Asking and answering queries semantically

P. Bouquet, G. Kuper, S. Zanobini
Department of Information and Communication Technology
University of Trento
Via Sommarive, 14
38050 Trento (Italy)
{bouquet,kuper,zanobini}@dit.unitn.it

***Abstract*— In this paper we propose a new method, called SEMQUERY, for querying information sources whose data are organized according to different schemata. The method is based on the idea of *semantic elicitation*, namely a process which takes in input the structural part of a query (e.g. the XPath part of an XQuery) and returns an expression in a logical language which represent the meaning of the query in a form which ideally is independent from its original syntactic formulation. Since the same process of elicitation can be performed on any path of schemata used to organize data, the decision on whether there is any logical relation between a query and a path in the schema is made via logical reasoning.**

I. INTRODUCTION

The distribution of knowledge across a large number of different and autonomous providers raises the problem of retrieving information from semantically heterogeneous sources. A crucial issue is how to allow users to query heterogeneous information sources without assuming that they know their conceptual structure. The problem, of course, is not new. It has been studied for a long time in the database community, in the form of querying distributed and heterogeneous databases (e.g. [3]). However, the proposed solutions either cannot be straightforwardly extended to other domains (e.g. querying document repositories based on a classification schema, or according to a hierarchy of web directories), or are based on assumptions which limit their applicability (e.g. assuming that mappings across schemata are available from the start). In this paper, we propose a new approach, which builds on our experience in the Semantic Web, but can be generalized to any information source which is structured according to some explicit schema, such as databases, product and service catalogs, document directories (e.g. web directories in search engines like Google or Yahoo), file systems (e.g. in peer-to-peer file sharing applications). As a concrete example, and without any loss of generality, we will discuss the problem, and present our results, using XMLSchema [8], [9] as a syntax for schemata, and XQuery [10] as a query language (more precisely, the XPath [7] fragment of an XQuery).

The main contribution of the paper is a method, called SEMQUERY, which, given a query containing an XPath expression q and a collection of XMLSchema specifications Σ, computes the set of *semantically equivalent* rewritings of q with respect to each $\sigma \in \Sigma$. SEMQUERY is based on the concept of *semantic elicitation*, which is a process that takes an XPath expression and encodes its meaning in a logical language L[1]. This encoding, which in SEMQUERY is performed fully automatically, makes explicit the meaning of an XPath expression in a form which is (relatively) independent from its original syntactic form, and ideally is logically equivalent to any other semantically equivalent XPath expression. To achieve this result, we assume that the names of elements and attributes in a schema are meaningful noun phrases of some natural language (e.g. English). As we argued in [1], this assumption is crucial, as it allows us to exploit lexical and domain knowledge to construct a deep interpretation of XPath expressions.

Semantic elicitation can be applied both to XPath expressions occurring in a query and to XPath expressions which describe a path in a XMLSchema. Therefore, the way a query q is processed against a schema is the following: the meaning of the XPath part of the query is elicited, the meaning of each path in the schema is elicited as well, and then the decision on whether there is any logical relation between the query q and a path in the schema is made via logical reasoning (in the paper, we check for concept equivalence or subsumption, but of course this is only a special case). As we shall show, SEMQUERY can be implemented quite efficiently, as the semantic elicitation of a schema's paths can be performed at design time and stored with the schema (this enriched version of a schema is what we call a *context*). At execution time, we only need to elicit the meaning of the query and match it against the concepts (already) available in a schema.

The structure of the paper is as follows: Section II defines the problem, and Section III describes our method, SEMQUERY, for solving the problem. Finally, Section IV provides a detailed description of the semantic elicitation phase.

II. THE PROBLEM

Imagine that we have two schemata σ_1 and σ_2 such as those depicted in Figure 1, and suppose they are used to structure two multimedia document repositories. Consider the paths which lead to the node `LANDSCAPES` in the schema on the left hand side and to the node `JPEG` in schema on the right hand side. Despite their syntactical difference, they seem

[1]In this paper we use Description Logic, as we deal mostly with concepts and attributes. However, in previous work on semantic coordination, a much simpler encoding in propositional logic was used [1].

