An algorithm to enhance queries over a geo database

Miriam Baglioni[1], Maria Vittoria Masserotti[2], Chiara Renso[2], Laura Spinsanti[3]

[1]KDDLab Computer Science Department – University of Pisa, IT
baglioni at di.unipi.it
[2]KDDLab ISTI, CNR, Pisa, IT
masserotti, renso at isti.cnr.it
[3]LBD, EPFL, CH
laura.spinsanti at epfl.ch

Abstract. Geospatial semantic querying to spatial databases has been recognized as a hot topic in GIS research, although no standardized approaches have been proposed so far. However, a common solution is to adopt an ontology as a knowledge representation structure on top of a spatial database to support user queries. In this context, we propose an approach to build an ontology which, not only represents the concepts stored in the database and their semantic abstractions, but it is also capable of managing the defined specializations of such concepts. The methodology introduced in this paper aims at building this richer ontology and the associated materialized views to handle the semantic query translation.

Keywords: geodatabase, semantic ontology enrichment, spatial queries

1 Introduction

The exponential growth of positioning technologies, ranging from satellite images to GPS personal devices and Wi-Fi connections, tend to produce a huge amount of geographical referred data. This, in turn, calls for methods and techniques capable of managing and querying this large quantity of data. While data management recent developments in spatial and spatio-temporal databases converge towards some well defined proposals [OraSpa, PostGIS, Guting05], the query capabilities in terms of semantically enhanced user query language have not produced so far any standardized approach.

In the last few years, ontologies have gained increasing interest in the GIS community [Mark06, Fonseca02, Spaccapietra04], because they are essential to create and use data standards as well as human computer interfaces and to solve heterogeneity/interoperation problems. The relevant literature exploits ontologies [Gru08] to map data sources with explicit ontology concepts [Bishr08]. The use of ontologies as a middle layer between the user and the database adds a conceptual level over the data and allows the user to query the system on semantic concepts without having any specific information about the database at hand [Peuquet02]. This ontology should be capable to represent both high level semantic concepts and concepts that have a correspondence to database tables. This allows us to build a mapping between ontological concepts and data. Such an ontology can be constructed

automatically (or semi-automatically) from a database schema and a representation of domain knowledge. Indeed, the role of the ontology is to formalize knowledge about a domain of interest in terms of concepts and relationships between them. Moreover, ontology formalisms typically support the implicit definitions of concepts as logical formulas. The aim of the current work is to exploit these concept definitions to further enrich the semantics of the underlying geographical database.

Our long term goal is to develop a semantic geospatial natural language interface to spatial databases. A first step in this direction has been the definition of a mapping module, that, supported by a geospatial ontology, translates a natural language semantic geospatial user question to an appropriate spatial SQL query on the database ([Baglioni07, Baglioni08]). In this context, the use of ontologies has already been experimented (see for example [Viegas06]). Indeed, having a conceptual and taxonomical representation of spatial data provides the query system with a semantic representation of spatial concepts. This enables the user to pose "semantic geospatial queries" instead of the classical "geospatial queries" provided by the query language of the DBMS.

The contribution of the present work aims at exploiting the ontology concept definitions to provide the user query language with a number of concepts not directly represented in the database. These definitions represent specialization of a given concept defined as a logical combination of other ontology concepts and their associated relationships. Hence, the semantic of the underlying database is enhanced by explicitly representing information which is implicit in the data. The basic idea is to map these ontology definitions into appropriate materialized views which can be consequently queried by users.

This paper is organized as follows. Section 2 presents the related work, Section 3 shows the overview of the approach. Then, Section 4 regards the enhancing process: starting from a general description comprehending some example scenario, continuing with the module specification and concluding with the formalized implementation algorithm. Finally, Section 5 draws conclusions and future work.

2 Related Work

The partnership between ontologies and GIS has seen a growing interest in the last decade [Fonseca99, Mark06]. The role of ontologies in geographical information science can be manifold. A well known topic is ontology support to spatial querying, as witnessed for example by the results of the SPIRIT project [Jones05], where methods for ontology-based spatial query expansion for geographical search engines were studied. For example, in this context, [Cardoso07] proposes a method for the geographical expansion of queries exploiting spatial relationships.

A recent approach [Peachavanish07] proposes a methodology that exploits multiple ontologies for the interpretation of geospatial queries. However, they propose a mediation between ontologies that we are not facing, and in general their approach is more conceptual and not based directly on querying database tables. The work in [Torres05] has some similarities with ours, since the authors propose an ontological semantic layer to query a geographical database. In particular, this approach allows different community users to access the same geographic database.

However it focuses on the representation of spatial relationships, such as the topological ones (e.g. *touches*), and does not consider specifically the problem of representing the location of a geographical object, neither the ontology specialization definitions which we are handling here.

Similarly to our approach, the work in [Lüscher 08] aims at enriching the semantics of geodatabase for enhanced user queries. However, the authors propose a complementary approach, compared to our, since they infer semantic information about spatial objects exploiting pattern recognition techniques.

The work of [Zhao08] shares with our approach the use of ontologies as a query interface towards spatial data, but the focus is on data integration and they don't consider the use of a domain ontology to further enhance the geospatial semantics of queries.

