Domain Specific Ontologies for Semantic Information Brokering on the Global Information Infrastructure

E. Mena[1] V. Kashyap[2] A. Illarramendi[3] A. Sheth[4]

[1] *IIS depart., Univ. de Zaragoza. Spain. jibmenie@si.ehu.es*

[2] *MCC. Austin, TX 78759-6509. kashyap@mcc.com*

[3] *LSI depart., UPV. San Sebastián. Spain. http://siul02.si.ehu.es/~jirgbdat*

[4] *LSDIS Lab, Univ. of Georgia, Athens, GA 30602. http://lsdis.cs.uga.edu*

Abstract

Recent emerging technologies such as internetworking and the World Wide Web (WWW) have significantly expanded the types, availability, and volume of data accessible to an information management system. In this new environment it is imperative to view an information source at the level of its relevant semantic concepts. We propose that these semantic concepts be chosen from pre-existing domain specific ontologies. Domain specific ontologies are used as tools/mechanisms for specifying the *ontological commitments* or agreements between information users and providers on the information infrastructure. We use domain specific ontologies to tackle the information explosion by the: (a) Re-use and organization of knowledge in pre-existing real world ontologies, achieved by mapping semantic concepts in the ontologies to data structures in the underlying repositories; and (b) Knowledge integration and development of mechanisms to translate information requests across ontologies. We thus provide support for multiple domain specific ontologies as alternate world views on the vast amounts of data. Semantic information brokering is implemented by brokering across domain ontologies based on interontology relationships such as *synonyms, hyponyms* and *hypernyms* defined between terms in different ontologies. Information requests are rewritten using these relationships to obtain translations across ontologies. These ideas have been implemented in the OBSERVER system, the algorithms for which are discussed in this paper.

1 Introduction

Recent emerging technologies such as internetworking and the World Wide Web have significantly expanded the types, availability, and volume of data accessible to an information management system. In the diversified and unmanaged Web-centric environment, it becomes impossible for users to be aware of the locations, organization/structure, query languages and semantics of the data in the various repositories. Use of domain specific ontologies is an appealing approach to allow users to express information requests at a higher level of abstraction as compared to keyword based access. It provides the ability to view an information source at the level of its relevant semantic concepts and supports information requests that are specified generically independent

of the structure, location or even existence of the requested information. In particular, domain specific ontologies can help tackle information explosion by:

- Determining the relevance of an information source before accessing the underlying data. Representation of ontologies in a knowledge representation system can enable the use of the underlying inference mechanisms for determination of relevance.

- Supporting wider accessibility of data via multiple ontologies (representing new and different world views) and interoperation across them based on semantic relationships such as *synonyms, hyponyms* and *hypernyms*.

An ontology may be defined as the specification of a representational vocabulary for a shared domain of discourse which may include definition of classes, relations, functions and other objects [7]. A significant amount of work has been done in designing and creation of ontologies in AI and Knowledge Representation using semi-automatic techniques [14].

In an open and dynamic environment such as the Web, it is infeasible to expect different communities of users and information providers to conform to the same vocabulary to describe and represent their information. We believe that domain ontologies can play a critical role in supporting semantic information brokering for providing access to numerous data repositories accessible on this global information infrastructure enabled by the Web. This would require the support for multiple domain ontologies that may pre-exist and may be developed independently of the data repositories. Two critical issues involved in this information brokering paradigm are support for ontological commitments and support for query processing over multiple ontologies.

Ontological commitments are agreements between information users and information providers. Information providers support ontological commitments by providing mappings from the data structures in the repositories to semantic concepts in the domain ontologies. Domain specific ontologies thus enable the re-use, organization and communication of knowledge and semantics between information users and providers. Approaches for mapping database objects to semantic contextual expressions constructed from ontological terms have been proposed in [8, 5, 2].

Our query processing approach enables the user to subscribe to the vocabulary (characterized by a domain ontology) he is familiar with. Even when the user poses a query using terms from one ontology, relevant concepts may be described in other ontologies, and the corresponding information may be accessible through semantic relationships associated with those ontologies. This requires a strategy that expands a query on one ontology to other relevant ontologies. In this context, we need a strategy to support this query expansion, but restrict it appropriately to manage the quality (relevance) of information returned.

The OBSERVER[1] system is an implementation of the above approach. We use a system based on Description Logics (DL) [3] to represent the domain ontologies. Information requests to OBSERVER are specified as a DL expression based on terms chosen from some domain ontology. These requests are then translated by using terms of other target domain ontologies. A limited implementation that supports translations

[1]OBSERVER (*Ontology Based System Enhanced with Relationships for Vocabulary hEterogeneity Resolution*) is our approach of using multiple pre-existing ontologies to access heterogeneous, distributed and independently developed data repositories [13].

obtained by using *synonym* relationships has been described in [13]. Here we extend that to support combinations of *synonym*, *hyponym* and *hypernym* relationships to access information not available through a single ontology. By the use of these semantic relationships, OBSERVER determines the relevant data repositories and translates the DL expressions into the local query languages of the data repositories. Domain ontologies such as *Bibliographic Data* (DARPA Knowledge Sharing Effort) and *WordNet 1.5* are well known and publicly available on the Web. To demonstrate our approach and techniques, we use these and other ontologies that have been developed independently by different organizations to describe data repositories containing bibliographical references.

