OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies

E. Mena[1,2*] V. Kashyap[1,3] A. Sheth[1] A. Illarramendi[2]
[1]LSDIS Lab, http://lsdis.cs.uga.edu, Univ. of Georgia, Athens, GA 30602
[2]Facultad de Informática, Universidad del País Vasco. San Sebastián. Spain
[3]Dept. of Computer Science, Rutgers University, New Brunswick, NJ 08903

Abstract

The huge number of autonomous and heterogeneous data repositories accessible on the "global information infrastructure" makes it impossible for users to be aware of the locations, structure/organization, query languages and semantics of the data in various repositories. There is a critical need to complement current browsing, navigational and information retrieval techniques with a strategy that focuses on information content and semantics. In any strategy that focuses on information content, the most critical problem is that of different vocabularies used to describe similar information across domains. We discuss a scalable approach for vocabulary sharing. *The objects in the repositories are represented as intensional descriptions by pre-existing ontologies expressed in Description Logics characterizing information in different domains. User queries are rewritten by using interontology relationships to obtain semantics-preserving translations across the ontologies.*

1. Introduction

We are witnessing today an exponential growth of data accumulated within universities, corporations and government organizations. Autonomous repositories that store different types of digital data in multiple formats are becoming available for use on the fast evolving global information infrastructure. The resulting information overload makes it impossible for users to be aware of the locations, organization/structure, query languages and semantics of the data in various repositories. One classification of various approaches for query processing in global information systems is as follows:

- Syntactic keyword-based and navigational approaches in which the query is a set of keywords. There are little or no semantics associated with this approach and the user has to do most of the information filtering and correlation. However, this approach is simple to use and support.

- A global (common) ontology-based approach which supports expression of complex constraints as a part of the query. This involves development and integration of domain-specific ontologies into a common global ontology and partitioning it into micro-theories. This approach transfers the burden of information correlation and filtering on the query processing system. However it can be very difficult to support because of the complexity involved in integrating the ontologies and maintaining consistency across concepts (originally) from different ontologies.

- A group of "loosely coupled" approaches advocated in the paper, where instead of integrating the *pre-existing* ontologies, interoperation across them is achieved via **terminological relationships** represented between terms across the ontologies. We expect answers to be inferior (*wrt* the previous approach) as we *approximate* semantic relationships using terminological ones, but loosely coupled approaches are *scalable, extensible* and easier to support.

Browsing and navigation tools available on the WWW [3] belong to the first group and include among many others, *WAIS* [12], *Archie* [8] and *Gopher* [18]. However, these tools require the user to be aware of the possible locations (URLs) where they might be able to find relevant information. An important next step should be to support location and repository-independent queries. Early steps in this direction are implemented using associative access in [28]. In *Nomenclator* [24], **metadata** about the various repositories is cached to help constrain the search space for a query. The

*This work was supported in part by a grant of the Basque Country Government and was mostly performed at the LSDIS Lab as a part of the InfoQuilt project.

Rufus [30] and the *InfoHarness*[1] [29] systems use automatically generated metadata to access and retrieve heterogeneous information independent of type, representation and location. An approach using a global ontology divided into micro-theories is discussed in [7].

We extend or build upon some of the above approaches by using metadata to capture the *information content* of the repositories. We represent intensional descriptions to *abstract from* the structure and organization of the individual repositories as **intensional metadata**. The user queries the system by expressing his information needs using intensional metadata descriptions represented using Description Logics (DLs) [6], in our case, *CLASSIC* [5].

The most critical problem in characterizing the information content is that of different vocabularies used to describe similar information across domains. This leads to different terms[2] and constraints being used to characterize similar information. Interoperation across ontologies is achieved by traversing *semantic* relationships defined between terms across ontologies. User queries are rewritten in a semantics-preserving manner by replacing them with synonym terms from different ontologies; hyponym and hypernym terms can also be used and the loss of information measured.

The key objective of our approach is to reduce the problem of knowing the structure and semantics of data in the huge number of repositories in a global information system to the significantly smaller problem of knowing the synonym relationships between terms across ontologies.

We have developed a prototype system which supports querying of real-world repositories. Some data has been down-loaded into local databases, some are in plain files and others are remote repositories accessed on-line through WWW supported forms. The queries are constructed using terms from one of the pre-existing ontologies available on the WWW. Section 2 discusses our approach, the associated architecture and a motivating example. We discuss at a high level the basic elements of the architecture– *the Query processor*, the *Ontology Server* and the *Interontology Relationships Manager (IRM)*. We also discuss in this section the ontologies and their underlying repositories. In Section 3 we discuss the query processing steps, such as translation of the query, data access and correlation. In Section 4, we present the conclusions and future work.

2. A motivating example

In this section, we discuss a concrete query example that explains the general problems of query processing in a global information system. Answers to queries similar to the example given below can be obtained from our prototype. Subsequent subsections discuss our architecture and general approach to solve some of the problems outlined.

We have chosen the domain of bibliographic information as a test case for our prototype query processing system. Consider the following query which we will use as a running example to illustrate the various issues and solutions.

'Get the titles, authors, documents and the number of pages of doctoral theses dealing with "metadata" and that have been published at least once.'

- **Resource Discovery.** The user has to first locate the repositories relevant to the query, (e.g., which repositories are likely to have information about doctoral theses?).
- **Structure/Format Heterogeneity.** Different repositories may have different data organizations (e.g. relational database, file system), formats and media (e.g., 'document' values are Postscript documents), and may support different applications and query languages.
- **Modeling of Information Content.** We represent queries/information as a conjunction of constraints expressed using DLs. The information may be modeled such that it may not be possible to evaluate some of the constraints (e.g., the information about the number of pages may not be modeled even though doctoral theses are modeled).
- **Querying of the Information Content.** Using constraints in DLs to express a query enable us to capture information content as opposed to checking for the presence or absence of keywords or a limited form of concept match. In the latter case, if keywords do not appear in the document it will not be retrieved even though it may be relevant (e.g. the word "automobile" may not appear in a document describing cars).
- **The Vocabulary Problem.** Current Internet tools and query processing systems are unable to support heterogeneous vocabularies used to describe the same information. In the case of keyword-based systems, if a synonym of the keyword present in the document is used as a part of the query, the document may not be retrieved. When we attempt to capture and query the answers in an intensional manner, different but related terms may be used to describe similar information at the intensional (e.g. the term for "pages" may be modeled as "leaves" at a different ontology) as well at the extensional level (e.g. semantically heterogeneous keys such as SS# and Employee No. may be used to identify instances at different repositories).