The approach proposed in this paper enhances the work presented in [Baglioni07, Baglioni08] where ontologies are exploited to support semantically enriched natural language queries to geo-database. The overall objective of our work is to define a methodology, and an associated architecture, to translate natural language semantic geospatial user queries into the corresponding spatial SQL statements. This allows the user to pose queries based on concepts that may not have an explicit representation in the database.

One of the main contributions of [Baglioni07] was the definition of the semantic enrichment process of a spatial database by means of a geospatial domain ontology which results in adding a conceptual map to the spatial database. In other words, database tables are linked to their correspondent ontology concepts and consequently to the corresponding taxonomy, which represents the semantic abstraction of them. This enables the user to refer to semantic concepts that do not have a direct mapping to database tables, by referring to any of the concepts of the hierarchy. Furthermore, [Baglioni08] introduced a natural language semantic component managing natural language queries and translating them, by means of the enriched ontology, into spatial SQL queries.

3 The Overview of the Approach

Our current work proposes an evolution of [Baglioni07] where the resulting ontology captures also implicit spatial definitions wherever they can be translated into a spatial SQL query on the underlying geographical database.

The new architecture is shown in Figure 1. Here, the basic steps of the automatic ontology extraction from the spatial database are summarized, while introducing a new module, called *Selection&View Module*.

The *Application Ontology* represents the structure of the database and it is built from the database schema via the *Extraction Module* [Baglioni09]. This ontology is then combined (via the *Enriching Module*) with a *Domain Ontology* to produce the

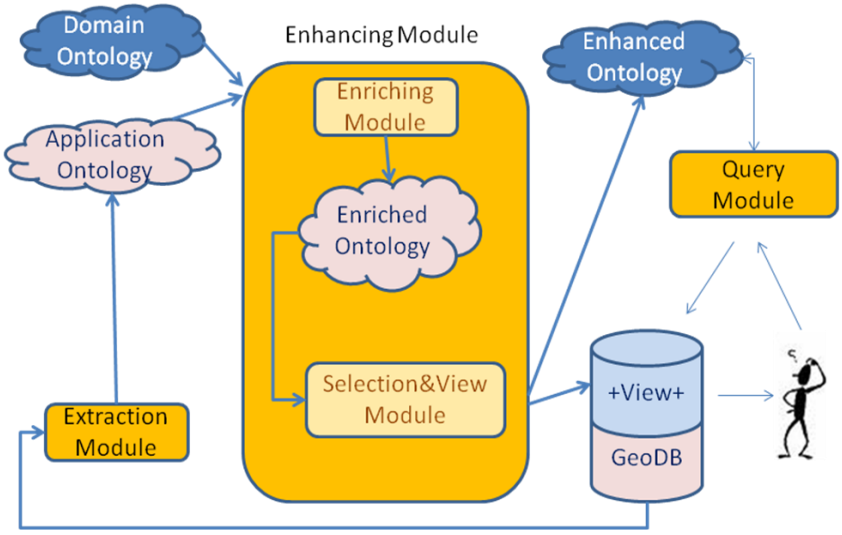

Fig. 1. The proposed architecture

Enriched Ontology. The enrichment process adds to the ontology all the domain concepts which are semantic abstractions of database tables. This enrichment process performs a semantic match of concepts from the Application Ontology with the Domain Ontology. For all the domain classes that matches, the entire upper part of the hierarchy is taken as part of the Enriched Ontology, thus connecting each database concept with a number of semantic abstractions of it. The query component implements a rewriting method to map each concept of the Enriched Ontology to a specific (set of) table(s) in the database. It is worth pointing out that only the leaves of the resulted ontology (corresponding to the Application Ontology) have a direct mapping to database tables.

Now, we propose an additional step which aims at further increasing the semantic richness of the resulting ontology, introducing the *Selection&View Module* which exploits the Domain Ontology specialization definitions, with the aim at extending the Enriched Ontology concepts with a number of sub-concepts that further characterize the concepts themselves. This step produces two outcomes: a number of materialized views are added to the spatial DBMS, and the new *Enhanced Ontology* is built to support the user queries on these views. This resulting ontology acts as a knowledge representation structure which is the basis of the query component, representing all the concepts that the user can mention in the query. In this paper, we focus on the *Selection&View Module,* which analyzes the Domain Ontology to identify implicit classes definitions and, while adding the new concepts to the final Enhanced ontology, it translates them into the appropriate database views to support the user SQL queries on the introduced concepts.

The specialization classes of the ontology are defined by *axioms*, a combination of classes and relations with logical operators, which depend on the specific logic at hand. Here, we base our approach upon the OWL DL [OWL], a W3C standard arisen from research on the Semantic Web based on Description Logics [Baad03]. The kind

of axioms that we consider have a specific form to ensure an easy translation into spatial SQL. Indeed, mapping OWL into SQL is a complex task and is object of current research [Acci05]. Here, we restrict the forms the axioms must have in order to be properly translated, since the only allowed properties are the spatial relationships, which ensure an immediate correspondence to OpenGIS topological relations. Details on the axioms form restrictions are reported in Section 4.

