**)
8. **“typicalOf”** (inversed as **“hasTypical”**)
9. **“wasIn”** (inversed as **“hosted”**).
    a. **“reason”**, type string, not mandatory.
    b. **“date”**, type date, not mandatory.

As it can be seen some relationships were present in the tourism domain like **“isA”** or **“part of”.** As the food ontology is not the central one in this documentation all relationships will be presented in one table similarly as in section 5.10.

Table 19 – All defined relationships among the top-level classes of the food domain									
Object/ Subject	**User**	**Person**	**FoodBeverage**	**Ingredient**	**Nutrient**	**Menu**	**MenuItem**	**PointOfInterest**	**Place**
User			dislikedBy/ dislikes - interestedIn/ interests	dislikedBy/ dislikes - interestedIn/ interests			dislikedBy/ dislikes - interestedIn/ interests	wasIn/ hosted	
Person								wasIn/ hosted	
Food Beverage			associatesWith/ associatesWith - isA/ kindOf						typicalOf/ hasTypical
Ingredient			associatesWith/ associatesWith - partOf/ hasPart	associatesWith/ associatesWith - isA/ kindOf - partOf/ hasPart			partOf/ hasPart		typicalOf/ hasTypical
Nutrient				partOf/ hasPart					

Menu									
menuItem			instanceOf/ hasInstance - isA/ kindOf			partOf/ hasPart			
Ponts OfInterest						Purposes/ puroposedBy			wasIn/ hosted
Visit									

9 Proposal for expansion

This section encircles the figures and tables of expansion. They can be three-fold:

1. Addition entirely new classes with their attributes and subclasses.
2. Addition of new subclasses to already existing classes.
3. Addition of new attributes to already existing classes.

Each detail (class or attribute, etc.) will be supported with a brief explanation of the expansion purpose.

9.1 Proposal for a food domain

9.1.1 FoodBeverage

Starting with topmost class of **"FoodBeverage".** Besides its current attributes, the purposed addition would mark it in the context of different cuisine. Like **'Person"** who has the attribute for his nationality – the food or beverage may also have such attribute. Also for the same reason attribute of "placeOfOrigin" is purposed.

Table 22. – "FoodBeverage" attribute addition

XML name	Mandatory	Data type	Enumeration	Described as another class of
placeOfOrigin	No	entity	-	**Place**
cuisine	No	string	-	**-**

The class as described has 2 subclasses of food and beverage. For the latter is purposed to add the self-describing boolean attribute of "isAlcoholic".

Table 23. – "Beverage" attribute addition

XML name	Mandatory	Data type	Enumeration	Described as another class of

isAlcoholic	No	boolean	-	-

9.1.2 A purposed new class of "Characteristic"

The current state of the food domain does not have something that could represent a set of characteristics to describe the food in terms of basic "sensations" like flavor (e.g., salty), texture (e.g., crunchy), and state (e.g., solid). This preset is purposed as a class with corresponding subclasses.

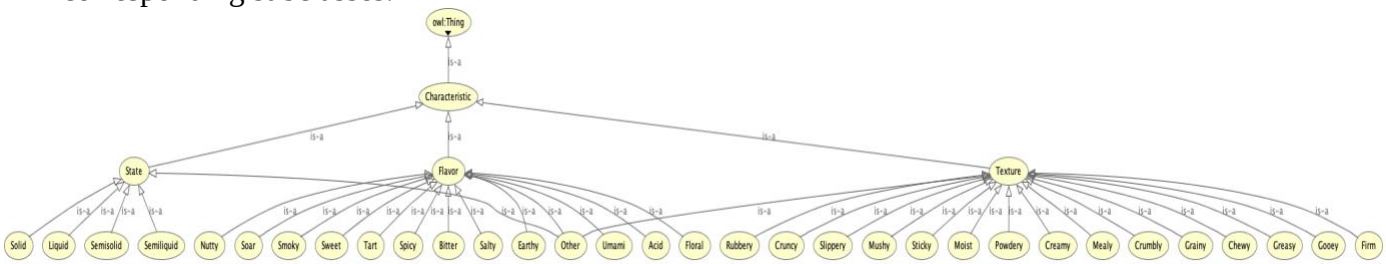


Figure 27. Taxonomy of class "Characteristic".

No additional attributes are purposed, only an "Id" for a technical reason. To use it accordingly there is a relationship defined between **"Ingredient"**, and **"FoodBeverage"** (introduced in a later section).

Also, a more abstract but noticeable characteristic of the foods or ingredients is a classification of type in terms that is it something like hamburger which is a common food for fast food restaurants or could a specific food be placed under the category of "kosher" (Jewish dietary regulations). The full set can be seen in 28. Figure.

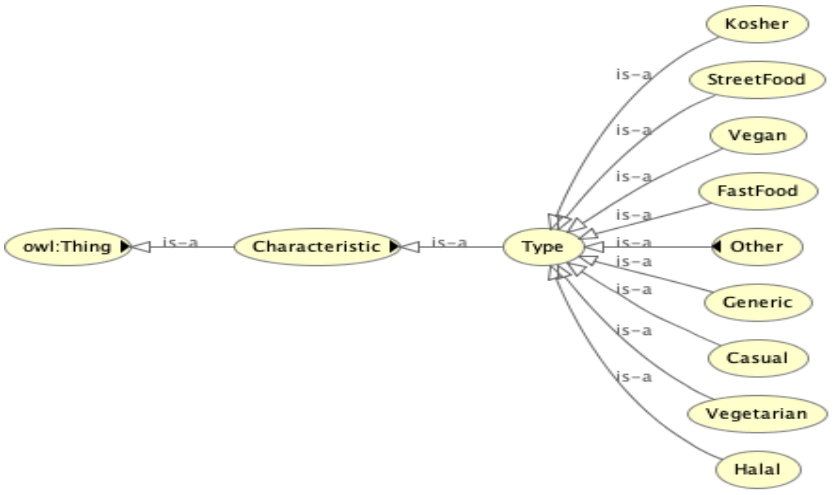


Figure 28. Taxonomy of class "Type".

