ISSN 2229-5518

Ontology Based Data Integration in Federated Databases and Its' Issues

J.Sharmila, Dr.Subramani

Abstract: In this paper, we discuss the use of ontologies for data integration. We consider two different settings depending on the system architecture: central and peer-to-peer data integration. Within those settings, we discuss five different cases studies that illustrate the use of ontology in metadata representation, in global conceptualization, in high-level querying, in declarative mediation, and in mapping support. Each case study is described in detail and accompanied by examples.

Many applications, e.g., data/information fusion, data mining, and decision aids, need to access multiple heterogeneous data sources. These data sources may come from internal and external databases. They have to evolve due to requirement changes. Any change in an application domain induces semantics change in the data sources. The integration of these data sources raises several semantic heterogeneity problems. This has traditionally been the subject of data/schema integration and mapping.

However, many heterogeneity conflicts remain in information integration due to lack of semantics. Therefore, richer semantics of data are needed to resolve the heterogeneity problems. Ontological approaches now offer new solution avenues to this interoperability limitation. In this perspective, we propose an ontology based information integration with a local to global ontology mapping as an approach to the integration of heterogeneous data sources.

The term "Federated Databases" refers to the data integration of distributed, autonomous and heterogeneous databases. However, a federation can also include information systems, not only databases. At integrating data, several issues must be addressed. Here, we focus on the problem of heterogeneity, more specifically on semantic heterogeneity – that is, problems related to semantically equivalent concepts or semantically related/unrelated concepts. In order to address this problem, we apply the idea of ontologies as a tool for data integration. In this paper, we explain this concept and we briefly describe a method for constructing ontology for Data Integration.

Key words: Federated, Fusion, Mapping, Mediation system, Ontology, Schema Integration, Semantic.

1. INTRODUCTION

Data integration provides the ability to manipulate data transparently across multiple data sources. It is relevant to a number of applications including enterprise information integration, medical information management, geographical information systems, and E-Commerce applications. Based on the architecture, there are two different kinds of systems: central data integration systems and peer-to-peer data integration systems. A central data integration system usually has a global schema, which provides the user with a uniform interface to access information stored in the data sources. In contrast, in a peer-to-peer data integration system, there are no global points of control on the data sources (or peers). Instead, any peer can accept user queries for the information distributed in the whole system.

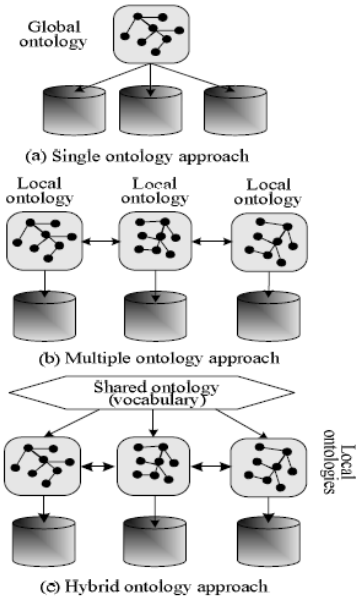
- *J.Sharmila, Bharathidasan University College(W), orathanadu, Taminadu,India.*
- *Dr.A.Subramani,KSR College of Engg. And technology,Thiruchengodu,tamilnadu,India*

1.1 Ontology

An Ontology is a formal, explicit specification of a shared conceptualization. In this definition, "conceptualization" refers to an abstract model of some domain knowledge in the world that identifies that domain's relevant concepts. "Shared" indicates that an ontology captures consensual knowledge, that is, it is accepted by a group. "Explicit" means that the type of concepts in an ontology and the constraints on these concepts are explicitly defined. Finally, "formal" means that the ontology should be machine understandable. Typical "real-world" ontologies include taxonomies on the Web (e.g., Yahoo! categories), catalogs for on-line shopping (e.g., Amazon.com's product catalog), and domain-specific standard terminology (e.g., UMLS and GeneOntology). As an online lexicon database, WordNet3 is widely used for discovery of semantic relationships between concepts.

1.2 Ontologies for Data Integration

Ontologies have been extensively used in data integration systems because they provide an explicit and machine-understandable conceptualization of a domain. They have been used in one of the three following ways.



