DO-ROAM: Activity-Oriented Search and Navigation with OpenStreetMap

Mihai Codescu[1], Gregor Horsinka[1], Oliver Kutz[2],
Till Mossakowski[1,2], and Rafaela Rau[1]

[1] DFKI GmbH Bremen
[2] Research Center on Spatial Cognition (SFB/TR 8),
University of Bremen, Germany

Abstract. We develop a web service focusing on finding places not (only) by their address, but by systematically relating the places to activities that a person could perform there. This is helpful if a person wants to explore a new city, or plans leisure activities. OpenStreetMap provides a rich set of tags that can be used for activity-oriented search. We propose the use of several ontologies that are related to each other using matching tools to cope with the evolving nature of the tags available in social media.

1 Introduction

OpenStreetMap has evolved into a rich source of geodata that in some aspects (like e.g. the level of detail for certain pedestrian lanes) even gets ahead of Google maps. When searching and navigating through a map portal like `http://www.openstreetmap.org`, semantic metadata could greatly help with providing an *intention* and *activity*-based access to the data. In the case of OpenStreetMap, the metadata is provided in the form of *tags* that are entered into the database in a Social Web and wiki-like manner. Metadata obtained through such Social Web, collaborative and community based efforts have specific characteristics, namely evolve in a bottom-up way, contain a lot of noise (typos, redundancies, etc.) and are subject to constant change. A main challenge now is how to use such metadata in flux in a meaningful way for an activity-based search and navigation tool. In this paper, we use ontologies and (semi-automatically generated) ontology mappings for bridging the gap between (a single) user's intentions and the (community generated) metadata tags. This approach provides a relatively simple, yet effective solution to the generally rather hard problem of how to relate data to ontologies (see [14]).

Based on this, we have developed an open source tool—DO-ROAM[3]—which is a prototype providing, beyond the usual search facilities inherited from the OpenStreetMap portal, an ontology-based search for *located activities* and *opening hours*.

[3] Freely available at `www.do-roam.org`.

1.1 Related Work

The GeoShare project did pioneering work on ontology-based integration of geodata sources and services [9]. Their system performs ontological, spatial and temporal reasoning when processing user queries. However, as the authors note, "all application ontologies have in common that they are based on the same vocabulary". We here follow a more flexible approach based on ontology matching. Moreover, the authors of the GeoShare/Buster software told us that it is not used by any web service, and it would be a great effort to get the software running again. Indeed, we have the impression that the user interface was too complex—a stripped down version, however without the ontology-based search, is still online[4].

Google maps[5] obviously uses a mixture of full-text search and search in a taxonomy of categories; unfortunately, only parts of the taxonomy are openly available. While full-text search can in some cases provide extra value, sometimes it can also produce misleading results. For example, when entering "new york barber restaurant" or "new york barber near restaurant", you get results in the category "restaurant" for which the word "barber" occurs in a related text document (or vice versa), but only few restaurants near barbers. Searching for "charging station" in most cities does not deliver any results at all.

In comparison, whilst searching for "charging station" in OpenStreetMap does in fact not deliver any results at all, OSM's internal data is open and publicly accessible, and thus better search methods, employing e.g. the OSM tags, can be utilised in order to semantically enrich queries. Thus, we intend to realise an activity-oriented search where several activities can be combined, thereby leading to various possibilities for searching for nearby places, or for the restriction of a search to certain opening hours.

Google city tours[6] suggests touristic tours starting from a given point; however, the user cannot enter specific activities. Other works that stress the importance of activities and actions in GIS include [10, 16], who argue that in order to make geographical information really useful, corresponding ontologies would have to be designed with a focus on human activities rather than being 'static and entity-based'. Moreover, similar to our approach mapping metadata tags to activities in an ontology, e.g. [10] proposes to exploit textual descriptions of activities in order to derive domain ontologies. However, unlike the present paper, these works do neither employ statistical matching methods to link these two layers, nor do they use the OWL language, nor apply ideas from recent progress in ontology-based data access.

Our work has been much inspired by an activity-oriented interactive route planning system [18], see Fig. 1 and `http://www.digitaltravelmate.net`. This system allows the user to specify interesting locations via holiday activities, and routes are planned along locations where the selected activities can be performed.

[4] `http://www.geoshare.umwelt.bremen.de`
[5] `http://maps.google.com`
[6] `http://citytours.googlelabs.com`

Fig. 1. Activity-oriented interactive route planning system

Routes can also be interactively corrected. However, the system is based on a *fixed* fictitious map, and on a small predefined set of activities.

1.2 Organisation of the Paper

The aim of our work reported in this paper is to provide activity-oriented map search and navigation with an *evolving* set of activities based on OpenStreetMap's tags, which are continuously changing due to the wiki-nature of OpenStreetMap.

The paper is organised as follows. Section 2 provides a set of use cases, and in Section 3 we describe the overall architecture of our DO-ROAM system. The main parts of this system are then described in further detail in the subsequent sections: Section 4 recalls some technical background on the web ontology language OWL, Section 5 introduces the ontology of activities and Section 6 the ontology of OpenStreetMap tags, Section 7 discusses the mapping between these two ontologies, and Section 8 finally contains a discussion on how the ontology search is integrated with the actual representation of data. Section 9 concludes and discusses future work.

2 Motivating Use Cases

In this section, we describe some scenarios in which people search for locations where certain activities take place and for routes that include such activities.

One important such application scenario is electric mobility, in particular due to the limited battery reach of electric automobiles and relatively long charging times. Notice that we assume that the map presented to the user is taken from OpenStreetMap; the approach is, however, flexible and it could also use multiple data sources. [1] provide different navigation scenarios based on an ontology for GIS, which inspired the format of our use cases.

- *Scenario 1:* Alan spends some days in a city and he want to charge his electric car. First, he wants to know where he can find a charging station. Second, he is interested in which activities he could do within walking distance from the location of the charging station.
- *Scenario 2:* Betty is new in town. She wants to know which activities are offered within her neighbourhood. She also knows she will be getting hungry soon, so she searches for all restaurants close to home and which will be still open within the next two hours.
- *Scenario 3:* Maria wants to visit her friend. On her way she needs to stop at a supermarket, an ATM and a post office. She needs a system which will generate and present to her a route, including all these stops, in any order. She also wants to be able to modify the resulting route.
- *Scenario 4:* Tom wants to travel from A to B. He wants to take the most scenic route possible. He also wants to see displayed all places of his interest within a certain area that he can choose and modify. Furthermore, he wants to get a route suggestion which is still flexible and can be modified at a later stage.