[1] InfoHarness is a trademark of Bellcore and is now available in commercial form as Adapt/X Harness.

[2] In this paper we shall use "terms" to mean "concepts" as well as "roles".

The problems relating to modeling and querying the information content is collectively referred to as the query processing problem in this paper. This will involve **information focusing** (determining the relevant information in a particular repository) and **information correlation** (combining relevant information from different repositories).

2.1. OBSERVER: An architecture for Global Information Systems

In this section we describe OBSERVER[3], an architecture for query processing in global information systems motivated by the problems discussed in the previous section. The basic elements of the architecture illustrated in Figure 1 are:

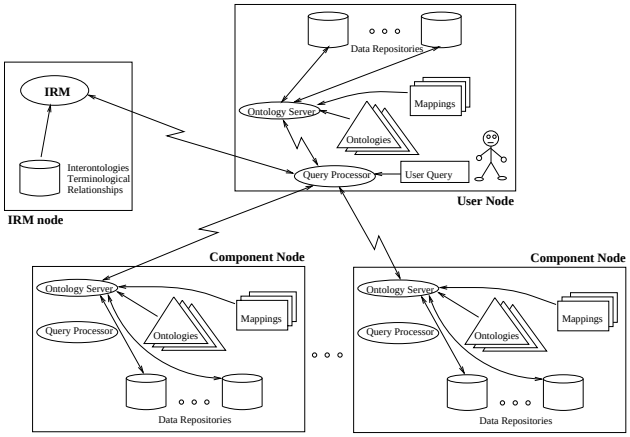

Figure 1. OBSERVER: An architecture to support Query Processing

- **Query Processor.** It takes as input a user query expressed in DLs using terms from a chosen *user ontology*. The query processor navigates other component ontologies of the global information system and translates terms in the user query into the component ontologies preserving the semantics of the user query. Our focus in this paper is supporting "semantically rich" queries in an environment where different vocabularies are used. The resulting (possibly partial) translation of the query at the component ontology enables identification of relevant information at the underlying data repositories providing a solution to the *information focusing* problem. It also combines the partial translations at the present ontology with those determined at previous ontologies such that all constraints in the user query are translated. Constraints not modeled at a particular ontology may be modeled at another. Thus the combination of translations provides a solution to the **information modeling** problem. The data corresponding to different constraints retrieved at different ontologies are then correlated to give the final answer (**information correlation**).

- **Ontology Server.** The Ontology Server provides information about ontologies to the Query Processor. It provides the definitions of the terms in the ontology and retrieves data underlying the ontology. **Mappings** that link each term in an ontology with structures in data repositories are combined in order to access and retrieve data from the repositories. This addresses the *structure/format heterogeneity* problem.

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

- **Ontologies.** Each Ontology is a set of terms of interest in a particular information domain, expressed using DLs in our work. They are organized as a lattice and may be considered as semantically rich metadata capturing the information content of the underlying data repositories. These semantically rich descriptions can be used to query the global information system providing a solution to the *querying information content* problem.

2.2. The Query Processing Approach

In this section we give a broad overview of our query processing approach based on the elements of the architecture described in the previous section. The main steps illustrated in Figure 2 are described below.

Step 1: *Connection to the system.* The user chooses and connects to one component ontology (referred to as *the user ontology*). This implies that the user subscribes to the terminology and the model of the domain as captured by the chosen ontology.

Step 2: *Query metadata construction.* Appropriate terms from the user ontology are chosen. The intensional query expressed in CLASSIC is constructed using a GUI.

Step 3: *Resource Discovery and Information Focusing.* The query is translated into terms in the component ontology using synonym[4] relationships (from

[3] *Ontology Based System Enhanced with Relationships for Vocabulary hEterogeneity Resolution.*

[4] Although our current work is related to hypernym and hyponym relationships space limitations prevent us from discussing them in this paper.

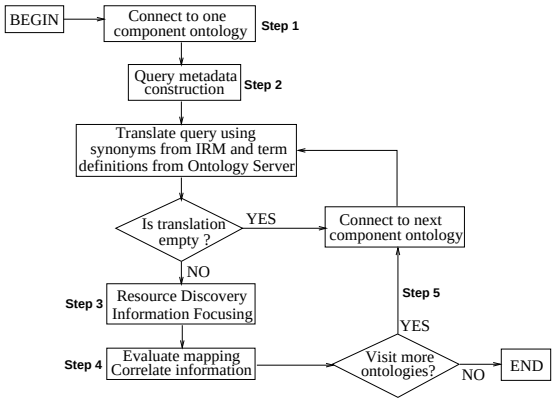

Figure 2. Query Processing for Global Information Systems: A high level approach

the IRM) and term definitions (from the Ontology Server). If there exists a (partial/complete) translation, then the repositories under that ontology are relevant. Furthermore, the query constraints translated at the component ontology enable identification of the relevant subset of the data in the repositories.

Step 4: *Information Access and Correlation*. If there is a complete translation into a particular ontology; or if the current partial translation in conjunction with previously generated partial translations are equivalent to the original query, the data is retrieved from the relevant ontologies and appropriately combined (following the query evaluation plan) to give the final answer.

Step 5: *Iteration*. If the user is not satisfied with the answer, (s)he can access new ontologies (steps 3 and 4 are repeated).

2.3. Component Ontologies: Motivation

There have been proposals of similar architectures where mappings are maintained between terms in the user ontology and data structures in the underlying repositories [2, 16]. We discuss how the use of component ontologies can help eliminate some disadvantages in our approach.

