A Cooperation Model to Support Distributed Global Schemes

Helena Graziottin Ribeiro[1], Daniel Luis Notari[1], Joel Luis Carbonera[1]

UCS - University of Caxias do Sul, Centro de Ciencias Exatas e Tecnologia (CCET), Rua Francisco Getulio Vargas, 1130 - B. Petropolis, 95070-560, RS, Brazil

Abstract. The integration of heterogeneous data (issued from different sources) can be made by one or more global schemes constructed in a authomatic or semi-automathic way. HetSIS is an heterogeneous system based on cooperative data agents defined to integrate heterogeneous data. Each data agent represents an specific data source, which cooperates with other data agents using distributed global schemes to access integrated data. A cooperation model, composed by cooperation protocols and a metadata model, is proposed in order to structure cooperative agents. The protocols are implemented by means of web services. HetSIS architecture and the cooperation model are presented in this paper.

Keywords: heterogeneous databases, global schema, metadata, schema mapping expressions, cooperation model, database integration.

1 Introduction

The integration of data stored in multiple heterogeneous sources is both a challenge issued in the construction of global information systems and a complex task to manage. Besides the autonomy of data source managers, the main problems that have to be faced are due to the structural and semantic differences (representation conflicts) resulting from the heterogeneity of sources. Most of research in this area is related more recently with the matching of structured and semi-structured data supported by mappings or conversions between either different data or representing models, query languages, execution models, etc. The main results obtained in last years are schema matching and schema merging techniques that work individually or may be combined in hybrid approaches [10, 5]. Such proposals are developed for heterogeneous system components that generally manage global schemes or views. These components can be implemented in different architectures as federated databases, data warehouses or mediators [7], according to the goals of data using, the need of integration transparency and the implementation strategies to support materialized or virtual integration. In the virtual integration approach [3], physical data remain stored in local databases while users have single, integrated and non-redundant views over them. Mediators [12] generally concentrate the knowledge about integration, centralizing information about data sources and executing mappings and conversions in a

manual or semi-automatic way. Thus, local data source managers (as database management systems) are isolated from each other, and consequently they are not able to cooperate during the integration process, just receiving queries from and returning results to a wrapper/mediator.

Different systems can cooperate in order to execute a task that can not be accomplished individually or to reach a shared objective. In order to cooperate, systems must be interoperable, know the goals of cooperation and be integrated in the cooperation process. A recent approach to provide interoperability between systems are services. Actually the most popular services are the web services [1, 8] that enable to access systems in a remote way using a communication pattern interface.

HetSIS is a system to integrate heterogeneous data based on cooperative data agents. It has a metadata model to represent global schemes resulting from schema merging. Our integration approach considers the co-existence of multiple global schemes distributed among local data sources, that we call *data agents*. The rest of this paper is organized as follows. Section 2 introduces the approach of multiple global schemes, and presents the general architecture of HetSIS, our heterogeneous data system, and the cooperation model. In section 3 the metadata model we propose to represent global schemes and mapping expressions is described. The mapping expressions and the set of operators used to define them are presented in Section 4, and Section 5 shows the main operations and services proposed to support the cooperation protocols. Section 6 presents some related works, and finally Section 7 shows the conclusions and propose some future works.

2 The HetSIS System

Mediators generally centralize the knowledge about integration and coordinates the process. In this way, data source managers are isolated from each other, and thus they are not able to cooperate during the integration process, just receiving queries from and returning results to a wrapper/mediator. In the Heterogeneous Source Integration System (HetSIS) there is not a centralized mediator. Each data source has an interface that enables it to cooperate with other sources to access integrated data. Next sections present our approach and the HetSIS architecture.

2.1 Multiple global schemes

A typical global schema approach to integrate heterogeneous databases proposes the construction of a single structure (global schema) to unify different local schemes developed individually for specific data sources (databases or another data sources). An heterogeneous data source integration system can provide several global views over the global schema in order to present different options of integrated data organization for different users or applications. In our proposal, that considers virtual integration approach, these views are called global

schemes. For instance, in Fig. 1 are defined n global schemes: *global schema 1*, *global schema 2*, ... *global schema n*. *Global schema 1* is composed of data issued from *DB A* and *DB B*, while *global schema 2* contains data from *DB A*, *DB B* and *XML file*. Thus, our approach considers that a system can support multiple global schemes those may share local or global terms or attributes.

