Enabling Semantic Referencing of Selected Travel Related Resources

Maciej Gawinecki
Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań, Poland

Minor Gordon
Computer Science Department, Technical University of Berlin, Berlin, Germany

Marcin Paprzycki
Computer Science Department, Oklahoma State University, Tulsa OK, USA and Computer Science, SWPS, Warszawa, Poland
marcin@cs.okstate.edu

Michał Szymczak
Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań, Poland

Zygmunt Vetulani
Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań, Poland

Jimmy Wright
Computer Science Department, Oklahoma State University, Tulsa, OK, USA

Abstract

This paper describes our efforts to semantically describe travel-related documents on the Internet. We focus on the areas of lodging and gastronomy, with allowance for further expansion into other aspects of the world of travel. Based on an analysis of currently-available data, we develop hotel ontology. Furthermore, we explicitly define an ontology for the Chefmoz project data, which has no official schema. Finally, we sketch a way to combine these two ontologies for use in an integrated travel framework.

1. Introduction

The most important of the expected changes in information management in the next decade is the transition from information managed for **display** (e.g. using

Witold Abramowicz (ed.), Business Information Systems, Proceedings of BIS 2005, Poznań, Poland

HTML) in a human-centric "eye-pleasing" form to availability of "intelligent" information ripe for **semantic-level processing**. Toward this end we have begun to investigate issues related to semantic representation of travel-related information ([1],[2],[3]). This paper presents an extension of this work, including a solidified design for an ontology of a *hotel* as well as a re-engineered ontology of a *restaurant* (based on the RDF-demarcated data recorded by the Chefmoz project).

We proceed as follows. In the next section we introduce the basic concept of ontologies. A description of the proposed *hotel ontology* follows. In Section 4, we analyze and interpret the publicly-available Chefmoz *restaurant ontology*. Finally, in Section 5, we discuss possible approaches to connecting these two ontologies.

2. Concept of an ontology

The term *ontology* originates from philosophy, where it denotes a branch of philosophy devoted to answering two basic questions: (a) what exists? and (b) if what exists is (at a given level) divisible into parts, then what are these parts and what are the relationships between them? In the world of computing these questions may be interpreted as: (a') what objects, concepts, and other entities are assumed to exist in the context of an area under consideration, and (b') what are the relationships between these objects? ([4],[5]). The resulting conceptualization is an abstract, simplified view of a part of the "world" to be represented for some purpose.

Ontologies are often equated with taxonomic hierarchies of classes, class definitions, and the subsumption relation (similarly to the structuring of the world represented by Google and Yahoo directories [1]). However, this is a rather narrow understanding of a much richer concept. Ontologies are not limited to definitions in the traditional logic sense, i.e. only introducing terminology and not adding any knowledge about the world [6]. In an ontology, definitions are associated with names of entities in the universe of discourse (e.g., the set of objects and their relations that can be represented) along with formal axioms that constrain the interpretation and specify well-formed use of these terms. Finally, an ontology defines the basis of the vocabulary with which queries and assertions are exchanged between parties (such as software agents) ([7], [8]), and ontological commitments are agreements by parties to use the shared vocabulary in a coherent and consistent manner.

There are two basic approaches to developing ontologies. One is to try to create a "top-down" ontology that is robust enough to encapsulate an extremely large number of "notions;" starting from the most general concepts and descending towards more specific ones [28]. The second approach (which is also the approach of our project) is to follow a "bottom-up" route. Here, domain-specific ontologies are designed first and then linked / combined to form a whole (this is the vision of the Semantic Web [9]). In [3] we have presented an

overview of best known top level ontologies as well as travel-related minimalist ontologies.

Our interest in ontologies relates to the project of developing an agent based travel support system. While the design of the project changed already a number of times, in each iteration, the core consisted of a collection of ontologically demarcated resources collected for intelligent browsing and personalized content delivery ([29],[30],[31],[32]).

Here, we proceed by describing in detail the proposed hotel ontology.

3. Hotel ontology

The starting point for our *hotel* ontology is the general travel ontology described in [3] (and this source should be used to provide further details on the general level). For the design of that ontology we considered business aspects (i.e. *customers* – hotel users; and *service providers* – hotels; and their relationships) and real-world aspects (the concept of a site) of the "world of travel." Here we have to stress that our current hotel ontology is based on conceptualizing the *user who will stay in the hotel* $\leftrightarrow$ *hotel as a service provider* relationship. Obviously, different versions of a hotel ontology would have to be developed if the view of the hotel from the "janitorial perspective" was considered (e.g. hotel room as a place that needs to be cleaned and prepared for the next guest when the guest checks-out or room that needs to be made-up when the guest stays) or if a hotel is viewed as a conference / event site (for more details how multiple ontologies interact with each other to provide different "views" of the same "data / object" see [24]).

A further methodological comment is in order. There are many ways to develop an ontology. We have proceeded with a particular goal in mind – to use our ontology to (1) organize information gathered form the Internet and (2) to facilitate delivery of personalized content through an "intelligent" search engine. This being the case we have analyzed the way in which a *hotel* is described in Internet documents and tried to formalize these descriptions in our ontology. Obviously, other possible hotel ontologies could have been created (e.g. if a starting point was a set of dictionary definitions and theoretical considerations). In the case when a number of hotel ontologies were to be created, the well known problem of ontology-matching would have to be solved. However, this problem and its possible solution reside outside of the scope of our current interest.