- **Redundancy.** If more than one user ontology has semantically equivalent terms, the same mapping will be defined more than once. In our approach, we represent *synonym* relationships between equivalent terms across ontologies. We thus need to define the mapping once.
- **Extensibility.** Every time there is a change in the structure of an underlying repository, the mappings to the associated user ontologies also need to be changed. In our case, all we need to do is modify the mappings of the component ontology that describes that concrete repository; the synonym relationships across ontologies remain the same.
- **Query Decomposition.** In the case of a direct connection between terms in the user ontology and the data structure of underlying repositories, the complexity and heterogeneity of the mappings is very high when a term is supported by several repositories. In our approach, the complexity in the mappings is replaced by the simplicity of representation of synonym relationships.

In section 2.3.1 we describe briefly the features of DL-based systems used to define the component ontologies. In sections 2.3.2, 2.3.3 and 2.3.4 we present the pre-existing real-word ontologies designed independently by researchers in linguistics and knowledge representation that we **reuse** after representing them in CLASSIC. The hierarchies of concepts can be found in Appendix B; the DL definitions of these ontologies have not been included due to space limitations.

2.3.1 CLASSIC: A system based on Description Logics

Systems based on DLs, also known as terminological systems, are descendants of *KL-ONE* [6]. Some systems based on DLs are *CLASSIC* [5] (used in our prototype), *BACK* [32], *LOOM* [17] and *KRIS* [1]. The main features of the DL systems are described below:

- The language contains unary relations called *concepts* which represent classes of objects in the domain and binary relations called *roles* which describe relationships between objects. Concepts and roles are created via *terminological descriptions* built from preexisting concepts, roles and a set of operators (ALL, ATLEAST, ATMOST, etc.). A distinguished role called *self* stores the id of each object belonging to each concept (in Section 3 the utility of such a role is described).
- *Primitive and defined terms*. Terms are *primitive* if their descriptions specify only the necessary conditions and are *defined* if their descriptions specify both the necessary and sufficient conditions.
- *Subsumption of concepts* allows to determinate whether a term is more general than another. The *subsumption* relation is exploited by the DL system to maintain a classification hierarchy/lattice of terms (which is useful in dealing with large collections of definitions) and to *classify* new terms as well as

queries[5]. This classification mechanism allows the system to detect *incoherent* and *disjoint* descriptions.

2.3.2 WN: A subset of WordNet 1.5

WN is an ontology we have built by re-using a part of the WordNet 1.5 ontology [21]. The concepts in the WN ontology are a subset of terms in the hyponym tree of the noun "print media" [22]. As no roles are defined in WordNet we had to define some (name, ISBN, type, pages, etc.). This is a case where we represent a *linguistic-based* ontology using a knowledge representation language: the concepts of WN ontology correspond to the nouns in WordNet 1.5 and the hyponym/hypernym relationships in WordNet 1.5 are modeled as subsumptions in WN. The underlying data are MARC [25] records from the University of Georgia Main Library stored in plain files.

2.3.3 Stanford-I and Stanford-II: ARPA Knowledge Sharing Effort

Two of our other ontologies, Stanford-I and Stanford-II, are subsets of the Bibliographic-Data ontology [10] developed as a part of the ARPA Knowledge Sharing Effort (http://www-ksl.stanford.edu/knowledge-sharing). The Stanford-II ontology corresponds to the sub-tree under the concept 'reference' of the Bibliographic-Data ontology. Stanford-I corresponds to the rest of the ontology. The data underlying Stanford-I are MARC records from the Library at Monterrey Institute of Technology stored and managed by the object-relational DBMS, Illustra. The data corresponding to Stanford-II is accessed directly through the Z39.50 Web gateway of the Library of Congress [23], so no data was downloaded locally. The format and data organization of the Library of Congress repository are unknown and irrelevant for our system. We thus take an *operational view* of this repository.

2.3.4 The LSDIS ontology

The LSDIS ontology is a "home-grown" ontology which represents our view of our Lab's publications. The data is composed of several text, HTML and Postscript documents of the LSDIS Lab (http://lsdis.cs.uga.edu/publications/) and is distributed over various files.

2.4 The Interontology Relationships Manager (IRM)

The IRM is the critical component which supports ontology-based interoperation. It also enhances the **scalability** of the query processing strategy by avoiding the need for: (a) designing a common global ontology containing all the relevant terms in the global information system; and (b) investing time and energy for the development of an ontology specific for your needs when "similar" ontologies are available. The main assumption behind the IRM is that **the number of relationships between terms across ontologies is an order of magnitude smaller than the number of all the terms relevant to the system**.

Hammer and McLeod [11] have suggested a set of relationship descriptors to capture relationships between terms across different (locally developed) ontologies. A set of terminological relationships have been proposed in [21]. In this paper we focus on the *synonym* relationships to represent when two terms in different ontologies have the same semantics[6]. The types of relationships will be extended in the future e.g., *hyponyms* and *hypernyms*. These will be consulted by the Query Processor to solve the vocabulary problem at the intensional level. Such relationships should be defined when a new ontology is added to the Information System. To address the vocabulary problem at the extensional level, transformer functions between roles of different ontologies can be defined in the IRM.

If the IRM repository becomes so large and its centralized nature discussed here affects the efficiency of the System, its independence *wrt* the system enables its partitioning or mirroring without affecting the rest of the system.

2.4.1 Services provided by the IRM

The IRM stores information about the component ontologies of the Global Information System. The following IRM services can be used by the Query Processors:

- **Get-ontologies()** returns the name of all the component ontologies of the Global Information System. For example, in our prototype system,
  Get-ontologies() $\rightarrow$ {WN, Stanford-I, Stanford-II, LSDIS}.
- **Get-node(ont)** returns the node where that ontology and its Ontology Server are located. For example:
  Get-node(WN) $\rightarrow$ ra.cs.uga.edu
- **Synonym-term(term1, ont1, ont2)** returns the term in ontology *ont2* which is a synonym of *term1* in ontology *ont1*.
  Synonym-term(pages, WN, Stanford-I)$\rightarrow$number-of-pages
- **Transform-value(val,role1,ont1,role2,ont2)** returns the equivalent value of *val* stored in *role1* (ontology *ont1*) but for *role2* in the ontology *ont2*. If no transformer function is defined between those roles the same value will be returned.
  transform-value('d', content, WN, type-of-work, Stanford-II) $\rightarrow$ 'dictionary'

[5] Queries are considered as concepts whose constraints represent the properties that the objects in the answer set must satisfy.