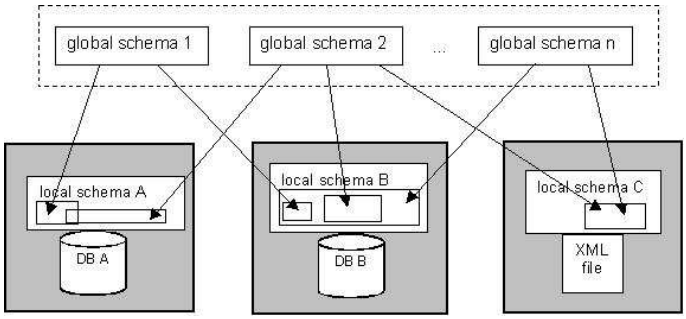

Fig. 1. Multiple global schemes integrating local sources

2.2 HetSIS Architecture

The Heterogeneous Source Integration System (HetSIS) architecture is presented in Fig. 2. HetSIS is composed by a set of data agents that cooperate to answer global queries. Each data agent encapsulates a local data source (as DB1, DB2, XML file, etc.) and manages the access to local data. Local data sources are data repositories (as databases) associated with an structure describing their contents (a local schema) that is represented in a canonical model (XML-Schema).

The main components of a data agent are repositories of partial global schemes and repositories of global schema members with their respective managers. The description of what local data is considered in global schemes is done in *partial global schemes*. Each data agent has one repository of partial global schemes, and one partial global schema contains the description of local data considered in global schemes already defined in the system. The global information maintained by an agent is about its local data, and may represent only parts of different global schemes (GS). HetSIS supports a set of n global schemes: $GS = \{GS_1, GS_2, ..., GS_n\}$. Each global schema GS_i can be composed by a set of *k, l, m, ...* partial global schemes (PGS) from different data agents:

$GS_1 = PGS_{1.1} + PGS_{1.2} + ... + PGS_{1.k}$

$GS_2 = PGS_{2.1} + PGS_{2.2} + ... + PGS_{2.l}$

...

$GS_n = PGS_{n.1} + PGS_{n.2} + ... + PGS_{n.m}$

In this way, a data agent knows what of its local data items are being considered in global schemes. *Global schema members* are repositories containing the