Since in our case the concept of a hotel reflects descriptions of hotels as found within web pages, to develop the hotel ontology we started by analyzing the better known travel-related web sites:

- http://www.travelocity.com
- http://www.venere.com
- http://www.hotelclub.net

- http://www.hrs.de
- http://www.web-hotels.com
- http://www.travelweb.com
- http://www.travelciti.com.

These sites belong to the "top-10" of Google's response to the query "hotel reservation" (and thus the core of the relationship: *prospective user* ↔ *prospective service provider*). We looked at descriptions of hotels in general and, in order to make our study more focused, we paid especial attention to data for the Warsaw Le Royal Meridien Bristol hotel. We found that **all** sites carry practically the same information about hotels, which is summarized in Table 1.

Table 1. Summary of typical, WWW available, information about hotels

Element of description	Did we include it
Name	Yes
Address	Yes
Rating (number of stars)	Yes
Total number of rooms	Yes
Room types with their prices	Yes
Amenities	Yes
Way of reaching the place	Yes
Dining	Yes
Nearby attractions	No
Transportation	Yes
Accepted means of payment	Yes
Accepted currencies	No
Check in/out time	Yes
Booking and cancellation policy	No
Guaranteed rates & other	No
Pets accepted	Yes
Photos of the property	No
City map with hotel location	No

Let us make a few comments about the characteristics that we have identified and used:

(1) thanks to the flexibility of ontology demarcation languages, omitted characteristics can be easily added later, e.g. nearby attractions may be just an additional set of the Site class elements etc.,

(2) since geospatial information processing (maps and photos) constitutes a very large and separate (active research) area, we decided for the time being to omit this category,

(3) as far as the information about rooms and their prices was concerned, we found very little consistency among sites: most pages provided only simple information, e.g. room with double queen bed; in other cases only general written comments about room types were

provided; furthermore, in some sites a distinction between a hotel's amenities and room amenities was made, while in other cases it was not; finally not all types of room were listed and we were not sure about the reasons for this situation. Based on these experiences, we expect that some of the data related to the rooms may remain unfilled, or partially filled.

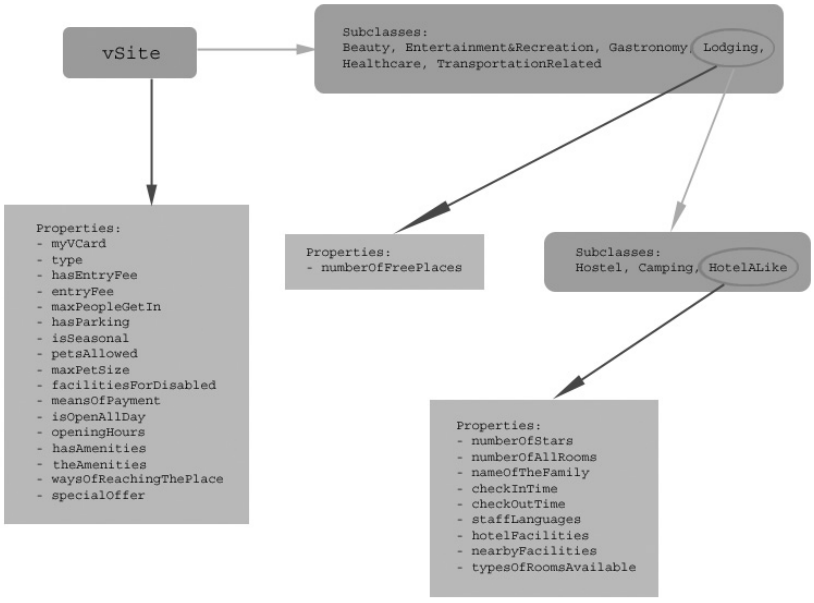

Figure 1. Structure of the top-level class *site*

3.1. The general ontology architecture

At the top level the proposed *hotel ontology* consists of the following general classes:

- *site* – different types of places to visit (not necessarily travel related), where hotel is one of possible examples of sites, while a restaurant would be another,
- *hotel room* – types of rooms in particular hotel,
- *amenities* – available within the site and in the hotel room.

First, we describe in detail each of these aspects and then combine them in order to derive a complete description of a hotel.

3.1.1. Site

As noted above, the site represents the real-world related characteristics of places such as cinemas, hospitals, bars, arenas, hotels or restaurants. Those properties are inherited by all subclasses of site included in the ontology, which, additionally, have their own specific characteristics, e.g. camping site can have support for campers (large camping vehicles) or not, while a hotel instance includes the number of rooms in the hotel, availability of a health center etc. Figure 1 depicts the structure of this class.