As a motivating example, let us consider the ontology shown in Figure 2. Here, the top-level geospatial ontology is depicted in grey hexagons. The *spatialRelation* property has, as sub-properties, the OpenGIS topological relations *equals*, *disjoint*, *touches*, *within*, *overlaps*, *crosses*, *intersect*, *contains* [OpenGIS]. The grey rectangular boxes indicate the part derived from the Domain Ontology, thus each of these concepts does not have a direct correspondence to database tables and represents a semantic abstraction of data. White continues boxes represent concepts derived from the Application ontology, thus having a one-to-one correspondence to database tables. Eventually, dotted rectangles indicate domain ontology concepts which are defined by axioms, and are not included in the resulting Enriched Ontology. For example, CentralAccomodationPlace is defined as a subclass of the AccomodationPlace class, but it is not included by the Enrichment step since no database table exactly matches with this concept. However, this concept can be easily defined as an axiom stating that each accommodation place located within the historical center is a *central accommodation place*.

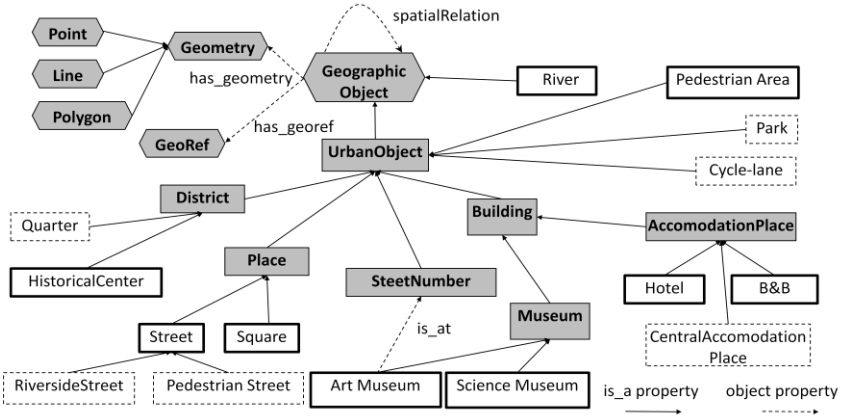

Fig. 2. An example of Enriched Ontology in the urban domain (continue rectangles) with defined concepts (dotted rectangles)

The main contribution of this work is the definition of a method to capture in the enrichment process all axioms defining new concepts in terms of topological relations between others. At the same time, the method creates a database support for these new concepts. Since these new concepts do not have a corresponding database table, but can be obtained by applying a topological operation to database objects, a number of materialized views are created. Thus, the final Enhanced Ontology supports the user queries providing a number of concepts otherwise excluded since they are not

directly represented by database tables and they are not included in the previously defined Enriched ontology.

In the next section we introduce the details of the new semantic enhancing process.

4 The Semantic Enhancing Process

The Enhancing process aims at extending the number of concepts the query language can support, by creating a set of materialized views associated to the Domain Ontology specialization concepts. The general idea is to define new concepts using OWL axioms. For example, back to the AccomodationPlace definition introduced above, the concept CentralAccomodationPlace can be expressed combining AccomodationPlace with HistoricalCenter through the spatial property *within*. As an OWL axiom it will be formalized as

```
CentralAccomodationPlace ≡ AccomodationPlace and
                           (within some HistoricalCenter).
```

where *within* is defined in the ontology as a sub-relation of the *spatialRelation* property between GeographicObjects.

For simplicity, we assume that every table has *the_geom* attribute. To resolve this axiom, that is to create a view containing the accomodation place instances that are located in the center of the town, the concept AccomodationPlace must belong to the Enriched Ontology, thus mapped to at least one table of the database. Note that the concept AccomodationPlace can be directly or indirectly associated to data. There is a direct association between the concept and the data if the concept is a leaf of the Enriched Ontology. There is an indirect association when the concept is a node in the Enriched Ontology. In this latter case, the concept is rewritten in the set of its leaves children, and so associated to each of the tables related to this set. Moreover, HistoricalCenter must belong to the Enriched Ontology or be already associated to a materialized view (thus the result of a previously translated axiom). Note that, also in this case, the concept could have data directly or indirectly associated using the leaves of its corresponding sub-tree. The different cases are best described, by examples, in section 4.1.

Let us now briefly consider how the *Enhancing Module* works. It takes as input the Application Ontology and the Domain Ontology, and returns two different outputs. On the one hand, it returns a set of materialized views associated to some of the concepts in the Domain Ontology that defined specialization of concepts and are not already present in the Enriched Ontology, and, on the other hand, it returns an Enhanced Ontology including the Enriched ontology and extended with the new selected specialization concepts.

The *Enhancing Module* is composed of two sub components: the *Enriching Module* described in details in [Baglioni07], and the *Selection&View Module*, described in this work. The objective of the Selection&View Module is to select the concepts of the Domain Ontology defining spatial specializations and extending the Enriched Ontology consequently, thus producing the new Enhanced Ontology. The concepts that can be used to extend the Enriched Ontology are those defined by an axiom whose defining concepts refer to classes of the Enriched Ontology or classes

referring to newly materialized views. To be pointed out that the only concepts considered in the selection module for the ontology extension, are those directly connected, in the Domain Ontology, to the selected class, namely the direct children and siblings of the selected leaf in the Domain Ontology. The Selection&View module is responsible also for the automatic creation of the views corresponding to the new concepts.