[6] It does not mean that they have the same extension.

- **Transform-table(table,roles1,ont1,roles2,ont2)**, given *table* containing a list of values for the roles in *roles1* of ontology *ont1* it returns another table in which, if there exists a transformer function between $role1_i$ and $role2_i$, all the values in $column_i$ are substituted by the result of Transform-value(*value*,$role1_i$,*ont1*,$role2_i$,*ont2*).

2.4.2 Storage of the relationships

We store the relationships in an independent repository that is consulted only by the IRM for requests from the Query Processor. When new ontologies join the system we only need to update the IRM repository. Since the synonym relationships are symmetric in nature (if *a synonym b* then *b synonym a*) they are stored in the following manner:

$< canonical\text{-}term, term, ontology >$

Each new term is related to a *canonical term* representing a generic concept or role. If the new term does not *fit* any preexisting canonical term, a new one will be added to represent that concept/role. The IRM infers that terms with the same canonical term are synonyms. This also helps avoid the redundant representation of these relationships. The transformer functions between values in different roles are defined in this format:

$< function\text{-}name, domain, range >$

where *domain* and *range* are sets of pairs of the format $<role, ont>$ and *function-name* is the name of the function that translates values of the roles in *domain* into semantically equivalent values of the roles of *range*. The implementations of such functions are accessible to the IRM.

Example: FUNCTION: Transform-type-to-WN
DOMAIN: <type, LSDIS>, <type-of-work, Stanford-II>
RANGE: <content, WN>

3. Query Processing

In this section we discuss in detail the query processing approach introduced in Section 2.2, that involves the *reuse of pre-existing ontologies* and interoperation across them. The Query Processor performs the following important steps:

- Translation of terms in the query into terms in each component ontology (Section 3.1). The query processor obtains information from the IRM (discussed in Section 2.4) and the Ontology Server (discussed in Section 3.3).
- Combining the partial translations, in such a way that the semantics of the user query is preserved (Section 3.2).
- Accessing the Ontology Server to obtain the data under the component ontology that satisfy the translated query. This step is discussed in detail in Section 3.3.
- Correlation of the objects retrieved from the various data repositories/ontologies (Section 3.4).

Constructing Query Metadata Using Ontological Terms

The user query will be expressed in the format:

$<list\text{-}of\text{-}roles>$ for $<classic\text{-}expression>$

where *list-of-roles* is a list of roles to be projected (the roles for which the user asks about) and *classic-expression* is a list of constraints expressed in DL (the conditions that the answer must satisfy). If *list-of-roles* is empty, the distinguished role *self*, will be included as the only projection. Consider the example query in Section 2. Let Stanford-II (Section 2.3.3) be the user ontology. The user can construct the query expression as follows:

[title author document pages] for (**AND** doctoral-thesis-ref
(**FILLS** keywords "metadata") (**ATLEAST** 1 publisher))

3.1. Translation into Component Ontologies

We now discuss query re-writing using terms from different component ontologies. The goal is to obtain the same query but expressed in terms of a component (target) ontology and preserving the semantics of the user query. This is achieved as we use *synonym* relationships between terms in different ontologies, thus preserving the meaning of the query. The translation of the roles to be projected is also discussed.

3.1.1 Semantics-preserving translations into Component Ontologies

Intuitively, the algorithm replaces each concept and role in the user query by their corresponding constraints in the component ontology. If a translation is not found for a term, it is substituted by its definition and then the translation algorithm is executed on the definition. We illustrate the algorithm using the example query in Section 2. The detailed algorithm is described in Appendix A.1. Synonyms and transformed values will be obtained from the IRM. To obtain the definition of a term the Ontology Server of the user ontology is consulted. The translation process is applied iteratively at each component ontology as described in Section 2.2. Some important definitions are as follows:

Translation: A translation into a component ontology is represented as:
$<TargetOntology, TranslatedRoles, TranslatedSet, NonTranslatedSet>$

Partial Translation: If some constraints cannot be expressed in the target ontology, i.e. if NonTranslatedSet $<> \phi$, then it is a partial translation.

Full Translation: If all the constraints can be expressed in the target ontology, i.e. NonTranslatedSet = ϕ, then it is a full translation. A full translation may be obtained by combining partial translations (see Section 3.2).

Non-relevant Ontology: If no constraint of a query can be translated at a component, i.e. TranslatedSet = ϕ, then that ontology is not relevant for the query.

Examples: Consider the example query expressed using terms from the Stanford-II ontology earlier in this section. The translation of the query into the component ontologies is as follows:

- **Note that the user query always represents a full translation into the user ontology**.
  < Stanford-II, [title author document pages], (**AND** doctoral-thesis-ref (**FILLS** keywords "metadata") (**ATLEAST** 1 publisher)), ϕ >
- This is an example of a **partial translation**.
  < Stanford-I, [title author NULL number-of-pages], (**AND** doctoral-thesis (**ATLEAST** 1 publisher)), (**FILLS** keywords "metadata")>
- This is an example where a term is substituted by it's definition.
  doctoral-thesis-ref $\equiv$ (**AND** thesis-ref (**FILLS** type-of-work "doctoral"))
  thesis-ref $\equiv$ (**AND** publication-ref (**FILLS** type-of-work "thesis"))
  < WN, [name creator NULL pages], (**AND** print-media (**FILLS** content "thesis" "doctoral") (**ATLEAST** 1 publisher) (**FILLS** general-topics "metadata")), ϕ >
- This is an example of a **partial translation** where the value of the role-filler of the role *keywords* is transformed by the transformer function between the roles *keywords* (Stanford-II) and *subject* (LSDIS).
  < LSDIS, [title authors location-document NULL], (**AND** publications (**FILLS** type "doctoral" "thesis") (**FILLS** subject "METADATA")), (**ATLEAST** 1 publisher)>

3.1.2 Translation and projection of roles

Consider the list of roles of a user query to be projected and the translation of the example query into the WN ontology discussed in the previous section. It is still a full translation (all the instances of print-media retrieved would satisfy the constraints in the user query) but only information about the name and creator can be provided from the underlying repositories. In this case, roles to be projected with no translation will be represented as NULL values. After accessing the data corresponding to that translation the answer from WN can be correlated with answers from other ontologies (e.g. LSDIS, Stanford-II) and the NULL columns will be overwritten with other values in the same role for the same object.