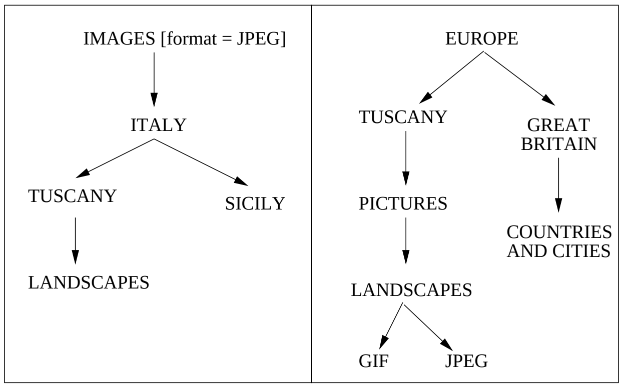

Fig. 1. Two simple schemata

to have the same meaning, something like *images of Tuscan landscapes in JPEG format*. For classification schemata, the intuition behind the notion of "having the same meaning" is that a human user would classify the same documents under the two nodes. However, XPath expressions refer directly to the syntactical features of schemata; as a result, there is no single XPath expression that can be used in a query to refer to the two semantically equivalent nodes LANDSCAPES and JPEG, and therefore to retrieve the associated documents.

We therefore need a way of recognizing that the two XPath expressions

$$/\mathtt{IMAGES}[\mathtt{format} = '\mathtt{JPEG}']/\mathtt{ITALY}/\mathtt{TUSCANY}/\mathtt{LANDSCAPES} \quad (1)$$

and

$$//\mathtt{IMAGES}[\mathtt{about} = '\mathtt{Tuscany}']/\mathtt{LANDSCAPES}/\mathtt{JPEG} \quad (2)$$

are semantically equivalent, regardless of their concrete syntactic form, and therefore that an XQuery expression containing the first path should allow us to retrieve not only documents from the corresponding path in the first schema, but also document from the path in the second schema.

In addition, one might want to recognize that, for example, a query containing the XPath expression

$$/\mathtt{IMAGES}[\mathtt{format} = '\mathtt{JPEG}']/\mathtt{ITALY}/\mathtt{FLORENCE}/\mathtt{LANDSCAPES}$$

can be also a valuable answer for a query containing the expression

$$/\mathtt{IMAGES}[\mathtt{format} = '\mathtt{JPEG}']/\mathtt{ITALY}/\mathtt{TUSCANY}/\mathtt{LANDSCAPES}$$

even though in this case the relation would not be semantic equivalence, but rather subsumption (after all, JPEG pictures of landscapes of Florence are a special case of JPEG pictures of landscapes of Tuscany).

To sum up, the examples show that syntactically different XPath expressions may be used to refer to *semantically equivalent* concepts. The problem we address is to define an automated method for *asking and answering queries semantically*, namely to ask queries which are (relatively) independent from their syntactic form, and to answer a semantic query by looking for semantic relations between concepts.

III. SEMQUERY: A SHORT DESCRIPTION

SEMQUERY is a method for asking semantic queries based on what we call the *meaningfulness hypothesis*, namely that the schemata which are used to organize information sources have meaningful labels[2]. As we discussed in [1], there are two reasons for making this hypothesis in a framework in which semantic relations between schema elements are to be discovered and exploited in a principled way:

1) Firstly, if we didn't assume that labels were meaningful, there would be no reason to say, for example, that there is a relation between IMAGES and PICTURES. Indeed, as mere strings, there is no similarity, and synonymy is definitely a semantic relation between meaningful words. Conversely, we don't want to conclude that DIG an DOG are more similar than DIG and EXCAVATION, though the first two are syntactically much more similar than the others. We stress this issue, as many approaches to semantic interoperability, e.g. those based on graph matching, use thesauri or other type of lexical information in a way which is sound only if one makes the assumption of meaningfulness;
2) The second, more important observation, is that if we ignore the meaning of lablels, we will not be able to discover relations between paths in schemata that depend only on the meanings of these labels. For example, consider the two pairs of isomorphic schemata in Figure 2. Intuitively, the relation between the two pairs of