Single ontology approach. All source schemas are directly related to a shared global ontology that provides a uniform interface to the user. However, this approach requires that all sources have nearly the same view on a domain, with the same level of granularity. A typical example of a system using this approach is SIMS.

Multiple ontology approach. Each data source is described by its own (local) ontology separately. Instead of using a common ontology, local ontologies are mapped to each other. For this purpose, an additional representation formalism is necessary for defining the inter-ontology mappings. The OBSERVER system is an example of this approach.

Hybrid ontology approach. A combination of the two preceding approaches is used. First, a local ontology is built for each source schema, which, however, is not mapped to other local ontologies, but to a global shared ontology. New sources can be easily added with no need for modifying existing mappings. Our layered framework is an example of this approach.

The single and hybrid approaches are appropriate for building central data integration systems, the former being more appropriate for GaV systems and the latter for LaV systems. A hybrid peer-to-peer system, where a global ontology exists in a "super-peer" can also use the hybrid ontology approach. The multiple ontology approach can be best used to construct pure peer-to-peer data integration systems, where there are no super-peers.

2. SEMANTIC DATA INTEGRATION

Integration describes the features to reconcile and condense collected or stored data. Semantic integration is the process of matching/merging/fusing individual data/schemas in order to provide a global and integrated (unified) view. Therefore, it involves the transformation of data sources into an integrated view and the resolution of heterogeneity conflicts.

2.1 Data integration operations

There are many integration operations on data sources. There are situations in which data are collected unchanged from data sources and others in which semantic integration of higher levels is also performed, e.g., in decision aids and data/information fusion. In addition to data/schema matching (alignment), other data integration operations are:

1) Collection: Data objects are collected unchanged. There is no matching with equivalent other data objects coming from diverse sources.

2) Fusion: This term introduced by [15] is much narrower than its application in the Fusion model. The integration of data objects is done by a simple extraction; no further abstracting computations are done. In contrast to the data collection approach, data object fusion is performed in conjunction with semantically equivalent data objects coming from different sources. Furthermore, the fusion processes try to determine consistent representations, i.e., if data sources report contradicting values for the same data item, the fusion uses mapping rules and heuristics to remove data conflicts. It should be noted that even the integration at the data fusion level may be very difficult. Frequently, it is impossible to identify data objects or to decide which data value is correct.

3) Abstraction: Transformations may be applied to the source data to match the abstraction level. Abstraction encompasses functions for aggregating data, reclassifying entities, or even more complex reasoning processes.

4) Supplementation: Data are not only derived from object data, but other data are added which describe the content or context of the object data (e.g., semantic metadata). Such integration is used to handle implicit semantics of objects. This

operation is necessary whenever data sources provide no schema, and the integration bases on a metadata schema.

Aggregation and grouping are other well-known data integration operation types. Note that many data transformations often prevent possible mapping between original sources data and the resulting global schema.

2.2 Integration system architectures

Integration means a form of cooperation between several users and several sources of data. There are several taxonomies of integration systems. For the current purpose, we will consider that the integration architecture is regardless of the notion of centralized or distributed information systems.

Multibase system

A multibase (multiple database) system allows the users to view the database through a single global schema, simulating to users that a federated data base exists. However, in this architecture there is no attempt to unify the semantics of data from the various sources. The integration is considered as dynamic as mapping links between schemas are not predefined but established as required. The multibase system has two components: a schema design tool and a query processing system. The first component provides the tools needed by the database designer for designing the global schema and defining a mapping from the local databases to the global schema. The second component uses the mapping definition to translate global queries into local queries.

Federated systems

The architecture of federated systems has been defined by Sheth and Larson. It is characterized by the existence of a federated schema which establishes the interface to this integrated system. The integration is achieved at the schema level of each data source. The design of a federated schema supposes to unify source schemas and to handle their heterogeneousness. It is necessary to identify the mapping and to resolve the conflicts between the schema elements. This mapping can be expressed, for example, by means of various languages, by means of rules or, as we will see below, through an ontology. Thus, there is, in this architecture, a unified vision by data sources. Integration offers a common access and a common representation to data sources. In this architecture, the integration of federated systems is considered as static as mapping links between schemas are predefined.