Other relevant projects that also subscribe to the idea of using ontologies to describe data repositories are SIMS [1], Information Manifold [9] and InfoSleuth [15]. The key contributions of our work are: use of *multiple* ontologies developed independently of the information management system; and support for *incremental* query expansion based on semantic relationships in a manner that minimizes the loss of information.

The rest of the paper is organized as follows. In Section 2 we discuss in detail the role of domain ontologies in re-using and organizing knowledge. Section 3 introduces the global framework and the main steps of the query processing. In Section 4 we explain in detail the incremental query expansion to multiple ontologies. Conclusions are presented in Section 5.

2 Re-use and Organization of Knowledge using Domain Ontologies

In this section we discuss first the use of ontologies in describing information repositories, and second, the definitions of the *mapping information* that links ontologies to underlying data repositories.

2.1 Ontological Commitments for specifying agreements

An ontology has been defined as "a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents" [7]. From our point of view, an ontology is a set of terms of interest in a particular information domain and the relationships among them. Ontologies and the interontology relationships between them are created by experts in the corresponding domain. As indicated earlier, we express ontologies using DL descriptions. These descriptions are organized as a lattice and may be considered as *semantically rich metadata capturing the information content of the underlying data repositories.*

We advocate an approach based on multiple ontologies as opposed to a global integrated ontology as managing a global ontology involves problems of administration/maintenance, consistency and efficiency. A very large ontology may also be very hard for a user to navigate and comprehend. However, different ontologies described using different vocabularies can satisfy users needs in a better way and problems of consistency and efficiency get reduced.

Different ontologies are not completely orthogonal, however. Nor is it likely that a user's information need is satisfied by accessing the data repository accessible through mappings associated with a single ontology. To support this, ontologies are virtually

linked by interontology relationships. These relationships can be used for the translation of user queries from one ontology into another as we will explain later.

Ontologies sharing a great similarity can be organized in clusters in order to make easier choosing the most appropriate ontology for our needs. Clusters can represent general knowledge areas like "Animals", "Libraries", "Arts", etc. Anyway, there can exist, and it is very frequently the case, relationships among ontologies in different clusters in the same way that experts in one area sometimes need information managed by people in a different area. Since some clusters can be more general than others, they could be organized in hierarchies. More detailed comments about how to create clusters are out of the scope of this paper.

The ontologies we have used in our prototype can be found in [10][2]. All of them describe different data repositories containing bibliographical references (they all belong to the cluster that we call "Libraries") and have been developed independently by different organizations. The semantic relationships defined between them can also be consulted in the same place.

2.2 Mapping Information: Links to Data repositories

Each term in an ontology has associated a *Mapping Information* that relates such a term to data structures in the underlying data repositories. Our approach involves using different ontologies that are linked by semantic relationships, and where each ontology describes a small set of data repositories. In an alternative approach [9], mappings are maintained between terms in each ontology and data structures of *all* the related repositories. The main disadvantages of the latter approach (redundancy, complexity), and the advantages of the use of component ontologies on top of each data repository (extensibility) are described in [13].

The mapping information is represented as a tuple which involves the use of Extended Relational Algebra (ERA) expressions [4]. These mappings play a key role in encapsulating the heterogeneity due to different formats and organization of the data in the various repositories. The mappings subscribe to the idea of viewing a data repository as a set of entity types and attributes (or relations and attributes in ERA), independently of the concrete organization of the data in the repository. This gives an homogeneous view of the description of the data repositories without capturing any characteristic specific to the individual data repositories. Our representation is expressive enough to capture the complex associations of *concepts* and *roles*[3] with entity types and attributes.

The grammar and the formal definition of the mapping information used and its application to relational databases can be found in [5]. The mappings defined for ontologies in our prototype can be found in [10].

Logical Schemas, Data repositories and Data Sources

From the point of view of mapping expressions, a data repository is seen as a set of entity types and attributes that describe a logical schema. Mappings act as an intermediary language between Description Logics expressions and the concrete query languages of the local repositories. Only when data has to be accessed do we tackle the problem of

[2]WN, Stanford-I and Stanford-II ontologies are based on subsets of WordNet and Bibliographic Data (DARPA Knowledge Sharing Effort).

[3]Concepts and roles are the data elements in Description Logics; by 'terms' we mean both kind of data elements.