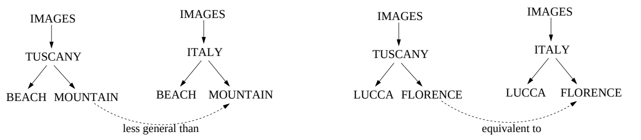

Fig. 2. Relations across schemata with meaningful labels

nodes is different, subsumption on the left hand side and equivalence on the right hand side, even though the two schemata are isomorphic. The explanation is that we use what we know about the concepts corresponding to the labels, in order to decide what a node really means.

Closely related to the meaningfulness hypothesis is the idea that SEMQUERY should use knowledge about labels to improve the quality of its results. Indeed, only domain knowledge can allow us to realize that the concept of 'images of Florence' is less general than the concept of 'images of Tuscany', no matter how the two concepts are expressed syntactically. This again is crucial to discover semantic relations across paths, which do not depend only on what is explicitly said, but also on what we know about the corresponding concepts.

We now turn to a general description of our method. Let Λ be the set of noun phrases that can be built in English from a set λ of English words, and Λ^* the set of all finite XPath expressions using only elements of Λ as tags, the child and descendant axes, and the wild card *. $\mathcal{C}$ is the set of terms that

[2]More precisely, we assume that they would be interpreted as meaningful by humans via some simple manipulation; for example, labels like `ProgrammingLanguages`, `Programming_Languages` or even `ProgLang` would be easily recognized as meaningful – and basically equivalent – by humans on the web site of, say, a Computer Science Department. In the rest of the paper, we will pretend that labels are English noun phrases, but in many real applications it may be necessary to go through a normalization phase in which labels are transformed into correct English words and noun phrases.

can be built in a Description Logic language like $\mathcal{ALC}$[3] from a set T of primitive terms and a set R of primitive roles, and $\mathcal{O}$ is a (possibly empty) set of axioms defined over $\mathcal{C}$. The process of semantic elicitation can be viewed as a function $\Upsilon : \Lambda^* \rightarrow \mathcal{C}$ which takes as input an element of Λ^* and returns a (complex) term in $\mathcal{C}$ which expresses its meaning.

How to compute this function is a crucial issue of our work, and this will be discussed in Section IV. For now, suppose that Υ is defined. We can then divide the set Λ^* of all the possible XPath expressions into sets of semantically equivalent expressions, namely expressions with an equivalent meaning. Formally:

Definition 1 (Equivalence class): Let p and p' be two XPath expressions from Λ^*, Υ the semantic elicitation function, and let $L = \langle \mathcal{C}, \mathcal{O} \rangle$ be a T-Box containing terminological axioms. We say that p and p' belong to the same equivalence class $|\Lambda^*|_{(\Upsilon,L)}$ of Λ^* with respect to Υ and L iff:

$$\mathcal{O} \models (\Upsilon(p) \equiv \Upsilon(p'))$$

We write $\mathcal{J}(p)$ to denote the equivalence class containing p.

We can now define the set of *semantically equivalent rewritings* of a query over a collection of schemata. Intuitively, given an XPath expression p and a set of schemata Γ, the problem of answering queries semantically can be defined as the problem of determining the set P of XPath expressions occurring in Γ which belong to the same equivalence class of p. Formally:

Definition 2 (Semantically equivalent answer): Let p be an XPath expression occurring in a query, Γ a set of schemata, and $P \subseteq \Lambda^*$ the set of all the XPath expressions which denote a path occurring in at least one schema in Γ. Then P' is the set of *semantically equivalent answers* for p if it is the maximal subset of P such that

$$\text{for all } q \in P',\ \mathcal{J}(q) = \mathcal{J}(p)$$

A weaker, but still useful, notion of semantic answer can be defined as follows. Suppose that a query containing the XPath expression `/IMAGES/JPEG/ITALY/LANDSCAPES` is performed over the structure on the left in Figure 1. Intuitively, the associated concept, ‘JPEG images of Italian landscape’, is subsumed by the concept ‘JPEG images of Tuscany’s landscape’ corresponding to the path `/IMAGES[format = 'JPEG']/ITALY/TUSCANY/LANDSCAPE`. Therefore, the corresponding XPath expression is not semantically equivalent to the query, but can be considered as a *semantically less general* answer.