3 General Tool Architecture of DO-ROAM

We have designed and implemented the prototype of a tool DO-ROAM for answering such requests and for assisting the users in spatio-temporal planning of activities. DO-ROAM is an acronym for *Data and Ontology driven Route-finding Of Activity-oriented Mobility*.

Currently, only the search component is implemented and the route finding integration is in progress. Therefore, only Scenarios 1 and 2 from those mentioned in Section 2 are currently supported. The general GUI of the tool is illustrated in Fig. 2. The tool displays a map (based on OpenStreetMap) with a zoom functionality which allows the user to focus on a certain area of interest. Searching for locations which allow to perform desired activities can be done either in a guided way, using the ontology navigation bar on the left of the map, or in a less contrained way, using a text field for introducing the query. In the latter case, address and opening hours can also be taken into account.

The tool is implemented as a Web application, using Ruby on Rails[7], a popular and powerful web application framework. We have built our tool on top of the existing Rails portal for OpenStreetMap[8]. It must solve a data integration

[7] `http://rubyonrails.org/`
[8] `http://wiki.openstreetmap.org/wiki/The\_Rails\_Port`

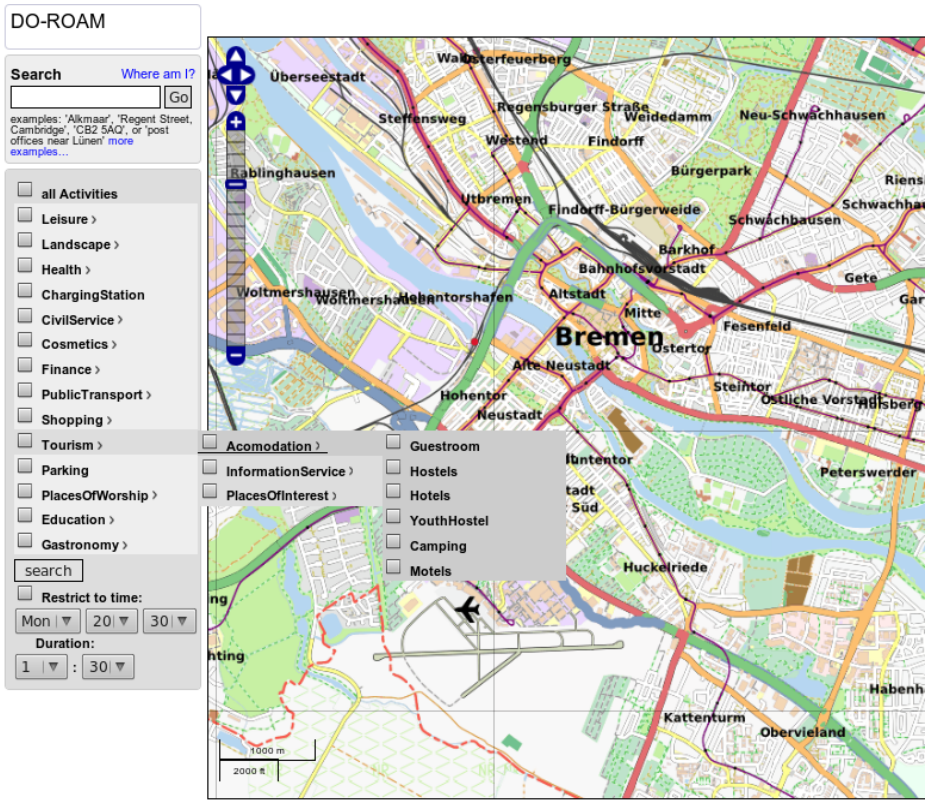

Fig. 2. User interface of DO-ROAM prototype

problem in the sense that the way the OpenStreetMap data is represented should not be directly visible to the user, and the interaction with the user should be as facile as possible. The solution employed is *ontology-based data access* (OBDA) [15, 4], where the domain of interest is modelled as an ontology which is connected with the data in a way that allows queries expressed in terms of the ontology to be translated to queries in the database. Therefore, we introduced an ontology of spatially located activities playing a central role and connecting the user interface with the data integration management system. The ontology will be discussed in detail in Section 5. Interestingly, the access to data is achieved by introducing another ontology for OpenStreetMap tags, which will be presented in Section 6. The two ontologies are connected via an ontology mapping, which relates the concepts/roles in the ontology of activities with corresponding concepts/roles in the ontology of OSM tags. We will discuss some of the fundamentals of ontology mappings and the means for generating such mappings automatically in Section 7. Moreover, we give a brief intuition on OBDA and its implementation in our tool in Section 8. Finally, the results of the queries are displayed on a map

using OpenStreetMap layers: each location of a certain activity is marked with a distinctive icon. This is realized dynamically in the sense that the markers are only introduced for locations within the current view.

The architecture of our approach is depicted in Fig. 3.