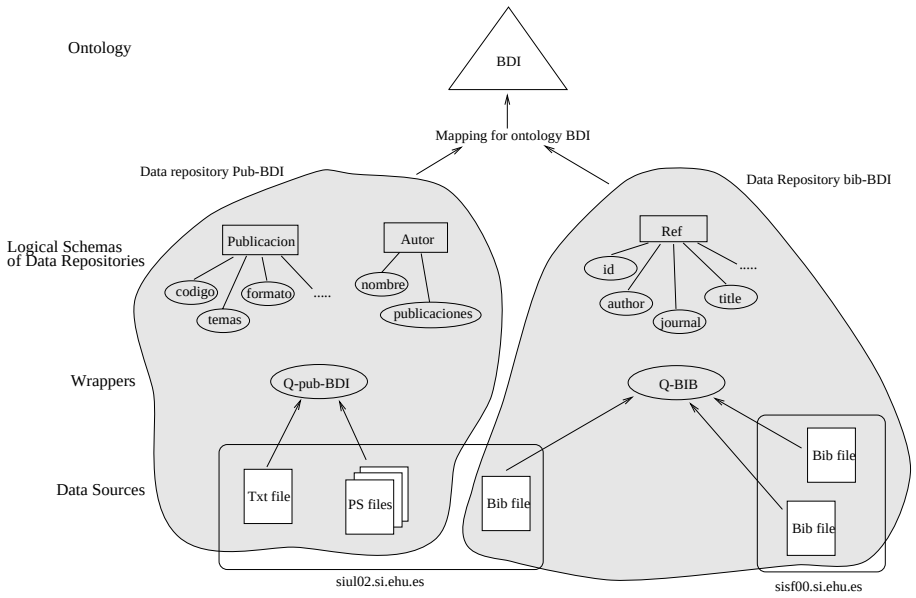


Figure 1: Data repositories, wrappers and data sources for ontology BDI

the specific data organization of the repository. We view a data repository as a data pool with a specific data organization. A data repository may or may not have a manager. It can be composed of several data sources which actually store data. Each data source can also be distributed. The simplest data source is a system file. It would be possible to speak of a data organization composed by a plain file and a database if each of those two elements needs the other to make sense. Almost everything can be a data repository– a set of files of different format, an HTML page, a database (which may have a DBMS as a manager), and any combination of them.

A *wrapper* is a module which understands a specific data organization of a repository. It knows how to retrieve data from the underlying repository and hide the specific data organization to the rest of the Information System. For repositories without a manager the wrapper has to access data sources directly. It is necessary to have a wrapper for each different data organization but the same wrapper can be used for accessing different data repositories with the same organization. Extensive work has been performed on generating/designing and using wrappers; hence we do not discuss this aspect of our work which is not particularly novel. An example of all these different levels existing under one of the ontologies of our prototype (BDI) appears in Figure 1.

3 Access to data repositories under an ontology

In this section we present first the global framework and then the query processing strategy.

3.1 Architecture of OBSERVER

The architecture of the OBSERVER system that provides incremental answers to user formulated queries in a Global Information System was explained in detail in [13] so we present the main modules here briefly (see Figure 2).

Query Processor. It takes as input a user query expressed in DL using terms from a chosen *user ontology* and access the underlying data stored in the repositories under

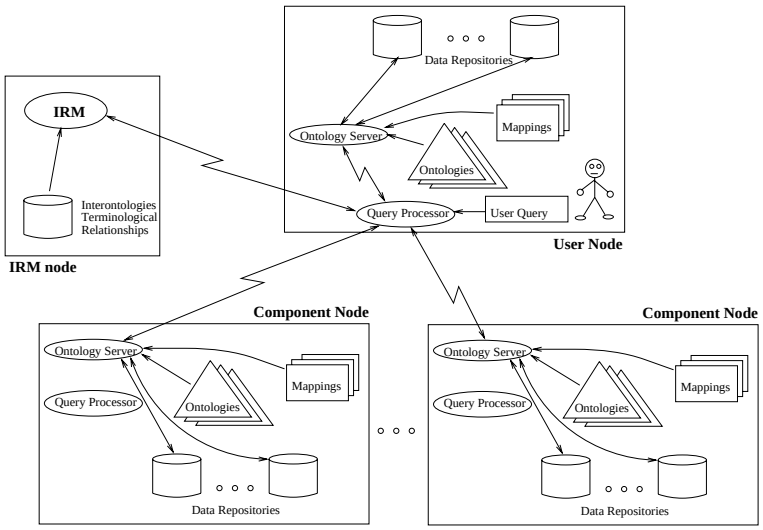


Figure 2: OBSERVER Global Architecture

the user ontology. If the user is not satisfied with the answer, the query is translated into the "language" of another (target) ontology by utilizing predefined terminological (interontology) relationships between the user and the target ontology. This process, explained in detail in Sections 3.2 and 4, is repeated until the answer satisfies the user.

Ontology Server. The Ontology Server provides intensional as well as extensional information about the ontologies residing in his node.

Interontology Relationships Manager (IRM). Interontology relationships relating the terms in various ontologies are represented in a declarative manner in an independent repository. This enables a solution to the *vocabulary sharing problem.*

Ontologies. As said before, each ontology is a set of terms of interest in a particular information domain; in our work, such terms are expressed using DL. They are organized as a lattice and may be considered as semantically rich metadata capturing the information content of the underlying data repositories.

3.2 Query Processing in OBSERVER

The main steps of query processing in OBSERVER are (Figure 3): *Query Construction*, *Access Underlying Data* and *Controlled and Incremental Query Expansion to a New Ontology.* In this section we describe briefly the first two steps. The controlled and incremental query expansion to a new ontology is the key to enhancing the scalability of query processing and is discussed in detail in the next section. We use the following example query to illustrate the various steps of our approach.