Formally, let $\leq$ be a partial order over the set $|\Lambda^*|_{(\Upsilon,\mathcal{O})}$ w.r.t. Υ and $\mathcal{O}$, let X and Y be two equivalence classes in $|\Lambda^*|_{(\Upsilon,\mathcal{O})}$, and let x and y be the witnesses of X and Y respectively. Then

$$x \leq y \quad \text{iff} \quad \mathcal{O} \models \Upsilon(x) \sqsubseteq \Upsilon(y)$$

We can then define the set of all the semantically related answers as follows:

Definition 3 (Semantically related answer): Let p an XPath expression occurring in a query, Γ a set of schemata, and $P \subseteq \Lambda^*$ the set of all XPath expressions which denote a path occurring in at least one schema in Γ. Then P' is the set of *semantically related answers* for p if it is the maximal subset of P such that, for all $q \in P'$, one of the following conditions hold:

1) $\mathcal{J}(p) \leq \mathcal{J}(q)$ (semantically less general answer)
2) $\mathcal{J}(q) \leq \mathcal{J}(p)$ (semantically more general answer)

In most real applications, only less general answers are likely to be used; however, we cannot exclude that in some situations one might be interested in broadening the scope of a search and look for concepts that are more general than the initial one.

IV. Semantic elicitation

A crucial issue for our approach is the definition of a reasonable implementation of the semantic elicitation function Υ. In this section we provide an algorithm which approximates Υ under the assumptions of meaningfulness. The current version is adapted from [6].

Semantic elicitation is not just a (whatever complex) syntactic rephrasing an XPath expression into an expression of some formal language. To explain what we mean, consider the two following XPath expressions:

$$\texttt{IMAGES/JPEG/TUSCANY} \tag{3}$$
$$\texttt{IMAGES/ITALY/TUSCANY} \tag{4}$$

Intuitively, the two XPath expressions could be translated into the two DL terms respectively:

$$\mathsf{Image} \sqcap \exists \mathsf{format.JPEG} \sqcap \exists \mathsf{about.Tuscany} \tag{5}$$
$$\mathsf{Image} \sqcap \exists \mathsf{about.}(\mathsf{Tuscany} \sqcap \exists \mathsf{partOf.Italy}) \tag{6}$$

where Image is the concept of “a visual representation of an object or scene or person or abstraction produced on a surface” (from WordNet2.0, sense 1), JPEG is a format for electronic images, Tuscany is the Italian region, and so on and so forth.

Despite their isomorphic syntactical structure, the XPath expressions (3) and (4) do no have an isomorphic semantic structure. Indeed, in (3), the second and the third elements `JPEG` and `TUSCANY` are modifiers of the element `IMAGES`, while in (4) the first element, `IMAGES`, is modified by the third element, `TUSCANY`, which is in turn modified by the second element, `ITALY`. Thus, the process of semantic elicitation should be a process of deep interpretation of an XPath expression, as a human being would do. [2] argues that such a deep interpretation must take into account two general kinds of knowledge:

[3]The choice of the logical language depends on what kind of structures one is querying. Indeed, it’s all very well to say that we deal with XPath expressions, but one thing is to query a hierarchical classification, and one thing is to query a service description. Indeed, it is well-known that the sub-element relation in XML does not have any pre-defined meaning, and can be used to organize concepts in a taxonomy, objects in a partonomy, or even to decompose actions in a service description. A method for semantic elicitation must take into account this pragmatic aspect, and choose the most appropriate language for each case. In the situation we describe below, we are interested in querying classifications, where each node corresponds to a (complex) concept (e.g. ‘photos of my holidays in Italy’), and therefore we will adopt the language $\mathcal{ALC}$; however, no DL logic language would not do for a service description.