If no role from a user query is translated into some ontology (suppose the user query only asks about the role 'document') the corresponding ontology is not relevant for the user query although all the constraints were translated. This is a case in which we can obtain the objects related to the query but not the information about the objects the user is interested in.

3.2. Combining Partial Translations

As illustrated in the previous section, there are cases when the user query is only partially translated into some ontologies. We now present an interesting theorem which enables us to determine when a combination of partial translations are logically equivalent to a query. The theorem has been rigorously proved in [20].

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 $\mathcal{Q}$.*

Example: Consider the partial translations of the user query at the ontologies Stanford-I and LSDIS (Section 3.1.1). As the intersection of the non-translated parts of the partial translations into Stanford-I and LSDIS is empty, the intersection of both partial answers must satisfy all the constraints in the query. Intuitively:

- From Stanford-I, doctoral theses about any subject which have been published at least once will be retrieved;
- From LSDIS, documents about metadata which may have not been published will be retrieved.
- The intersection of the above will be those documents classified as doctoral theses about metadata and have been published at least once, which is exactly the user query.

In Appendix A.2, we present an algorithm which, given a new partial translation, tries to determine whether it can be combined with any of the partial translations into previously visited ontologies. It also tries to combine the new partial translation with any combination of the previously obtained partial translations which is not a full translation. If the maximum[7] 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. This

[7] Since the original constraints can be substituted by others constraints when using definitions of defined terms.

reduces the explosion of the search space. We also maintain the different combinations of ontologies that can form new full translations and only minimal full translations[8] are returned by the algorithm.

3.3. Ontology Server: Accessing the Data Repositories

In this section we discuss the Ontology Server described in Section 2.1. Only one Ontology Server is needed for all the ontologies residing on its node. The services provided to the Query Processor are:

- To provide the definition of *defined* terms in the query by consulting the user ontology and invoking the appropriate functions of the DL system.
  Get-definition(dictionary, WordNet) $\rightarrow$ (**AND** print-media (**FILLS** content "d"))

- To retrieve data corresponding to a query over a component ontology. Given a query and an ontology name it returns the corresponding data stored in the repositories underlying the ontology as a relation. E.g.: Get-extension('[pages] for dictionary',WN) $\rightarrow$ <relation>
  The Ontology Server utilizes the **mappings** between terms in the ontology and data structures in the underlying repositories. These mappings play a key role in encapsulating the heterogeneity due to different formats and organization of the data in the various repositories. They subscribe to the **idea of viewing a data repository as a set of entities and attributes, independently of the concrete organization of the data in the repository**. They act as an intermediary language between the DL expressions and the query languages of the local repositories.

In the following we illustrate the (combined) mappings corresponding to each of the translations in Section 3.1.1 and the resulting translations into the local repository query language. A detailed discussion of the modules and mechanisms that are used will be available in future papers due to space limitations.

- Stanford-II:

  [self title author document pages] for (**AND** doctoral-thesis-ref (**FILLS** keywords "metadata") (**ATLEAST** 1 publisher))
  Mappings:

```
< [SELECTION,stanford-II.doc,
    [AND,[=,stanford-II.doc.Series,"doctoral"],
         [=,stanford-II.doc.Series,"thesis"],
         [=,stanford-II.doc.Subjects,"metadata"],
         [NOT-NULL,stanford-II.doc.Publisher]]],
  [stanford-II.doc.LC_Call_No,
   stanford-II.doc.Title, stanford-II.doc.Author,
   stanford-II.doc.document,
   stanford-II.doc.Description],
  [string, string, string, postscript, string] >
```

  Local Repository Language (Z39.50 Gateway to Library of Congress):

  firstrecord = 1 & maxrecords = 1000 & dbname = BOOKS & term_term_1 = doctor al &
  term_use_1 = Series Title & term_struct_1 = Word & operator_2 = and & term_term_2 = thesis & term_use_2 = Series Title & term_struct_2 = Word & operator_3 = and & term_term_3 = metadata & term_use_3 = Subjects & term_struct_3 = Word & operator_3 = and not & term_term_4 = NULL & term_use_4 = publisher & term_structure_4 = Word & port = 2210 & esn = F host = ibm2.loc.gov& attrset = BIB1 & rtype = USMARC & DisplayRecord-Syntax = HTML

- Stanford-I:
  [self title author NULL number-of-pages] for (**AND** doctoral-thesis (**ATLEAST** 1 publisher))
  Mappings:

```
< [SELECTION,stanford-I.document,
   [AND,[=,stanford-I.document.series_title,
          "doctoral thesis"],
        [NOT-NULL,stanford-I.doc.publisher]]]
  [stanford-I.document.loc,
   stanford-I.document.title,
   stanford-I.document.name,
   NULL, stanford-I.document.pages],
  [string, string, string, NULL, string] >
```

  Local Repository Query Language (SQL):

```
SELECT loc, title, name, "NULL", pages
FROM document
WHERE doc_type like "%doctoral thesis%"
and publisher NOT NULL;
```

- WN:
  [self name creator NULL pages] for (**AND** print-media (**FILLS** content "thesis") (**ATLEAST** 1 publisher) (**FILLS** content "doctoral") (**FILLS** general-topics "metadata"))
  Mappings:

```
< [SELECTION,wn.record,
   [AND,[=,wn.record.008$[24-27],"doctoral"],
        [=,wn.record.008$[24-27],"thesis"],
        [NOT-NULL,wn.record.260$b],
        [=,wn.record.650$a,"metadata"]]],
  [wn.record.010$a, wn.record.245$a, wn.record.100$a,
   wn.record.300$a],
  [string, string, string, NULL, string] >
```