'Get title and number of pages of books written by Carl Sagan'

Query Construction

The two main tasks in this step are:

1. Select User Ontology. The user browses the available clusters of ontologies and chooses one. Then, the user navigates the ontologies in the cluster in order to select one *user ontology* that contains terms to express the semantics of her/his

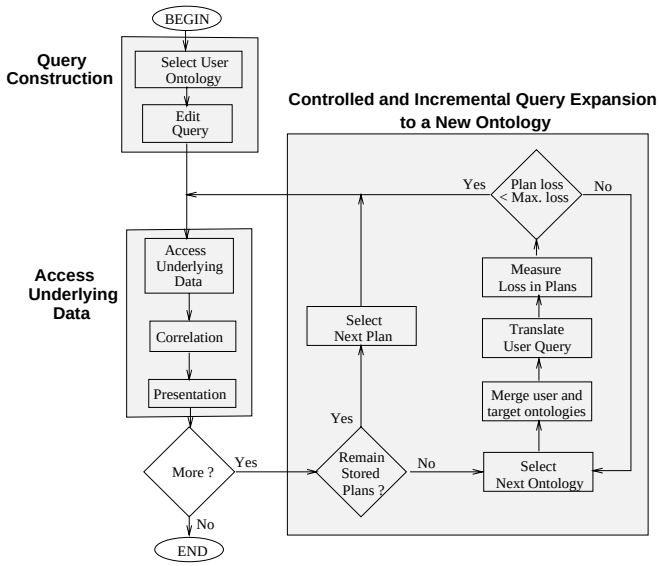


Figure 3: Detailed Query Processing in OBSERVER

information needs. For the example query, the WN ontology (see [10]) is selected since it contains all the terms needed to express the semantics of the query, i.e., terms that store information about titles ('name'), number of pages ('pages'), books ('book') and authors ('creator').

2. Edit Query. The user chooses terms from the user ontology to build the constraints and projections that comprise the query: '[name pages] for (**AND** book (**FILLS** creator "Carl Sagan"))'.

In general we assume that users formulate queries over only one ontology. OBSERVER incorporates a mechanism to automatically transform queries formulated over multiple ontologies into queries formulated over only one ontology [12].

Access Underlying Data

The Query Processor invokes the Ontology Server that manages the user ontology. The Ontology Server uses relevant *mapping information* ([5, 11]) that relates terms in the ontology to data structures in data repositories underlying such an ontology. Since mappings are defined for terms and queries are considered as concepts in DL systems, OBSERVER also uses a mechanism to combine mappings of terms to obtain the mappings of the whole query. With the help of the corresponding wrappers the underlying data are retrieved. After that, the Ontology Server correlates the information coming from different repositories and returns the result to the Query Processor which presents the new answer to the user. For brevity, the details of this step are not given.

4 Controlled and Incremental Query Expansion to Multiple Ontologies

Initially, a query is processed using the user ontology and associated repositories. If the user is not satisfied with the answer, the Query Processor incrementally retrieves more relevant data reachable through other ontologies in the system to "enrich" the answer.

Each incremental step involves selecting a new component ontology called the *target ontology*. In the example, we choose Stanford-I as target ontology to demonstrate the process.

Once a target ontology is selected, the user query has to be expressed in terms of that target ontology. For that task, the user and target ontologies are integrated automatically *wrt* the interontology relationships stored in the IRM at that time. During this process the user query is rewritten in terms of the integrated ontology "language". If all the terms in the user query have not been rewritten in this process then the translation is called a *partial translation*. In that case, either that partial translation is combined with partial translations obtained when visiting other target ontologies, or each "conflicting" term (a non translated term) is substituted by the intersection of its immediate parents or by the union of its immediate children until the whole query is completely expressed in terms of the target ontology.

4.1 Integration of the user and target ontologies

Two types of relationships are considered to integrate the user and target ontologies.

1. Synonym, hyponym and hypernym relationships between terms in the user ontology and terms in the target ontology.

2. Synonyms, hyponyms and hypernyms in both the user ontology and the target ontology.

The first three types of relationships are those stored in the IRM repository. The second three types are relationships between terms in the same ontology; synonyms are equivalent terms, hyponyms are terms subsumed by a given term and hypernyms are terms that subsume a given term. We use the above two kinds of relationships to integrate the user and target ontologies by using the deductive power of the DL system. Thus no user intervention is required. The integration process is as follows:

Initial definitions:
$\mathcal{T} = Ont_{user} \bigcup Ont_{target}$
$\mathcal{R} =$ { semantic relationships between Ont_{user} and Ont_{target} stored in the IRM }
t^{descr} = Description of term t
Redescribe-Term(t, new-descr) substitutes the description of t by the new one
Delete-Term(t) removes t from all the descriptions in which it appears and then removes the term t

1. Renaming of terms in user and target ontology:
   $\forall t \in Ont_{user} \Longrightarrow$ Rename-term(t, $Ont_{user}\#t$)
   $\forall t \in Ont_{target} \Longrightarrow$ Rename-term(t, $Ont_{target}\#t$)

2. Rewriting of term descriptions based on semantic relationships:
   $\forall r \in \mathcal{R}$

   (a) If $r \equiv t_1$ synonym t_2 $(t_1, t_2 \in \mathcal{T}) \Longrightarrow$
   $\forall t_1/t_1 \in t^{descr} \Longrightarrow$ Redescribe-Term (t, (**AND** t^{descr} t_2))
   Delete-Term(t_1)[4]

[4]We are only interested in terms in the target ontology as we are looking for substituting conflicting terms of the user ontology by terms in the target ontology.