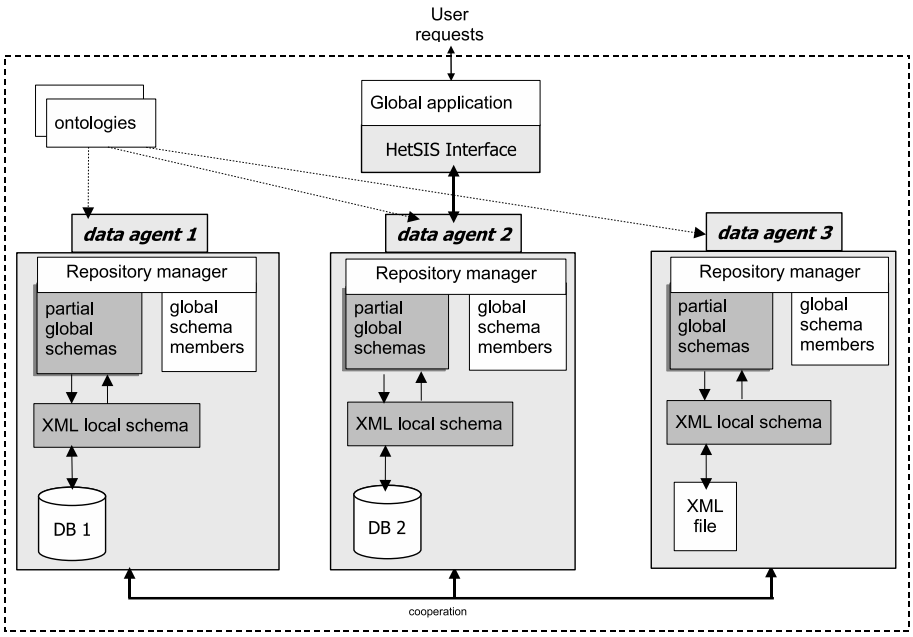

Fig. 2. Architecture of the HetSIS System

identification of global schemes, and lists of all data agents concerned. For one global schema, each data agent referenced *knows* all the others also referenced in the same global schema. These repositories are replicated in all agents and must be updated when a global schema is created or modified. The construction of a global schema is coordinated by the HetSIS interface, and the access to integrated data is a cooperative process. In order to improve global schema construction, we are studying how to integrate ontologies in the data agents. Global applications receive global requests from users. A global application interacts with data agents by a HetSIS Interface that selects a data agent defined in a global schema being used. This agent consults a repository of global schema members to know the other agents managing data for the same global schema and starts the cooperation sending them the global request. A local source may be active or not in the integrated system. When it is not active, its data are not considered to answer the global request. An specific global request will be executed in all local data sources actives at the moment the request arrives.

2.3 Cooperation model

Each data agent has the knowledge necessary to communicate with other agents, that is, it knows *"why, how and when to cooperate"*. For instance, to access data through a global schema GS1(name, address, status) - *why* - the data agent that

receives the request - *when* - send messages to other data agents having local data accessed by means of the same global schema - *how* - asking for information to be sent to HetSIS interface. More then communicate, agents cooperate to provide access to the integrated data. The "knowledge" about integration is structured by means of a cooperation model composed by:

- a metadata model (further explained in section 3), that represents data mappings among heterogeneous data sources. Instances of metadata are stored in distributed repositories managed by agents, and each time a new relation among elements or attributes is defined these repositories are updated;
- cooperation protocols (further explained in Section 5), that are defined to create and update global schemas, and to answer global queries. We are currently working in the protocol to update local data when global updates are requested by applications. The cooperation between the data agents is defined by some protocols and it is supported by web services.

3 Metadata model

The main problems in schema integration are schema matching and schema merging [11]. Schema matching concerns the identification of correspondences, or mappings, between schema objects (like entities), and schema merging is the process to create a unified schema based on the identified mappings. The execution of schema matching and merging is studied in different works as [11, 10, 5] and it is not focused in our work. For now, we consider that one or more global schemes are the result of a schema merging process.

The metadata model represents information about local sources (by means of classes *Source, Objects, Attribute, Reference* and *Collection*), the mappings defined to relate them (class *Relation*), and global terms associated with these relations (class *Global_Terms*). *Partial global schemes*, that are instances of the metadata model containing mapping information between global terms and local data, describe for each data agent what of its local data is concerned by global schemes. Local data sources are not associated with specific representations (relational, object-oriented, XML, etc), they are just a set of elements related by some structure. We consider that local schemes are translated (before the integration) to a common representation for all local schemes (a canonical model), defined in XML-Schema. Figure 3 presents the UML class diagram of the metadata model. A global schema is a set of relations (mappings) between elements (objects or attributes) of local schemes and global terms. In the following, classes representing mappings are detailled:

- *Global_schemes*: defines global schemes for an integrated system. A global schema is composed by *Global_Terms* that relate elements of local schemes stored in different *Sources*;
- *Global_Terms*: defines global schema elements according to an specific *Vocabulary*. As a global term establishes binary *Relations* between local attributes and/or another (global) terms, relations between many attributes can be defined through the composition of other relations.