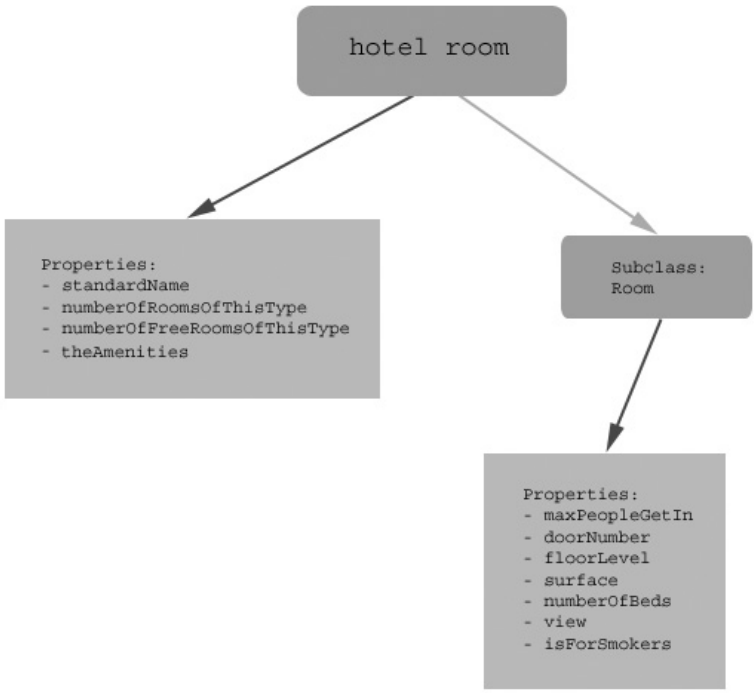

Figure 2. General structure of the top-level class *hotel room*

3.1.2. Hotel room

Hotel room is the fundamental class that defines the concept of the hotel (the most basic intuition is that hotel = place with a large number of rooms for rent). Again, on the basis of our analysis of information from the web we have specified a number of properties directly defining this class (e.g. room standard or number of rooms in a given hotel). Additionally, we have created a subclass

room, with its own, specific, properties. The general structure of the *hotel room* class is depicted in Figure 2.

3.1.3. Amenities

The *amenities* class defines the possible amenities available in a given hotel and/or room. Figure 3 presents the general schema of the proposed class, while its utilization is discussed in the subsequent sections (see also Figure 4).

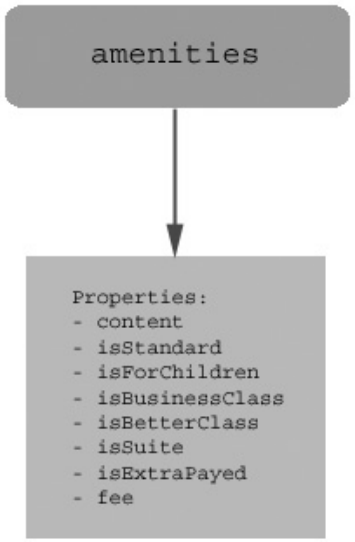

Figure 3. General structure of the *amenities* class

3.1.4. Putting it all together

In order to obtain the complete hotel description we combine the three classes. To show all the amenities of the hotel, e.g. fax machines, phone booths in the main hall or a playground for children, an instance of hotel has properties that refer to objects derived from the class amenities. To display a hotel and its nearby available facilities, e.g. golf course, bar or swimming pool, we simply refer to the sites attached to a hotel. When we want to provide details of room types, we refer to the description based on the hotel room class again. To present certain room type amenities such as a phone, a safe deposit box or an alarm clock, we again use subclasses in a manner similar to the way we use subclasses of hotel amenities. In Figure 4 we illustrate the "merged" ontology together with an example of possible values that together describe an imaginary hotel.

Note that some properties may have multiple objects, i.e. one property may contain several exclusive objects. An example of such a container in Figure 4 is: *meansOfPayment*, which can store multiple objects representing various means of payment. To implement such properties we utilize the *Bag* class from RDF Schema. Details of the implementation follow in the next section.

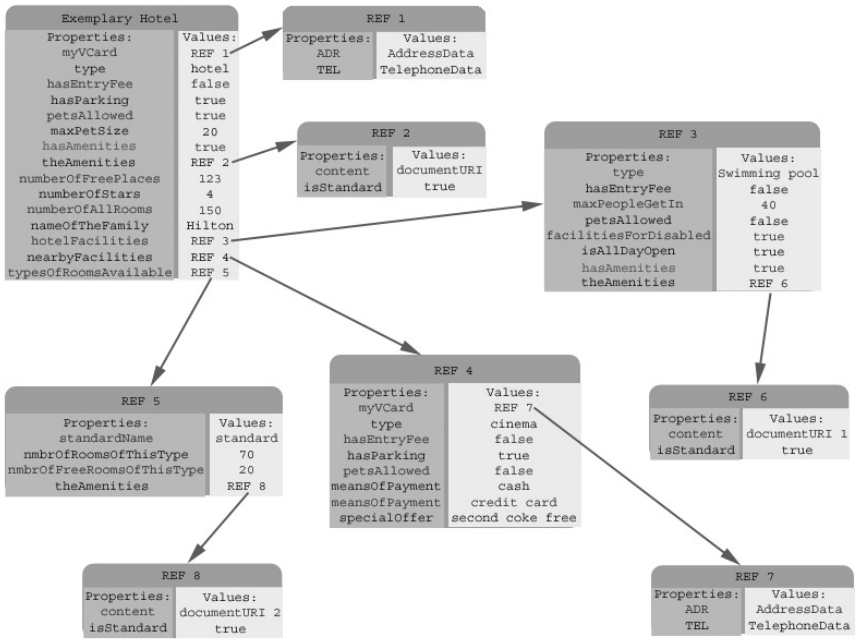

Figure 4. Example of the general structure of a hotel described ontologically using combined classes *site*, *hotel room*, and *amenities* as well as sample on a hotel defined through values assigned to particular properties