  Local Repository Query Language:

```
FILES: /home/grad/mena/MARC/UGA/oclcwkly.unicat
PROJECTIONS: 010$a | 245$a | 100$a | NULL | 300$a
CONDITIONS: 008$[24-27] = doctoral | 008$[24-27] = thesis
          | 650$a = metadata | 260b$ <> NULL
```

- LSDIS:
  [self title authors location-document NULL] for (**AND** publications (**FILLS** type "doctoral" "thesis") (**FILLS** subject "METADATA"))
  Mappings:

[8] If translations at ontologies A and B, and at ontologies A, B and C can be combined to obtain a full translation, then the combination A and B is minimal, whereas the combination A, B and C is not.

```
< [JOIN [SELECTION, lsdis.pub,
   [AND,[=,lsdis.pub.type,"doctoral"],
        [=,lsdis.pub.type,"thesis"],
        [=, lsdis.pub.subjects, "METADATA"]]],
        lsdis_html.pub,
        [=, lsdis.pub.id, lsdis_html.pub.id]]
  [lsdis.pub.id, lsdis.pub.title,
   lsdis.pub.authors, lsdis_html.pub.document, NULL],
  [string, string, string, postscript, NULL] >
```

Local Repository Query Language:

```
FILES: /home/grad/mena/PROGS/publication-list.txt
   | /research2/www/htdocs/publications/pub_ALL.html
PROJECTIONS: id | title | authors | location-document
           | NULL
CONDITIONS: subjects = METADATA | publisher <> NULL
```

Since the mappings are defined for terms, the Ontology Server also uses a mechanism to combine mappings of terms to obtain the mapping of the whole query. The details of this mechanism can be found in [9].

3.4. Correlation

After obtaining the corresponding data for each ontology involved in the user query, that data must be combined to give an answer to the user. First, the data retrieved is checked for format and value heterogeneity. For each answer (represented as a relation), the Query Processor will invoke the service 'Transform-table' described in Section 2.4.1 to transform the values in the format of the user ontology. After this initial step, the different partial answers can be correlated since all of them are expressed in the *language* of the user ontology. In the following we describe how partial answers can be combined.

- Let Objects(C, Ont) be the set of objects underlying the ontology Ont that satisfy the constraints C; constraints C are expressed in terms of ontology Ont.
- Let C be the set of constraints in a query Q constructed from a user ontology Ont. Let C' and C" be **full translations** of the query Q at ontologies Ont' and Ont" respectively. Then the final answer is given as:
  Objects(C, Ont) = Objects(C', Ont') ∪ Objects(C", Ont")
- Let C' be a partial translation of C at ontology Ont' and C" be a partial translation of C at ontology Ont" respectively, where the combination of C' and C" is a **full translation**.
  The final answer is then given as:
  Objects(C, Ont) = Objects(C', Ont') ∩ Objects(C", Ont").

We now present the correlation plan which is applied to the translations of Section 3.1.1.

User_Query_Objects = Objects('[self title author document pages] for (**AND** doctoral-thesis-ref (**FILLS** keywords "metadata") (**ATLEAST** 1 publisher))')

Stanford-II_Objects
= Objects('[self title author document pages] for (**AND** doctoral-thesis-ref (**FILLS** keywords "metadata") (**ATLEAST** 1 publisher))', Stanford-II)

Stanford-I_Objects = Objects('[self title author NULL number-of-pages] for (**AND** doctoral-thesis (**ATLEAST** 1 publisher))', Stanford-I)

WN_Objects = Objects('[self name creator NULL pages] for (**AND** print-media (**FILLS** content "thesis" content "doctoral") (**FILLS** general-topics "metadata"))', WN)

LSDIS_Objects = Objects('[self title authors location-document NULL] for(**AND** publications (**FILLS** type "doctoral" "thesis") (**FILLS** subject "METADATA")', LSDIS))

User_Query_Objects = Stanford-II_Objects ∪ WN_Objects ∪ [Stanford-I_Objects ∩ LSDIS_Objects]

We can see that the final answer is composed of two full translations (Stanford-II which plays the role of the user ontology and WN) and two partial translations (Stanford-I and LSDIS) combined to give a third full translation.

Correlation with projections

When the user query is the projection of some roles of the objects satisfying the specified constraints (if not, only the distinguished role *self* will be retrieved), an intermediate step is needed before presenting the answer to the user. To perform correlation between data from different ontologies we must be able to identify common objects retrieved from different ontologies. For intersection, we show only the common objects; and for union, we eliminate the duplicate objects. The queries sent to the Ontology Servers always include the distinguished role *self* (see examples in the previous section) so that correlation can be performed based on that column to identify different instances.

4. Conclusions and Future Work

We have described an architecture for Global Information Systems that is especially tailored to address the challenges discussed in Section 2. Our approach is based on:

- Use of intensional metadata descriptions to model and query the information content in various repositories, and
- Ontology-based interoperation by navigating terminological relationships, to handle the vocabulary problem.

Novel contributions in this paper include the representation of the synonym relationships between terms across ontologies, an algorithm for (partially) translating the intensional query expression into different ontologies, and an algorithm to combine the partial translations in different ontologies such that they satisfy the constraints in the

original query. The heterogeneity in the values is managed by using *transformer functions* stored by the IRM. Unlike a regular thesaurus, the expressiveness of the DL systems allows using descriptions when a defined term has no translation. The methods described in this paper are implemented in a prototype system developed at the LSDIS lab, OBSERVER, accessible from WWW browsers at http://lsdis.cs.uga.edu/~mena/OBSERVER). This prototype accesses information in real-world data repositories using pre-existing real-world ontologies in the domain of bibliographic information. OBSERVER, by using pre-existing real-world ontologies and real-world repositories, helps the user to observe a *semantic conceptual view* of a global information system by giving her/him the ability to **browse multiple domain specific ontologies as opposed to individual heterogeneous repositories**. OBSERVER uses the CLASSIC system and demonstrates a practical use of DLs for interoperation across domain specific ontologies to support querying and information organization in a global information system. Our architecture is *extensible* and *scalable* in the following respects:

- The extensions of semantically equivalent terms can be appropriately combined using the relationships stored and managed by the IRM.
- The number of relationships across terms between different ontologies are expected to be *an order of magnitude less* than the terms in all the ontologies.