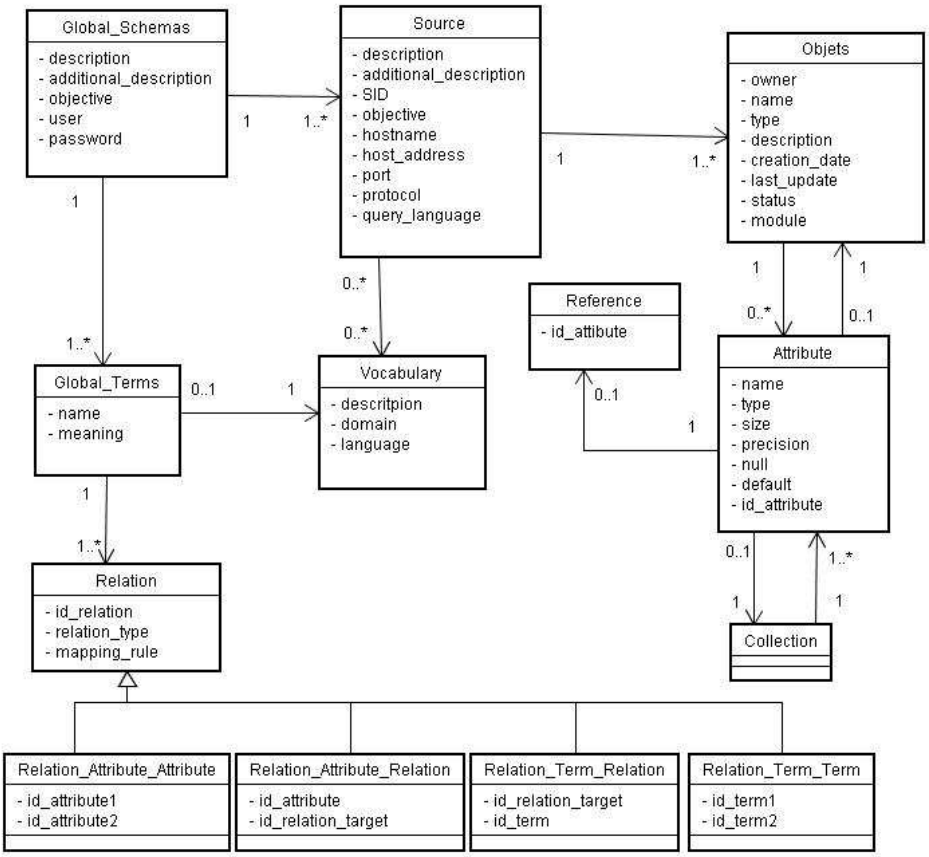

Fig. 3. The metadata model

– *Relation*: supports the relations established between different *Global_Terms* and/or *Attributes*. Relation types can be association, union, equality, equivalence, composition, concatenation, etc. The subclasses of *Relation* support binary relations in order to express more complex expressions, with many mapping operators and many attributes or other relations, grouped two by two: relations between two attributes (*Relation_Attribute_Attribute*), one attribute and one relation (*Relation_Attribute_Relation*), one term (*Global_Term*) and one relation (*Relation_Term_Relation*), and between two terms (*Relation_Term_Term*). Each binary relation composes an intermediate relation that can be named and stored to be reused. The mapping rule between terms/attributes can be described as an expression. Section 4 presents the mapping operators considered and an example.

The *Source* class describes data sources that store data accessed through of global schemas. A *Source* contains *Objects* (tables, views, classes, XML elements, etc) and may be described using different *Vocabularies*. An *Object* is described by its *Attributes* (class attributes, table columns, etc). An *attribute* can be referenced by another attribute, or can be defined as a *Collection* of other *Attributes* or even as another *Object* (object composition).

4 Mapping Expressions for Schema Matching

The resulting global schema is the correspondence established between local schema elements. This correspondence concerns the schema elements (entities, classes, attributes, etc) and the mapping rules defined between these elements. We consider the mapping operators detailed in the Tab. 1: union, concatenation, equivalence, condition and relational operators.

Operator	*Symbol*	*Description*
union	$\bigcup$	join instances of different local attributes
concatenation	+	concatenate instances of different local attributes
equivalence	=	associate one or more instances or relations to a global term
condition	where	restrict the instances to be integrated
conditional operators	==, >=, <=, <>, <, >, not, and, or,	used in conditions: equal, greater/less or equal, different, less, greater, etc.

Table 1. Mapping operators supported by the metadata model

Consider two data sources, DB1 and XMLDB2, that store data about people. DB1 is a relational database with the table **People**, and XMLDB2 is a XML file that defines the elements **People** and **PersonIdentification** to represent similar data. These sources are integrated in the global schema *GlobalPerson* including global terms *name, cpf* and *rg* (Fig. 4). Local schemas are firstly converted to corresponding descriptions in XML-Schema and after integrated in a global schema represented also in XML-Schema.