3.2. Implementation details

The ontology proposed in the previous section was implemented using the Resource Description Framework [26]. Obviously, we cannot present here the complete code and thus we will only illustrate (with snippets of the RDF code) some of the more important features of the implementation (a complete code can be found in [27]). Let us start with the heading of the *vSite* class, which is the main site class in our ontology (hence other classes do not involve all the namespaces, while this one does):

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF
```

```
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:vCard="http://www.w3.org/2001/vcard-rdf/3.0#"
 xmlns:gastronomy="http://www.agentcities.org/EURTD/Ontologies/restaurant.v4"
 xmlns:localTransport="http://www.agentcities.org/EURTD/Ontologies/local-
transport.v2"
 xmlns:calendar="http://www.agentcities.org/EURTD/ontology/calendar"
 xmlns:HRoom="http://atos.wmid.amu.edu.pl/~d124124/ontologies/hotelroom"
 xmlns:amenities=http://atos.wmid.amu.edu.pl/~d124124/ontologies/amenities>
```

Here, the first two namespaces are standard declarations of coding a schema in the RDF language. The *vCard* provides an identity description. It is important to note that the *gastronomy*, *localTransport* and *calendar* concepts are examples of taking advantages of schemas existing in other well-defined ontologies. In this case we have included namespaces that originate from the Agentcities project [25][1]. The last two namespaces refer to the remaining two classes of our ontology (*hotel room*, and *amenities*, see above; Figure 4 in particular). In comparison to the Travel Ontology introduced in [3] we used a larger number of data types:

```
<rdfs:Datatype rdf:about="&xsd;integer"/>
<rdfs:Datatype rdf:about="&xsd;string"/>
<rdfs:Datatype rdf:about="&xsd;boolean"/>
<rdfs:Datatype rdf:about="&xsd;float"/>
<rdfs:Datatype rdf:about="&xsd;time"/>
<rdfs:Datatype rdf:about="&xsd;dateTime"/>
<rdfs:Datatype rdf:about="&xsd;anyURI"/>
```

The last type is particularly interesting as it allows including any URI as an object. We used it in the class *amenities* to link a document with a list of potential amenities described somewhere else on the Internet. It has been defined as follows:

```
<rdf:Property rdf:ID="content">
        <rdfs:label xml:lang="en-US">List of facilities</rdfs:label>
        <rdfs:comment xml:lang="en-US">
                URI of the list of facilities.
        </rdfs:comment>
        <rdfs:domain rdf:resource="#amenities"/>
        <rdfs:range rdf:resource="xsd;anyURI"/>
</rdf:Property>
```

Here, the domain field specifies "where from" the subject of the statement originates and the range gives a similar information about the object, which is of the URI type declared in the domain field. Let us now look into the definition of *myVCard* property:

```
<rdf:Property rdf:ID="myVCard">
        <rdfs:label xml:lang="en-US">Business card</rdfs:label>
        <rdfs:comment xml:lang="en-US">
```

[1] In the case of the *gastronomy* ontology, at this stage it is a "placeholder." This is because, as it will be seen in the next section, we have decided to re-create and utilize the *Chefmoz* restaurant ontology, which already provides a very large body of ontologically demarcated data. Further discussion of this point is presented below.

```
                Fax and phone numbers, name, address.
        </rdfs:comment>
        <rdfs:domain rdf:resource="#Site"/>
        <rdfs:range rdf:resource="vCard#vCard"/>
</rdf:Property>
```

The next example shows how classes originating from different ontologies can be utilized:

```
<rdf:Bag rdf:ID="theAmenities">
        <rdfs:label xml:lang="en-US">Amenities inluded</rdfs:label>
        <rdfs:comment xml:lang="en-US">
                Objects which list different types of amenities
                (standard, suite, for children, etc.)
        </rdfs:comment>
        <rdfs:domain rdf:resource="#Site"/>
        <rdfs:range rdf:resource="amenities#Amenities"/>
</rdf:Bag>
```

Let us compare the last two code snippets as they illustrate our ontology structure and use of the *Bag* construction. Let us assume that we want to describe a hotel. It is obvious that it has only one name, address and set of telephone numbers. At the same time the same hotel may have several "selections" of amenities, e.g. standard, a set prepared for business guests, or those for children. That is why *myVCard* is defined as a simple Property and *theAmenities* is defined as a *Bag*, i.e. a container of objects associated with a given property. Moreover, both properties are examples of references to different classes in the *range* field, which means that our object may have properties of that class.

Finally, let us present code snippets that define hotel facilities and types of rooms:

```
<rdf:Bag rdf:ID="hotelFacilities">
      <rdfs:label xml:lang="en-US">Hotel facilities</rdfs:label>
      <rdfs:comment xml:lang="en-US">
               Swimming pools, golf courses, restaurants,çetc.
      </rdfs:comment>
      <rdfs:domain rdf:resource="#HotelAlike"/>
      <rdfs:range rdf:resource="#Site"/>
</rdf:Bag>