Mediation system

The idea of a mediator was introduced by Wiederhold who defined it as follows: "A mediator is a software module that exploits encoded knowledge about some sets or subsets of data to create information for a higher layer of applications." Since then, it has been used in many data integration projects and techniques. We may perceive a mediator as a software component that mediates between the user and physical data sources. Data are not stored in the mediation system but remain at their sources. Interrogation of data sources is made by wrappers which establish an interface to the various data sources. These wrappers translate the sub-requests expressed in the language request specific to every source. The results are then returned to the wrappers who integrate them before presenting them to the user.

Data Warehouse

In this architecture, data are accessed, transformed and stored in a single location, the data warehouse. Once the data are extracted, they are then processed by analytics tools, data fusion and data mining tools, decision aid tools and query languages. There is one global schema and there is no need to return to the original data sources. This architecture is often privileged in large organizations.

3. ISSUES

3.1 Developing ontologies

In any reasonably realistic e-commerce scenario involving interoperability between systems, semantic heterogeneity is a significant problem and will continue to be so in the future. A solution to this problem based on the use of formal ontologies will need to accommodate different types of ontologies for different purposes. For example, we may have resource ontologies, which define the terminology used by specific information resources. We may also have personal ontologies, which define the terminology of a user or some group of users. Another type is shared ontologies, which are used as the common terminology between a number of different systems. The problem of developing ontologies has been well studied and a number of methodologies have been proposed.

A comparative analysis of these can be found in [Jones etal., 1998].) One of the major conclusions of this study was that the best approach to take in developing an ontology is usually determined by the eventual purpose of the ontology. For example, if we wish to specify a resource ontology, it is probably best to adopt a bottom-up approach, defining the actual terms used by the resource and then generalising from these. However, in developing a shared ontology it will be extremely difficult to adopt a bottom-up approach starting with each system, especially where there are a large number of such systems. Here, it is most effective to adopt a top-down approach, defining the most general concepts in the domain first.

3.2 Mapping Between Ontologies

In order to resolve the problems of semantic mismatches discussed above, we will often need to translate between different terminologies. While it would be ideal to be able to automatically infer the mappings required to perform such translations, this is not always possible. While the formal definitions in an ontology are the best specification of the meaning of terms that we currently have available, they cannot capture the full meaning. Therefore, there must be some human intervention in the process of identifying correspondences between different ontologies. Although machines are unlikely to derive mappings, it is possible for them to make useful suggestions for possible correspondences and to validate human-specified correspondences. Creating mappings is a major engineering work where re-use is desirable. Declaratively-specifying mappings allow the ontology engineer to modify and reuse mappings. Such mappings require a mediator system that is capable of interpreting them in order to translate between different ontologies. It would also be useful to include a library of mappings and conversion functions as there are many standard transformations which could be reused e.g. converting kilos to pounds, etc.

Mapping between ontologies is not an exact science. Certain semantic mismatches cannot be resolved exactly but may involve some loss of information e.g. when translating from a colour system based on RGB values to one which uses terms such as 'red', 'blue', etc. Whether or not the loss of information is an issue varies between applications. In some domains, precision of information is more important than in others. For example, in e-commerce, imperfect information is generally unacceptable, whereas it is widely accepted that internet search engines will return many irrelevant results.

3.3 Ontologies and Resource Information

It is generally acknowledged that we have more information than we know what to do with. This proliferation of data means that often, for any information query we might have, there are a variety of resources available that store data about the same domain and which are of varying quality. A distributed query engine needs to decide which of the many available resources to use in finding the solution to a query. In addition to finding the resources that have the required information, it may also be necessary to decide between different resources that have the same information available. In order for a distributed query engine to understand what information is available, the resources need to make descriptions of their contents available in a meaningful way. If the terms using in such a description are formally defined in an ontology, the query engine has access to the meaning of the terms in the description. This allows the query engine to make fully informed decisions about which resources are relevant to resolving a particular query. There are a number of pragmatic issues in locating the resources that will be used to answer a query. For example, a particular user may - for whatever reason - prefer one resource over another as the source of some information. Such personal preferences can be taken into account by the distributed query engine if a personal profile of a user's preferences is maintained. The query engine can make better informed decisions if the definitions of the terms used in such a profile are available to it in the form of a user ontology, which defines the terminology of a user or usergroup.