9.1.3 A purposed new class of "Processing"

Another seemingly significant way of deserving the food is the actions that are applied to it. We may look for an ingredient with the same name, but differently preserved (e.g., by freezing or drying). Similarly, one ingredient can be cooked differently (e.g., grilled or broiled). To enrich the food description the class of **"Processing"** is purposed.

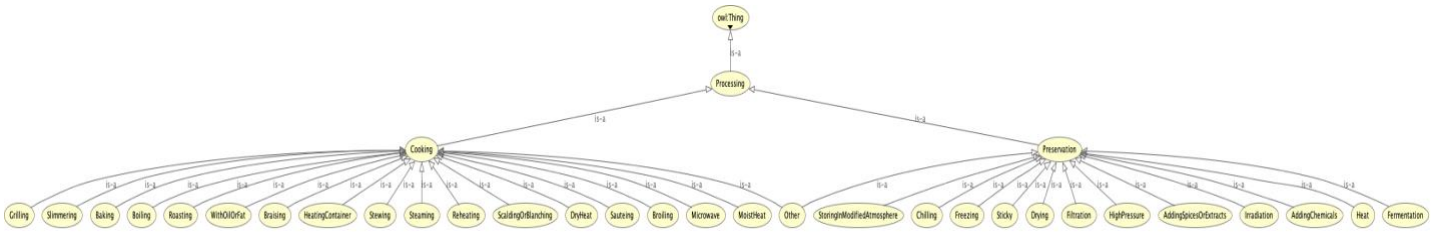


Figure 29. Taxonomy of class "Processing".

No additional attributes are purposed, only an "Id" for a technical reason. To use it accordingly there is a relationship defined between **"Ingredient"**, and **"FoodBeverage"** (introduced in a later section).

9.1.4 A purposed new class of "Recipe"

2 equal sets of ingredients or foods can be cooked differently resulting in 2 different dishes. For such reason, class "Recipe" is purposed. It does not have a sub-taxonomy, just the attributes.

Table 24. – "Recipe" class attributes

XML name	Mandatory	Data type	Enumeration	Described as another class of
name	Yes	string	-	-
description	No	string	-	-
hasCookingTemprature	No	integer	-	-

hasCookingTime	No	integer	-	-

To use it accordingly there is a relationship introduced in a later section.

9.1.5 Ingredient

One of the main classes in the food domain is **“Ingredient”**. Attribute wise it only has a name and to enrich it the purposed attributes are 3 – “placeOfOrigin”, “hasGluten”, “hasGlycemicIndex”.

Table 25. – “Ingredient” attribute addition

XML name	Mandatory	Data type	Enumeration	Described as another class of
placeOfOrigin	No	entity	-	**Place**
hasGluten	No	boolean	-	-
glycemicIndex	No	integer	-	-

If the purpose of the “placeOfOrigin” is the obvious then “hasGluten” and “glycemicIndex” are meant as regular information for people that have gluten intolerance or who suffer from diabetes. Other such attributes can be possible, but the mentioned ones are the most common. Taxonomy of **“Ingredient”** currently holds plain classes of “Animal”, “Chemical” and “Vegetal”. All of them are extended followingly. In the case of “Animal” the 30. The figure is introduced.

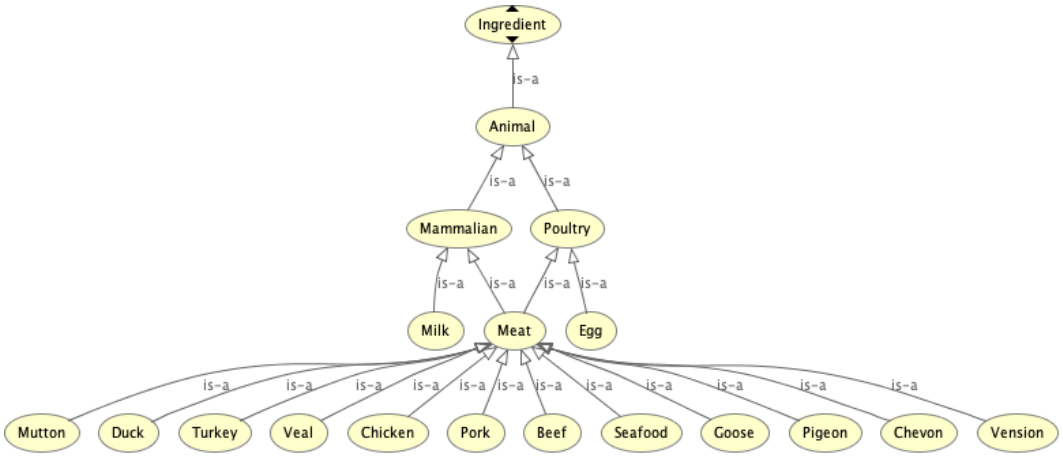


Figure 30. Taxonomy of class "Animal".

The proposal seeks to separation in mammalian and poultry types of animal ingredients. Both of them may have their corresponding products like milk or egg, but both have a class of meat which have values for different but most common classifications of meat products like beef, pork, or seafood.

For the "Chemical" class it is proposed to extend it with a subclass of "Additives" as the ingredients something like corn syrup, vinegar, or yeast extract. The most common ones are shown in 31. Figure.

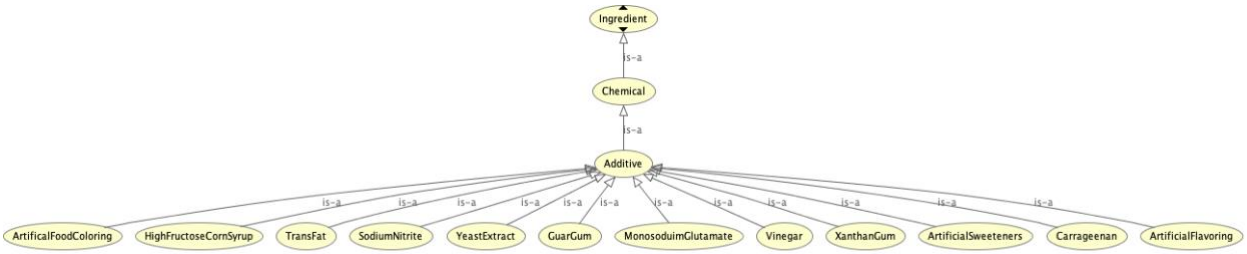


Figure 31. Taxonomy of class "Chemical".