<rdf:Bag rdf:ID="typesOfRoomsAvailable">
      <rdfs:label xml:lang="en-US">Types of rooms available</rdfs:label>
      <rdfs:comment xml:lang="en-US">
               Group of provided room types with their properties.
      </rdfs:comment>
      <rdfs:domain rdf:resource="#HotelAlike"/>
      <rdfs:range rdf:resource="HRoom#HotelRoom"/>
</rdf:Bag>
```

When the above-described features are combined, we obtain the *hotel ontology* that provides a common framework that hoteliers may use to describe their hotels, rooms and amenities associated with both hotels and rooms available in them. It also enables developers and travelers to work with concepts that exist within the proposed ontology.

4. Restaurant ontology

In addition to hotels, one of the more important classes of travel-related objects is restaurants. Contrary to the hype of the Semantic Web project, there exist very few ontologically-demarcated sources on the public Web. Restaurants, fortunately, are an important exception. The Chefmoz project [10], an outgrowth of the DMOZ open source directory project [11], provides a very large volume of RDF demarcated data about restaurants around the world. Unfortunately, while the Chefmoz site offers an RDF/XML dump of its data and some description of its construction, it does not constitute a true ontology. Thus we have defined a formal ontology for *Chefmoz* centering on the class of *restaurants*. At the same time, among the namespaces of the most general class of the *hotel ontology* we have included the AgentCities-defined *gastronomy*. This kind of situation is likely to remain fairly typical in the early days of development of ontological foundations of the "new Internet." On the one hand we have a purely theoretical (in the sense that no substantial quantity of real-world data demarcated in this way exists on the Web) ontology of *gastronomy*. On the other hand we have a large amount of data demarcated within an implicit and much narrower ontology of a *restaurant*. This means that the problem of ontology matching has to be addressed, but this problem is outside of the scope of this paper. Here we will present the main precepts of the ontology that we have re-covered from the Chefmoz project.

In the Chefmoz project a restaurant is characterized by a specific address, cuisine, parking facilities, accepted credit cards, required dress code and whether smoking is allowed. The address is broken up into address, neighborhood, city, state, country and zip components. An additional, interesting element is the *CrossStreet* feature, which points to the nearest street that crosses the street that the restaurant is on. This approach could be expanded by pointing to the nearest: subway station, tram stop landmark, hotel etc. and, furthermore, naturally combined with the GIS component of a travel support system. Within the Chefmoz data an attempt was made to capture hours of operation of a restaurant. Interestingly, two different versions of this aspect of a restaurant are available. There is an element called *Hours* which contains a human-readable description of the opening times, and a *ParsedHours* element which represents a machine-readable version: the hours are converted to the 24-hour clock and split into seven days of the week, delimited by pipes, for example:

```
<Hours>Monday to Friday 12:00noon - 2:30pm, 6:00pm - 10:00pm; Saturday,
      Sunday, Public Holiday 6:00pm - 10:00pm</Hours>
<ParsedHours>18-22|12-14.5,18-22|12-14.5,18-22|12-14.5,18-22|12-14.5,
      18-22|12-14.5,18-22|18-22</ParsedHours>
```

Our proposed restaurant class consists of a list of specific properties. Each one is unique and clarified by providing and identifier and a comment element. The latter suggests not only the description of the property but also the suggested format that its value should have, for instance:

```
<rdf:Property rdf:ID="Location">
      <rdfs:label xml:lang="en-US">Location</rdfs:label>
      <rdfs:comment xml:lang="en-US">
               Location of the restaurant, represented as a category path.
               For example, a restaurant in New York city would get the
               category path "United_States/NY/New_York"
      </rdfs:comment>
      <rdfs:domain rdf:resource="#Restaurant"/>
      <rdfs:range rdf:resource="&xsd;string"/>
</rdf:Property>.
```

Certain properties may be repeated a few times to list some elements, e.g.

```
<c:Accepts>MasterCard/Eurocard</c:Accepts>
<c:Accepts>bank debit cards</c:Accepts>
<c:Accepts>Japan Credit Bureau</c:Accepts>
<c:Accepts>Visa</c:Accepts>
```

And thus we provide fixed list of possible values for the *Accepts* property:

```
<rdfs:Class rdf:ID="ACCEPTVALUES"/>
      <ACCEPTVALUES rdf:ID="Visa"/>
      <ACCEPTVALUES rdf:ID="Mastercard"/>
      <ACCEPTVALUES rdf:ID="American Express"/>
      <ACCEPTVALUES rdf:ID="Diners' Club"/>
      <ACCEPTVALUES rdf:ID="checks"/>
      <ACCEPTVALUES rdf:ID="gift certificates"/>
      <ACCEPTVALUES rdf:ID="bank debit cards"/>
      <ACCEPTVALUES rdf:ID="Carte Blanche"/>
      <ACCEPTVALUES rdf:ID="Japan Credit Bureau"/>
      <ACCEPTVALUES rdf:ID="EnRoute"/>
```

Note that this approach is different from the *meansOfPayment* proposed above for the hotel ontology. This issue should be resolved in such a way that both ontologies use the same way of defining payment methods (most likely a separate ontological entity “means of payment” should be defined).

It is worth mentioning that, typically, the choice of the restaurant involves a specific context and thus restaurant description should provide the user not only with “objective” information (such as its address or the menu) but also with features such as food reviews, clientele type and rating of food and service. This fulfills the requirement of comprehensiveness of the ontology.

To close this section we present an example of the complete instance of the restaurant class (this example is fictitious, but consists of true data belonging to multiple Chefmoz listed restaurants):