(b) If $r \equiv t_1$ hyponym t_2 $(t_1, t_2 \in \mathcal{T}) \Longrightarrow$ Redescribe-Term $(t_1,$ (**AND** t_1^{descr} $t_2))$

(c) If $r \equiv t_1$ hypernym t_2 $(t_1, t_2 \in \mathcal{T}) \Longrightarrow$ Redescribe-Term $(t_2,$ (**AND** t_2^{descr} $t_1))$

3. Assertion of updated terms from user and target ontologies in the DL system

Although some of the semantic relationships can be redundant, the DL system will classify the terms at the right place in the integrated ontology. As we use the deductive features of DL systems we avoid defining new *costly deductive algorithms* to determine the immediate hyponyms and hypernyms of a term. Rewritting terms by adding other DL constraints would enable representations of relationships other than synonymy, hyponymy and hypernymy.

Apart from the complexity of the third step, which depends on the specific DL system, the complexity of this algorithm is $\mathcal{O}(kn)$ where $k = \mid \mathcal{R} \mid$ and $n = \mid \mathcal{T} \mid$.

Studies about the performance of the DL systems [16] show that it is possible to integrate two ontologies of around a thousand terms in less than a minute. In our example, the process of integrating in CLASSIC [2] (Lisp version) the ontologies WN (73 terms) and Stanford-I (50 terms) taking into account 18 interontology relationships takes less than a second. Ontologies describing specific domains, as opposed to a global ontology, are not expected to be huge since knowledge is distributed among several ontologies and combined when needed by our system. Thus the integration can be performed at run time and has a positive impact on the scalability of our query processing strategy.

The result of integrating the WN and Stanford-I ontologies by applying the relationships defined between them is shown in Appendix B. Terms without parents in the figure are actually the children of *Anything* (the top term in any ontology expressed in DL). Terms from the ontology WN are in uppercase and terms from Stanford-I are shown in lowercase. The user query has also been rewritten (Q='*(***AND** *BOOK (***FILLS** *doc-author-name "Carl Sagan"))*') and classified in the right place by the DL system . Notice that in the case of synonyms, (document, PRINT-MEDIA), (journal, JOURNALS), (newspaper, NEWSPAPER), (magazine, MAGAZINE) and (doc-author-name, CREATOR), only the terms from Stanford-I (those in lower case) appear because we are translating from the WN (user ontology) into the Stanford-I (target) ontology.

Hereafter, the Query Processor will only deal with the integrated ontology since it contains all the needed information.

4.2 Plans with no loss of information

A complete translation of the user query into the terms of the target ontology can be achieved in one of the following ways:

1. Using synonym relationships. When the user and target ontologies are integrated, all the terms in the user query may be translated by their corresponding synonyms. Consider the user query formulated over the WN ontology '*[PAGES] for (***AND** *MAGAZINE (***FILLS** *NAME "Time"))*', where the user is interested in the number of pages of the magazine "Time". If Stanford-I is chosen as the target ontology, the new description of this query in the integrated ontology would be '*[number-of-pages] for (***AND** *magazine (***FILLS** *title "Time"))*'. All the terms in this query are from the target ontology as there exist corresponding synonyms defined in the IRM. There is no need for traversing the hyponym and hypernym relationships and the plan incurs no loss of information.

2. Combining partial translations. After integration, the user query may not be completely translated (some terms in the query do not have synonyms in the target ontology). In this case, the partial translation obtained could be combined with partial translations into other target ontologies to get a new plan with no loss of information. Consider the user query formulated over the Stanford-II ontology '*(AND doctoral-thesis-ref (FILLS keywords "metadata") (ATLEAST 1 publisher))*', formulated for the retrieval of the available doctoral thesis about metadata with at least one publisher. Partial translations into Stanford-I and LSDIS ontologies obtained using only synonyms are:

   - (AND doctoral-thesis (ATLEAST 1 publisher)), where the constraint about keywords could not be translated into Stanford-I.
   - (AND publications (FILLS type "doctoral" "thesis") (FILLS subject "METADATA")), where the constraint about publisher could not be translated into LSDIS.

   In Appendix A we present the proof of the following theorem:
   Theorem: *Given a user query $\mathcal{Q}$ and a set of partial translations of that query, if the intersection of the non-translated parts is empty then the intersection of the objects of the translated parts will satisfy all the constraints in $\mathcal{Q}$.*

   In the previous example, this theorem is satisfied, therefore one of the generated plans (with no loss of information) when translating into LSDIS would be:

   $<$ Objects((AND doctoral-thesis (ATLEAST 1 publisher)), Stanford-I) $\bigcap$ Objects((AND publications (FILLS type "doctoral" "thesis") (FILLS subject "METADATA")), LSDIS) , 0 $>$

   This checking can be performed in parallel with a translation process (explained later) that uses hyponym and hypernym relationships and incurs a loss of information as these relationships result in changing the semantics of the query. The reason to deal with loss of information is that sometimes there may not exist synonym relationships between terms in independently developed ontologies. In that case, it is better to return some information with an estimate of information loss rather than no information at all.