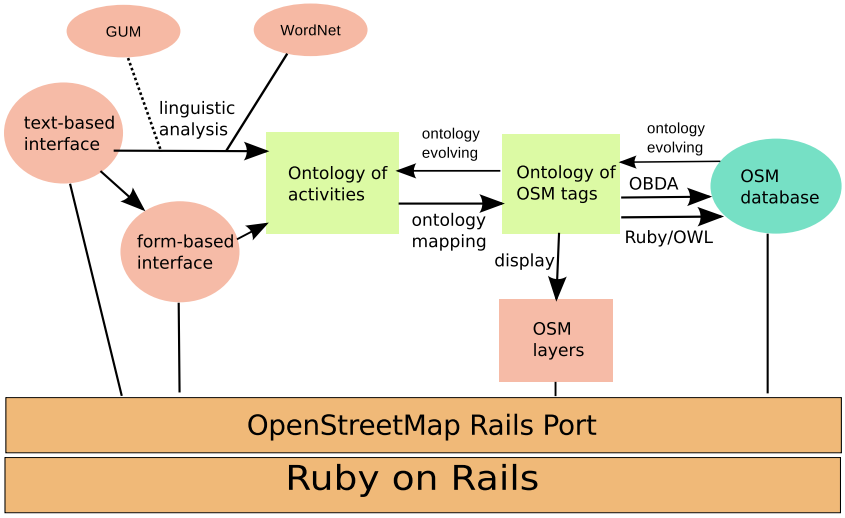

Fig. 3. Architecture of DO-ROAM's activity-based search

The user interaction is handled via two alternative interfaces, which we will now motivate and describe in some detail. The first one is a simple text-based interaction for free search, similar to the one existing in tools like Google Maps or OpenStreetMap, while the second provides a better structuring of the query with the help of a Web form. While the first interface seems more intuitive, the second has the advantage that it is easier to relate with the concepts in the ontology of activities. In the case of the former, the text input by the user needs to undergo a process of linguistic analysis which extracts from the query the concepts which are matched.

For the linguistic analysis we currently use WordNet [5] synsets. That is, the user need not exactly match the concept names of the activities ontology with his query, but can also enter synonyms. For example, take "eating place". If you enter "eating place New York" into Google maps, you only get a few restaurants, however, if you enter "restaurant New York", you get plenty of restaurants. With our connection to WordNet, we get the same set of restaurants for both queries since WordNet knows that "eating place" and "restaurant" are synonyms. Another example would be "clothing": when typing in "clothing London" into Google Maps, you get plenty of results; however, when entering

"dress" or "vesture", Google Maps delivers very few outputs; in case of "vesture", not even one. WordNet in contrast knows all words as synonyms of clothing.

The linguistic analysis also needs to divide the user's input into class names (from the activities ontology), address parts, and time information, and it even needs to infer role names in some cases (like the case with restaurants having a cuisine of a given nationality). This is currently done with a simple matching in a comma-separated list, then the synonyms of the activity are obtained from WordNet and used to generate a list of queries. Addresses are resolved using the search engines of OSM, e.g. Nominatim[9]. In the future, in particular in connection with way finding, we will also use the linguistic ontology GUM [3], since it provides a more detailed semantics for linguistic spatial expressions.

We will now illustrate the way the user interacts with the tool with a stepwise description for the case of Scenario 1 in Section 2 . First, Alan finds the city by using the free search text field. Afterwards, he chooses "Charging Station" in the ontology navigation bar. He can then select one charging station of his choice and zoom in to the desired scale (which is indicated using the functionality of the OSM Rails Port in the bottom-left of the map), obtaining thus an estimate of the area reachable by foot. Then he can get displayed markers for the locations of the free time activities or even all possible activities using again the navigation bar. The results can be seen in Fig. 4. There is one charging station, marked with a plug, and various activities marked with different icons.

In Scenario 2, Betty searches for her address using the free search text field, then she selects in the ontology navigation bar "Gastronomy/Restaurants" and enters the current time and two hours for duration. The results of this query are shown in Fig. 5. There are several restaurants, a café (at the lower right corner) and some fastfoods.

4 Ontologies and the OWL language

Ontologies are formal descriptions of the concepts in a certain domain of discourse and can be informally understood as fixing a meaning for the terms of a particular field. Ontologies are used in artificial intelligence, the semantic web, systems engineering, software engineering, biomedical informatics, library science, enterprise bookmarking, and information architecture as a form of knowledge representation about the world or some part of it. Domain ontologies are typically formulated in the web ontology language OWL[10]. The relation of our domain ontology introduced in the next section to a suitable foundational ontology (typically formulated in a richer logical language) is left for future work.

Formally, an OWL ontology signature consists of sets of *atomic concepts*, *roles* and *individuals*, which fix the vocabulary. Sentences that can be expressed are of two types: TBox sentences are subsumption relations between concepts which are defined inductively from atomic concepts using the universal concept,

[9] `http://wiki.openstreetmap.org/wiki/Nominatim`
[10] `http://www.w3.org/TR/owl2-overview/`

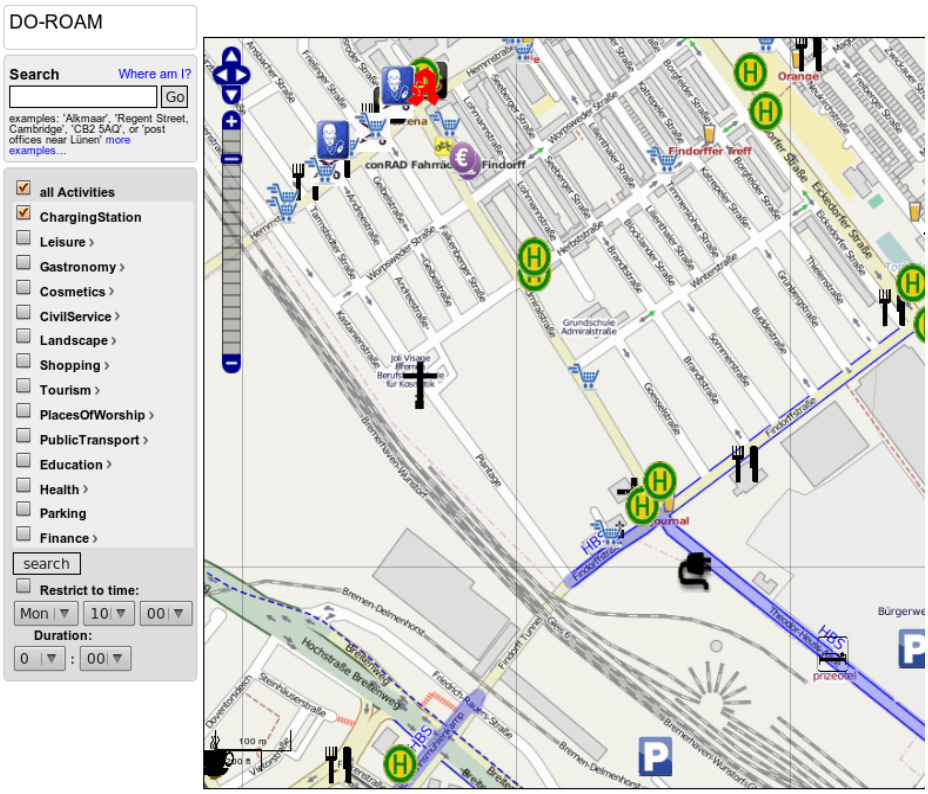

Fig. 4. User interface of DO-ROAM prototype: looking for activities near charging stations.

the empty concept, unions, disjunctions, negations and universal and existential quantification over roles. ABox sentences contain assertions saying that certain individuals belong to certain complex concepts expressible in the vocabulary. Since the ontologies we use here do not contain individuals, we will concentrate on presenting TBox sentences.