We use **real-world** ontologies (developed independently of the real-world repositories) to describe real-world repositories from the same domain (bibliographic data), and provide **different (independently designed) conceptual views of the same data**.

Future Work

Our approach for querying a global information system depends very crucially on interoperation across pre-existing ontologies. The following on-going research activities are expected to make our solution more comprehensive.

- Development of an algorithm to support the interoperation across ontologies when the terms have hyponyms and hypernyms in other ontologies. The system tries to substitute the conflicting terms by its immediate parents (generalization) or by its immediate children (specialization) to get a full translation into a component ontology. The resulting loss of information is measured.
- Support for synonym relationships between expressions instead of terms.
- Extensional relationships (Disjoint, exactly-the-same) between terms can be defined in the IRM and used by the Query Processor to determine redundancy and minimize access to component ontologies as well as measure the information loss.
- Modification of the Query Processor to support composition of transformer functions so that the values of synonym roles in different ontologies can be appropriately transformed.
- Addition of important tools for user and administrators to make the system easier and more convenient to use. Specifically,
  - a **query editor** which helps the user to write a DL expression. A context sensitive GUI is being developed with automatic syntax checking.
  - an **ontology editor** to help create and edit ontologies and to map the ontological terms to the underlying repositories. Some tools to create ontologies expressed in DLs over relational databases; and to define mappings for the terms in the ontologies are described in [4] and [19] respectively. We need to extend them to work with other data organizations.

Acknowledgments

We would like to thank Marty Tanner Hughes, Anne Hope and Brad Baxter for their priceless help in getting real data in MARC format from the University of Georgia Main Library. The CLASSIC system, donated by AT&T Bell Labs and the Illustra DBMS from Illustra Information Technologies have been valuable tools.

References

[1] E. Achilles, B. Hollunder, A. Laux, and J. Mohren. KRIS: Knowledge Representation and Inference System. Technical Report D-91-14, DFKI Kaiserslautern-Saarbrucken, 1991.

[2] Y. Arens, C. Chee, C. Hsu, and C. Knoblock. Retrieving and Integrating Data from Multiple Information Sources. *International Journal of Intelligent and Cooperative Information Systems*, 2(2), June 1993.

[3] T. Berners-Lee et al. World-Wide Web: The Information Universe. *Electronic Networking: Research, Applications and Policy*, 1(2), 1992.

[4] J. Blanco, A. Illarramendi, and A. Goñi. Building a Federated Database System: An approach using a Knowledge Based System. *International Journal on Intelligent and Cooperative Information Systems*, 3(4):415–455, December 1994.

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

[6] R. Brachman and J. Scmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2), Feb. 1985.

[7] C. Collet, M. Huhns, and W. Shen. Resource Integration using a Large Knowledge Base in Carnot. *IEEE Computer*, Dec. 1991.

[8] A. Emtage and P. Deutsch. Archie: An Electronic Directory Service for the Internet. In *Proceedings of the Winter 1992 Usenix Conference*, 1992.

[9] A. Goñi, J. Blanco, and A. Illarramendi. Connecting Knowledge Bases with Databases: A complete mapping relation. In *Proc. of the 8th ERCIM Workshop. Trondheim, Norway*, 1995.

[10] T. Gruber. Theory BIBLIOGRAPHIC-DATA, Sept. 1994. http://www-ksl.stanford.edu/knowledge-sharing/ontologies/html/bibliographic-data/index.html.

[11] J. Hammer and D. McLeod. An approach to resolving Semantic Heterogeneity in a Federation of Autonomous, Heterogeneous, Database Systems. *International Journal of Intelligent and Cooperative Information Systems.*, Mar. 1993.

[12] B. Kahle and A. Medlar. An Information System for Corporate Users : Wide Area Information Servers. *Connexions - The Interoperability Report*, 5(11), Nov. 1991.

[13] V. Kashyap and A. Sheth. Semantic and Schematic Similarities between Databases Objects: A Context-based approach. *The VLDB Journal*. To appear; http://www.cs.uga.edu/LSDIS/~amit/66b-VLDB.ps.

[14] V. Kashyap and A. Sheth. Semantics-based Information Brokering. In *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM)*, Nov. 1994.

[15] D. Lenat and R. V. Guha. *Building Large Knowledge Based Systems : Representation and Inference in the Cyc Project*. Addison-Wesley Publishing Company Inc, 1990.

[16] A. Levy, D. Srivastava, and T. Kirk. Data Model and Query Evaluation in Global Information Systems. *Intelligent Information Systems*, 5(2), Sept. 1995.

[17] R. MacGregor. A deductive pattern matcher. In *Proceedings AAAI-87*, 1987.

[18] M. McCahill. The Internet Gopher Protocol : A Distributed Server Information System. *Connexions - The Interoperability Report*, 6(7), July 1992.

[19] 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.

[20] 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. Technical report, TR-CS-96-001 LSDIS Lab, UGA, 1996.

[21] G. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11), Nov. 1995.

[22] G. Miller. World Wide Web interface to WordNet 1.5, June 1995. http://www.cogsci.princeton.edu/~wn/w3wn.html.

[23] T. L. of Congress. Library of Congress WWW Z39.50 gateway, 1995. http://lcweb.loc.gov/z3950/.

[24] J. Ordille and B. Miller. Distributed Active Catalogs and Meta-Data Caching in Descriptive Name Services. In *Proceedings of the 13th International Conference on Distributed Computing Systems*, May 1993.

[25] S. Piepenburg. Easy MARC: A simplified guide to creating catalog records for library automation systems: Pre-format integration, 1994.

[26] E. Pitoura, O. Bukhres, and A. Elmagarmid. Object Orientation in Multidatabase Systems. *ACM Computing Surveys*, 27(2), June 1995.

[27] E. Sciore, M. Siegel, and A. Rosenthal. Context Interchange using Meta-Attributes. In *Proceedings of the CIKM*, 1992.

[28] M. Sheldon et al. Semantic File Systems. In *Proceedings of the 13th ACM Symposium on Operating System Principles*, 1991.