Most simplicity for the "Vegetal" class there is an extension of most common representations like fruit, nut, vegetable, etc.

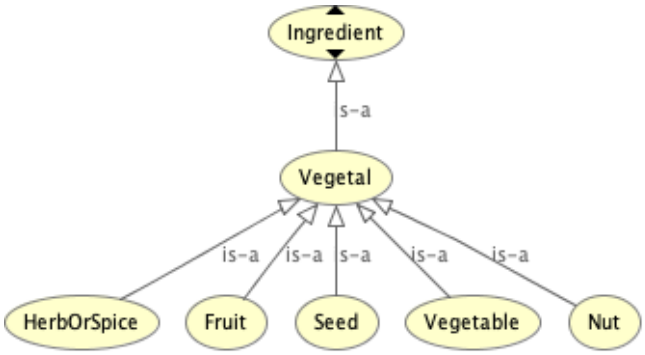


Figure 32. Taxonomy of class "Chemical".

9.1.6 Menu

Proposal for class "Menu" is to add additional types of menus as currently, it holds only "Lunch", "Dinner", "Kids". Besides these categories, others most often are "Branch" and "Breakfast".

Table 25. – "Menu" attribute addition

XML name	Mandatory	Data type	Enumeration	Described as another class of
Tyle	No	select	1. Lunch 2. Dinner 3. Kids 4. Brunch 5. Breakfast	-

9.1.7 Relationship extensions

Previously described classes that were proposed - there is the following necessity to introduce new relationships and definitions.

For **"Recipe"** it is proposed to reuse the definition of "**associatesWith**" to establish the relationship between **"MenuItem"** and **"Ingredient"**.

- from **"Recipe"** to **"MenuItem"** by **"associatesWith"** (inverse as **"associatesWith"**).
- from **"Recipe"** to **"MenuItem"** by **"associatesWith"** (inverse as **"associatesWith"**).

It is proposed to introduce new relationship definitions like:

1. **“causeAllergyTo”** (inversed as **“isAllergicTo”**)

   Applied to:

   - from **“User”** to **“FoodBeverage”** by **“isAllergicTo”** (inverse as **“causeAllergyTo”**).
   - from **“User”** to **“Ingredient”** by **“isAllergicTo”** (inverse as **“causeAllergyTo”**).
   - from **“User”** to **“MenuItem”** by **“isAllergicTo”** (inverse as **“causeAllergyTo”**).

2. **“hasCharacteristic”** (inversed as **“characteristicOf”**)

   - **“kind”**, type select, not mandatory. Enumeration:
     i. **Flavor**
     ii. **Texture**
     iii. **State**
     iv. **Type**

   Applied to:

   - from **“Characteristic”** to **“FoodBeverage”** by **“characteristicOf”** (inverse as **“hasCharacteristic”**).
   - from **“Characteristic”** to **“Ingredient”** by **“characteristicOf”** (inverse as **“hasCharacteristic”**).
   - from **“Characteristic”** to **“MenuItem”** by **“characteristicOf”** (inverse as **“hasCharacteristic”**).

3. **“hadProcessedBy”** (inversed as **“processingAppliedTo”**)

   - **“kind”**, type select, not mandatory. Enumeration:
     i. **Cooking**
     ii. **Preservation**

   Applied to:

   - from **“Processing”** to **“FoodBeverage”** by **“processingAppliedTo”** (inverse as **“hadProcessedBy”**).
   - from **“Processing”** to **“Ingredient”** by **“processingAppliedTo”** (inverse as **“hadProcessedBy”**).

- from **“Processing”** to **“MenuItem”** by **“processingAppliedTo”** (inverse as **“hadProcessedBy”**).

4. **“substitutesFor”** (inversed as **“substitutesFor”**)

   Applied to:

   - from **“FoodBeverage”** to **“FoodBeverage”** by **“substitutesFor”** (inverse as **“substitutesFor”**).
   - from **“Ingredient”** to **“Ingredient”** by **“substitutesFor”** (inverse as **“substitutesFor”**).

9.2 Proposal for tourism domain

9.2.1 Attraction

For a better description of attraction few attributes are proposed. One is the “author”, which is currently missing in a set of attributes. As in this ontology attraction is applied in the context of the art object - second and third are the “style” and “genre”. The usual mistake is to think about these terms as synonyms. The genre answers question of what is represented – landscape, still life, portrait. The style is answering questions of how it was represented – cubism, dadaism, rococo or baroque.

Table 26. – “Attraction” attribute addition

XML name	Mandatory	Data type	Enumeration	Described as another class of
author	No	entity	-	**Person**
genre	No	string	-	**-**
style	No	string	-	**-**

The taxonomy of attraction dictates 3 possibilities of such objects – “Handicraft”, “IndustrialWork” and “Artwork”. For “IndustrialWork” it is proposed to extend with the sub-taxonomy of “Installation” as art installations are usually described as industrial.

Figure 33. Taxonomy of class "IndustralWork".

Artwork is proposed to extend with the "Artwork" subclass of "Attraction" with more specific classifications that are shown in 32. figure.

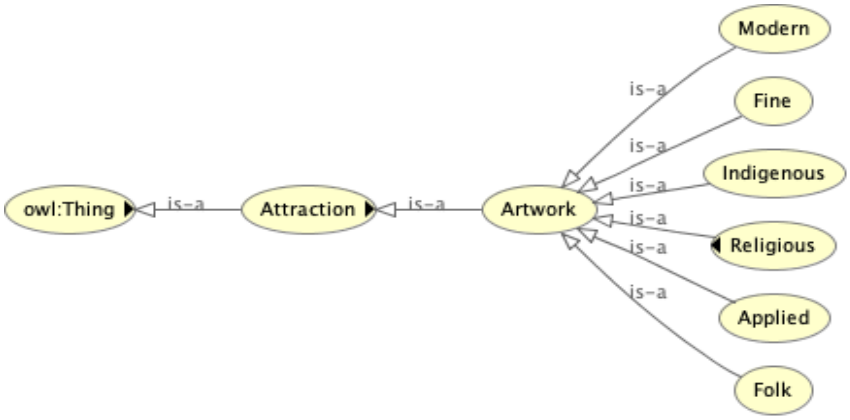


Figure 34. Taxonomy of class "Artwork".