- **Lexical knowledge:** it allows us to determine the (set of) concept(s) possibly denoted by a lemma[4]; for example, the fact that the lemma 'image' can mean 'a visual representation' and 'a standard or typical example'. Conversely, it can be used to recognize that two different lemmas may refer to the same concept; for example, the words 'image' and 'picture' can both denote the concept of a visual representation, and therefore – under this interpretation – they are to be taken as synonyms. Formally, let m be the set of lemmas that can be denoted by words occurring in Λ. A lexicon $\mathcal{L} : m \to 2^{T \cup R}$ is a function that associates each lemma to a set of primitive concepts or roles belonging to the signature of the T-Box L. In the current version we shall use $\mathsf{Image\#n}$ for the n-th concept that can be denoted by the lemma 'Image'.
- **Ontological/World knowledge:** this type of knowledge concerns relations between primitive concepts. For example, the fact that there is a PartOf relation between the concept $\mathsf{Italy\#1}$ ('a republic in southern Europe') and the concept $\mathsf{Tuscany\#1}$ ('a region in central Italy'). We formally define the ontological knowledge $\mathcal{O}$ to be a set of axioms of the T-Box L. In the this paper we will assume that we have a "black box" function $\mathcal{R} : T \times T \to R$ which takes as input two concepts and returns a role which holds between them. For further details, see [6].

For the sake of simplicity, we shall assume the set Λ consists of single words in English[5]. Furthermore, we shall use WORDNET[6] as our source $\mathcal{L}$ of lexical knowledge; finally, the terms T and the roles R of the signature S will be interpreted as WORDNET synsets; R contains two predefined roles, IsA and PartOf; the set of concepts $\mathcal{C}$ is the set of all the allowed expression built using the signature S; finally, the ontological knowledge $\mathcal{O}$ contains the IsA and PartOf relations defined over WORDNET, and possibly other relations from some domain ontology. To make the presentation clearer, we show how the process works with a running example over the X-Path (2). The process of semantic elicitation is split into four main steps:

1) Local Interpretation: In the first phase, we try to build the space of all possible interpretations for each element of the query. Each element consists of a label and (possibly) a set of attributes. We interpret an attribute as an object which qualifies the meaning expressed by the label. Formally, we interpret a node by the expression

$$\mathsf{label} \sqcap \exists \mathsf{attName}_1.\mathsf{filler}_1 \sqcap \ldots \sqcap \exists \mathsf{attName}_n.\mathsf{filler}_n$$

In particular, the attribute name is interpreted as a role and the attribute filler as a range. We obtain the space of all possible interpretations of a node by replacing the words that occur in the pattern elements by all concept that could possibly be denoted by the words, with respect to the lexicon $\mathcal{L}$. For example, our lexicon provides 7 concepts for the lemma 'image', 7 for 'about', 1 for 'Tuscany', 4 for 'landscape' and 1 for 'JPEG'. As a result, the space of the possible interpretations for the elements of example (2) is:

$li(\mathtt{IMAGES})$	$li(\mathtt{LANDSCAPE})$	$li(\mathtt{JPEG})$
$\mathsf{Images\#1} \sqcap \exists \mathsf{about\#1}.\mathsf{Tuscany\#1}$	$\mathsf{Landscape\#1}$	$\mathsf{JPEG\#1}$
$\mathsf{Images\#1} \sqcap \exists \mathsf{about\#2}.\mathsf{Tuscany\#1}$	$\vdots$	
$\vdots$	$\mathsf{Landscape\#4}$	
$\mathsf{Images\#7} \sqcap \exists \mathsf{about\#7}.\mathsf{Tuscany\#1}$		

2) Semantic Enrichment: We now look for semantic relations that hold between the concepts defined in the previous step. This is done by accessing the ontological knowledge $\mathcal{O}$ using the $\mathcal{R}$ function. In particular, we search for the relations that hold between two different kinds of elements:

- **Attribute Roles:** Consider $\mathtt{IMAGES}[\mathtt{about} = '\mathtt{Tuscany}']$ in our example. In the previous step, we built the set $li(\mathtt{IMAGES})$ of all the possible interpretations for this node. We now use the ontology $\mathcal{O}$ to determine if it *explicitly supports* one or more of these possible interpretations. For example, we discover that $\mathcal{R}(\mathsf{Image\#2}, \mathsf{Tuscany\#1}) = \mathsf{about\#1}$, i.e., that the first interpretation is supported by the ontology.
- **Structural roles:** Here, we search for semantic relations between different elements in the same XPath expression. In our example the relation $\mathcal{R}(\mathsf{Image\#2}, \mathsf{JPEG\#1}) = \mathsf{format\#1}$ holds.