   In [13], we presented an algorithm which, given a new partial translation, tries to determine whether it can be combined with any combination of the previously obtained partial translations. If the number of constraints of a given user query is k, the previous algorithm will never construct combinations of more than k-1 elements/partial translations since only non-redundant combinations[5] are considered. This reduces the explosion of the search space. So the algorithm returns a list of minimal combinations which are equivalent to full translations (the combinations returned satisfy the theorem). If k is the number of constraints in the user query and i is the number of partial translations found[6], the complexity of the algorithm is the following:

$$complexity = \begin{cases} O(k2^i) & 1 \leq i \leq k \\ O(ki2^k) & i > k \end{cases}$$

[5]Each partial translation of a combination translates at least one constraint of the user query that the others in the same combination do not.

[6]Remember that the system obtains a new (partial) translation each time it translates the user query into a new target ontology.

This means that the algorithm is exponential only in the first k ontologies visited. Notice that the number of ontologies available in a Information System can be huge but the number of constraints in a query is usually a small number, for example, less than ten.

4.3 Plans with loss of information

After translation of the user query into the integrated ontology, there may be terms of the user ontology for which there did not exist synonyms into the target ontology. Each conflicting term in the user query is then replaced by the intersection of its immediate parents or by the union of its immediate children. This method is applied recursively until a translation of the conflicting term is obtained using only the terms of the target ontology. It may be noted that it is always possible to get at least one full translation of any conflicting term in both the directions since the terms *Anything* and *Nothing*[7] exist in any ontology.

For each visited term the system stores its plans/translations so the possibilities of each term are explored only once. The complexity of this algorithm is $\mathcal{O}(e)$ where e is the number of edges in the integrated ontology where the translation is performed.

Traversing hypo and hypernym relationships as described above can result in several possible translations for each conflicting term. All the possibilities are explored and the result is a list of tuples in the format $< Plan, Loss >$, where *Plan* is a DL expression using only terms from the target ontology, and *Loss* is a number between 0 and 100 representing the percentage loss of information of *Plan* with respect to the original user query. All the possible plans are evaluated and the loss of information incurred by each plan is estimated [6]. Redundant plans are removed following this rule:

$$< Plan1, Loss1 > \subset < Plan2, Loss2 > \Leftrightarrow Plan1 \subset Plan2 \wedge Loss1 > Loss2$$

Based on the above rule, $< Plan1, Loss1 >$ will be eliminated. If the first condition is not satisfied, *Plan1* could bring new relevant objects; if the second one is not satisfied, *Plan1* will be chosen before *Plan2* as it has less loss. After the removal of redundant plans, the one with less loss is chosen to access new relevant data. The rest of the plans are stored and used if the user wants more data.

In Appendix C we present the steps given to translate a conflicting term ('BOOK') of the query Q that appears in Appendix B into the language of the target ontology. The computation of the loss [6] incurred in each plan obtained is calculated based on extensional information of underlying data repositories combined by metrics like *precision* and *recall*. A more detailed description is not feasible due to space limitations.

5 Conclusions and Future Work

The data/information explosion on the Global Information Infrastructure (GII) has left most of us with information overload problems. Moreover, the problem is only bound to increase in severity. In this paper, we have proposed an approach based on re-use, organization and integration of knowledge to help tackle the problem of information overload on the GII.

[7]The term *Anything* and *Nothing* are the top and the bottom, respectively, of any ontology defined in a system based on Description Logics.

One important underlying reason for the information overload is the differences in world views and vocabularies used by the people providing information (information providers) and the people desiring information (information users) to describe the information. We propose an approach based on the use of multiple domain specific ontologies to tackle this problem. We invoke the property of an **ontological commitment** associated with a domain ontology to help bridge the communication gap between the information provider and the user.

Thus our approach predicates the use of domain specific ontologies to export and query information in the GII. Admittedly, it is infeasible to expect the millions of users and providers to align themselves to a common set of ontologies. In order to make this approach more feasible we propose the following:

- Use of pre-existing (independently developed) real world domain specific ontologies to describe information on the GII.
- Organization of information in terms of domain specific ontological concepts. Mechanisms and techniques enabling the mapping of ontological concepts to data structures in the underlying repositories and retrieval of data corresponding to information requests using concepts were discussed in the paper.
- Support for multiple domain specific ontologies to enable the use of different world views and vocabularies by information providers and users. Mechanisms for translating information requests across different ontologies involving knowledge integration and the ability to combine and reformulate (partial) translations were also discussed in the paper. This provides an automatic way of determining the relevance of an ontology and its underlying repositories.

We believe that Knowledge Representation Technology has a key role to play in handling the information explosion on the GII and especially issues related to re-use, organization and integration of knowledge will be of critical importance. The approach presented by us in the paper has been implemented in the OBSERVER project thus demonstrating its feasibility.

Acknowledgements

We would like to thank Jesús Bermudez his help in the calculus of the complexity of the algorithms.