[29] L. Shklar, A. Sheth, V. Kashyap, and K. Shah. Infoharness: Use of Automatically Generated Metadata for Search and Retrieval of Heterogeneous Information. In *Proceedings of CAiSE '95*, June 1995. Lecture Notes in Computer Science, #932.

[30] K. Shoens, A. Luniewski, P. Schwartz, J. Stamos, and J. Thomas. The Rufus System: Information Organization for Semi-Structured Data. In *Proceedings of the 19th VLDB Conference*, Sept. 1993.

[31] M. Siegel and S. Madnick. A Metadata Approach to resolving Semantic Conflicts. In *Proceedings of the 17th VLDB Conference*, Sept. 1991.

[32] K. von Luck, B. Nebel, C. Peltason, and A. Schmiedel. The anatomy of the BACK system. Technical Report KIT Report 41, Technical University of Berlin, Berlin, F.R.G., 1987.

A. Algorithms used by the Query Processor

A.1. Algorithm for semantics-preserving translations

```
/* Given a user query and a target ontology it transforms
   the user query in an equivalent one (using synonym
   relationships, definitions of terms, and equivalent
   values) expressed in terms of the target ontology */

TRANSLATE-PRESERVING-SEMANTICS (user-query,target-ontology)
{ fills=0
  FOR each constraint in the user query DO
     FOR each component of the constraint DO
         CASE component is:
              term: /* concept or role */
               IF exists synonym from
                    user to target ontology THEN {
                         substitute(term, synonym-of-term)
                         IF fills THEN {
                                role=term
                                new-role=synonym-of-term
                               }
                        }
                ELSE IF it is a defined term THEN {
                          substitute(term, definition-of-term)
          translate-preserving-semantics(definition-of-term,
                                         target-ontology)
                     }
              value:
               IF fills AND exists transformer
                  function between role and new-role THEN {
                         substitute(value, equivalent-value)
                         fills = 0
                   }
```

```
            operator: /* ALL, AT-LEAST, FILLS, ... */
                 IF it is 'fills' THEN
                              fills = 1 ELSE fills = 0
}
```

A.2. Algorithm for combining the partial translations

```
COMBINE_PARTIAL_TRANSLATIONS ( non_full_combs, new_partial)
/* Non_full_combs: previous combinations of partial
   translations which do not satisfy all the constraints
   in the user query. The list is in increased order based
   of number of partial translations involved. Each partial
   translation translates at least one constraint of
   the user query that the others in the same combination
   do not.
   New_partial: the new partial translation the system has
   just obtained */
{
  full={} /*  new full translations resulting of the use
              of the new partial translation */
  new_fulls={} /* name of the component ontologies involved
                  in each new full translation */
  n_f_c = non_full_combs UNION new_partial
              /* New partial is a non full combination */
  WHILE not_empty(non_full_combs) DO {
       comb = first(non_full_combs)
       new_comb = comb UNION new_partial
       IF  (#_non_translated(new_comb) <
           #_non_translated(comb)) AND
           (ontologies(new_comb) is not a superset of any
           element in new_fulls) THEN {
/* some of the nontranslated constraints in the
   combination is translated in the new partial
   translation */
             IF full(new_comb) THEN {
                /* equiv. #_non_translated(new_comb)=0 */
                          full = full UNION new_comb
                          new_fulls = new_fulls UNION
                                      ontologies(new_comb)
                        }
                  ELSE n_f_c = n_f_c UNION new_comb
             }
          /* ELSE The new partial is not interesting for
                  that combination or it is not minimal) */
       non_translated_combs =
                     remove_first(non_translated_combs)
      }
  return < full, n_f_c >
}
/* Returns new full translations when using the new partial
   and the new interes ing non full combinations */
```

B Component Ontologies in the prototype

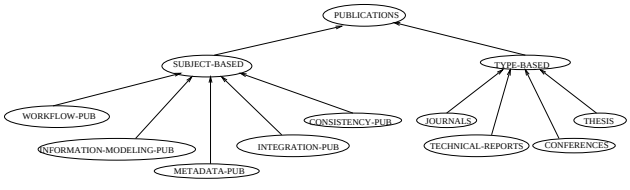

Figure 3. The LSDIS ontology

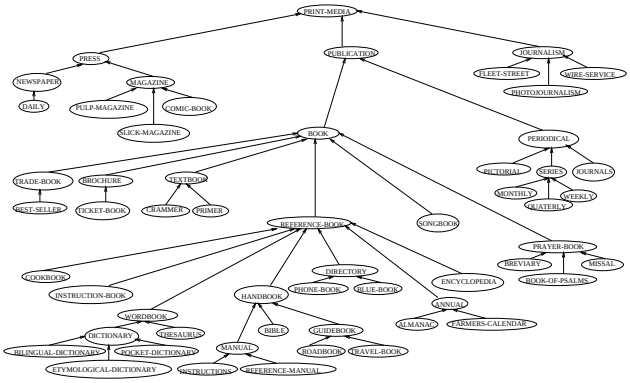

Figure 4. WN: A subset of the WordNet 1.5 ontology

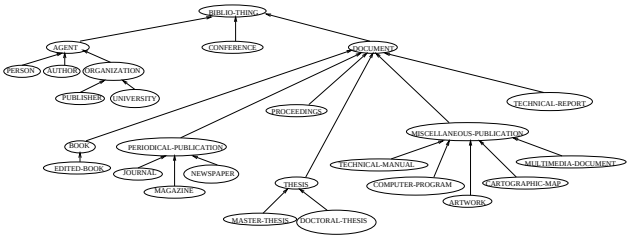

Figure 5. Stanford-I

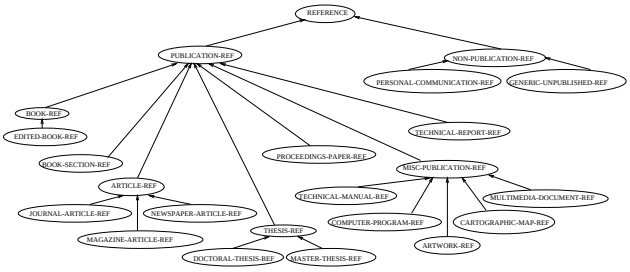

Figure 6. Stanford-II