3.4 Ontologies and Database Schemas

Ontologies and database schemas are closely related and people often have trouble deciding which is which. There is often no tangible difference, no way of identifying which representation is a schema and which is an ontology. This is especially true for schemas represented using a semantic data model. The main difference is one of purpose. An ontology is developed in order to define the meaning of the terms used in some domain whereas a schema is developed in order to model some data. Although there is often some correspondence between a data model and the meaning of the terms used, this is not necessarily the case. Both schemas and ontologies play key roles in heterogeneous information integration because both semantics and data structures are important. For example, the terminology used in schemas is often not the best way to describe the content of a resource to people or machines. If we use the terms defined in a resource ontology to describe the contents of a resource, queries that are sent to the resource will also use these terms. In order to answer such queries, there needs to be a relationship defined between the ontology and the resource schema. Again, declarative mappings that can be interpreted by some mediator system are useful here. The structural information provided by schemas will enable the construction of executable queries such as SQL queries. This is related to the discussion earlier about XML, where a database schema is analogous to an XML schema or DTD. As pointed out above, using XML is insufficient for determining the semantics of resources. A schema, whether specified using XML or some database schema language, needs an associated formal ontology in order to make the semantics of the resource clear. When the meaning of data and schemas is made explicit using an ontology, programs can be designed that exploit those semantics.

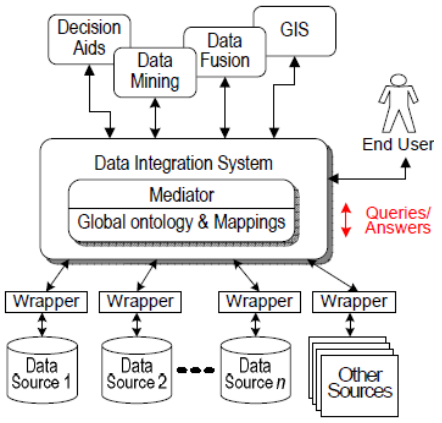
3.5 Entity Correspondence

Ontologies are used in e-commerce environments where data is scattered across heterogeneous distributed systems. In order for the consumer to have access to the maximum amount of available information, we want to be able to retrieve information from various systems and to integrate it. For example, we might want to integrate information from a supplier's product catalogue with customer reviews produced independently. To gather all the information relevant to an entities, the correspondence between entities across resources must be established. For example, the academic records and criminal records of a person are likely to be stored in separated data resources. However, the way in which different resources identify individuals varies. For example, in relational databases entities are identified using key attributes. There is no guarantee that different relational databases use the same key attributes. Even when the same key attribute is used, different terms may be used to denote the attributes. How our systems can determine whether entities from different resources are the same or not is crucial to fusing information. Standard schemas do not provide a full solution here since many systems (e.g. KBSs, object oriented databases) often do not have key attributes at all.

3.6 Ontology-based data integration architecture

Starting on the technological advances, especially at the level of semantic Web and ontology mapping, we describe an ontology-based data integration system which consists in building a global ontology from the local ontologies corresponding to the data sources as opposed to a federated system approach. We now develop a local ontology for each data source and a global ontology. The role of the data integration system, which may be designed as a semantic portal for end users at the organization level is to exploit the global ontology and its integration with the local ontologies of data sources, as illustrated in the following Figure

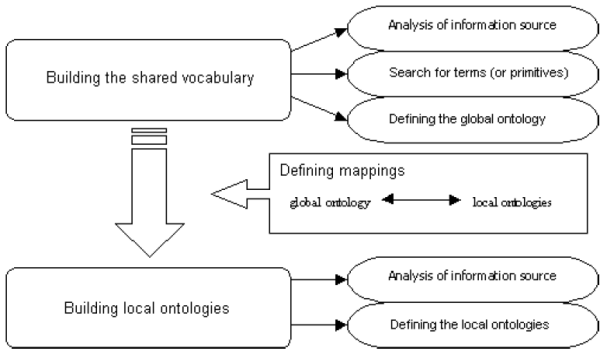
In this architecture, the data integration system constitutes a virtual database as opposed to a data warehouse, which copies data from several data sources in a single database. Now, the mediator maps the requests and answers between the global ontology/schema and the local ontologies with their associated source schemas.