In the enhancing process we distinguish three different cases, depending on which ontology node can be selected for the translation. Let us now describe better the mechanism by explaining each case with an example.

Case A – Sibling

The selected Enriched Ontology (EO) leaf corresponds to a leaf in the Domain Ontology. We select, in the Domain Ontology, all the siblings of the leaf that are defined by an axiom, and are not already present in the EO. This means that the father of the node is already present as a node in the EO, so no table is associated to this concept. The only data available to express this concept are those associated to its children that have a direct mapping to a database table. Then, the spatial operator defining the new concept will be applied only to these siblings to create the materialized view. If the selected node has more than one sibling not present in the EO, then the same mechanism is applied and only the concepts sibling associated to database tables will be considered in the creation of the view (since considering also the views will produce only a duplication of the data).

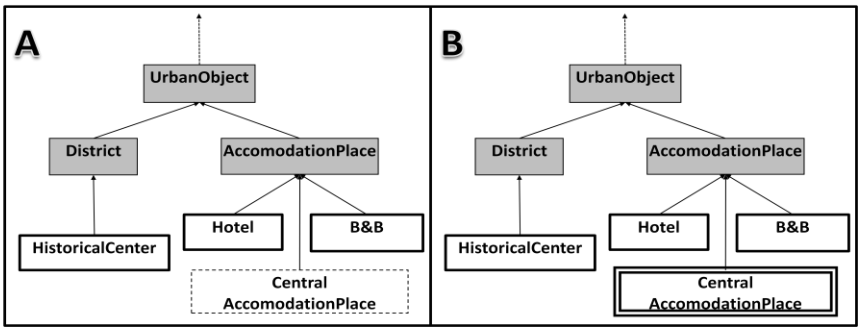

Fig. 3. Extension of the definition of AccomodationPlace, in figure (A) it is represented as an axiom, while in figure (B), after the node selection, it becomes part of the Enhanced ontology

Example 1.

Consider Figure 3, where the CentralAccomodationPlace concept is defined by an axiom. As previously defined, an Accomodation Place located in (geographical coordinates are located *within*) the Historical Center, is a CentralAccomodationPlace. Since AccomodationPlace is a node of the Enriched Ontology, no database tables are directly associated to it, thus it represents the abstract concept of its children classes Hotel and B&B that, in turn, are leaves and therefore they are directly associated to a database table. The axiom translation results is the creation of a materialized view (evidenced in double line in the ontology figure 3-B) with the same name of the

axiom class. The data contained in this view are the union data of the subset of the Hotel instances located in the HistoricalCenter with the subset of the B&B instances located in the HistoricalCenter. It is important to highlight that the resulting view contains *new information*, even if it does not contain really new data. In fact, the semantics of the view data makes explicit the *central* accommodation places.

Note that when the second concept used in the axiom, (in this example HistoricalCenter), is not a leaf, the spatial relation is applied to all the children of its sub-tree.

Case B – children

The selected Enriched Ontology leaf is a node in the Domain Ontology. In this case all the children of the concept in the Domain Ontology that are defined by an axiom are selected. This step produces one more concept child of the selected node, that is associated to all the instances not belonging to any of the children of the concept. We call this new class *Complement* class. This class is needed because the selected concept was a leaf in the Enriched Ontology and therefore associated to a table. Since we aim at specializing the concept, we create partitions of the associated instances, therefore the Complement concept includes all the remaining records of the original table. When the specialization partitions covers all the instances, then the Complement class has no associated data.

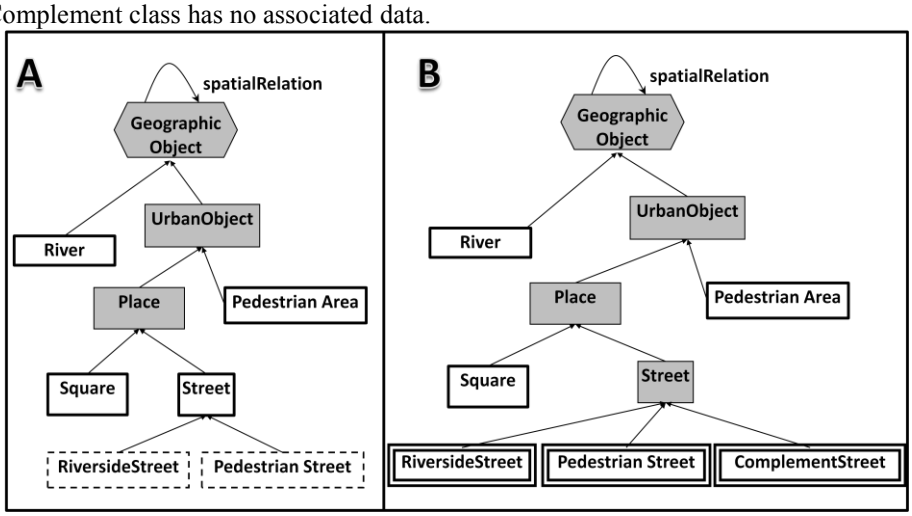

Fig. 4. Specialization of the concept Street. The new concept ComplementStreets is added to the ontology as the complement class.