Several syntaxes have been designed for ontology languages; in this paper we prefer to use Manchester OWL syntax [7] which provides, for the fragment corresponding to the description logic $\mathcal{ALC}$, the following grammar for concepts:

$$C ::= A \mid Thing \mid Nothing \mid C\ and\ C \mid C\ or\ C \mid not\ C \mid R\ some\ C \mid R\ all\ C$$

where R is a role and A is an atomic concept.

The semantics is set-theoretical: an interpretation I consists of a non-empty set W (the universe) and an interpretation function $.^I$ which assigns a subset of the universe to each atomic concept, a binary relation to each role and an element of the universe to each individual. The interpretation extends from atomic

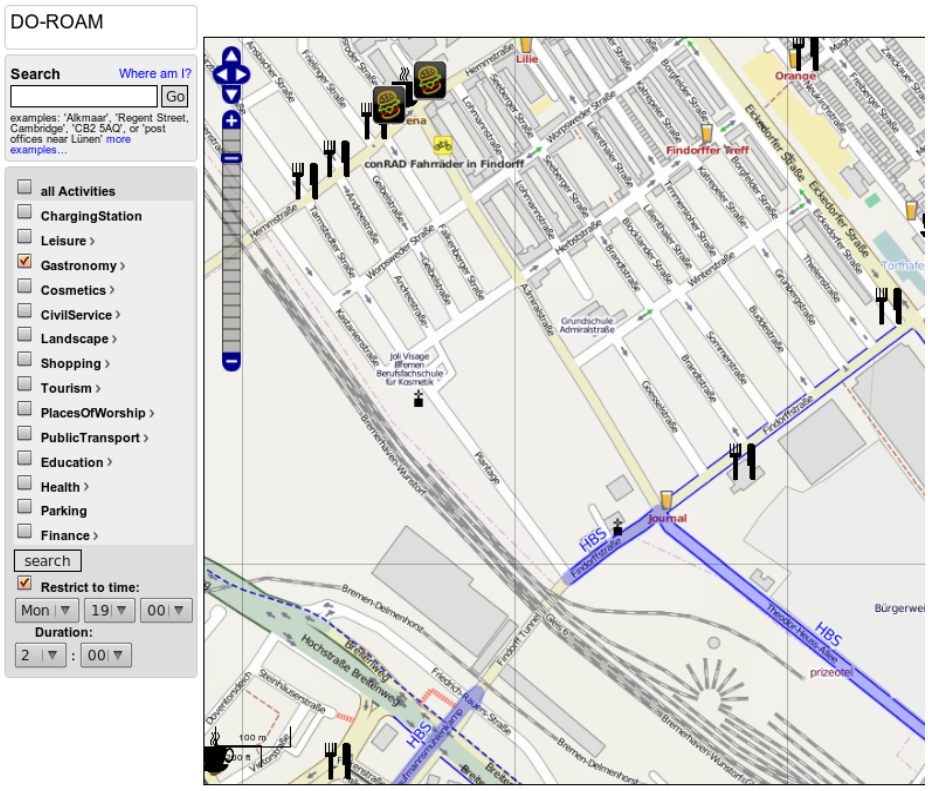

Fig. 5. User interface of DO-ROAM prototype: looking for restaurants open in the next two hours.

concepts to complex concepts in the expected set-theoretic way following the grammar, more precisely: the top concept *Thing* is interpreted as the universe W, *Nothing* as the empty set (bottom concept), a *conjunction* C *and* D by the intersection of the interpretations for C and D, a *disjunction* C *or* D by the union of the interpretations, *not* C by set-theoretic complement, and finally *universal* $(R\ all\ C)$ and *existential* $(R\ some\ C)$ *role restrictions* as follows:

$$(R\ all\ C)^I = \{x \in W \mid \forall y \in W\ .\ R^I(x,y) \text{ implies } y \in C^I\}$$

and

$$(R\ some\ C)^I = \{x \in W \mid \exists y \in W\ .\ R^I(x,y) \text{ and } y \in C^I\}$$

Two ontologies can be related by an *ontology mapping*, sending atomic concepts, roles and individuals of the source ontology to (not necessarily atomic) concepts, roles and individuals of the target ontology. Among many other applications, ontology mappings are important for extracting modules from large ontologies.

5 An Ontology of Spatially Located Activities

Since the scenarios presented in Section 2 are centred on activities, we develop an ontology of *spatially located activities*, which means that the concepts of the ontology refer to locations where a certain activity takes place. This provides an abstraction level from the representation of the data in the databases and thus the user can express queries using a vocabulary closer to natural language. Notice that from an ontological perspective, the ontology developed is a *task ontology*: here the main motivation is not to create and specify a model of a domain, but to solve a well-defined task, namely, searching locations.

We found some interesting guidelines for building ontologies in [11]. In particular, they say

> We advise only creating hierarchies when necessary for describing the domain [...]. The modeller should consider whether an alternative relationship can be used instead. [11]

This means that instead of subclass relations, sometimes it is more useful to use roles. For example, instead of turning "French" into subclass of "Cuisine", it is better to introduce a role "hasNationality" between Cuisine and Nationality.

The ontology has been designed in several steps. The initial design was based on common sense reasoning about the domain, incorporating ideas from the spatial ontology of UbisWorld[11] (the SpatialPurpose concepts) and the taxonomy of medical specialisations from the Bremen city portal[12]. Moreover, the choice of names for the classes of the ontology was inspired by the OSM tags; this helps in generating the ontology mapping automatically.