There are a lot of advantages in the use of ontologies for data integration. Some of them are: the ontology provides a rich, predefined vocabulary that serves as a stable conceptual interface to the databases and is independent of the database schemas; the knowledge represented by the ontology is sufficiently comprehensive to support translation of all the relevant information sources; the ontology supports consistent management and the recognition of inconsistent data; etc. Research works using ontologies to solve problems about data integration can be found in Then, we describe our method for building the structure of the ontology. The following Figure shows the algorithm designed to do that.
As we can see, the method has three main stages: building the shared vocabulary, building local ontologies and defining mappings. Each stage embodies a set of tasks that must be achieved. We will briefly explain each stage by using an example. We have used Ontolingua to represent the ontology example.

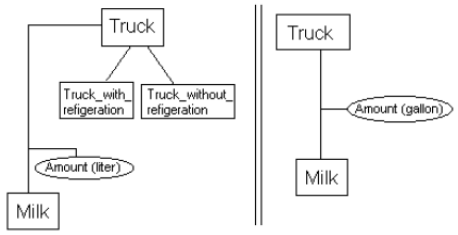
First stage:
Building the shared vocabulary: As Figure shows, this stage contains three main steps: analysis of information sources, search for terms (or primitives) and defining the global ontology.

The first step implies a complete analysis of the information sources, e.g., what information is stored, how it is stored, the meaning of this information (the semantic), etc.



It must localize the problems about semantic heterogeneity previously explained. For example, the next Figure shows an example of two similar systems containing information about transporting milk. In this example, we can clearly see two semantic problems: property-type mismatch and different classification.

The first problem is reflected by the amount attribute in both systems because there are two classes with the same meaning but with different representations: liter and gallon.



Two Systems with different Representation

System 1 has the milk and truck classes with the amount attribute to represent the amount of transported milk. The instances of the amount attribute are in liters. On the other hand, System2 has the same classes but the amount attribute is represented in gallons. The second problem, different classification, is reflected by the truck class in System 2 and, in System 1 by the hierarchy of trucks: truck_ with_ refrigeration and truck_without_refrigeration. Both systems use different classifications to denote the same things. The truck class in System 2 includes the two sub clasiffications of System1.

The second step, search for terms (or primitives), implies the choice of the list of terms or concepts in agreement with the shared vocabulary. In our example, the list of terms can be: milk, gallon, amount, truck, truck_with_refrigeration and truck_ without_ refrigeration. We include the terms of the hierarchy because this classification is more descriptive. As we have previously mentioned, the truck class in System 2 includes both trucks with refrigeration and without refrigeration.

The inclusion of this hierarchy into the global ontology provides more semantic information. Later we will seewhat happens with the liters of System 1. The third and last step, defining the global ontology, uses the terms chosen in the last step to create the global ontology. The following procedures shows the global ontology generated from the two systems defined in the above mentioned Figure.

---------- Classes ----------
Milk
(Define-Class Milk (?X) "the set of types of milks" :Def (And (Thing ?X)))

Gallon
(Define-Class Gallon (?X) "the set of gallons" :Def (And (Thing ?X)))

Truck
(Define-Class Truck(?X) "the set of trucks" :Def (And (Thing ?X)))

Truck_With_Refrigeration
(Define-Class Truck_With_Refrigeration (?X) "the set of trucks with refrigeration" :Def (And (Truck?X)))

Truck_Without_Refrigeration
(Define-Class Truck_Without_Refrigeration (?X)"the set of trucks without refrigeration" :Def (And (Truck ?X)))

---------- Relations ----------
The_Amount
(Define-Relation The_Amount (?Frame ?Value) "the amount expressed in gallons" :Def (And (Gallon ?Frame) (Number ?Value)))

Transporting
(Define-Relation Transporting (?Truck ?Milk?Gallon) "the amount of milk transported by a truck" :Def (And (Truck ?Truck) (Milk ?Milk) (Gallon ?Gallon)))

Global Ontology

We could have represented the ontology without the gallon class. Thus, the transporting relation would be:

Transporting
(Define-Relation Transporting (?Truck ?Milk ?Number) "the amount of milk transported by a truck" :Def (And (Truck ?Truck) (Milk ?Milk)
(Number ?Number)))

The primitive type Number is replacing the gallon class. This representation, although acceptable for Ontolingua, is not clear enough and does not provide the whole semantic information available in the system. In fact, inclusion of classes describing attribute types in the ontology is the best choice for providing the semantic information required.

