Hybrid Architecture of OWL-Ontologies for Relational Data Sources Integration

Nora Maiz, Laboratory ERIC, Université Lumière Lyon2, 5 avenue Pierre Mendès France, 69676, Bron Cedex, France; E-mail: nmaiz@eric.univ-lyon2.fr
Omar Boussaïd, Laboratory ERIC, Université Lumière Lyon2, 5 avenue Pierre Mendès France, 69676, Bron Cedex, France; E-mail: omar.boussaïd@univ-lyon2.fr
Fadila Bentayeb, Laboratory ERIC, Université Lumière Lyon2, 5 avenue Pierre Mendès France, 69676, Bron Cedex, France; E-mail: bentayeb@eric.univ-lyon2.fr

ABSTRACT
Data integration is one of the most important tasks in the data warehousing process. The use of ontologies in the mediation process allows semantic and structural integration. In this paper, we propose a new mediation system based on an hybrid architecture of ontologies modelled according to GLAV (Generalized Local As View) model. The hybrid architecture builds a local ontology for each data source and a global ontology viewed as a mediator. The integration model defines how sources, local and global ontologies are mapped. So we propose an ascending method for building ontologies, which facilitates the semantic reconciliation between data sources. Moreover, we use OWL (Ontology Web Language) for defining ontologies and mappings between data sources and ontologies. User queries are expressed in our specific language which handles global ontology concepts and local ontologies properties since we assume that the user is expert in its domain. Queries are decomposed by the rewriting algorithm to obtain a set of equivalent subqueries that are sent to the corresponding sources for execution, and after that recomposed to obtain the final result.

1. INTRODUCTION
In a data warehousing process, data integration is an important phases. Centralized data warehouse is a solution for companies that handle static data. However, when data change, this solution becomes not practical because of the refreshment cost.
We think that data integration by mediation can solve this problem and allows to construct a mediation system for building analysis context on-the-fly using data from their real sources.

In this paper, we treat only the first part, which concerns the building of the mediation framework. It consists in creating a mediator based on ontologies. The use of ontologies in the integration by mediation is not recent [3, 4], it allows to implement a structural and semantic integration. There are several architectures based on ontologies in integration systems [1, 2, 16], approaches with only one ontology as in the case of system SIMS [5], approaches with multiple ontologies as in OBSERVER [6] and hybrid architecture which associates a local ontology for each data source and a global ontology to link them [7]. The later is interesting because it is flexible for updates and there is no need to define mappings between local ontologies. Several structural models can be applied on this architecture: GAV (Global As View) [9, 10, 11], LAV (Local As View) [12, 13, 14, 6, 8]. The advantages and disadvantages of these two approaches are opposite [8]. LAV is flexible for updates but the construction of query’s answers is complex, contrary to the construction of answers in a system adopting an approach GAV which simply consists in replacing the predicates of the query global concepts by their definition. GLAV (Global-Local As View) [15] is the combination of GAV and LAV. It inherits the query unfolding property of GAV, maintains independence between data sources and allows to indirectly computing mappings between them. It uses views in local and global levels. The query processing in this model is only feasible when the query is expressed in a language that takes into account global and local levels.

In this context, we propose an ascending method for building ontologies starting from the local ones, then we use these ontologies to build manually the global ontology and define mappings between global and local ontologies (figure 1). We use OWL (Ontology Web Language) to define ontologies and their mappings. Our goal is to use the ontologies terminology to formulate user queries. To reach this goal, we propose a query language based on global ontology concepts and local ontologies properties. The problem of mediator using several ontologies according to the GLAV model is the query rewriting and the way how the obtained results are combined. For this end, we propose our query rewriting algorithm, which enables to re-formulate user queries to queries comprehensive by the mediator.

Our work lies within the scope of a project of virtual data warehousing of banking data in LCL - Le Crédit Lyonnais (French bank). The purpose of the project is to manage and improve the decision process in LCL in the direct marketing activities domain. It contains many management applications and databases. The banking data are heterogeneous and change much, so the construction of cubes on-the-fly is pertinent. Each cube represents a specific analysis context.