Example 2

Let us consider the ontology depicted in Figure 4, and let PedestrianStreet be defined as a Street whose geographical coordinates are located *within a* PedestrianArea area. Similarly, suppose RiversideStreet be defined as the streets whose geographical coordinates have a *touch* relation with a river. In this example, the concept Street is a leaf in the Enriched Ontology, therefore a database table is directly associated to it. The axiom translation results is the creation of three

materialized views (in double line in the ontology figure 4-B). These views refer to the subset of Street records contained in a PedestrianArea and the records of Streets located near the River, respectively. A third view, called ComplementStreet is created to collect the records of Street not included in the two views. As a consequence, Street is transformed in a node (in grey in the ontology figure 4-B).

Case C – Merge siblings and children

The selected EO Leaf is a node in the Domain Ontology that has both siblings and children not present in the EO and defined by an axiom. Both the previous steps are applied in this case.

These three cases are managed by the *Selection&View Module*, to select the appropriate node and to build the materialized views respectively.

4.1 Selection&View Module

The *Selection&View Module* is responsible for the selection of the ontology specialization classes that can be solved and consequently for the production of the materialized views associated to these concepts. This process results in the construction of the new Enhanced Ontology which will be the knowledge representation structure underlying the query system. The process starts considering all the leaf concepts in the Enriched Ontology to verify whether they can be exploited for the semantic enhancing step or not. It is worth noticing that, we limit the process to the first level, that is to consider the siblings and the direct children of the selected leaves.
Figure 5 shows a generic ontology schema. Let us assume that the dark circles represent the Enriched Ontology concepts, whereas the white circles represent the implicit concepts. Therefore, the leaves of the Enriched Ontology are {J, L, N, O}. Because of the limitation fixed in the explored levels, the leaf J will cause the selection of the concept I as its sibling. The leaf O will cause the selection of the nodes R and S since both R and S are direct children of O. Finally, the leaf L (which corresponds, in the Domain Ontology, to a node having both siblings and children) will cause the selection of the classes {K,M,P,Q}. The leaf N does not have in the Domain Ontology any direct children, and the only sibling is already part of the Enriched Ontology, so no class will be selected, and the set of the selected nodes is {I, K, M, P, Q, R, S}.

As a second step, the Selection&View module checks if these concepts can be mapped to a spatial SQL query. In fact, let us remind that an axiom can be mapped to SQL only when it refers to a spatial relation and to a concept already included in the ontology. For example, if the concept I is defined in terms of a spatial relation with G, it will be added to the final ontology. On the contrary, if M is defined by an axiom with a spatial relation with B, then it is not added to the ontology since it refers to a concept not already materialized. Suppose now Q is defined by an axiom having a spatial relation with I, then Q can be added to the final ontology only after the view for I is materialized.

This process is recursive, and it stops because the number of considered concepts is finished.

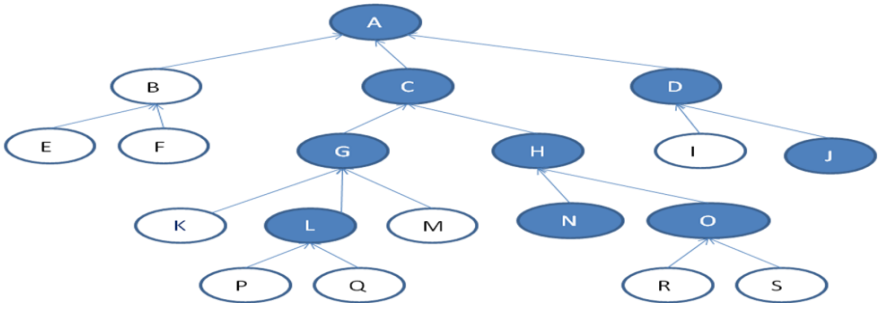

Fig. 5. Example of the extension process of the Selection Module

Let us now consider in more detail the creation of the materialized views. The basic idea is that axioms abstractly define materialized views in the database. Obviously, we are assuming some restrictions on the form the axioms can take. Indeed, the expressive power of OWL DL overcomes the SQL expressive power therefore it is not always possible to map OWL axioms into SQL queries [Acci05]. Here, we are assuming axioms have a basic simple structure that makes the SQL translation simple. It is worth noticing that we are not relying on a specific OWL DL subset, such as OWL-Lite [OWL], since we are not interested in the computational expressiveness of the language. Instead, we use OWL axioms only from a syntactical point of view in order to associate the appropriate SQL query.