One main concept in the ontology is *Activity*, which implicitly means a location where an activity takes place. The selected activities are dependent on the particular scenario chosen for the application. We have currently concentrated on daily life activities and tourism. Notice however that the approach is flexible and the ontology can be easily adapted if an alternative scenario is chosen (e.g. business activities). Related activities are grouped and form subclasses of a certain activity type; this provides the advantage that the search can be done in such a way that all found locations can be displayed with a single search.

Locations can have associated addresses or opening hours. We decided to model these as abstract concepts *OpeningHours* and *Address*, and to make the analysis of the query at the string level. In the case of the opening hours, for example, they are usually stored in the OSM database as strings containing assignments of time intervals to the days of the week. The query given by the user will also be retrieved as a set of time intervals and the test whether a certain location satisfies a certain time restriction is done using Allen's interval calculus [2], in particular, using the relations "contains" and "overlaps" between intervals.

To illustrate the ontology design, we present in the following the Restaurant concept.

[11] `http://ubisworld.ai.cs.uni-sb.de/index.php`
[12] `http://www.bremen.de/gesundheit\_und\_soziales/aerztesuche`

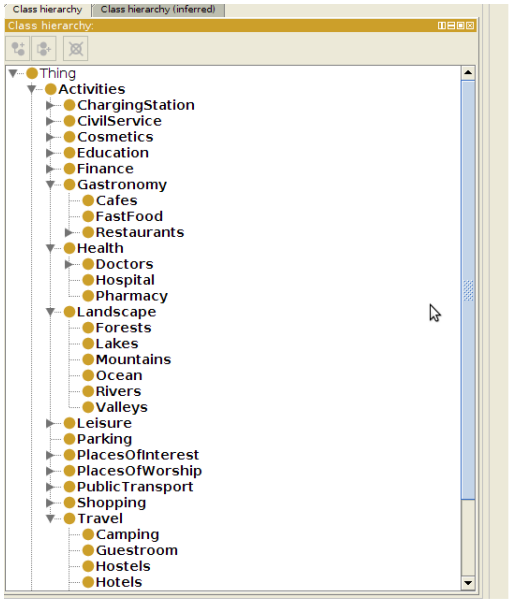

Fig. 6. Ontology of activities.

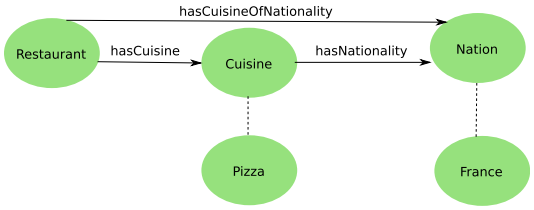

Fig. 7. Restaurants with cuisine.

The restaurants can be categorized according to their cuisine (see Fig. 7, where concepts are represented as discs, roles as arrows and subconcepts as dotted lines). The cuisine is either a food speciality, like pizza or seafood, or nation-specific, say French or Italian specialities. We model this by introducing concepts Restaurant, Cuisine and Nation, together with corresponding roles: a restaurant can have a cuisine and a cuisine can have a nationality. For cuisines and nationalities, we introduce the corresponding subconcepts. Strictly speaking, France should be an individual of the class Nation, but for keeping the symmetry with the ontology of tags, we prefer to introduce it as a singleton class. Notice that French restaurants could be expressed in OWL as the concept

```
Restaurant and hasCuisine some (hasNationality some France)
```

However, a conceptual design problem in the structure of tags in OpenStreetMap requires the presence of the role $hasCuisineOfNationality$ as com-

position of *hasCuisine* and *hasNationality* and thus French restaurants are equally represented as

```
Restaurant and hasCuisineOfNationality some France
```

The entire ontology has been subject to an evolving process. As mentioned, the ontology of spatially located activities will be related to an ontology of OpenStreetMap tags. Some tags may refer to activities that were not taken into account when designing the initial ontology. Of course, the challenge is to add such new concepts to the ontology of activities in an automatic manner. We will address this in the next section. Moreover, other data sources could be plugged in, like e.g. Google Maps. The situation repeats: the new database may contain data that was previously not considered to relate to interesting activities, but has since become relevant in the new context. It is then sensible to add a corresponding concept also at the abstract level, namely the ontology of activities.

6 An Ontology of OpenStreetMap Tags

OpenStreetMap's internal files are lists of nodes, ways and relations, which can be *tagged* with information about the map element. The convention is that any user is free to introduce his own tag, but it is recommended to use existing tags and only have new ones if they are not already covered by the existing ones. The tags of the map elements are represented as pairs $(key, value)$ and an element of the map may have multiple tags (see Figure 8 for the example of a OSM node with its tags in an XML representation. This format has been developed by the OSM community. The listed tags vary from node to node).

The purpose of the ontology of tags was to stay as close as possible to the structure of the OSM files in order to facilitate database querying. This means that we do not try to correct any possible conceptual mistakes in the taxonomy of OSM tags, but rather have it reflected faithfully in the structure of the ontology.

When designing the ontology, it makes sense to decompose the tags into a hierarchy according to the keys: the key becomes a superconcept of its values. We have followed this approach whenever the value was an OSM constant rather then a string/numeral. Since it is possible that a key and a value have the same name whilst the names of the concepts are required to be unique in OWL (OSM has "station" as value of the key "railway" but also a key named "station"), we decided to prefix all keys with "k_" and all values with "v_", e.g.:

```
k = "amenity" v = "charging_station"
```

would introduce a concept "k_amenity" with a subconcept "v_charging_station". Moreover, another problem is that some values are subclasses of more than one key. E.g. "v_no" is a subclass of "k_smoking" but also of "k_smoking_outside".[13]

[13] We maintain our design uniform, so "v_no" must be a concept; other choices would also be available.

```
<node id="834034642"
   lat="53.0871310" lon="8.8091071"
   version="7" changeset="6027662"
   user="Kerridge" uid="324245"
   timestamp="2010-10-13T09:51:39Z">
    <tag k="addr:city" v="Bremen" />
    <tag k="addr:country" v="DE" />
    <tag k="addr:housenumber" v="20" />
    <tag k="addr:postcode" v="28215" />
    <tag k="addr:street" v="Theodor-Heuss-Allee" />
    <tag k="amenity" v="charging_station" />
    <tag k="name" v="Elektrotankstelle swb" />
    <tag k="note" v="telephone reservation necessary" />
    <tag k="opening_hours" v="Mo-Fr  6:00-18:00; Sa off; Su off" />
    <tag k="operator" v="swb" />
    <tag k="phone" v="+49 421 3593186" />
  </node>
```