The remainder of this paper is organized as follows. Section 2 presents our mediation system starting by our approach, which allows to create various ontologies applied to the case of the sources of the LCL. Next, we present our query language. After that, we present our query rewriting algorithm and give an example. The architecture and the implementation of our mediator are exposed in section 3. We finish this article by the section 4 which concludes our work and presents the prospects on new generated problems.

2. ONTOLOGY-BASED MEDIATION SYSTEM
The construction of the mediation system is decomposed into three steps:
1. definition of local and global ontologies
2. definition of a query language
3. definition of the query rewriting algorithm

2.1 Ontologies Development Approach
In this section, we present our approach of ontologies construction using the hybrid architecture modelled according to GLAV model. We also use OWL for the ontologies description. In fact, OWL is capable to describe data and metadata in RDF (Resource Description Framework), so it gathers the RDF description power and the mechanism of reasoning. The approach we propose consists in creating ontologies in an ascending way. We start from local ontologies, and extract a global ontology from the local ones in order to facilitate the semantic reconciliation between sources.

• The first phase consists in creating local ontologies. It contains two steps: (1) the analysis of sources; and (2) the definition of ontology concepts. The first step is a complete analysis of each source independently. The analysis consists in searching primitives used in sources, implicit information, its storage, its significance and its relation with other terms. After that, we define concepts which will constitute the ontology hierarchy, their relations and constraints on their use.

• The second phase is the extraction of the global ontology starting from various concepts used in local ontologies. It contains two steps: (1) local ontologies analysis; and (2) selection of all concepts and solving semantic conflicts. The first step is a complete analysis of local ontologies. Note that, ontologies analysis is easier than that of data sources. After concepts selection, the expert solves all kinds of heterogeneity (naming conflicts, confounding conflicts and/or scaling conflicts) to determine global ontology concepts.

• The third phase, which represents the core of the system, consists in defining mappings between the global and local ontologies. The global ontology is built...
from local ontologies. So, in order to identify the original ontological source of concepts, we use annotations. OWL enables the annotation of concepts and properties according to predefined meta data schema.

Our study is limited on relational data sources, where tables are represented by OWL classes. Relationships between classes are represented in OWL by owl:ObjectProperty and owl:DatatypeProperty. OWL properties can represent various attributes and constraints in the relational schema. They also represent attributes by Datatype. If the attribute is a primary key constraint, then a functional characteristic will be added. In addition, we use owl:ObjectProperty to represent foreign keys attributes. Therefore, we obtain two ontologies representing the two relational data sources. The process of ontologies development must be particularly reliable for the global ontology construction. In fact, this ontology ensures the connection between various local ontologies and contains the knowledge for the query formulation and the data warehouse construction. The LCL has two relational data sources, each one contains two tables. The OWL schema is represented in the following table:

<table>
<thead>
<tr>
<th>Tables</th>
<th>Equivalent OWL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaborator</td>
<td>owl:Class rdfs:ID=&quot;Collaborator&quot;</td>
</tr>
<tr>
<td>MarketingDemand</td>
<td>owl:Class rdfs:ID=&quot;MarketingDemand&quot;</td>
</tr>
<tr>
<td>Person</td>
<td>owl:Class rdfs:ID=&quot;Person&quot;</td>
</tr>
<tr>
<td>Profile</td>
<td>owl:Class rdfs:ID=&quot;Profile&quot;</td>
</tr>
</tbody>
</table>

2.2 Query Language

The use of the global ontology as a model for query reformulation is not new. It can be more intuitive for the users. Our system allows queries to exploit concepts of the global ontology and properties of local ontologies. A basic user query is in the form:

Concept ^ Property ^ Concept or only Concept.

2.3 Query Rewriting

The GLAV Approach corresponds to each concept ConceptG or Vg from the global ontology a concept ConceptL or a view Vg from the local ontology. A query expressed in terms of global ontology can not be always reformulated in a view from the local ontology only if the query is expressed in terms of the global and the local schemas. For that we propose the preceding query language (see example in section 2.3.2) and the following rewriting algorithm.