The explanation for them is that they are the most common categories in which artwork can be classified. Nerveless it is proposed to introduce another attraction classification of "Historical" for archeological attractions like fossils and artifacts.

Figure 35. Taxonomy of class "Historical".

It is still encouraged to loosen attraction affiliation only to the objects as there could be attractions in form of artistic actions like ones that are known as "Happening" or "Performance".

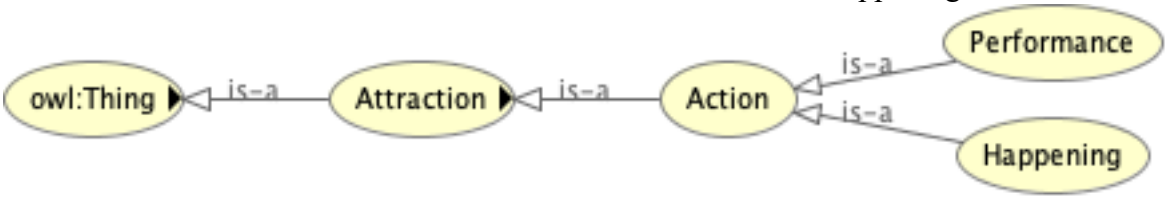


Figure 36. Taxonomy of class "Action".

9.2.2 Person

For the **"Person"** class, which is an attribute-wise most rich entity, only one attribute is offered as the proposal and it is the enumeration of the person's education level. Enumeration consists of four values – kindergarten, primary, secondary or higher.

Table 27. – "Person" attribute addition				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
educationalLevel	No	select	1. Kindergarten 2. Primary 3. Secondary 4. Higher	-

9.2.3 Place

The proposal for the "Place" class consists of adding two more attributes – phone and website - for the opportunity to set contact information. The class originates from the "General" domain and for its definition, the same attributes and taxonomy are also updated.

Table 28. – "Place" attribute addition				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**
phone	No	string	-	-
website	No	string	-	-

Taxonomy of **"Place"** dictates that it can be administrative, building, geographic, and mansion. The expansion is offered for "building" and "geographic". The building is simply an addition of touristic places like the mall, shopping center, and farm (35. figure).

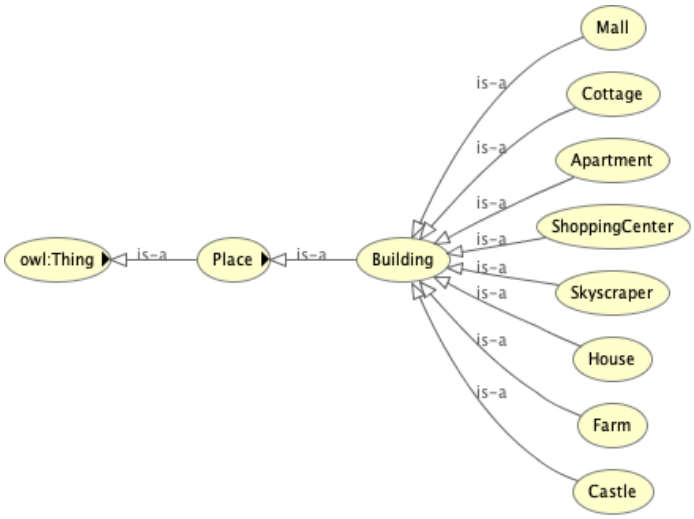


Figure 37. Taxonomy of class "Building".

For the "geographic" it is proposed to add the taxonomy of 2 new subclasses – coast and forest (36. figure). Additionally, for this subclass, it is purposed the attribute for setting the habitat for more description. It is an enumeration of 3 values – terrestrial, marine, and desert.

Table 29. – "Place" attribute addition

XML name	Mandatory	Data type	Enumeration	Described as another class of
habitat	No	select	1. Terrestrial 2. Marine 3. Desert	-

9.2.4 PointOfInterest

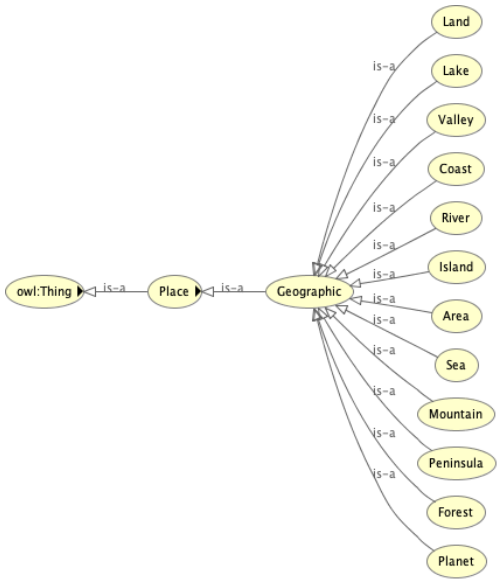


***Figure 38. Taxonomy of class* "Geographic".**

There are multiple proposals for the **"PointOfInterest"** class. Attribute-wise similarly to the "Place" class as the museums and castles usually has a website – such attribute is proposed. Attribute for a phone number is already present in the definition.

Table 30. – "PointOfInterest" attribute addition

XML name	Mandatory	Data type	Enumeration	Described as another class of
website	No	string	-	-

The taxonomy for **"PointOfInterest"** splits into **"Service"** and **"PlaceToVisit".** Under service, there is a set containing ones like banks, gas stations, hotels, and shops. One particular extension proposed is a subclass of **"Eating/Drinking"**. Its subclasses dictate that it can be four-

fold – bar, pub, market, or restaurant. It could be enhanced with other variations like bistro, café, canteen, or pizzeria (38. Figure).