The global attribute `name` of the global schema GS1 corresponds to local attribute `name` of table People in the source DB1, plus the concatenation of local attributes <`firstname`> and <`lastname`> of the element <`People`> in the source XMLDB2. The resulting data must be the union of all `name` data in DB1 and all `firstname` data concatenated with `lastname` data in XMLDB2. Once established, this relation is stored as the following rule in the metadata repository:

`name.GlobalPerson` = `name.People.DB1` U
`(firstname` + `lastname).People.XMLDB2`

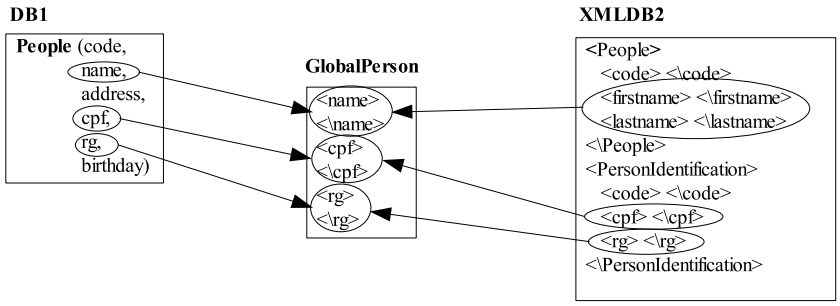

Fig. 4. Global schema GlobalPerson integrating data from DB1 and XMLDB2

The representation of n-ary relations in the metadata model is made by the binary composition of other relations. These kind of representation is usefull in the parser implementation. Thus, *Global_Term* class can be used to represent intermediate relations that are themselves composed two-by-two in order to support mappings with many attributes. For instance, the concatenation between *firstname* and *lastname* can be represented as the intermediate global term *global_term1*, which is associated with the DB1 attribute *name* by a union relation in order to define the global term *global_term_nameperson*:

`global_term_nameperson` = `name.People.DB1` U `global_term1`

`global_term1` = `(firstname + lastname).People.XMLDB2`.

The global term *global_term_nameperson* can be used as an internal symbol by the parser, and is equivalent to global term `name.GlobalPerson` defined above.

5 Cooperation protocols

The cooperation protocols are defined by means of web services, in order to support the answer a global query operation. The main operations already defined to be executed by data agents are:

- create a global schema: a data agent calls the CreateGlobalSchemaService and receives the corresponding partial global schemas, and the user executes data mappings;
- update a global schema: a data agent calls the UpdateGlobalSchemaService and receives the corresponding partial global schemas, and the user updates data mappings;
- answer a global query: a data agent calls the GlobalQueryService.

The Fig. 5 shows the sequence of web services invocated to execute the request of global data, by means of a global query: a data agent (host A), after reading the partial global schema, calls the GlobalQueryService that send global queries to data agents concerned, and receives the corresponding data.