2.3.1 Rewriting Algorithm

The user query expressed in our language query will be rewritten by our algorithm to obtain a set of linked subqueries. If a concept in the user query is not linked with the preceding ones, it will be excluded. Semantically, this exclusion tends to make the query coherent. A coherent query is decomposable into subqueries, and of which its results can be recomposed. The query rewriting can be seen as a mapping between the global ontology and local ontologies.

- **Formal Algorithm:** Formally, a based-ontology mediation system O is a triplet (G, S, M_g,s) where G is the global ontology, S is the set of local ontologies and M_g,s are mappings between the global ontology G and local ontologies S in O.
- **Global ontology:** Let C_g be the set of the hierarchic concepts of the global ontology, An_g the set of annotations, and Annotation a function defined from C_g to An_g.
- **Local ontologies:** Let S be a set of n local ontologies S1, S2, ..., Sn. We note A_i the set of a local ontology concepts. A_i is the union of the A_i of the n local ontologies. Local ontologies concepts are linked by a set Rs of properties defined in A_i * A_j. Let Rs be the set of all properties sets Rs. Let An_i be the set of annotations and let Wrapping be the function defined from An_i to An_g which associates to each concept an annotation.
- **Mappings:** The mapping M_g,s defines how the concepts of the global ontology G and concepts of the local ontology S_i are linked. M_g,s is a function from C_g to S_i.
- **Query language:** Queries are expressed in terms of a query language Q_g. In our system, queries are conjunctions of global ontology concepts and local ontologies properties, so we obtain two types of queries:

### Algorithm 1 Query rewriting

1. Entry: $Q_u =$ query
2. Result: $Q[k] =$ The set of k queries deduced from $Q$
3. $k = 1$, $Q[1] = Q$, NoCorrespondent $\rightarrow$ True
4. for all $Q_i (i = 1, \ldots, n) \in Q[0]$ do
5. 6. if $(i)$ is Local then
7. for all $Q_j (j = 1, \ldots, m) do$
8. 9. $\Psi$: The set of subsumed, subsuming or equivalent concepts $C_j$ of $Q_i$
10. $\Omega$: The set of all $C_j$ concepts obtained using $C_i$ concepts of $\Psi$ such as:
11. Annotation ($C_j$): Annotation ($C_i$) and $C_j \in Q_i$
12. for all $C_j \in \Omega do$
13. 14. if CorrespondConcept($C_j$, $C_i$) $\neq$ False then
15. 16. RsCorrespondent $\rightarrow$ False
17. 18. $k^{++}$
19. 20. $Q[k] \rightarrow Q[k] \cup Q_i$ in the role which links $C_j$ and $C_i$
21. 22. $\forall l \in RsCorrespondent (C_j, C_i) \in T[l]$
23. 24. end if
25. 26. end for
27. 28. if NoCorrespondent then
29. 30. $Q = Q - Q_i$
31. 32. end if
33. 34. else
35. 36. if CorrespondSubQuery($Q_i$, Q) $\neq$ False then
37. 38. add the correspondence ($C_j$, $C_i$) in $T[k]$
39. 40. end if
41. 42. end if
43. 44. $Q = (Q - Q_i)$
45. 46. end if
47. 48. end for
49. 50. end for
51. 51. Else ($Q_i$ is Global $\rightarrow$
52. 53. for all $Q_j (j = 1, \ldots, m) do$
54. 55. if CorrespondSubQuery($Q_i$, Q) $\neq$ False then
56. 57. add the correspondence ($C_j$, $C_i$) in $T[k]$
58. 59. end if
60. 61. end for
62. 62. end for
63. 63. end if
64. 64. return ($Q_u, T$)
The reasoning mechanism of OWL, helps our algorithm to obtain a set of $Q_{GAV}$ and/or $Q_{LAV}$ subqueries, which are equivalent semantically. The goal of the user query rewriting is to eliminate $Q_{GAV}$ subqueries, which have not any relationship with other ones in the same query. Function2 has as parameters two concepts $C_i$ and $C_j$ and gives, as result the role (if it exists), which links them. Function1 returns two equivalent concepts or two concepts linked by a role. Our algorithm uses all global ontology concepts and local ontologies roles to provide a set of equivalent subqueries. For each subquery $Q_i$ of the user query $Q$.