Fig. 8. A node in an OSM file

In this case, we extended the value to "v_smoking_k_no". Another design decision was to take into account tag dependencies. For example, when a node is tagged with `k = "amenity" v = "restaurant"` it is possible (but not mandatory) that the cuisine is also tagged: `k = "cuisine" v = "seafood"`. In such cases, we introduce a role *hasCuisine* with "v_restaurant" in domain (it is also possible that "v_fast_food" is tagged with "k_cuisine") and range "k_cuisine" in order to be able to select only those restaurants with a certain cuisine.

Recalling the example of French restaurants of the previous section, notice that nation specific cuisines are added directly as subconcepts of "k_cuisine"—see Fig. 10. This is conceptually a mistake in the design of OpenStreetMap's tags, which we here reflect in the ontology which is meant to be very close to the OSM tags.

In order to create a realistic ontology of OSM tags, one faces the problem of an open project where everyone is allowed to contribute—which is also an OSM strength. This has the effect that the data source can be regarded as dynamic, not only at the level of entries, but also at the level of tags. In the OSM wiki page[14], there exists a list of tags, but this list does not reflect the status quo of the actual OSM databases. Some tags which are in the wiki page are not yet tagged by the community, some tags which were abolished through discussion in the wiki or the mailing lists are still used by the mappers. Therefore, the wiki provides only an overview of the available tags; to have a more realistic estimation, one should use websites like Taginfo[15], where the OSM data of the whole world is searched and a list of tags in use, sorted by the number of occurrences, is provided as a

[14] `http://wiki.openstreetmap.org/wiki/Map\_features`
[15] `http://taginfo.openstreetmap.de/`

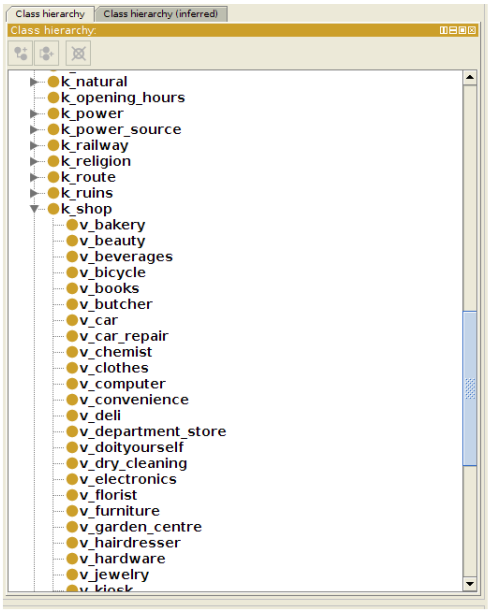

Fig. 9. Ontology of OSM tags.

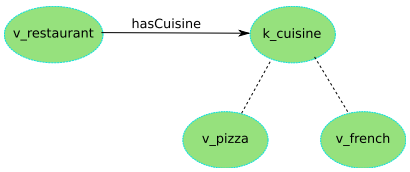

Fig. 10. Restaurants with cuisine.

result. Of course, this list will also contain spelling errors or falsely used tags. The most straightforward solution here is to consider relevant those tags that have a certain, high occurrence in the database, using the list provided by Taginfo. This strategy could result in a limitation using a certain percentage (e.g., all values with, say, more than 0.3 % occurrence rate for the respective key are included), but this approach fails to capture all interesting values in the cases where some keys appear with a far higher occurrence and thus the percentage of important values is low. Also, some keys have far more values (e.g., amenity with 7714 values in use according to Taginfo) than others (e.g., smoking with 22 values), so that the percentage of each value naturally is quite low, which is another point against a certain percentage as a limit for inclusion. This is why a limitation based on the absolute occurrence of a value makes more sense. In our case, we decided to select all values which occur more than 100 times in the database. Spelling errors are thus excluded as well (there is never 100 times the

same mistake), and still all relevant values will be in the database. Theoretically, this threshold could be exceeded by mistakes created during automatic tagging procedures. In reality, there is no evidence in Taginfo that this is the case. It is either prevented by the professionalism of those using automatic tagging, or mistakes of such quantity are quickly noticed by the community and repaired. After this procedure, we added the tags that are in the wiki but not covered through our search of Taginfo. This guarantees that we also include tags which are not yet used by the mappers, but in the future shall be implemented or will replace other tags. To keep this ontology of tags up-to-date, one option would be to make it available to the OSM community. People creating new tags could include them themselves into the ontology as well. Another option is automation, e.g., programs searching regularly through Taginfo for new tags.

7 Mapping the Ontologies

The connection between the two ontologies is bi-directional. In one direction, we map atomic concepts and roles from the ontology of activities to concepts and roles in the ontology of tags. This is the first step towards assigning the elements of the ontology of activities to queries over the database, and will be completed in the next section.

We allow the possibility that the ontology of activities contains concepts which are not related with OpenStreetMap tags. The reason for this is that it is possible to extend and complement the tool with a similar construction for other geographical database systems—indeed, the integration with Google Maps is currently in progress, and this means that some activities with no counterpart in OpenStreetMap may still be found using another database. Thus, the mapping we obtain is partial, and we can see this as having a sub-ontology of activities which is then mapped totally to the ontology of OSM tags.

Since the number of concepts and roles is quite large, providing such a mapping manually would be a very tedious process. We can, however, use an ontology matching tool to obtain a list of pairs of concepts that are in correspondence. This approach is very effective—with the ontology matcher Falcon [8], the degree of automation reaches 80%. This means that the user is still required to verify and confirm the matches produced with the tool, and possibly introduce new matchings between concepts that were not identified by the tool's analysis.