Second stage:
Building local ontologies:

As Figure 1 shows, this stage contains two main steps: analysis of information source and
defining the local ontologies. The first step is similar to the first stage previously explained. A complete analysis of the information sources must be made. This analysis is performed independently, that is, without taking into account the other information sources. With this analysis, the second step can be performed. The next Figure shows the two ontologies about the systems described in the previous procedure. Each Ontology defines its own classes and relationships. Ontology 1 has milk, liter and truck (with the subclasses) classes and the the_amount relation to represent the domain. Ontology 2 has the same classes and relations except for liter class that is replaced by the gallon class indicating different milk measures.

---------- Classes ----------
Milk
(Define-Class Milk (?X) "the set of types of milks"
:Def (And (Thing ?X)))

Liter
(Define-Class Liter (?X) "the liters" :Def (And (Thing ?X)))

Truck
(Define-Class Truck(?X) "the set of trucks" :Def (And (Thing ?X)))

Truck_With_Refrigeration
(Define-Class Truck_With_Refrigeration (?X) "the set of trucks with refrigeration" :Def (And (Truck ?X)))

Truck_Without_Refrigeration
(Define-Class Truck_Without_Refrigeration (?X) "the set of trucks without refrigeration" :Def (And (Truck ?X)))

---------- Relations ----------
The_Amount
(Define-Relation The_Amount (?Frame ?Value) "the amount expressed in liters" :
Def (And (Liter ?Frame)
(Number ?Value)))

Transporting
(Define-Relation Transporting (?Truck ?Milk ?Liter) "the amount of milk transported by a truck" :
Def (And (Truck ?Truck) (Milk ?Milk) (Liter ?Liter)))

Ontology 1

---------- Classes ----------

Milk
(Define-Class Milk (?X) "the set of types of milks"
:Def (And (Thing ?X)))

Gallon
(Define-Class Gallon (?X) "the set of gallons"
:Def(And (Thing ?X)))

Truck
(Define-Class Truck(?X) "the set of trucks" :
Def (And (Thing ?X)))

---------- Relations ----------

The_Amount (Define-Relation The_Amount (?Frame ?Value) "the amount expressed in gallons" :Def (And (Gallon ?Frame) (Number ?Value)))

Transporting
(Define-Relation Transporting (?Truck ?Milk ?Gallon) "the amount of milk transported by a truck"
:Def (And (Truck ?Truck) (Milk ?Milk) (Gallon ?Gallon)))

Ontology 2

Third stage:
Defining Mappings: In this stage we define the mappings (and relations) between the concepts defined in the global ontology and in the local ontologies. This stage must solve the semantic heterogeneity problems making connections between the two stages.

In our example, the global ontology has the gallon class to represent the metric measure of the milk. The liter class of Ontology1 is not represented in the global ontology and we must include an axiom to relate these classes. A liter equals 0.22 gallon.

(<=>(Liter ?x) (Gallon ?x * (0.22)))

This mapping is performed because users could query the integrated system by asking for information about the amount of milk in liter measure. And the global ontology only has the gallon class. Therefore, when users make queries, the global ontology and the mapping are used to retrieve the information needed.

No mapping is needed for the truck classes in both systems because they have the same name and they denote the same things.

4. CONCLUSION

Our current research on data integration uses the "ontology" concept to help solving the semantic heterogeneity problems. We create a useful and practical method for the construction of a hybrid ontology approach. The method has three main stages: building the shared

vocabulary, building local ontologies and defining mappings. Each stage embodies a number of steps that must be followed. Each step serves like a guide to identify all the cases of semantic heterogeneity and the ways to solve them.

Semantic interoperation is one of the main obstacles to free and full electronic commerce. Understanding what is available is a necessary prerequisite to a successful business transaction. We have described a number of issues involved in supporting the interoperation of computer systems at the semantic level.