Allowed axioms define specialization classes by restricting the *spatialRelation* property (or its subproperties) with the "exists" (also called "some") operator. Indeed, this logical expression may be semantically mapped to a SQL SELECT statement involving the OpenGIS topological relations. We remind that we explicitly refer to a spatial relation which can be immediately one-to-one mapped in a spatial OpenGIS operator. We also recall the spatial relations considered here: *equals*, *disjoint*, *touches*, *within*, *overlaps*, *crosses*, *intersect*, *contains*, modelled in the ontology as subproperties of *spatialRelation*. At this stage, we are not considering more complex axioms, such as logical expressions with union and intersection, neither including the "forall" and the max cardinality operators since their correspondence to SQL statement is not clear and requires more investigation.

Therefore the general form of the axioms is:

$$SpecializationClass \equiv Class_1 \text{ and } (spatialRelation \text{ some } Class_2)$$

Let us consider the examples presented in the previous section. CentralAccomodationPlace has been selected by the selection module and it is defined as a restriction of AccomodationPlace that is within HistoricalCenter. The corresponding axiom is:

$$CentralAccomodationPlace \equiv AccomodationPlace \text{ and } (within \text{ some } HistoricalCenter).$$

Since AccomodationPlace is a node class in the ontology, we expand it into its children concepts, namely Hotel and B&B which are leaves in the ontology, thus obtaining the following SQL view:

```
CREATE VIEW CentralAccomodationPlace as
 (SELECT * FROM Hotel h, HistoricalCenter hc
  WHERE intersect(h.the_geom, hc.the_geom)
                UNION
  SELECT * FROM B&B b, HistoricalCenter hc
  WHERE intersect(b.the_geom, hc.the_geom))
```

Notice the use of the `UNION` operator to include data selected from both tables[1].

Another example is related to the case where a complement class is created. Consider the specialization of the Street class with the two new concept children RiversideStreet and PedestrainStreet (see Example B), expressed by the already cited axioms and corresponding SQL queries. To cover all data on Streets, a complement class has to be created after the materialization of the two views:

```
CREATE VIEW ComplementStreet as
   SELECT * FROM Street
   WHERE not exist (select * from PedestrianStreet)
   AND not exist (select * from RiversideStreet)
```

Once each view is materialized, the Selection&View Module updates the Enriched Ontology with the new selected classes.

4.2 Implementation of the Selection&View Module: the algorithm

In this section we present the details of the *Selection&View Module* algorithm, that first selects the ontology nodes to be expanded and then creates the corresponding SQL views.

The algorithm is structured in two main steps. The first step consists in selecting all the leaf concepts of the Ontology mapped to database tables (so the DBOntology Leaves) and to expand them, as much as it is possible, with input domain Ontology (called Domain Ontology) concepts defined as axioms, following the rules presented previously. The algorithm selects all the Domain Ontology concepts that are direct children or siblings, if they are not already included in the DBOntology, of the selected DBOntology leaf. The second step consists in filtering, from the set of concepts previously selected, only those whose axiom can be solved by an SQL query. This means that we must be sure that all the concepts present in the axiom belongs to the DBOntology that is ase associated to batabse tables or have been already materialized as views. In this case the algorithm creates the corresponding SQL view, and updates the classes and is_a relationships in the DBOntology. The process stops when the set of concepts to be materialized is empty, or there is no

[1] Note that in the general case we cannot assume that the tables have the same structure. Anyway it is possible to create a 'merged' table starting from the two SELECT statements. The procedure is omitted for simplicity of reading.

change in this set from one step to the next one. If the process stops and the set is not empty, all the remaining concepts will be discarded from the enhancing process.

Before going to see the algorithm structure in more detail, let us briefly remind the definition of ontology. Formally, an ontology is a 5-tuple $O:=\{C, R, HC, rel, A_0\}$, where C is a set of concepts, which represent the entities in the ontology domain; R is a set of relations defined among concepts; HC is a taxonomy or concept-hierarchy, which defines the *is_a* relations among concepts (HC($C1$, $C2$) means that $C1$ is a sub-concept of $C2$), rel: $R \rightarrow C \times C$ is a function that specifies the relations on R. Finally, A_0 is the set of axioms expressed in a logical language, such as first order logic.

Let us now introduce some basic definitions and acronyms used through the algorithm: DBO indicates the DBOntology, DO the Domain Ontology, CA stands for Concept Axiom indicating the set of Domain Ontology concepts that do not belong to the DBOntology, and that are defined by an axiom. We remind that the only concepts of the Domain Ontology considered to be part of CA, will be those associated to the direct children and siblings of the initial DBOntology leaves. Finally CC is the set (ConceptComplement) of the concept to be complemented. A number of predefined functions are exploited, such as DC(c), DefiningConcept, that is a function which, given a concept c, return the set of concepts used in the axiom to define c. h_O indicates the Hierachy of the ontology O ($h_O(c)$ returns the set of children of c, whereas $h_O^{-1}(c)$ returns the concept father of c). Eventually, Leaf(O) is a function that, given an ontology O, returns the set of its leaves. Analogously, Node(O) returns the set of node concepts in the ontology O. The function has_axiom(c) checks whether a given concept is defined by an axiom.