References

[1] Y. Arens, C.A. Knoblock, and W. Shen. Query reformulation for dynamic information integration. *Journal of Intelligent Information Systems*, 6(2-3):99–130, 1996.

[2] A. Borgida, R.J. Brachman, D.L. McGuinness, and L.A. Resnick. CLASSIC: A structural data model for objects. In *Proceedings ACM SIGMOD-89, Portland, Oregon*, 1989.

[3] R. Brachman and J. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9:171–216, 1985.

[4] R. Elmasri and S. Navathe. *Fundamentals of Database Systems.* Benjamin/Cummins, 1989.

[5] A. Goñi, J.M. Blanco, and A. Illarramendi. Connecting knowledge bases with databases: a complete mapping relation. In *Proc. of the 8th ERCIM Workshop. Trondheim, Norway*, 1995.

[6] A. Goñi, E. Mena, and A. Illarramendi. Querying heterogeneous and distributed data repositories using ontologies. In *Proceedings of the 7th European-Japanese Conference on Information Modelling and Knowledge Bases (IMKB'97), Toulouse (France)*, May 1997.

[7] T. Gruber. http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.

[8] V. Kashyap and A. Sheth. Semantic and schematic similarities between database objects: A context-based approach. *The International Journal of Very Large Data Bases (VLDB)*, 5(4), December 1996.

[9] A.Y. Levy, D. Srivastava, and T. Kirk. Data model and query evaluation in global information systems. *Journal of Intelligent Information Systems*, 5(2):121–143, September 1995.

[10] E. Mena. http://siul02.si.ehu.es/~jirgbdat/OBSERVER/ontologies.html.

[11] E. Mena, A. Illarramendi, and J. M. Blanco. Magic: An interface for generating mapping information between object-based and relational systems. In *Information Systems Design and Hypermedia*. Cepadues-Editions, 1994.

[12] E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. Scalable query processing in dynamic and open environments: An approach based on information brokering across domain ontologies. Technical report. To be published by LSI department, University of the Basque Country (SPAIN), 1997.

[13] E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. In *Proc. of the First IFCIS International Conference on Cooperative Information Systems (CoopIS'96), Brussels (Belgium), June*. IEEE Computer Society Press, 1996.

[14] S. Milliner and M. Papazoglou. Scalable information elicitation in large heterogeneous database networks. To be published in IEEE Internet Journal, 1997.

[15] R. Bayardo, W. Bohrer, R. Brice, A. Cichocki, G. Fowler, A. Helai, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. Infosleuth: Semantic integration of information in open and dynamic environments. In *Proceedings of the 1997 ACM International Conference on the Management of Data (SIGMOD), Tucson, Arizona.*, May 1997.

[16] P.H. Speel. *Selecting Knowledge Representation Systems.* PhD thesis, University of Twente, Enschede, the Netherlands, 1995.

A Proof of the theorem for combining partial translations

Initial definitions

- User query:

$$\mathcal{Q} = \{Q_1 \dots Q_n\} \qquad Q_i \; constraints$$

- Partial Translation: ($\equiv$ means "semantically equivalent")

$$\mathcal{P} =< \mathcal{P}_{\mathcal{T}}, \mathcal{P}_{\mathcal{NT}} > \left\{ \begin{array}{ll} P_T = \{P_{T_1} \dots P_{T_i}\} & P_{T_k} \equiv Q_m \\ P_{NT} \subseteq \mathcal{Q} & \forall Q_i \in P_{NT} \Longrightarrow \nexists P_{T_j} \in P_T / P_{T_j} \equiv Q_i \end{array} \right.$$

$$P_T \bigcup P_{NT} \equiv \mathcal{Q}$$

- Full translation:

$$\mathcal{P} =< \mathcal{P}_{\mathcal{T}}, \phi > \qquad \mathcal{P}_{\mathcal{T}} \equiv \mathcal{Q}$$

- Extension:

$$Objects(\mathcal{C}) = \{o \mid C_i(o) \; \forall i \; C_i \in \mathcal{C}\} \qquad C_i \; constraint$$

Theorem: Given a user query $\mathcal{Q} = \{Q_1, \dots, Q_n\}$ and a set of partial translations of that query, say $\mathcal{P} = < P_T^1, P_{NT}^1 >, \dots, < P_T^i, P_{NT}^i >$ if the intersection of the non-translated parts is empty then the intersection of the objects of the translated parts will satisfy all the constraints in $\mathcal{Q}$.

$$\bigcap_{1 \le j \le i} P_{NT}^j = \phi \Longrightarrow \bigcap_{1 \le j \le i} Objects(P_T^j) = Objects(\mathcal{Q})$$

Proof:

$$(\mathbf{1}) \bigcap_{1 \le j \le i} P_{NT}^j = \phi \wedge o \in \bigcap_{1 \le j \le i} Objects(P_T^j) \Longrightarrow o \in Objects(\mathcal{Q}) \;\; ?$$