Our research is ongoing and there are a number of aspects being analyzed. We aim at including the "context" concept to solve, for example, the homonym problem, different representations, etc. Also, we are working on including similarity functions to find similarity terms within the different local ontologies.

Finally, the method and its extensions need be validated by using more complex examples and real cases for study.

5. REFERENCE

[1] Integration in Biopharmaceutical R&D: Strategies and Technologies. A White Paper. http://www.3rdmill.com/. May 2002.
[2] Cali, A., Calvanese, D., De Giacomo, G.,Lenzerini., M., Naggar, P., Vernacotola, F.IBIS: Data Integration at Work (extended abstract). SEBD 2002.

[3] Cali, A., Calvanese, D., De Giacomo, G., Lenzerini., M. On the Role of Integrity Constraints in Data Integration. IEEE Computer Society Technical Committee on Data Engineering. 2002

[4]Arens, Y., Hsu, C., Knoblock, C. A. Query processing in the SIMS Information Mediator. Advanced Planning Technology, Austin Tate (Ed.), AAAI Press pp. 61-69, Menlo Park, CA, 1996.

[5] Brisaboa, N.R., Penabad, M.R., Places, A.S, Rodríguez, F.J. Ontologías en Federación de Bases de Datos. Novática (ISSN 0211-2124), pp. 45-53. Julio 2002.

[6] Buccella A., Cechich A. and Brisaboa N.R. "Applying an Ontology on Data Integration" , WICC'03, 5th Workshop de Investigadores de Ciencias de la Computacion. Universidad Nacional del Centro de Buenos Aires, Tandil –Argentina, pp. 99-102, Mayo 2003.

[7] Busse, S., Kutsche, R.-D., Leser, U., Weber H. Federated Information Systems: Concepts, Terminology and Architectures. Technical Report. Nr. 99-9, TU Berlin. April 1999.

[8] Carey, M. y colabor. Towards Heterogeneous Multimedia Information Systems: The Garlic Approach. Fifth International Workshop on (RIDE): Distributed Object Management. 1995.

[9] Chandrasekaran, B.; Josephson, R. What are ontologies, and why do we need them? In IEEE Inteligent systems, 1999.

[10] Chawathe, S., Garcia -Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman,
J., Widom, J. The TSIMMIS project: Integration of heterogenous information sources. 16th Meeting of the Information Processing Society of Japan, pp. 7-18, Tokyo, Japan. October 1994.

[11] Cheng Hian Goh. Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Sources. Phd, MIT, 1997. http://ccs.mit.edu/ebb/peo/mad.html - 03/2003.

[12] Gruber T. Ontolingua: A Mechanism to Support Portable Ontologies. Knwoledge Systems Laboratory, Stanford University, Stanford, CA, Technical Report KSL 91-66. 1992.

[13] Hasselbring, W. Information System Integration. Comminications of the ACM. June 2000.
IBM Federated Database Technology – www7b.boulder.ibm.com/dmdd/library/techarticle/ 023haas/0203haas.html – 07/2003.

[14] Haas, L. M., Miller, R. J., Niswonger, B., Tork Roth, M., Schwarz, P. M., Wimmers, E. L. Transforming Heterogeneous Data with Database Middleware: Beyond Integration.
www.almaden.ibm.com/software/km/clio/clio.pdf - 07/2003.

[15] Hasselbring, W. Information System Integration. Comminications of the ACM. June 2000.
IBM Federated Database Technology – www7b.boulder.ibm.com/dmdd/library/techarticle/ 023haas/0203haas.html – 07/2003.

[16] Haas, L. M., Miller, R. J., Niswonger, B., Tork Roth, M., Schwarz, P. M., Wimmers, E.L. Transforming Heterogeneous Data with Database Middleware: Beyond Integration.
www.almaden.ibm.com/software/km/clio/clio.pdf - 07/2003.

[17] Özsu, M.T., Valduriez, P. Principles of distributed database systems, 2nd edition, Prentice Hall, 1999.

[18] Quddus Chong, Judy Mullins, Rajesh Rajasekharan. An Ontology-based Metadata Management System for Heterogeneous Distributed Databases. CS590L – Winter 2002.

[19] Barlow, S. Data Integration. University of Passau. July 24, 2000.