Table I shows the semantic relations that hold between the terms in our example.

1	$\langle \mathsf{Image\#2}, \mathsf{Tuscany\#1}, \mathsf{about\#1} \rangle$
2	$\langle \mathsf{Landscape\#1}, \mathsf{Image\#2}, \mathsf{IsA} \rangle$
3	$\langle \mathsf{Images\#2}, \mathsf{JPEG\#1}, \mathsf{format\#1} \rangle$
4	$\langle \mathsf{Image\#2}, \mathsf{Landscape\#3}, \mathsf{about\#1} \rangle$

TABLE I

SET OF RELATIONS

3) Semantic Filtering: This step filters out the concepts and relations which do not seem to be the right ones for the XPath expression under analysis. Such a filtering applies the following rules to every concept extracted in the previous phase:

- **Weak rule:** A concept c associated to a word w occurring in an XPath tag n can be removed if c is not involved in any relation, and there is some other concept c' that is also associated to w in n, which is involved in some relations.
- **Strong rule:** A concept c associated to a word w occurring in an XPath tag n can be removed if c is not involved in any IsA or PartOf relation and there is some another concept c' associated to w in n which is involved in some IsA or PartOf relation.

An example of the use first rule is as follows. In Table I we see that $\mathsf{Image\#2}$ occurs in relations 1–4, while $\mathsf{Image\#1}$ and $\mathsf{Image\#3} \ldots, \mathsf{Images\#7}$ do not occur in any relation. It is therefore likely that the "right" concept expressed by the

[4]We assume that we are able to determine the lemma of each word occurring in a label through some standard lemmatizer.

[5]For the case when Λ contains complex noun phrases, and for further discussion, see [6].

[6]WORDNET [4], a well-known Lexical/Ontological repository which contains the set of concepts possibly denoted by a word (called synsets, i.e. set of synonyms), and a set of relations (essentially the IsA and PartOf) that holding between senses

lemma 'Image' in this context is the second one, and the other concepts can be discarded. The second rule is stronger, as here a concept can be discarded even if it is involved in some relation. The idea is that we consider IsA and PartOf relations be stronger than the other ones, and give priority to these over others. For example, consider relations 2 and 4. Because there is a IsA relationship between Landscape#1 and Images#2 (relation 2), we can discard the concept Landscape#3 even though this concept occur in an about#1 relation (relation 4).

The goal of this step is to reduce the space of possible interpretations of a node, by discarding some concepts which are unlikely to be relevant. In our example, we would obtain the following terms.

$li(\texttt{IMAGES})$	$li(\texttt{LANDSCAPE})$	$li(\texttt{JPEG})$
Images#2 $\sqcap$ $\exists$about#1.Tuscany#1	Landscape#1	JPEG#1

Note that if more than one concept satisfies our conditions, all of them are retained (ambiguity partially solved). Furthermore, if the concepts associated to a word are not involved in any relation, no filtering is done (ambiguity is not solved). The same filtering process is then applied to the set of relations, i.e., we discard all relations involving discarded concepts, as these ones refer to concepts that no longer exist. Table II shows the current set of relations.

1	$\langle$Image#1, Tuscany#1, about#1$\rangle$
2	$\langle$Landscape#1, Image#1, IsA$\rangle$
3	$\langle$Images#1, JPEG#1, format#1$\rangle$

TABLE II
SET OF FILTERED RELATIONS

4) Constructing the representation of the semantics: The final step is to construct the logical representation of the semantics of the query. This is done in two steps.