Enhancing Algorithm

\\ Selection of the possible nodes for the enhancing process

CA = {}

Leaf = Leaf(DBO) \\takes the leaves of the DBOntology. The DBOntology is initially composed of all the dark classes of Onto

Repeat

c ∈ Leaf

c' = h_{DO}^{-1}(c) \\ takes the father of the node

Leaf\{c}

forall c'' ∈ h_{DO}(c').c''<>c \\ the set of nodes siblings of c in DO

if has_axiom(c'') then

if not ((c'' ∈ Nodes(DBO)) or (c'' ∈ Leaf)) then

CA ∪ {c''}

forall c'' ∈ h_{DO}(c) \\ the set of nodes children of c in DO

if has_axiom(c'') then

CA U{c''}

until Leaf = {}

\\ Selection of the concept to be materialized, construction of the materialized view, update of the DBO, and update of the DB if needed

Repeat

CA_tmp=CA

Forall c ∈ CA.

c' = DC(c)\ h_{DO}^{-1}(c) \\takes the concept needed to define the new class

if c' ∈ DBO then

CA\{c}

Create_view(c)

add_to_DBO(c)

c" = h_{DO}^{-1}(c)

if c" ∈ Leaf(DBO) and not c" ∈ CC then

CC U {c"}

Until CA = CA_tmp

∀c ∈ CC

Remove_and_complement(c)

Before going to see how the function add_to_DBO(c) is defined, we introduce the predefined function Axiom(c) that, given a concept, returns the axiom that defines the concept itself and *null* if the concept has no associated axiom. Furthermore we exploit the definition of hierarchy to check if two concepts are in the *is_a* relation.

***add_to_DBO(c)* =**

Leaf = Leaf(DBO)

Repeat

c' ∈ Leaf

Leaf \ {c'}

c" ∈ h_{DO}^{-1}(c') \\ father of concept c' in the HC relation

if HC_{DO}(c,c") then \\c is sibling of c' if HC_{DO}(c,c") holds

C_{DBO} ∪ {c}\\adds the new concept to the set of DBO concepts

HC_{DBO} ∪ {(c,c")} \\adds the father relationship in DBO

If HC_{DO}(c,c') then \\c is child of c' if HC_{DO}(c,c') holds

C_{DBO} ∪ {c}

HC_{DBO} ∪ {(c,c')}

until HC_{DO}(c,c") or HC_{DO}(c,c')

For the create_view(c) we remind that the function DC(c) selects all the concepts used in the axiom that defines c. The function concept_name(c) returns the name of the table the concept c is mapped on. The function Axiom(c) returns the axiom that defines the concept c. The function get_predicate(axiom) returns the property involved in the definition of the axiom. The function DBConnection provides the connection to the spatial DB and executed the query that takes as input. The function $Leaves_O$(c) returns all the leaves children of the concept c in the ontology O.

***create_view(c)* =**

Conc = DC(c) //takes the father of c and the other concept in the

//Enhanced Ontology needed to define c.

ExpFa = {} // the set of concepts the father can be rewritten in

∀ c' ∈ Conc

if HC_{DO}(c,c') then

father = c'

```
            // verify if father has to be expanded
            if not isMappedToTable(father) then
                Leaves = Leaves_DBO(father)
                    ∀ c ∈ Leaves
                        If isMappedToTable(c) then ExpFa ∪ {c}
            Else ExpFa = {father}
    else Leaf = Leaves_DBO(c')//take the leafs of the concept c'
//production of the query for each Leaf related to the axiom
Q = {}
∀ f ∈ ExpFa
∀ c ∈ Leaf
        Q ∪ {"SELECT * FROM " & concept_name(f) & "AS a, " &
                concept_name(c) & "AS b WHERE " &
                                get_predicate(Axiom(c)) &
                                " (a.the_geom, b.the_geom)"}
//creation of the view
query = empty string
q ∈ Q
query = "CREATE VIEW " & concept_name(c) & " VIEW as ( "  & q
Q\{q}
∀ q ∈ Q
    query = query & "UNION " & q
query = query & ")"
DBConnection(query)//connection to the DB and  view materialization
```

Eventually the definition of the remove_and_complement(c) function follows:

Remove_and_complement(c)=

//selection of the children of c in the extended enriched ontology

Leaf = $Leaves_{DBO}$(c)

query = "CREATE VIEW Complement" & concept_name(c) & " as SELECT * FROM " & concept_name(c) & " WHERE not exist "

l ∈ Leaf

query = query & "(SELECT * FROM " & concept_name(l) & ")"

Leaf\{l}

∀ l ∈ Leaf

query = query & " AND not exist (SELECT * FROM " &

concept_name(l) & ")"

DBConnection (query)

DBConnection("DROP TABLE " & concept_name(c))

5 Conclusions and Future work

This paper introduced a methodology to build an ontology as a semantic layer between a user query language and a geodatabase. In particular, we focus on exploiting the semantic embedded in a domain ontology, representing knowledge of the handled application The constructed enhanced ontology is aimed at supporting the translation of user queries from natural language to spatial SQL. Furthermore, the resulting enhanced ontology represents the semantics of the stored data in two directions. In one direction, it provides a concept abstraction by exploiting the ontology upper hierarchy, on another direction, it provides a concept specialization, by exploiting the concepts associated to the materialized views in the formulation of the queries.