In the other direction concerning the connection of the ontologies, the evolving process for the ontology of activities has to be considered. The social character of OSM makes the tags subject to continuous change: new tags are added frequently and they are often modifying. An example particularly relevant for the topic of the paper is the tag for charging stations for electric cars: initially, a charging station was tagged as

```
k="amenity" v="fuel"
```

like any fuel station and with a supplementary tag

```
k="fuel:electricity" v="yes"
```

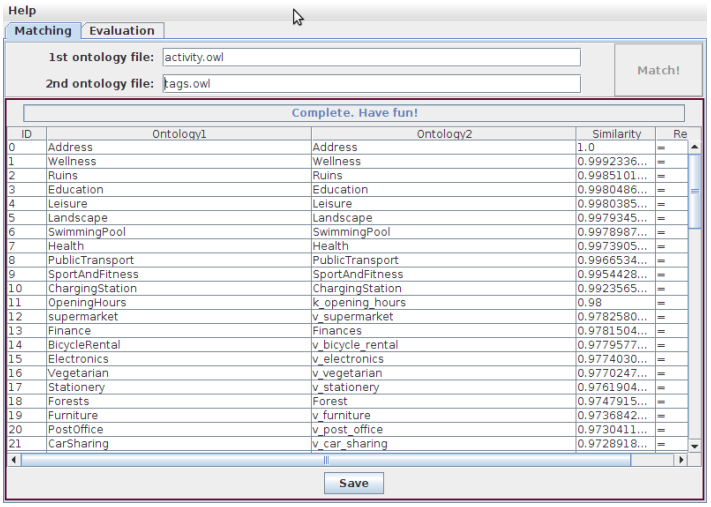

Fig. 11. Matching ontologies with Falcon.

This has later evolved into introducing a distinguished value for amenity:

`k="amenity" v="charging_station"`

but the two ways of tagging charging station still coexist. The dynamic character of the OSM database should be reflected in our tool as well. As more locations on a map are being tagged, it is reasonable to expect that tags that did not pass the criteria for being selected in the ontology of tags now become relevant and should be therefore included. We make use of this process to make the ontology of activities evolve as well: if a certain tag denotes an activity (this could be verified semi-automatically by inspecting its superconcept), it can be added to the ontology of activities.

7.1 The Heterogeneous Tool Set

The Heterogeneous Tool Set Hets [12] is a heterogeneous specification and proof management tool, providing support for a multitude of logics (including OWL in Manchester syntax) and interfacing various logic specific tools like theorem provers, consistency checkers etc. It relies on a heterogeneous specification language with an origin in CASL [13], a specification language developed within the IFIP working group 1.3 "Foundations of System Specifications", and is a de facto standard in the area of software specification. This is particularly relevant for ontology specification as this language can be employed for providing support for modularisation and structuring (see [**?**] for a detailed analysis). Also, when discharging a proof obligation in the system, if proof support is not directly available, a prover can be used by "borrowing" from another logic along a suitable translation of logics.

In the context of our application, HETS can be employed to verify semantic correctness of the ontology mapping produced by a matching. This means that the sentences of the sub-ontology of activities identified by the matching tool should translate along the mapping to logical consequences of the ontology of tags. The corresponding HETS specification can be obtained in an automatic manner: consider that ACTIVITIES is a named specification containing the initial ontology of activities and TAGS contains the ontology of tags.[16] The matching procedure returns a list of pairs having component concepts in the ontology of activities which have a corresponding match in the ontology of tags. This fits in exactly with the HETS syntax: ontology mappings are written as *symbol maps*, i.e., for a symbol of the source ontology, one must give its corresponding symbol along the mapping. Also, the source and target ontologies must be explicitly given; while the target specification is simply TAGS, the source specification is obtained from ACTIVITIES by *revealing* the symbols in the sub-ontology, an operation which hides the remaining ones. This finally gives a complete HETS specification of the ontology mapping and the tool can be used for verifying correctness.

The correctness of the mapping can be verified using HETS and the provers Pellet[17], Fact++[18] or also first-order provers like SPASS [19]. A Protégé [19] plugin for manipulating HETS-OWL specifications is under development.[20] In the current state of the ontologies, discharging the proof obligations is relatively simple. The added value of using a formal verification method for the view becomes visible in the presence of subsumptions implying more complex terms. Since such terms could be introduced by changes in the database or usage of another data source, we preferred to include this step as part of the tool methodology.

8 Ontology-Based Data Access

Ontology-based data access is a data integration methodology which separates the 'knowledge' about data from reasoning about it. This is achieved by providing an abstract representation of the application domain with the help of an ontology, a schema of the sources where the real data is stored, together with a mapping between the elements of the ontology and those of the data schema. Typically, the schema of the data is assumed to be a relational database schema, and the mapping provides a query in the database for each concept and each role of the ontology. The advantage of this approach is that we can use the knowledge base constituted by the TBox and the ABox sentences of the ontology to derive information about the data which is not present in the database, using query rewriting.

The data integration management component of our system follows the principles of OBDA: the domain of interest—spatially located activities—is modelled

[16] Notice that here ontologies are regarded as logical theories in the OWL logic.
[17] `http://clarkparsia.com/pellet/`
[18] `http://owl.man.ac.uk/factplusplus/`
[19] `http://protege.stanford.edu/`
[20] `https://github.com/pyneo/protege-hets`

as an ontology, the OpenStreetMap data is stored in a database, and the concepts of the ontology are related to queries in the database.

For the representation of and access to ontologies within the Ruby on Rails framework, we have developed a new library, Rails-OWL. Since OWL is represented in XML, our library is based on the existing library REXML[21] for reading in XML documents. Rails-OWL represents an OWL ontology in the Rails database. This allows programmers to easily and flexibly access ontologies in a way similar to the access of the geodata. Fig. 12 gives an overview of the different classes (which by the ActiveRecord framework of Rails simultaneously are database tables) used for representing ontologies, their classes and mappings between these. A "simple subclass relation" is one between named concepts, while in general, it can be postulated between arbitrary OWL class terms.

database table	represented contents
Ontology	ontologies
OntologyClass	classes (of various ontologies)
OntologySubclass	simple subclass relations
OntologyClassProperty	subclass (and other) relations
OntologyRole	roles (of various ontologies)
OntologyRoleProperty	role relations
OntologyMapping	ontology mappings
OntologyMappingElement	mapped pairs (of various mappings)

Fig. 12. Classes for representing OWL ontologies in Ruby on Rails

In ontology-based data access, usually, one SQL query per ontology class is designed manually, and this is used for the database interpretation of ontology terms, implemented by query rewriting. In case of OpenStreetMap, we would need to design dozens of such SQL queries, which is a tedious process. Instead, we use the OSM tag ontology, which is tailored towards the OSM database in such a way that the relation between classes in the OSM tag ontology and the OSM database is generic: since the basic classes directly correspond to keys and values of OSM tags, the corresponding SQL queries are simple, and this is then used for query rewriting of more complex class terms. This query rewriting is implemented in Rails-OWL easily, because classes, roles and such are first-class citizens. The involved OSM tables are shown in Fig. 13.