Let's suppose

$$\bigcap_{1 \le j \le i} P_{NT}^j = \phi \wedge \exists o \in \bigcap_{1 \le j \le i} Objects(P_T^j) \wedge o \notin Objects(\mathcal{Q})$$

$$o \notin Objects(\mathcal{Q}) \Longrightarrow \underbrace{\exists Q_k \in \mathcal{Q} \mid o \notin Objects(Q_k)}_{(\spadesuit)}$$

$$\exists o \in \bigcap_{1 \le j \le i} Objects(P_T^j) \Longrightarrow o \in Objects(P_T^j) \; \forall j \Longrightarrow \underbrace{o \in Objects(P_{T_m}^j)}_{(\spadesuit\spadesuit)} \forall j \forall m \; P_{T_m}^j \in P_T^j$$

$$\left. \begin{array}{c} (\spadesuit) \\ (\spadesuit\spadesuit) \end{array} \right\} \forall j \; \nexists P_T^j \mid Q_k \in P_T^j \underbrace{\Longrightarrow}_{P_T \cup P_{NT} \equiv \mathcal{Q}} \forall j \; Q_k \in P_{NT}^j \Longrightarrow Q_k \in \bigcap_{1 \le j \le i} P_{NT}^j \qquad \#$$

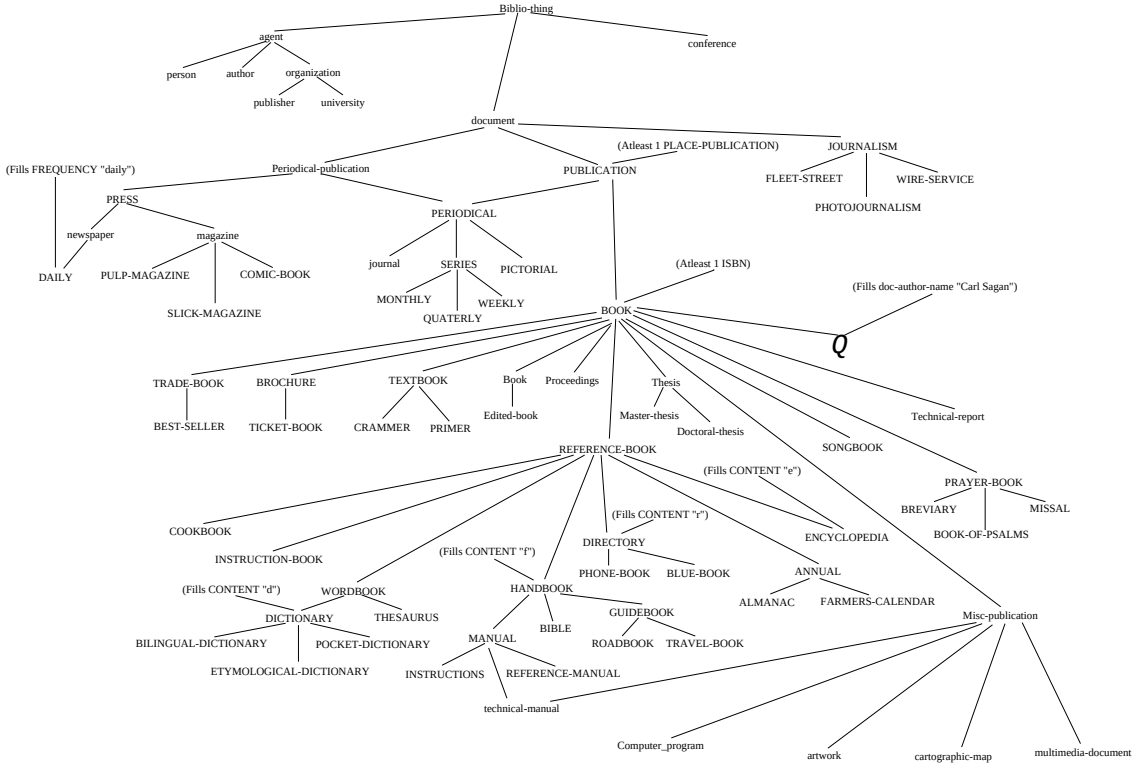
$$(\mathbf{2}) \; o \in Objects(\mathcal{Q}) \Longrightarrow o \in \bigcap_{1 \le j \le i} Objects(P_T^j) \;\; ? \Longleftrightarrow$$

$$\Longleftrightarrow o \in Objects(\mathcal{Q}) \Longrightarrow o \in Objects(P_T^j) \; \forall j \;\; ?$$

$$Taken \; any \; j, \; o \in Objects(P_T^j) \;\; ? \equiv o \in Objects(P_{T_k}^j) \; \forall k \mid P_{T_k}^j \in P_T^j \;\; ?$$

$$\left. \begin{array}{r} Taken \; any \; P_{T_k}^j \equiv Q_k \;\; 1 \le k \le n \\ If \; o \in Objects(\mathcal{Q}) \Longrightarrow o \in Q_m \forall m \;\; 1 \le m \le n \end{array} \right\} o \in P_{T_k}^j \;\; \forall k \forall j \qquad \square$$

B Integration of WN and Stanford-I ontologies



C Steps for translation into the target ontology

We illustrate with the help of a diagram the various steps involved in replacing the term 'BOOK' in the WN ontology by terms from the Stanford-I ontology. Arrows illustrate the direction in which the term/query is expanded. The iterative accumulation of all the plans is also illustrated.