An Ontology Approach to Data Integration

Agustina Buccella and Alejandra Cechich
Departamento de Ciencias de la Computación, Universidad Nacional del Comahue,
Buenos Aires 1400, Neuquén, Argentina
Email: abuccel,acechich@uncoma.edu.ar

Nieves R. Brisaboa
Departamento de Computación, Universidade de A. Coruña,
Campus de Elviña s/n, 15071 – A. Coruña, España
Email:brisaboa@udc.es

Abstract. 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 an ontology by using a hybrid ontology approach.

Keywords: Federated Databases, Ontology, Semantic Heterogeneity.

1. INTRODUCTION

Nowadays, large modern enterprise has different portions of the organization using different database management systems to store and search their critical data. Competition, evolving technology, geographic distribution and the inevitable growing decentralization, all contribute to this diversity. All of these databases are very important for the enterprise and they have different interfaces for their administration. It will be useful for the enterprise to retrieve the information through a common interface to realize, for instance, the full value of the data they contain [18]. The term **Federated Database** emerged to characterize techniques for proving an integrated data access, having a set of distributed, heterogeneous and autonomous databases [17]. We briefly explain these concepts:

- ***Autonomy***: The users and the applications can access to the data through a federated system or by your own local system. The autonomy can be classified in three types [7,20]: *design autonomy, communication autonomy* and *execution autonomy*.
- ***Distribution***: Nowadays most computers are connected to some type of network, especially the Internet, and it is natural to think of combining application and data sources that are physically located on different hosts, but that can communicate through the network.
- ***Heterogeneity***: it can be classified into four categories [31]: *structure, syntax, system, and semantic*. The *structure* heterogeneity involves different data models; the *syntax* heterogeneity involves different languages and data representations and the *system* heterogeneity involves hardware and operating systems. The semantic heterogeneity can be classified as: *semantically equivalent concepts* (the models use different terms to refer to the same concept, e.g. synonyms; the properties are modeled differently by distinct systems, etc.), *semantically unrelated concepts* (the same term may be used by distinct systems to denote completely different concepts) and semantically *related concepts* (generalization/specification, different classifications, etc.). Another similar classification of heterogeneity can be found in [11].

In order to address the problem of semantic heterogeneity previously described, we apply to the idea of ontologies as a tool for data integration. Section 2 introduces the concept of ontologies and discusses different approaches

for data integration. Then we describe our method to build an ontology. A discussion explaining the advantages and limitations of our method is described in Section 3. Future work and the conclusion are discussed in Section 4.

2. DATA INTEGRATION BASED ON ONTOLOGIES

The term "ontology" has long been used in many ways and domains [1,9,13]. In the computer science world the ontologies are introduced by Gruber [15] as an "*explicit specification of a conceptualization*". A *conceptualization* refers to an abstract model of how people commonly think about a real thing in the world, e.g. a chair. *Explicit specification* means that the concepts and relations of the abstract model have been given explicit names and definitions [30]. An ontology gives the name and the descriptions of the entities of specific domains using predicates that represent relationship between these entities. It provides a vocabulary to represent and communicate knowledge about the domain and a set of relationship containing the term of the vocabulary at a conceptual level. Therefore, an ontology might be used for data integration tasks because of its potential to describe the semantic of information sources and to solve heterogeneity problems [11,31].

On the other hand, the concepts *data integration*, *application integration* and *application interoperability* are similar but we must differentiate them [8]. ***Data integration*** is concerned with unifying data sharing some common semantics but are originated from unrelated sources. ***Application interoperability*** attempts to standardize the interfaces among stand-alone applications so that the data generated from one application can flow as the input to another application. ***Application integration*** involves aspects of data integration and of application interoperability. In this paper we mainly focus in the first one.

There are many systems designed to address the needs of data integration. The developers of each system have made different choices about the best way to provide the needed services. Some of the most popular systems are: the Garlic System [8], the TSIMMIS System [10], the ObjectGlobe System [26], the SIMS System [4], etc. In [22] there is a brief explication of the first three with an analysis of their advantages and disadvantages. All of them have been created to solve any heterogeneity level. As we have already said, we concentrate only in semantic heterogeneity and for that, there are two different branches: *with ontologies and without ontologies*. On the "without ontologies" branch, there are several research works with different level of detail, see [2,3,12,19,25].

Data Integration using Ontologies

There are a lot of advantages in the use of ontologies for data integration. Some of them are [22,27]: 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 [5,21,29,32].

Then, we describe our method [6] for building the structure of the ontology. Figure 1 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 [16] to represent the ontology example.