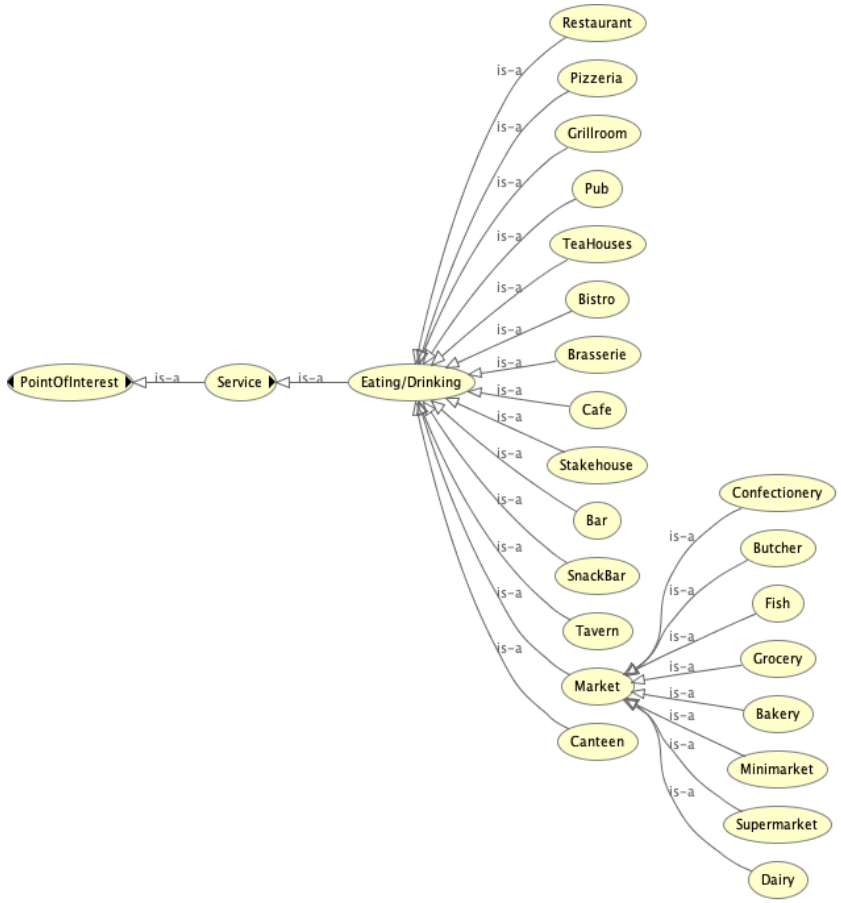


Figure 39. Taxonomy of class "Eating/Drinking".

Also, as the 39. figure shows there is a continuation for "Market" that specifies its type as grocery or fish market. The only proposal for "Market" is to add another subclass of "Confectionery".

Another subclass of **"Service"** is the "Station" that introduces transportation like air, rail, road, or water. For each of them are purposed additional varieties that are shown in 40. Figure. For example, as road transportation included the varieties of the bike, buses, and cars – it is purposed to include ones like motorcycles, vans, and scooters. For the car, it is purposed to add a set of classifications like sedan, hatchback, and cabriolet. The full set could be seen in 40. Figure.