\begin{algorithm}
\caption{Function1: CorrespondSubQuery($Q_i$,$Q_j$)}
\begin{algorithmic}
\STATE {1. Entry: $Q_i,Q_j : TwoSubqueries, G = \{C_1, C_2, \ldots, C_m\}, S = \{C_1, C_n, \ldots, C_p\}, R = \{r_1, r_2, \ldots, r_k\}$}
\STATE {2. Result: $(c,c_r): The relationships between Q_i and Q_j$}
\STATE {3. for all $C_i \in Q_i$ do}
\STATE {4. for all $C_j \in Q_j$ do}
\STATE {5. if $C_i = C_j$ then}
\STATE {6. return $(C_i,c_r)$}
\STATE {7. else}
\STATE {8. if CorrespondConcept($C_i$, $C_j$) then}
\STATE {9. return $(C_i,c_r)$}
\STATE {10. else}
\STATE {11. return $\Phi$}
\STATE {12. end if}
\STATE {13. end if}
\STATE {14. end for}
\STATE {15. end for}
\end{algorithmic}
\end{algorithm}

- If $Q_i$ is $Q_{GAV}$ subquery, that means it contains only one concept; the algorithm selects concepts $C_j$ into all previous subqueries $Q_j$ of $Q_i$. So, we obtain the set of all candidate concepts.
- If the algorithm finds a correspondence between $C_i$ and concepts $C_j$, then, for each concept $C_j \in \Omega$, it verifies if there is a correspondence between this concept and the concept $C_i$. If $C_j$ corresponds to $C_i$, then, it will be replaced by $C_i$. The result is a new rewritten subquery using the corresponding concept.
- If there is no correspondence, the concept $C_i$ is excluded.
- If $Q_j$ is $Q_{LAV}$ subquery, that means it contains two concepts and a role, the algorithm search in previous subqueries of $Q_j$ a corresponding subquery. If there is no one, $Q_j$ is excluded.
- The algorithm processes all subqueries into $Q$. After that, it processes new rewritten queries as the initial query.

\begin{algorithm}
\caption{Function2: CorrespondConcept($C_i$, $C_j$)}
\begin{algorithmic}
\STATE {1. Entry: $C_i,C_j : TwoQ_{LAV}$, $Subqueries, G = \{C_1, C_2, \ldots, C_m\}, S = \{C_1, C_n, \ldots, C_p\}, R = \{r_1, r_2, \ldots, r_k\}$}
\STATE {2. Result: $(r : The Role, which links c_i and c_j)$}
\STATE {3. if $r(C_i,C_j) \in Berr(C_i,C_j) \in R$ then} \STATE {4. return $r$}
\STATE {5. else} \STATE {6. return $\Phi$}
\STATE {7. end if}
\end{algorithmic}
\end{algorithm}

2.3.2 Example

Our approach is validated on LCL relational data sources. The following query concerns all collaborators having an address in Lyon, and a certain profile:

$((\text{Collaborator}(x) \land \text{hasAddress}(x,y) \land \text{Address}(y)) \land (\text{Address}(z) \land \text{hasAsTown}(z, \text{"Lyon"}) \land (\text{Profile}(p)))$

The mediator breaks up this query into three subqueries. The first two are sent directly to the sources to be executed because they are linked by the concept Address and thus, they can be recomposed by a classic join. However, the third subquery is not linked directly with the two previous subqueries. The mediator must find a link between Profile concept and concepts of the previous subqueries, if there is no link, it excludes this concept. In our example, the mediator must find a link between two concepts: Collaborator and Address, which is a property gathering directly these concepts with Profile concept. It can be also a property links Profile concept to another equivalent concept, subsumed or subsuming one of the two previous concepts: Collaborator or Address. In our case, Person concept is the concept subsuming Collaborator, and it has a link with Profile. The mediator must thus rewrite the third subquery Profile into “Person(r) \land \text{hasProfile(p)} \land \text{Profile(p)}”. It must add in his table of correspondence that Collaborator of the first subquery corresponds to Person of the third subquery. It will join its result with the two previous ones.