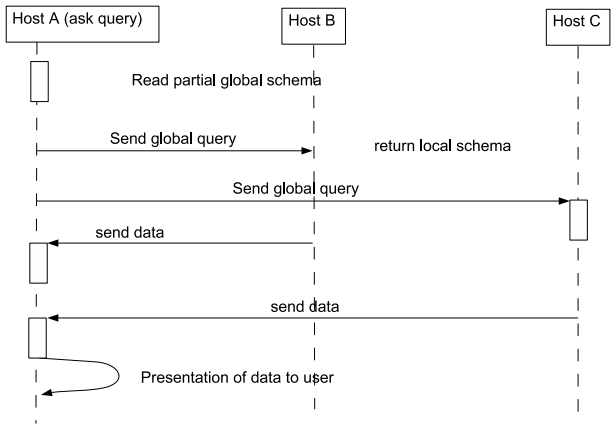

Fig. 5. The protocol to answer a global query

6 Related works

Many works propose models to represent global schemes. Tree or graph models as OEM [9] and YATTA [2] has been largely used to represent semi-structured data because they have a simple structure describing concepts as elements, tables or objects (nodes) and their relations (edges) in a high level. OEM is an object-oriented model that uses object labels to represent both class information and attributes of an object (it uses labels in a place of a schema). In data integration it is important that the model gives a precise definition of database schema. YATTA is a tree model that provides two levels of abstraction: the schema level and the data level. The metadata model we propose is simple, not dedicated to a specific domain but has a high expressivity. It is defined as a class model, represented in XML-Schema, and the mappings between integrated local schemes are maintened in a explicit way. These mappings are defined by transformations in YATTA. The use of agents is considered in recent works of data integration [4], and cooperation is an approach to support it [6]. We are currently studying cooperation models in the context of multiagent systems to improve our work.

7 Conclusions and future work

Our main contribution is a distributed structure to support distributed global schemes. We propose a metadata model to represent global schemes for the integration of heterogeneous data sources in the system HetSIS. Multiple global schemes can be defined and stored in distributed repositories. Thus, a global schema is not centralized in a single repository, but distributed between agents having data mapped in such schema. We are currently implementing a prototype using Java and specific APIs to manipulate XML documents containing

local schemes, in order to present these information to users by means an interface that permits the global schema construction. Now, the construction is executed manually, and in future works we intend to integrate matching schema algorithms [5] to realize this task in a semi-automatic way. In this work, we propose a set of operators used to define mapping expressions that are represented in a metadata model. A schema matching algorithm could use data represented in this model to execute automatic or semi-automatic mappings, for instance, with the aid of ontologies. We are implementing the repositories of partial global schemes and global schema members, and the mechanisms to manipulate them. A parser to analyse the mapping expressions used un global schema construction is also under implementation. As another future work, it is possible to export the global schemes represented in XML-Schema to a corresponding structure in RDF in order offer by means of services (web services) to other applications. We are also extending the metadata model to support the global schema evolution.

References

1. G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services: concepts, architectures and applications*. Springer-Verlag NY, 2004.
2. M. Boyd, C. Lazanitis, S. Kittivoravitkul, P. McBrien, and N. Rizopoulos. AutoMed: A BAV Data Integration System for Heterogeneous Data Sources. In *Proc. of 16th International Conference on Advanced Information Systems Engineering (CAISE04)*, LNCS, Rina, Latvia, June 7-11 2004. Springer-Verlag.
3. A.Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, 2001. Springer-Verlag.
4. L. Liu, H. Song, and L. Bai. Agent-based integration of heterogeneous database systems. In *Proc. of 9th International Conference on Computer Supported Cooperative Work in Design*, pages 1034–1037 Vol.2. IEEE, May 24-26 2005.
5. S. Mergen and C.A. Heuser. Matching of XML Schemas and Relational Schemas. In *Anais do 19 Simpsio Brasileiro de Banco de Dados*, pages 322–334, Braslia, October 18-20 2004. SBC.
6. C. Miao, M. Shi, and J. Shen. An Agent-Based Approach for Cooperative Data Management. In X.Zhou et al., editor, *Proc. of Frontiers of WWW Research and Development (APWeb2006)*, volume 3841 of *LNCS*, pages 133–144. Springer-Verlag, 2006.
7. H.G. Molina, J.D. Ullman, and J. Widom. *Database System Implementation*, chapter 11 - Information Integration. Prentice-Hall, 2000.
8. W3C org. Web Services Architecture, 2009. http://www.w3.org/.
9. Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object exchange across heterogeneous information sources. In Philip S. Yu and Arbee L. P. Chen, editors, *Proceedings of the Eleventh International Conference on Data Engineering, March 6-10, 1995, Taipei, Taiwan*, pages 251–260. IEEE Computer Society, 1995.
10. R.A. Pottinger and P.A. Bernstein. Merging Models Based on Given Correspondences. In *Proc. of the 29th. VLDB Conference*, pages 826–873, Berlin, Germany, September 9-13 2003. Morgan Kaufmann.
11. E. Raham and P.A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, 2001. Springer-Verlag.
12. G. Wiederhold. Mediators in the architecture of future information systems. In *IEEE Computer*, volume 25, pages 38–49. IEEE, March 1992.