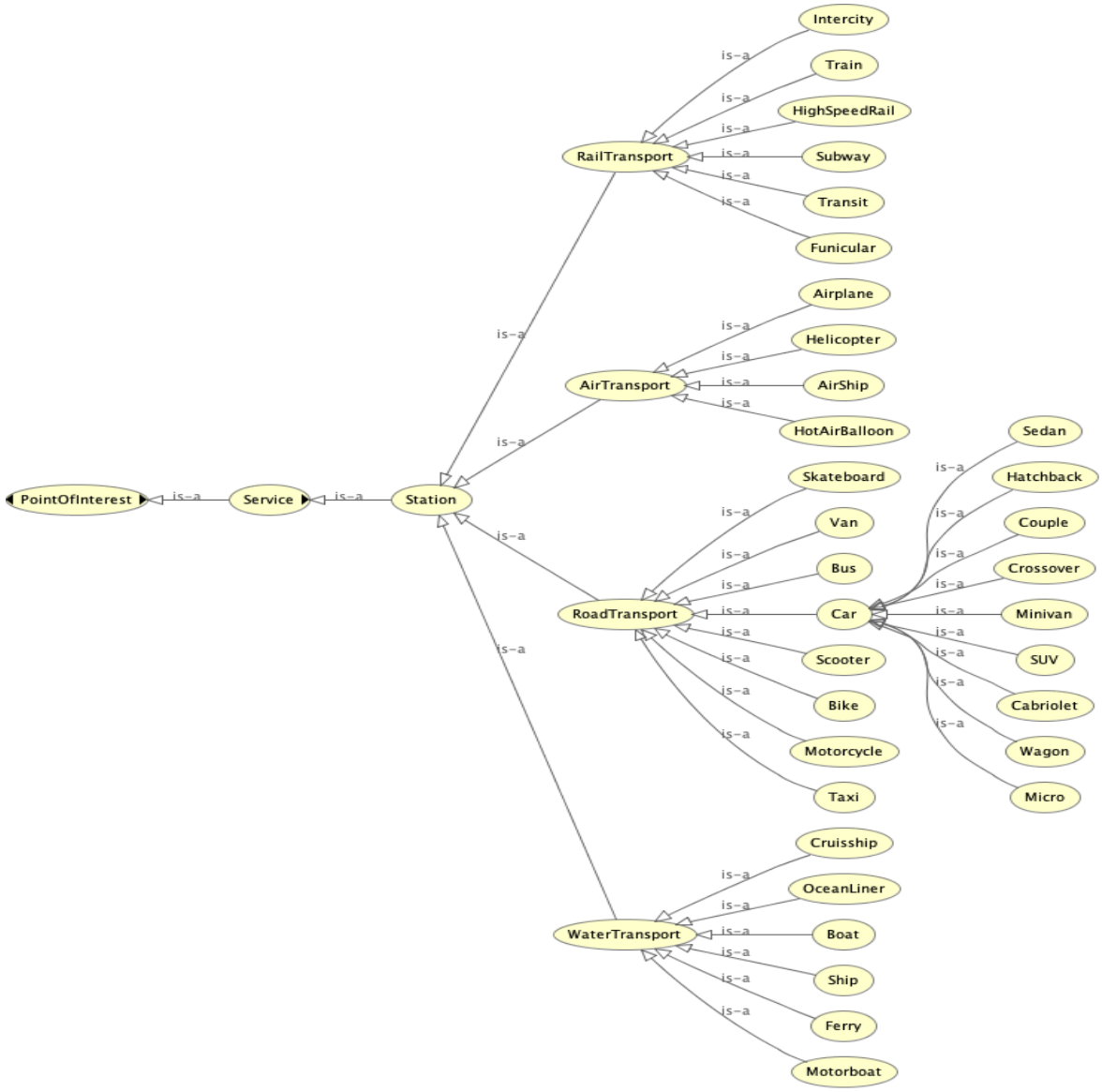


Figure 40. Taxonomy of class "Station".

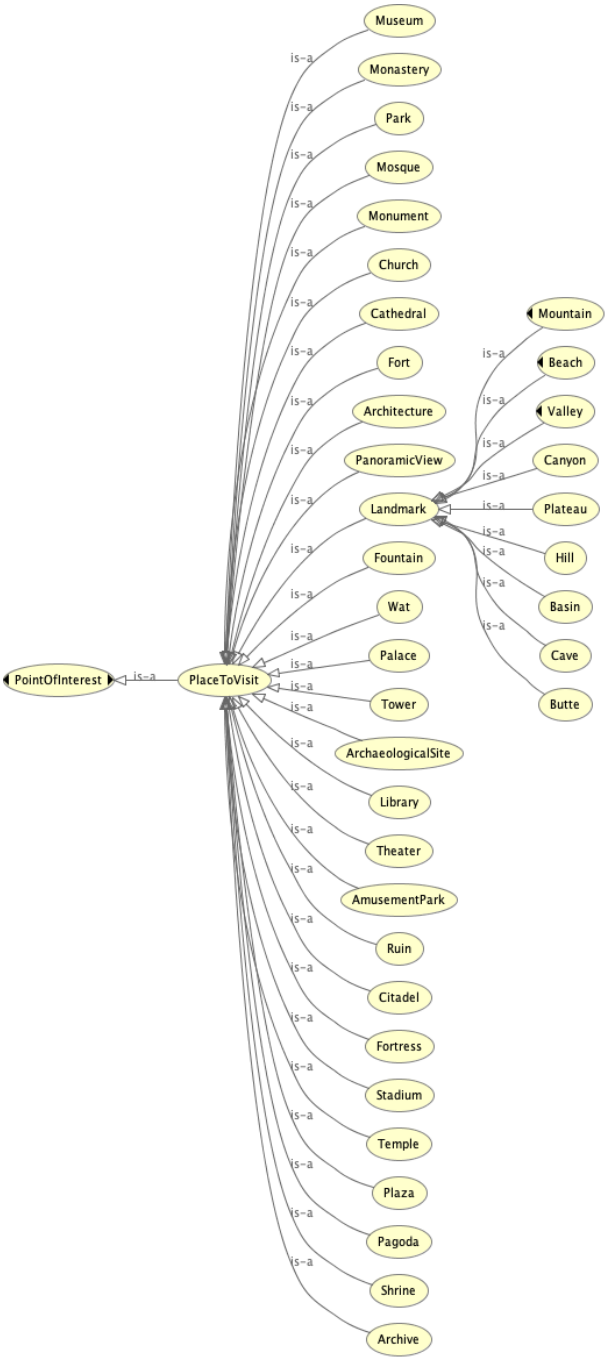
Additionally, for each transportation type ("RoadTransportation", "AirTransportation") it is purposed to add an attribute of availability (indicating is the instance of the transportation available for service), consumption, and type of fuel.

Table 31. – "Transportation" attribute addition				
XML name	**Mandatory**	**Data type**	**Enumeration**	**Described as another class of**

availability	No	boolean	-	-
consumption	No	string	-	-
typeOfFuel	No	string	-	-

As far as for new services it is proposed for 3 new ones – barber, hotel and beauty services. For example, “BeautyService” has a set of wellness procedures like pedicure or bath – also some of the sub services have further classifications like a massage that divides into eastern and western. The full proposal can be seen in 41. Figure.

Figure 41. Taxonomy of class “PlaceToVisit”.



That would conclude the purposes for the **"Service"** extension, but there is also a **"PlaceToVisit"** that has a few purposed subclasses additions. They can be seen in 41. Figure. Generally, it is a set of new touristic places like a fort, mosque, wat. A notable addition is the subclass of landmark, which introduces geographic objects like valleys and caves as they can also serve as touristic points of interest.

Figure 41. Taxonomy of class "PlaceToVisit".

9.2.5 A purposed new class of "Interest"

Figure 41. Taxonomy of class "Interest".

Another additional class for the purpose would be related to subjective aspects of tourism that are not represented in the ontology. They would also serve as additional categorization for most general cases. The purposed class for it would be **"Interests"** and its taxonomy is visualized in 42. figure. Such class would include interests classified as urban, romantic, family, business, and medical. One is the factual touristic action for some person for example who went to the Paris. But some tourists are only interested in French metropolitan gastronomy and tourists do not care about it at all as their interest lies in the city's architecture. There is of course a need for relationships which definition is presented in later sections. As the taxonomy is quite self-sufficient there would be only id as their attribute (22. table).