First, we construct the *local meaning* of an element of the query, namely its meaning considering only the label and the attributes. We define the $LM(n)$, the local meaning of an element n, as the disjunction ($\sqcup$) of all terms occurring in $li(n)$, the space of all the possible interpretations.

We then combine the local meanings to obtain the *global meaning* of a node. First of introducing the method, we want to make the following observation. Consider Table II: it essentially says that there is a relation IsA between the (concepts belonging to the interpretations of the) node LANDSCAPE and the (concepts belonging to the interpretations of the) node IMAGES, and that there is a relation format#1 between the (concepts belonging to the interpretations of the) node IMAGES and the (concepts belonging to the interpretations of the) node JPEG. In short, we have the following set of relations between nodes: $\texttt{LANDSCAPE} \xrightarrow{\text{IsA}} \texttt{IMAGES} \xrightarrow{\text{format\#1}} \texttt{JPEG}$. Essentially, axioms can be interpreted as edges relating nodes. Such chain of relations can be rephrased with the following pattern[7]:

$$\texttt{LANDSCAPE} \sqcap \texttt{IMAGES} \sqcap \exists \text{format\#1}.\texttt{JPEG}$$

[7]Ambiguity can arise in the axioms. As an example, two elements can be modifiers of each. See [6] for a set of heuristics for solving some ambiguity problems.

At this time is quite simple to build the *global meaning* of the X-Path: indeed we need just to substitute the node labels with the local meanings provided by function $LM()$. Going on with our example, the global meaning for the X-Path (2) is the following Description Logic term:

$$\begin{aligned} &\text{Landscape\#1} \sqcap \text{Images\#1} \sqcap \exists\text{about\#1.Tuscany\#1} \\ &\qquad \sqcap \exists\text{format\#1.JPEG\#1} \end{aligned} \tag{7}$$

5) Dealing with special symbols: The XPath symbols $*$ and $//$ do not have an explicit semantic counterpart in some concept in C. The first is a wild card, which can be replace by any tag; the second allows us to find elements at any depth in an XML document. Here we propose a simple treatment of thiese two special symbols in SEMQUERY.

From Section IV-.4 results that we essentially combine each element as a conjunction ($\sqcap$). Following this idea, we can argue that each element of an XPath is a specification of the meaning of the others elements. Consider the element $\texttt{IMAGES[about = 'Italy']}$. Its intuitive meaning is 'Images about Italy'. The further element LANDSCAPE reduces its meaning to the 'Images about Italy that are Landscapes', so as the last element JPEG reduces the meaning to the set of 'Images about Italy with JPEG format that are Landscapes'. Following this intuition, we can interpret a sign as $*$ as one potential element that reduces the meaning of the XPath, and the sign $//$ as a possibly empty set of potential elements that reduce the meaning of the XPath.

So, instead of introducing some redundant place-holder for that symbols, we prefer to play on the class of equivalence which an XPath where such symbols occur belong to.

Let q be an XPath where the sign $*$ ($//$) occurs. Let $Y \subseteq \Lambda^*$ be the set of all the XPaths allowed by Λ such that the symbol $*$ ($//$) in q is substituted with some element (a finite, possibly empty, sequence of elements) of Λ. Let q^* ($q^{//}$) be the XPath resulting by removing (substituting with $/$) element $*$ ($//$) from q. For each $y \in Y$, if

$$\mathcal{O} \models \Upsilon(y) \sqsubseteq \Upsilon(q^*) \qquad (\mathcal{O} \models \Upsilon(y) \sqsubseteq \Upsilon(q^{//}))$$

then $q^c = y^c$, namely q belongs to all the equivalence classes of equivalence to whom belongs the XPaths in Λ^* such that (i) they are generated by substituting $*$ ($//$) in q with an element (a set of elements) of Λ and (ii) their meanings are a specification of the meaning of q. Multiple occurrence of $*$ ($//$) can be defined recursively in the same way.