First stage: Building the shared vocabulary: As Figure 1 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, Figure 2 shows an example of two similar systems containing information about transporting milk. In this example, we can clearly see two semantic problems [31]: *property-type* mismatch and *different classification*.

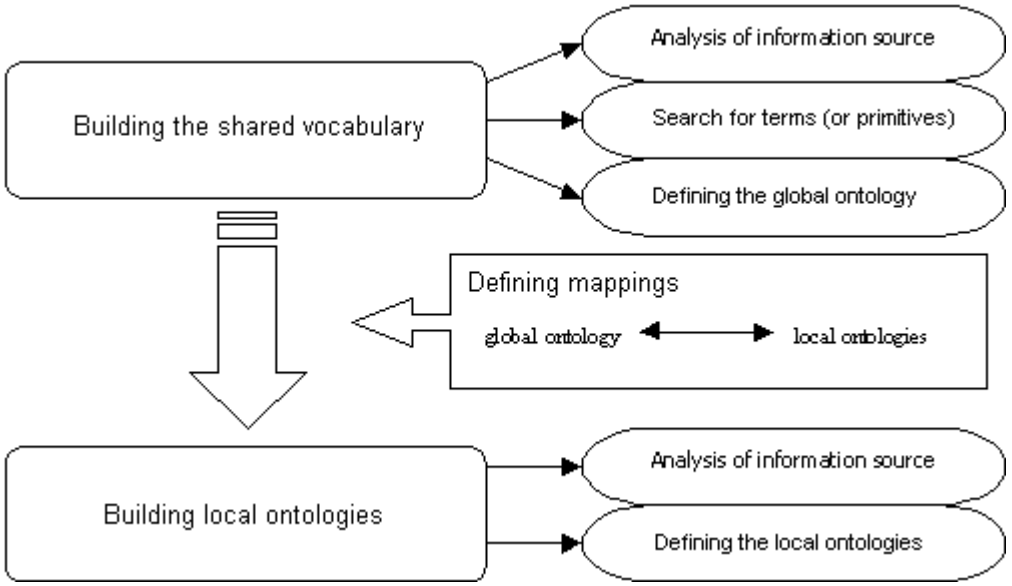

Figure 1: Ontology Construction Method

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.

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, System 2 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 subclasiffications of System 1.

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 see what 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. Figure 3 shows the global ontology generated from the two systems defined in Figure 2.

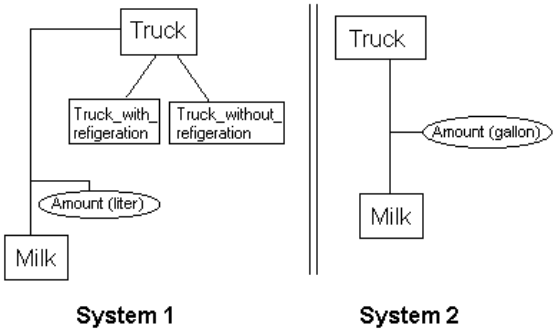

Figure 2: Two systems with different representations

```
;;; ------------------ 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)))
```

Figure 3: The 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. Figure 4 shows the two ontologies about the systems described in Figure 2. 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.

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 (Figure 3) has the *gallon* class to represent the metric measure of the milk. The *liter* class of Ontology 1 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.

3. DISCUSSION

Our method serves as a practical guide to analyze integrated data taking advantage of the use of ontologies. The ontologies allow the capturing of all the semantic information provided by the system. Several semantic problems can be localized and solved when our method is followed. Two of them, *property-type* mismatch and *different classification*, have been described in the last section.
As our proposal is based on an hybrid ontology approach, it has two main advantages:
(1) new information sources can be added without need of modification. Only the terms and relations (of the new source) that are not in the global ontology must be added.

Ontology 1	Ontology 2
;;; ---------- 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)))	;;; ---------- 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)))

Figure 4: Two ontologies

Also, the local ontology and the mappings among the new added terms must be defined.
(2) the shared vocabulary and the mappings among the local ontologies make them be comparables.
Our method must still solve some problems referred to the data integration, for example, when two terms are synonyms or homonyms. Synonyms correspond to the case when two different words have the same meaning and homonyms correspond when two systems use the same word to denote different meanings. We are working on finding specific methods to determine and solve these problems. We are implementing the use of the feature-based models [23] because they have been proposed by cognitive psychologists who judge similarity in terms of distinguishing features of concepts or objects, such as properties, roles and rules. These models are based on the Tversky's model [28] which defines a similarity measure as a feature-matching process. It produces a similarity value that is not only the result of common features, but also the result of the differences between two objects. Two functions are used to determine if two terms or relations are synonyms. Both functions relate two terms each bellowing to its ontology. These functions compare three different aspects between two terms: *parts*, *functions* and *attributes*. The *parts* are structural elements of a concept (or term), such as "roof" and "floor" of a building; the *functions* represent the purpose of the concept; and the *attributes* correspond to additional characteristics of a concept.
Also, we are extending our method to include the idea of *contexts* [24] assuming that each term is true or false according to the context it is in, that is, the context determines the truth or falsity of a statement as well as its meaning.