Table 32. – "Interest" class attributes

XML name	Mandatory	Data type	Enumeration	Described as another class of
id	Yes	string	-	-

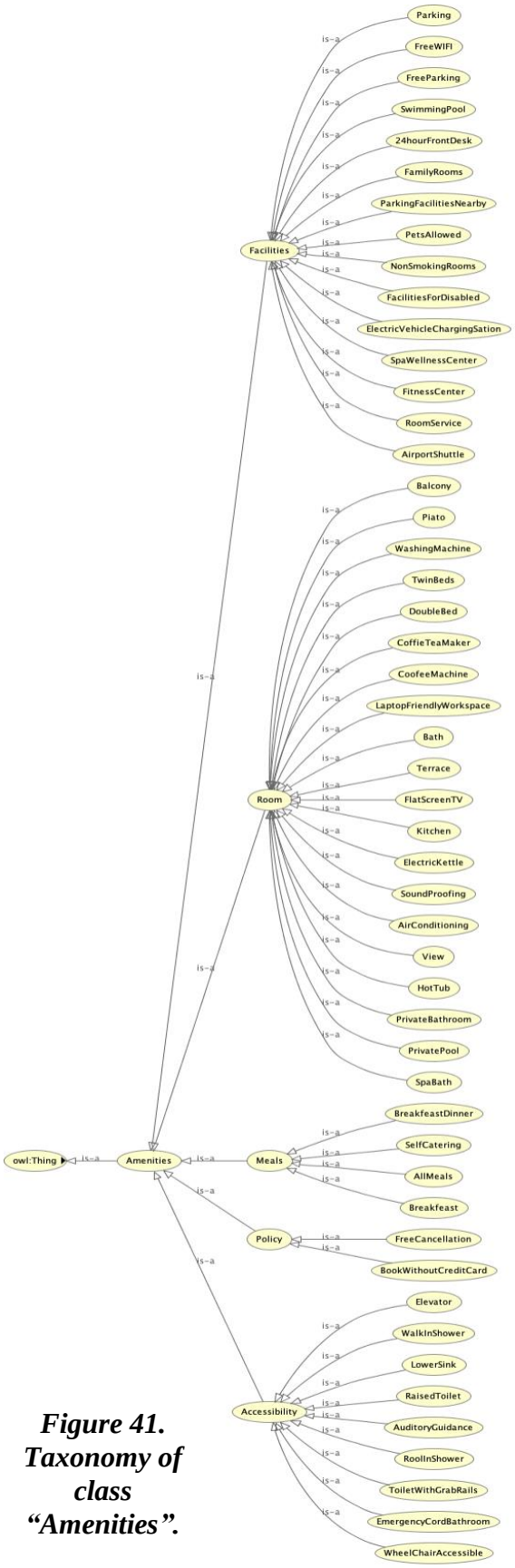


Figure 41. Taxonomy of class "Amenities".

9.2.6 A purposed new class of "Amenities"

Describing the ontology, it was noticed that there is no notion of amenities that would contribute to the characterization of accommodations. Air conditioner, double bed, private shower, and accessibilities for disabled ones are important details that are of experience. Such class is not restricted to just accommodations, but museums, apartments, and other establishments. As the taxonomy is quite self-sufficient there would be only id as their attribute (22. table).

Table 33. – "Amenities" class attributes

XML name	Mandatory	Data type	Enumeration	Described as another class of
id	Yes	string	-	-

9.2.7 A purposed new class of “Route”

In ontology, there is already a base feature of geolocation and it is a point that is represented by the class of **“Place”** as it allows via its attributes to set latitude and longitude. The class is also used in other top classes like **“PointOfInterest”**. The proposal is to extend the ontology's geolocation capabilities by introducing the second most basic feature – the line. It could mark a geoinformation from e.g., hotel to some point of interest. The attributes for such a class are following in 34. Table. The relationship definition is defined in later sections.

Table 34. – “Route” class attributes

XML name	Mandatory	Data type	Enumeration	Described as another class of
name	Yes	string	-	**-**
price	No	real	-	**-**
dateOfRequest	No	date	-	**-**
dateOfDischarge	No	date	-	**-**
pointOfDeparture	No	entity	-	**Place**
pointOfArrival	No	entity	-	**Place**

9.2.1 A purposed new class of “Review”

Another purposed class could represent user feedback from some museum, hotel, or restaurant as many use them in their touristic plans to use the service or to visit. For such a possibility the class “Review” is introduced. It does not have any taxonomy, but it has a set of attributes as shown in 35. Table.

Table 35. – “Review” class attributes

XML name	Mandatory	Data type	Enumeration	Described as another class of

id	Yes	string	-	-
title	Yes	string	-	-
reviewText	Yes	text	-	-
rating	Yes	integer	-	-
timeOfReview	No	date	-	-
author	Yes	entity	-	**Place**
targetOfReview	Yes	entity	-	**PointOfInterest**

9.2.1 Utilization of external domains

As it was already mentioned that the food plays significant part in touristic activity – some of the its domain things is purposed to be included in the tourism. Mainly 4 classes of **"FoodBeverage"**, **"Ingredient"**, **"Menu"** and **"Characteristic"**. They are imported in the schema file and used to form relationships like between market and what can it sell. Their definitions are present in next chapter.

9.2.2 Relationship extensions

Previously described classes that were proposed - there is the following necessity to introduce new relationships and definitions.

For **"Amenity"** and **"Route",** it is proposed to reuse the definition of "**belongsTo**" to establish the relationship between **"Amenity"** and **"Collection"**; **"Route"** and **"Collection".**

- from **"Amenity"** to **"Collection"** by **"belongsTo"** (inverse as **"includes"**).
- from **"Route"** to **"Collection"** by **"belongsTo"** (inverse as **"includes"**).

For **"Restaurant"** and **"Type",** it is proposed to reuse the definition of "**isA**" to establish the relationship between **"Restaurant"** and **"Type".**

- from **"Restaurant"** to **"Type"** by **"kindOf"** (inverse as **"isA"**).

It is proposed to introduce new relationship definitions like:

1. **“interests”** (inversed as **“interestedIn”**)

   Applied to:

   - from **“Person”** to **“Interest”** by **“interestedIn”** (inverse as **“interests”**).

2. **“associatesWith”** (inversed as **“associatesWith”**)

   Applied to:

   - from **“Visit”** to **“Interest”** by **“associatesWith”** (inverse as **“associatesWith”**).

3. **“includedIn”** (inversed as **“amenityIncluded”**)

   Applied to:

   - from **“PointOfInterest”** to **“Amenity”** by **“amenityIncluded”** (inverse as **“includedIn”**).