6) Concluding example: Following this approach, we can state that the class of equivalence where the XPath of example 3, namely $\texttt{//IMAGES[about = 'Tuscany']/LANDSCAPE/JPEG}$, belongs to is the same of the class of equivalence where the XPath $\texttt{/IMAGES[about = 'Tuscany']/LANDSCAPE/JPEG}$ belongs to.
Now consider the XPath expression

$$\texttt{IMAGES[format = 'JPEG']/ITALY/TUSCANY/LANDSCAPES}$$

from left hand schema of Figure 1. Running the semantic elicitation process, we obtain the following DL term:

$$\mathsf{Images\#1} \sqcap \exists \mathsf{about\#1}.(\mathsf{Tuscany\#1} \sqcap \exists \mathsf{partOf\#1}.\mathsf{Italy\#1}) \\ \sqcap \mathsf{Landscape\#1} \sqcap \exists \mathsf{format\#1}.\mathsf{JPEG\#1} \quad (8)$$

Imagine then to have an ontology $\mathcal{O}$ which contains the axiom $\mathsf{Tuscany\#1} \sqsubseteq \exists \mathsf{partOf}.\mathsf{Italy\#1}$ (such an axiom can be found, as an example, in WORDNET), then we can say that

$$\mathcal{O} \models (7) \equiv (8)$$

so they belong to the same class of equivalence. From Definition 2 we can conclude that the XPath $\mathtt{IMAGES[format='JPEG']/ITALY/TUSCANY/LANDSCAPES}$ of left schema of Figure 1 is a semantically equivalent answer for the query $\mathtt{//IMAGES[about='Tuscany']/LANDSCAPE/JPEG}$.

V. CONCLUSIONS

The main idea of the paper is that querying heterogeneous information sources requires to abstract from the syntactic form of local schemata and lift the representation to a level in which only semantic differences are preserved. This is what we called semantic elicitation. Here we proposed a general method for processing these type of semantic queries called SEMQUERY, and described a technique for semantic elicitation derived from our experience on the problem of semantic interoperability in the Semantic Web.

We are perfectly aware that this is only a starting point. Future work will explore the following directions. First, we will extend the approach to other kinds of data sources (e.g. relational databases) and other query languages (e.g. SQL). Second, we want to test the approach on real cases, and see how it performs from a user's point of view; however, we must say that similar tests have been done in our work on the Semantic Web (see e.g. [5]) in the domain of web directories and e-commerce catalogs, and the results were quite promising.

REFERENCES

[1] P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: a new approach and an application. In K. Sycara, editor, *Second International Semantic Web Conference (ISWC-03)*, Lecture Notes in Computer Science (LNCS), Sanibel Island (Florida, USA), October 2003.

[2] P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: a new approach and an application. In D. Fensel, K. P. Sycara, and J. Mylopoulos, editors, *The Semantic Web – 2nd international semantic web conference (ISWC 2003)*, volume 2870 of *LNCS*, Sanibel Island, Fla., USA, 20-23 October 2003.

[3] A. Elmagarmid, M. Rusinkiewicz, and A. Sheth, editors. *Management of Heterogeneous and Autonomous Database Systems*. Morgan Kaufmann, 1999.

[4] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, US, 1998.

[5] B. M. Magnini, L. Serafini, A. Doná, L. Gatti, C. Girardi, , and M. Speranza. Large–scale evaluation of context matching. Technical Report 0301–07, ITC–IRST, Trento, Italy, 2003.

[6] S. Sceffer, L. Serafini, and S. Zanobini. Semantic coordination of hierarchical classifications with attributes. Technical Report 706, Department of Informatics and Telecommunications, University of Trento, December 2004. http://eprints.biblio.unitn.it/archive/00000706/.

[7] W3C. XML Path Language (XPath). `http://www.w3.org/TR/xpath`, November 1999.

[8] W3C. XML Schema Part 1: Structures Second Edition. `http://www.w3.org/TR/xmlschema-1/`, October 2004.

[9] W3C. XML Schema Part 2: Datatypes Second Edition. `http://www.w3.org/TR/xmlschema-2/`, October 2004.

[10] W3C. XQuery 1.0: An XML Query Language. `http://www.w3.org/TR/xquery/`, October 2004.