```
<c:Restaurantr:id="Poland/ZP/Lobez/Caffe_Mexicos,_Restauracja_Kawiarnia1041">
    <c:Location>Poland/ZP/Lobez</c:Location>
    <d:Title>Caffe Mexicos, Restauracja Kawiarnia</d:Title>
    <c:Address>ul. Browarna 1457</c:Address>
    <c:City>Lobez</c:City>
    <c:Country>Poland</c:Country>
    <c:Phone>+48 (91) 197 44 84</c:Phone>
    <c:State>ZP</c:State>
    <c:Zip>73-150</c:Zip>
    <d:Description>Cafe and restaurant.</d:Description>
    <c:URL>http://cafe-mexicos.home.pl/</c:URL>
    <c:MenuURL>http://cafe-mexicos.home.pl/zapraszamy.htm</c:MenuURL>
```

```
    <c:Hours>11-22 daily</c:Hours>
    <c:ParsedHours>11-22|11-22|11-22|11-22|11-22|11-22|11-22</c:ParsedHours>
    <c:Cuisine>Continental / European / Mexican</c:Cuisine>
    <c:Accepts>American Express</c:Accepts>
    <c:Accepts>Diners&#039; Club</c:Accepts>
    <c:Accepts>MasterCard/Eurocard</c:Accepts>
    <c:Accepts>bank debit cards</c:Accepts>
    <c:Accepts>Japan Credit Bureau</c:Accepts>
    <c:Accepts>Visa</c:Accepts>
    <c:Review
r:resource="Poland/ZP/Lobez/Caffe_Mexicos,_Restauracja_Kawiarnia1041"/>
   </c:Restaurant>
```

This definition is followed by a review:

```
   <c:DinerReview
about="Poland/ZP/Lobez/Caffe_Mexicos,_Restauracja_Kawiarnia1041">
    <d:Description>The food was great, EXCELLENT decor, friendly staff great
music that wasn't intrusively loud, just really set the atmosphere, was quite
a shame when I had to walk out the door back into the hassle of a busy
street. A very pleasing experience and I look forward to my next visit ;-)
   </d:Description>
    <d:Date>2002-11-09</d:Date>
    <c:FoodRating>8</c:FoodRating>
    <c:ServiceRating>7</c:ServiceRating>
    <c:AmbianceRating>10</c:AmbianceRating>
    <c:OverallRating>8</c:OverallRating>
    <c:RecommendedDishes>Bistek Rancheros</c:RecommendedDishes>
   </c:DinerReview>
```

The proposed *restaurant ontology* enables a restaurateur to convey all the crucial concepts of a restaurant including the cuisine, the ambiance, the location and others, while providing the used of the travel support system with enough information to make an educated decision as to visiting a given restaurant.

4.1. Problems encountered while using the Chefmoz RDF/XML syntax

The ontology presented above is compatible with the RDF/XML dump of the Chefmoz data. In recognition of this Dmoz [11] (which Chefmoz is a part of) has been given many awards [13] and thus one would expect that the available data will be bug-free. However, one needs to understand that the Chefmoz project has many editors (57,251 volunteers who have reviewed and added websites to the directory – as of May 2004), and thus it is an enormous effort to assure even syntactical correctness of all data (which, in turn points to one of the important long-term problems in the development of a semantically demarcated Internet – when enormous amounts of legacy data will have to be correctly demarcated). Because numerous sites use the ChefMoz data [12], syntax validity is considered the most important problem listed on the Dmoz TO-DO List [13]. Unfortunately, this problem still has not been fully addressed. Attempting to use the Jena Semantic Web Framework [22] to browse and query the RDF/XML files obtained form the Chefmoz directory reveals the following problems:

- Illegal XML characters (entities)
- Well-formed XML: qualified name with a namespace prefix
- OWL validity

4.2. Solutions to the problems

Due to the very large quantity of available data (about 150 MB) we only process RDF statements pertaining to restaurants in Poland. We have prepared a Java application (using the Jena engine) to parse the RDF/XML data, correct each element and confirm changes, similar to the program described in [20]. Additional correcting procedures were implemented in Perl.

4.2.1. Illegal UTF-8 encoding and XML characters (entities)

Syntax validity problems in the ChefMoz data were previously analyzed by R. Steven Rainwater, one of the Dmoz editors, who implemented a program for checking XML and the UTF-8 characters [15]. Let us describe the problem in more detail. Chefmoz encodes its RDF data in UTF-8, one of the standard Unicode character sets [16]. Moreover, in accordance with the XML standard [19] Chefmoz exchanges some characters with entities, e.g. using *&* instead of the ampersand character (*&*). Unfortunately, this approach – called character escaping – has been performed incompletely.

The exceptions were found using Perl scripts focused on adherence to the UFT-8 rules [16]. When problems were found, non-Latin letters (ones with diacritic signs) were corrected. XML entity exchanges were performed according to [17]. The remaining part of the data was corrected "manually" (by applying a specifically-tailored Java application) to fix a number of additional, unreferenced sign problems (this manual approach is also often used by the Dmoz managers).

4.2.2. Well-formed XML - qualified names with namespace prefixes

XML use namespaces to qualify names into namespaces according to their meaning and definitions (`<c:Restaurant>` instead of `<Restaurant>`). The namespace of most of the Chefmoz elements is identified by the following URI: *http://chefmoz.org/rdf/elements/1.0/*. Similarly certain elements come from Dublin Core namespace. Namespace correlations were corrected using Perl script with regular expressions and manually.

4.2.3. OWL Validity

Since the RDF file is to be used within the Jena Semantic Web Framework it was tested against the Jena OWL Syntax Checker (OWL - Web Ontology Language [23]). Due to the returned warnings ("Unqualified use of rdf:about has been deprecated") we have exchanged all *about* string with *rdf:about*.

As a result of the above-described procedures we have obtained a clean set of ontologically demarcated data describing Polish restaurants.

5. Hotel-restaurant integration

Obviously, there exists a direct relationships between hotels and restaurants. For example, a hotel may have a restaurant as an amenity. At the same time it is possible that a restaurant is one of the sites in the proximity of a hotel. As mentioned above, one of the problems that we encountered in designing our hotel and restaurant ontologies is the existence of multiple ontologies for a given area and the need to integrate these potentially-conflicting views of the world. Let us assume for a moment that the problem of integrating the AgentCities *gastronomy* and Chefmoz *restaurant* classes has been resolved and sketch briefly the way in which the resulting ontology can interplay with our *hotel*. Here, we consider the case of integrating *restaurant* into the *hotel*. Obviously, the reverse process will proceed similarly. The first step is replacing the top-level