3. IMPLEMENTATION

To validate our approach, we develop a prototype that implements our architecture of mediation. Our system manages data sources independence and their distributivity. It manages also the interaction between global ontology and local ones during the query creation. Our prototype is based on Multi Agents Systems (MAS) since they are more adapted for distributed and cooperate environments. Our environment is distinguished from the existing integration systems by mediation by the fact that it enables to express descriptions of sources using the recent recommendation W3C for the ontologies description, which is OWL. It offers very interesting possibilities of descriptions and reasoning. Our objective is also to combine the power of expression and description of language OWL with the aspect communicating and cooperate Systems Multi Agents (MAS).

The mediator is an agent that communicates with other agents. It contains the global ontology and the rewriting module. The other agents are the sources agents. The process of query creation or rewriting is done by a dialogue between the agent mediator and the other agents. For the development of this environment, we used a certain number of tools: the ontology editor Protégé,4 the framework JADE for agents, the framework Jena for OWL-ontologies handling. Jena is a project of free source code developed by HP for the semantic Web. This framework offers us many advantages: it enables to have a uniform access for various ontologies because all information is stored in a Jena model. For the reasoning on OWL-ontologies, we use the free arguer Pellet, which allows to reason on the terminological part. Queries interface is presented in the form of a Java Web application based on the framework Struts.

Figure 2. Based-agent mediator architecture
4. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a new approach of data sources integration based on ontologies in data warehousing environment. Our approach is based on a hybrid architecture, using a global ontology for the mediator and local ontologies for the sources. It is important to create global ontology starting from local ontologies, because this facilitates and improves the resolution of semantic heterogeneity between data sources. We defined a method of ontologies construction, a language which guarantees the correct treatment of queries, by allowing their expression in terms of global and local ontologies. We also proposed a strategy of query rewriting, which ensures the user query coherence, by eliminating concepts not linked with others of the same user query. We applied our approach of ontologies creation on the relational sources of the LCL.

These ontologies are used in our system of integration, and were useful in the phase of creation and rewriting of queries. Various perspectives are considered. Initially, completing the implementation. Then the adaptation of the system to the various sources of information. It will be necessary to automatize the ontologies conception method. To reach this goal, we think to use data mining techniques to generate concepts classes and relationships in a formal way.

REFERENCES


ENDNOTES

1 http://Protege-standard.org
2 http://jade.tilab.com
3 http://jena.sourceforge.net
4 http://www.mindswap.org/2003/pellet/
5 http://jakarta.apache.org/struts/
0 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage: www.igi-global.com/proceeding-paper/hybrid-architecture-owl-ontologies-relational/33201

Related Content

Contemporary Reporting Practices Regarding Covariance-Based SEM with a Lens on EQS
www.irma-international.org/chapter/contemporary-reporting-practices-regarding-covariance/63263

A Practical Approach to Theory Structuring and Analysis: A way to Structure Research and Perform Sub-Problem Solving in a Changing IS Landscape
www.irma-international.org/chapter/practical-approach-theory-structuring-analysis/63268

Dynamic Taxonomies for Intelligent Information Access
www.irma-international.org/chapter/dynamic-taxonomies-for-intelligent-information-access/112829

Design of a Migrating Crawler Based on a Novel URL Scheduling Mechanism using AHP
www.irma-international.org/article/design-of-a-migrating-crawler-based-on-a-novel-url-scheduling-mechanism-using-ahp/169176

Mobile Sink with Mobile Agents: Effective Mobility Scheme for Wireless Sensor Network
www.irma-international.org/article/mobile-sink-with-mobile-agents/178160