4. **“hasPointOfInterest”** (inversed as **“alongTheRoute”**)

   Applied to:

   - from **“PointOfInterest”** to **“Route”** by **“alongTheRoute”** (inverse as **“hasPointOfInterest”**).

5. **“hasParticipant”** (inversed as **“participantIn”**)

   - **“role”**, type select, not mandatory. Enumeration:

     i. **Actor**
     ii. **Organized**
     iii. **Manager**
     iv. **Guide**
     v. **Musician**

   Applied to:

   - from **“Person”** to **“Event”** by **“participantIn”** (inverse as **“hasParticipant”**).

6. **“soldIn”** (inversed as **“sell”**)

   Applied to:

   - from **“Bakery”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).
   - from **“Butcher”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).
   - from **“Diary”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).
   - from **“Fish”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).
   - from **“Grocery”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).

- from **“Minimarket”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).
- from **“Supermarket”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).
- from **“Confectionery”** to **“FoodBeverage”** by **“sell”** (inverse as **“soldIn”**).
- from **“Bakery”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).
- from **“Butcher”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).
- from **“Diary”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).
- from **“Fish”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).
- from **“Grocery”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).
- from **“Minimarket”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).
- from **“Supermarket”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).
- from **“Confectionery”** to **“Ingredient”** by **“sell”** (inverse as **“soldIn”**).

7. **“menuOf”** (inversed as **“hasMenu”**)

   Applied to:

   - from **“Bar”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Pub”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Cafe”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Brasserie”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Canteen”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“SnackBar”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“GrillHouse”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Stakehouse”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Pizzeria”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Tavern”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“TeaHouse”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).
   - from **“Restaurant”** to **“Menu”** by **“hasMenu”** (inverse as **“menuOf”**).

10 References

[1] S. Ferilli and D. Redavid, “The GraphBRAIN System for Knowledge Graph Management and Advanced Fruition.” [Online]. Available: http://193.204.187.73:8088/GraphBRAIN/

11 Attachments

1. Attachment (Full XML schema of tourism domains ontology)

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="domain">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="entities">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="entity" maxOccurs="unbounded" minOccurs="0">
        <xs:complexType>
         <xs:sequence>
          <xs:element name="attributes">
           <xs:complexType>
            <xs:sequence>
             <xs:element name="attribute" maxOccurs="unbounded" minOccurs="0">
              <xs:complexType mixed="true">
               <xs:sequence>
                <xs:element name="values" minOccurs="0">
                 <xs:complexType>
                  <xs:sequence>
                   <xs:element name="value" maxOccurs="unbounded" minOccurs="0">
                    <xs:complexType>
                     <xs:simpleContent>
                      <xs:extension base="xs:string">
                       <xs:attribute type="xs:string" name="name" use="optional"/>
                      </xs:extension>
                     </xs:simpleContent>
                    </xs:complexType>
                   </xs:element>
                  </xs:sequence>
                 </xs:complexType>
                </xs:element>
```

```
        </xs:sequence>
        <xs:attribute type="xs:string" name="datatype" use="optional"/>
        <xs:attribute type="xs:string" name="mandatory" use="optional"/>
        <xs:attribute type="xs:string" name="name" use="optional"/>
        <xs:attribute type="xs:string" name="target" use="optional"/>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="taxonomy" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="value" maxOccurs="unbounded" minOccurs="0">
       <xs:complexType mixed="true">
        <xs:sequence>
         <xs:element name="attributes" minOccurs="0">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="attribute" maxOccurs="unbounded" minOccurs="0">
             <xs:complexType>
              <xs:simpleContent>
               <xs:extension base="xs:string">
                <xs:attribute type="xs:string" name="datatype" use="optional"/>
                <xs:attribute type="xs:string" name="mandatory" use="optional"/>
                <xs:attribute type="xs:string" name="name" use="optional"/>
               </xs:extension>
              </xs:simpleContent>
             </xs:complexType>
            </xs:element>
           </xs:sequence>
          </xs:complexType>
         </xs:element>
         <xs:element name="taxonomy" minOccurs="0">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="value" maxOccurs="unbounded" minOccurs="0"/>
```

```
                    </xs:sequence>
                  </xs:complexType>
                 </xs:element>
                </xs:sequence>
                <xs:attribute type="xs:string" name="name" use="optional"/>
               </xs:complexType>
              </xs:element>
             </xs:sequence>
            </xs:complexType>
           </xs:element>
          </xs:sequence>
          <xs:attribute type="xs:string" name="name" use="optional"/>
         </xs:complexType>
        </xs:element>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
     <xs:element name="relationships">
      <xs:complexType>
       <xs:sequence>
        <xs:element name="relationship" maxOccurs="unbounded" minOccurs="0">
         <xs:complexType>
          <xs:sequence>
           <xs:element name="references">
            <xs:complexType>
             <xs:sequence>
              <xs:element name="reference" maxOccurs="unbounded" minOccurs="0">
               <xs:complexType>
                <xs:simpleContent>
                 <xs:extension base="xs:string">
                  <xs:attribute type="xs:string" name="object" use="optional"/>
                  <xs:attribute type="xs:string" name="subject" use="optional"/>
                 </xs:extension>
                </xs:simpleContent>
               </xs:complexType>
              </xs:element>
             </xs:sequence>
```

```
            </xs:complexType>
          </xs:element>
          <xs:element name="attributes" minOccurs="0">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="attribute" maxOccurs="unbounded" minOccurs="0">
                  <xs:complexType>
                    <xs:simpleContent>
                      <xs:extension base="xs:string">
                        <xs:attribute type="xs:string" name="datatype" use="optional"/>
                        <xs:attribute type="xs:string" name="mandatory" use="optional"/>
                        <xs:attribute type="xs:string" name="name" use="optional"/>
                      </xs:extension>
                    </xs:simpleContent>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
        <xs:attribute type="xs:string" name="inverse" use="optional"/>
        <xs:attribute type="xs:string" name="name" use="optional"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute type="xs:string" name="name"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```