One main advantage of contexts is the avoidance of the homonym problem. If two systems use the same word (term) each denoting different meanings they will be in different contexts and they need not be compared.

Each ontology might be related to several contexts indicating the different roles of one database. For example, the *use cases* of a UML specification [14] might be the source to obtain some of the contexts. Each context contains a series of terms included in the ontology. Then, we will define relationships among the contexts of different ontologies. Thus, only the terms included in the related contexts will be compared.

Accordingly, these two new extensions to our approach help finding and solving more heterogeneity problems. The inclusion of the similarity functions gives a more precise comparison among the terms of different ontologies. Also, the combination of the use of ontologies and contexts provides a higher degree of semantic information needed for a consistent data integration.

4. CONCLUSIONS

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 d***efining 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.

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. REFERENCES

1. 3rd Millennium, Inc. Practical Data 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. Constantinescu, C., Heinkel, U., Rantzau, R., Mistchang, B. SIES: An Approach for a Federated Information System in

Manufacturing. www.informatik.uni-stuttgart.de/ipvr/as/personen/constantinescu/ise2001.pdf - 03/2003.
13. Event/Process-Based Data Integration for the Gulf of Maine. Campobello Island, New Brunswick. June 12 – 14, 2002. www.spatial.maine.edu/~bdei/bdeippr.pdf - 03/2003.
14. Fowler, M. and Scott, K. UML distilled, Addison-Wesley 1997.
15. Gruber, T. A translation approach to portable ontology specifications. Knowledge Acquisition 1993 -5(2): pp. 199–220. http://ksl-web.stanford.edu/KSL_Abstracts/KSL-92-71.html - 07/2003.
16. Gruber T. Ontolingua: A Mechanism to Support Portable Ontologies. Knwoledge Systems Laboratory, Stanford University, Stanford, CA, Technical Report KSL 91-66. 1992.
17. Hasselbring, W. Information System Integration. Comminications of the ACM. June 2000.
18. IBM Federated Database Technology – www7b.boulder.ibm.com/dmdd/library/techarticle/023haas/0203haas.html – 07/2003.
19. 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.
20. Özsu, M.T., Valduriez, P. Principles of distributed database systems, 2nd edition, Prentice Hall, 1999.
21. Quddus Chong, Judy Mullins, Rajesh Rajasekharan. An Ontology-based Metadata Management System for Heterogeneous Distributed Databases. CS590L – Winter 2002.
22. Barlow, S. Data Integration. University of Passau. July 24, 2000.
23. Rodriguez, A., Egenhofer, M. Determining Semantic Similarity among Entity Classes from Different Ontologies. IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 2, March/April 2003.
24. Rodriguez, A., Egenhofer, M. Putting Similarity Assessments into Context: Matching Functions with the User's Intended Operations. Context 99, Lecture Notes in Computer Science, Springer-Verlag, September 1999.
25. Tejada, S., Knoblock, C.A., Minton, S. Learning object identification rules for Information Integration. *Information Systems* Vol. 26, Nº 8, pp. 607-633, 2001.
26. The ObjectGlobe Homepage. http://www.db.fmi.uni-passau.de:8000/projects/OG/ - 05/2003.
27. Adams, T., Dullea, J., Clark, P., Sripada, S. and Barrett, T. Semantic Integration of Heterogeneous Information. Sources Using a Knowledge-Based System. In Proc 5th Int Conf on CS and Informatics (CS&I'2000), 2000.
28. Tversky, A. Features of Similarity. Psychological Rev., vol. 84, (pp. 327-352). 1977.
29. Visser, U., Stuckenschmidt, H., Schlieder, C. Interoperability in GIS – Enabling Technologies. 5th AGILE Conference on Geographic Information Science, Palma (Balearic Islands, Spain) April 25th-27th 2002
30. Visser, U. and Schlieder, C. Modelling with Ontologies. In: The Ontology and Modeling of Real Estate Transactions in European Juristictions Ashgate - 2002, to appear
31. Cui, Z. and O'Brien, P. Domain Ontology Management Environment. In Proceedings of the 33rd Hawaii International Conference on System Sciences - 2000
32. Cui, Z., Jones, D. and O'Brien, P. Issues in Ontology-based Information Integration. IJCAI – Seattle, USA • August 5 2001.