9 Conclusion and Future Work

We have presented an ontology-based tool prototype for searching locations for specific activities in OpenStreetMap. We have here concentrated on presenting the architecture and the general ideas underlying our tool, and could only sketch

[21] `http://www.germane-software.com/software/rexml/`

database table	represented contents
Node	geographical location with coordinates
NodeTag	tags for nodes with key and values
Way	polyline
WayNode	incidence relation between nodes and ways
WayTag	tags for way with key and values

Fig. 13. Classes used by OpenStreetMap for representing Geodata in Ruby on Rails

some of the technical details. The focus is not primarily on locations as such, but rather on (located) activities. Here, the central ontology of spatially located activities is subject to evolution due to the continuous development of the OpenStreetMap databases, and we have introduced an intermediate ontology of data source representation terms to facilitate querying.

Future work will enhance the ontology-based querying with a more sophisticated ontology navigation and query refinement, see [6] for an overview of existing approaches. Also, the searching of activities will be complemented with an activity-oriented route planning (such that Scenarios 3 and 4 in Section 2 will be supported by our tool as well). Our main topic of interest is electric mobility which requires special route-finding algorithms that take into account the energy consumption and the (projected) battery status at a given destination point. We also intend to complement this with a radius visualisation of the area within battery reach.

Acknowledgments

This work has been supported by the German Research Foundation (DFG) and Project I1-[OntoSpace] of the SFB/TR 8 "Spatial Cognition". We thank C. Maria Keet for proposing the name DO-ROAM and John Bateman and the anonymous referees for useful feedback. Moreover, Erwin R. Catesbeiana pointed out user queries that lead to no results. We also thank Shailen Sobhee and Shishir Gautam for implementation work.

References

1. Neeharika Adabala and Kentaro Toyama. Purpose-driven navigation. In Rodríguez et al. [17], pages 227–233.
2. James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26:832–843, 1983.
3. John A. Bateman, Joana Hois, Robert J. Ross, and Thora Tenbrink. A linguistic ontology of space for natural language processing. *Artificial Intelligence*, 174(14):1027–1071, September 2010.
4. Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. The MASTRO system for ontology-based data access. *Semantic Web Journal*, 2011. Forthcoming.

5. Christiane Fellbaum, editor. *WordNet: An electronic lexical database*. The MIT Press, 1998.
6. Hanh Huu Hoang and A Min Tjoa. The state of the art of ontology-based query systems: A comparison of existing approaches. In *In Proc. of ICOCI06*, 2006.
7. M. Horridge, N. Drummond, J. Goodwin, A. Rector, R. Stevens, and H. Wang. The Manchester OWL Syntax. In *OWL: Experiences and Directions*, 2006.
8. Wei Hu and Yuzhong Qu. Falcon-AO: A practical ontology matching system. In *Proc. of WWW-07*, pages 237–239, 2008.
9. Sebastian Hübner, Rainer Spittel, Ubbo Visser, and Thomas J. Vögele. Ontology-based search for interactive digital maps. *IEEE Intelligent Systems*, 19(3):80–86, 2004.
10. Werner Kuhn. Ontologies in support of activities in geographical space. *International Journal of Geographical Information Science*, 15(7):613–631, 2001.
11. Hayley Mizen, Catherine Dolbear, and Glen Hart. Ontology ontogeny: Understanding how an ontology is created and developed. In Rodríguez et al. [17], pages 15–29.
12. Till Mossakowski, Christian Maeder, and Klaus Lüttich. The Heterogeneous Tool Set. In Orna Grumberg and Michael Huth, editors, *TACAS 2007*, volume 4424 of *Lecture Notes in Computer Science*, pages 519–522. Springer-Verlag Heidelberg, 2007.
13. Peter D. Mosses. *CASL Reference Manual, The Complete Documentation of the Common Algebraic Specification Language*, volume 2960 of *Lecture Notes in Computer Science*. Springer, 2004.
14. Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. *J. on Data Semantics*, X:133–173, 2008.
15. Antonella Poggi, Mariano Rodriguez-Muro, and Marco Ruzzi. Ontology-based database access with DIG-Mastro and the OBDA Plugin for Protégé. In Patel-Schneider, editor, *Proc. of the 4th Int. Workshop on OWL: Experiences and Directions (OWLED 2008 DC)*, volume 496. CEUR-WS, 2008.
16. Martin Raubal and Werner Kuhn. Ontology-based task simulation. *Spatial Cognition & Computation*, 4(1):15–37, 2004.
17. M. Andrea Rodríguez, Isabel F. Cruz, Max J. Egenhofer, and Sergei Levashkin, editors. *GeoSpatial Semantics, First International Conference, GeoS 2005, Mexico City, Mexico, November 29-30, 2005, Proceedings*, volume 3799 of *Lecture Notes in Computer Science*. Springer, 2005.
18. Inessa Seifert. Region-based model of tour planning applied to interactive tour generation. In Julie A. Jacko, editor, *HCI (3)*, volume 4552 of *Lecture Notes in Computer Science*, pages 499–507. Springer, 2007.
19. C. Weidenbach, U. Brahm, T. Hillenbrand, E. Keen, C. Theobalt, and D. Topic. SPASS version 2.0. In Andrei Voronkov, editor, *Automated Deduction – CADE-18*, volume 2392 of *Lecture Notes in Computer Science*, pages 275–279. Springer-Verlag, July 27-30 2002.