```
xmlns:gastronomy="http://www.agentcities.org/EURTD/Ontologies/restaurant.v4
```

namespace by the newly resolved ontology, e.g.

```
xmlns:HRoom="http://atos.wmid.amu.edu.pl/~d124124/ontologies/gastronomy.
```

This will allow us to utilize the full set of concepts defined there.

In the case of a restaurant being one of the on-site facilities of the hotel, when constructing the ontology of a hotel it is important to define an amenity so that it can contain a link to the restaurant ontology (which has just been made available). In this way we make the restaurant ontology an essential part of our hotel ontology.

Since the Chefmoz ontology already has a GIS-like *CrossStreet* feature, we can slightly extend this to work with the *nearbyHotelFacilities* feature of our hotel class. For instance, the new concept *nearbyFacilities* that both ontologies share could become a Bag containing information about sites that are near a given site (we omit here all the possible problems with the fuzziness of the definition of being near-by and the special case of near-by = on-site). In this way the hotel could contain information about having a restaurant on-site (we may need to link the restaurant twice: (1) as an amenity and (2) through *nearbyFacilities*) and the restaurant information about being on-site of the hotel and a link to the hotel (through its URI).

Obviously, this is also a solution for the case when the restaurant is near-by the hotel (and vice-versa). In this case, the connection would run through the *nearbyFacilities* feature of both the hotel and the restaurant, which would point to the restaurant/hotel (similarly to the cinema being near-by the hotel in Figure 4). In the case of the hotel the connection would look as follows:

```
<nearbyFacilities>
 <facilities:Restaurant
r:resource="someURL#Poland/ZP/Hel/Caffe_Zyzyio,_Kawiarnia10410456"/>
</nearbyFacilities>
```

and within the restaurant instance:

```
<nearbyFacilities>
  <facilities:Hotel
  r:resource="someURL#Hel_Janina_Hotel"/>
</nearbyFacilities>.
```

The compete picture of the interactions between *hotel* and *restaurant* ontologies is presented in Figure 5. This figure can be also used to visualize the restaurant ontology discussed above.

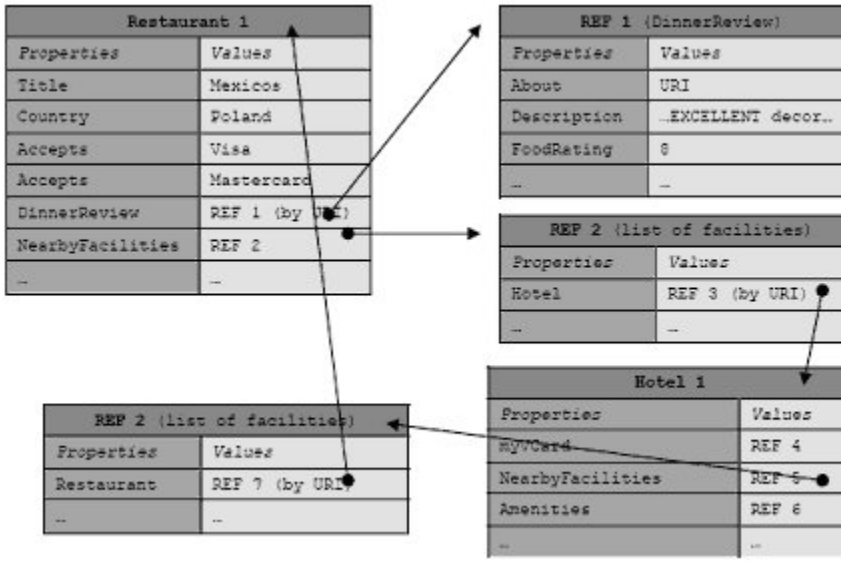

Figure 5. Example of the general structure of integration of hotel and restaurant ontologies

6. Concluding remarks

In this paper we have presented our attempt at developing framework for semantically demarcating travel-related concepts of a *hotel* and a *restaurant*. In the case of the *hotel* we have developed our own ontology based on hotel-related information available on the Internet. Furthermore, the proposed ontology focuses entirely on the *guest* ↔ *hotel* relationship. A *restaurant* ontology was re-engineered out of the implicit ontology underlying the Chefmoz project. Since these two developments proceeded semi-independently from bottom-up approaches, we ended up with two ontologies in need of integration. We have sketched some ideas as to how the integration may proceed on the top level, allowing hotel and restaurant ontology processors to "recognize" their opposites

as on-site and near-by facilities. However, even an initial analysis indicates that both ontologies utilize a number of shared concepts that should be encoded in a different way: e.g. GIS features or the payment methods. This seems to suggest that a number of mini-ontologies may need to be implemented to be shared across travel-related entities. Integration and development of ways for co-existence of hotel and restaurant ontologies as well as conceptualization of related concepts of: pub, bar, motel, B&B. are the subsequent planned steps of our project.

7. References

1. A. Gilbert, M. Gordon, M. Paprzycki, J. Wright, *The World of Travel: a Comparative Analysis of Classification Methods*, Annales UMCS Informatica, A1, 2003, pp. 259-270.
2. A. Gilbert, A. Nauli, M. Paprzycki, M. Gordon, S. Williams, J. Wright, *Indexing Agent for Data Gathering in an e-Travel System*, Informatica, Vol. 28, No. 1, 2004, pp. 69-78.
3. M. Gordon, A. Kowalski, M. Paprzycki, T. Pełech, M. Szymczak, T. Wąsowicz, *Ontologies in a Travel Support System*, submitted for publication.
4. M.R. Genesereth, N. Nilsson, *Logical Foundation of Artificial Intelligence*, Morgan Kaufmann, Palo Alto, CA, 1986.
5. http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.
6. H.B. Enderton, *A Mathematical Introduction to Logic*, Academic Press, N. Y., NY, 1972.
7. H.J. Levesque, *A Logic of Implicit and Explicit Belief*, in: Proceedings of the Fourth National Conference on Artificial Intelligence, Austin, TX, pp. 198-202.
8. H.J. Levesque, P.R. Cohen, J.H.T. Nunes, *On Acting Together*, in: Proceedings of the Eighth National Conference on Artificial Intelligence, Boston, MA, 1990, pp. 94-99.
9. D. Fensel, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, Springer, Berlin, Germany, 2001.
10. http://chefmoz.org.
11. http://dmoz.org.
12. http://freshlinks.net/odp.aspx/Computers/Internet/Searching/Directories/Open_Directory_Project/Tools_for_Editors/ChefMoz_Editors/Sites_Using_ChefMoz_Data/
13. http://dmoz.org/awards/.
14. http://www.rainwaterreptileranch.org/steve/sw/odp/rdflist.html.
15. http://www.utf-8.com/.
16. http://www.w3.org/International/questions/qa-forms-utf-8.
17. http://www.mountaindragon.com/html/iso.htm.
18. http://www.rainwaterreptileranch.org/steve/sw/odp/index.html.
19. http://www.virtualpromote.com/tools/validate-xml/.
20. http://www.topazguide.com/.
21. http://www.w3.org/1999/07/WD-xml-c14n-19990729.
22. http://jena.sourceforge.net.
23. http://www.w3.org/TR/owl-features/.

24. J. Jakubczyc, V. Galant, M. Paprzycki, M. Gordon, *Knowledge Management in an E-commerce System*, in: Proceedings of the Fifth International Conference on Electronic Commerce Research, Montreal, Canada, October, 2002, CD, 15 pages.
25. http://www.agentcities.org.
26. *RDF Primer*, http://www.w3.org/TR/rdf-primer.
27. http://atos.wmid.amu.edu.pl/~d124124/ontologies/.
28. http://www.cyc.com/.
29. M. Paprzycki, R. Angryk, K. Kołodziej, I. Fiedorowicz, M. Cobb, D. Ali and S. Rahimi, *Development of a Travel Support System Based on Intelligent Agent Technology*, in: S. Niwiński (ed.), Proceedings of the PIONIER 2001 Conference, Technical University of Poznań Press, Poznań, Poland, 2001, 243-255.
30. M. Paprzycki, P. J. Kalczyński, I. Fiedorowicz, W. Abramowicz, M. Cobb, *Personalized Traveler Information System*, in: B. F. Kubiak and A. Korowicki (eds.), Proceedings of the 5th International Conference Human-Computer Interaction, Akwila Press, Gdańsk, Poland, 2001, 445-456.
31. M. Paprzycki, V. Galant, *Information Personalization in an Internet Based Travel Support System*, in: Abramowicz (ed.), Proceedings of the BIS'2002 Conference, Poznań University of Economics Press, Poznań, Poland, 2002, 191-202.
32. M. Paprzycki, R. Angryk, V. Galant, M. Gordon, *Travel Support System – an Agent Based Framework*, in: H. R. Arabnia, Y. Mun (ed.), Proceedings of the International Conference on Internet Computing (IC'02), CSREA Press, Las Vegas, NV, 2002, 719-725.