The long term goal of the approach is to translate natural language queries into statements directly linked to database tables. In this context, the addition of a semantic abstraction layer on top of data allows the user to refer to domain concepts that are not explicitly represented in database table. This gives the user a greater expressive power and a semantic view of geographical data.

Some interesting open issue we intend to explore in the future are related to the limitations that we have posed to the present solution. For example, relaxing the limitations on the depth in the selections of concepts in the enhancing process. It would also be interesting to exploit the power of axioms to generalize the use of relations, and to use object property to extend axioms.

References

[Acci05] Acciarri A., Calvanese D., De Giacomo G., Lembo D., Lenzerini M., Palmieri M., Rosati R. QuOnto: Querying ontologies. In Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005), 2005.

[Baad03] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., & Patel-Schneider, P. The description logic handbook: Theory, implementation and applications. Cambridge: Cambridge University Press. 2003

[Baglioni07] Baglioni, M., Masserotti, M.V., Renso, C., Spinsanti, L.: Building Geospatial Ontologies from Geographical Databases. In: GeoS2007, Mexico City November 29-30, 2007. Lecture Notes in Computer Science, vol. 4853, pag. 195-209, Springer 2007.

[Baglioni08] Baglioni M., Giovannetti E., Masserotti, M.V., Renso, C., Spinsanti, L.: Ontology-supported Querying of Geographical Databases. In: Transactions of GIS,Volume 12, Issue s1, Date: December 2008, Pages: 31-44

[Baglioni09] M.Baglioni, M.V. Masserotti, C. Renso, L.Soriano, L. Spinsanti. A Tool for Extracting Ontologies from Geographical Databases, SEBD 2009, Convegno Italiano di Basi di Dati, Camogli, Genova, Italy.

[Bishr08] Bishr Y. The Geospatial Semantic Web: Applications. In Encyclopedia of GIS, Shekhar, Shashi and Xiong, Hui (Eds.) Springer Verlag, 2008

[Cardoso07] Cardoso N., Silva M.J.: Query Expansion through Geographical feature Types. In: GIR'07, November 9, 2007, Lisboa, Portugal.

[Fonseca99] Fonseca, F.T., Egenhofer, M.J.: Ontology-Driven Geographic Information Systems. In: ACM-GIS, ACM Press, New York (1999)

[Fonseca02] Fonseca, F.T., Egenhofer, M.J., Agouris, P., Camara, C.: Using Ontologies for Integrated Geographic Information Systems. Transact. GIS 6(3), 231–257 (2002)

[Gru08] Gruber, T. (2008). Ontology. In L. Liu, & M. T. Özsu (Eds.), Encyclopedia of database systems. Springer-Verlag.

[Guting05] Güting, R., Schneider, M. . Moving Objects Databases, Morgan-Kauffman, 2005

[Lüscher 08] Lüscher, P., Weibel, R. & Mackaness, W.A. (2008): Where is theTerraced House? On the Use of Ontologies for Recognition of Urban Concepts in Cartographic Databases. In: Ruas, A. & Gold, C. (eds.): Headway in Spatial Data Handling, Springer, 449-466.

[Mark06] Mark, D.M., Egenhofer, M.J., Hirtle, S., Smith, B.: UCGIS Emerging Research Theme: Ontological Foundations for Geographical Information Science (2006)

[OpenGIS] OpenGIS Simple Feature Access

http://portal.opengeospatial.org/files/?artifact_id=18242

[OraSpa] ORACLE, Oracle Spatial, http://www.oracle.com/database/spatial.html

[OWL] OWL Web Ontology Language http://www.w3.org/TR/owl-features/

[Peachavanish07] Peachavanish, R., Karimi, H. A.,: Ontological Engineering for Interpreting Geospatial Queries. In Transactions in GIS, 2007, Volume 11 Issue 1, Pages 115 - 130

[Peuquet02] Peuquet, D. Representations of Space and Time. The Guilford Press, N.Y. (2002)

[PostGis] PostGres database– PostGIS extension, http://postgis.refractions.net/

[Spaccapietra04] Spaccapietra, S., Cullot, N., Parent, C., Vangenot, C.: On Spatial Ontologies. In: GeoInfo (2004).

[Jones05] Fu, G., Jones, C.B., Abdelmoty, A. I.: Ontology-based Spatial Query Expansion in Information Retrieval LNCS, vol. 3761, pag. 1466-1482, Springer 2005.

[Torres05] Torres, M., Quintero, R., Moreno, M., Fonseca, F.T.: Ontology-Driven Description of Spatial Data for Their Semantic Processing. In: Rodríguez, M.A., Cruz, I., Levashkin, S., Egenhofer, M.J. (eds.) GeoS 2005. LNCS, vol. 3799, pp. 242–249. Springer, 2005.

[Viegas06] Viegas, R., Soares, V.: Querying a Geographic Database using an Ontology-Based Methodology. In: GEOINFO 2006 - VIII Brazilian Symposium on GeoInformatics, 2006.

[Zhao08] Zhao T., Zhang C., Wei M., Peng Z.: Ontology-Based Geospatial Data Query and Integration. In Cova T.J. et al. (Eds): GIScience 2008, LNCS 5266, pp. 370-392,Springer.