Semantic Mediator Querying

Béatrice Bouchou Markhoff, Cheikh Niang

Semantic Mediator Querying

Béatrice Bouchou
Université François-Rabelais de Tours
LI EA 6300, Tours, France
beatrice.bouchou@univ-tours.fr

Cheikh Niang
Université François-Rabelais de Tours
LI EA 6300, Tours, France
cheikh.niang@univ-tours.fr

ABSTRACT

We present the whole querying process of our ontology-based data integration proposal, that we call *Semantic Mediator*. The global schema (a TBox) is composed of the source schemas (also Tboxes) and a taxonomy, which links the sources to each other. The querying process is based on the global-schema's structure and consists of three steps: global query rewriting, source querying and global answer building. We describe the overall distributed system and the query-rewriting algorithm. Then we present an application of such a semantic mediation, the Personae project, which is for enabling historians to share their prosopographic data from the Middle Ages and the Renaissance.

Categories and Subject Descriptors

H.2.4 [**Systems**]: Query Processing, Distributed Databases

Keywords

Semantic Data Integration, Distributed Query Answering

1. INTRODUCTION

We devised an ontology-based data integration system whose query resolution component is detailed in this paper. Data integration is a broad research topic, that has received for many years the attention of researchers in databases and knowledge representation, and it is now revisited with the growth of the Semantic Web. As defined in [11], there are two main approaches for data integration, the data warehouse (the source databases are loaded in the warehouse) and the mediation. The mediation approach allows information to be retrieved dynamically from original databases at query time. It provides a unified global query-interface and relies on mappings between the global schema and each of the local source schemas. These mappings are used to rewrite the global query into a union of queries that match local schemas. They are directed, either from entities in the global schema to entities in the local sources ("Global As View" (GAV) mappings), or from entities in the local sources to the global schema ("Local As View" (LAV) mappings). LAV mappings require more sophisticated inferences to resolve a query on the global schema than GAV mappings, but they make it easier to add new data sources to the mediation framework.

We call our proposal *Semantic Mediator* because it is a mediation, in which the schemas are ontologies, more precisely they are the conceptual part of ontologies. Our semantic mediator is an application of the ontology-based data access (OBDA) paradigm, which has been proposed as a data integration solution that offers an efficient access to large quantities of data stored in relational databases, via a conceptual model of the data. Its principles, as illustrated on the left of Figure 1, have been introduced in [5, 3, 15], and then fully implemented as reported in [9, 17]. While leaving the RDBMs the tasks of efficient storage, maintenance and querying of the data, it allows (i) several relational databases to be integrated, and (ii) their querying to be enriched with the ontological knowledge.

In an OBDA system, the global schema $\mathcal{G}$ of a classical mediator data integration system $\mathcal{J} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ (as defined in [11]) is a TBox of the *DL-Lite* Description Logics [3]. As analyzed in [4], using a conceptual model rather than a *logical* one for the global schema, allows the users to easily manage their queries while the logical schema of each source is still used for its strenghts: optimizations of the storage, optimizations of the querying, etc. The declarative approach based on $\mathcal{M}$, set of semantic mappings between the conceptual global schema and the logical database schema implies that, when global and local schemas evolve, only those mappings must be updated. However, the implementation of a classical OBDA system architecture requires (i) the construction of a consensual ontology to represent the domain of integration, which is known as a difficult task [23], and (ii) the design of the mappings between this ontology and the heterogeneous sources' relational databases.

The query-answering process detailed in this paper is based on a more flexible architecture, illustrated on the right of Figure 1. It consists in keeping all the previously listed advantages while limiting the difficulty of constructing the consensual global ontology. In our architecture, each source is an OBDA system. With existing tools such as Ontop[1] [17] or Mastro[2] [9], installing an OBDA system on an existing database consists in (i) building the lightweight ontology that represents local data and (ii) defining the mappings

[1] `http://ontop.inf.unibz.it/`
[2] `http://www.dis.uniroma1.it/quonto/`

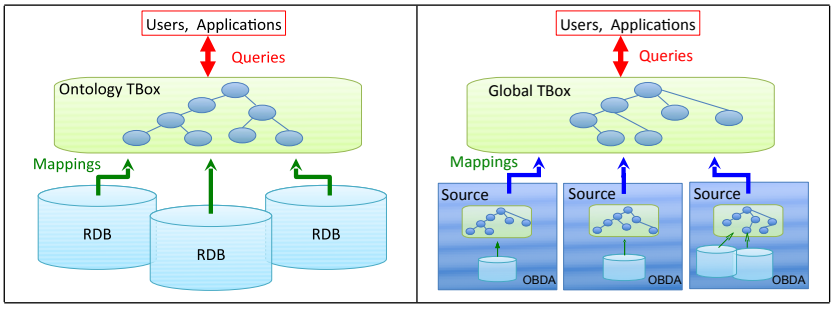

Figure 1: Ontology Based Data Access and OBDA Integration.

between this ontology and the local RDB. This is not such a difficult task for a local database administrator who knows her data well, compared to the task of building a global ontology from several relational databases, such as reported in [19]. As discussed in [23], it is simpler to obtain one ontology for each data source and then to relate these, either in a peer-to-peer network or in a "hybrid" architecture that introduces a global level, such as our proposed global ontology.

Moreover, our proposal includes the management of distribution. To the best of our knowledge, the existing OBDA systems rely on tools for federated relational databases in order to consider that the OBDA system comprises a single relational data source. By contrast, as presented in Section 2, our proposal uses the Sesame[3] architecture in order to implement a truly distributed mediator system.

The paper is organized as follows: in Section 2 we introduce the general architecture of our mediation system. In Section 3 we present the global schema's structure and we show in detail how this structure allows us to perform the query-answering task. Then we present our use-case in Section 4.

2. MEDIATOR ARCHITECTURE

An overview of our mediator system architecture is shown in Figure 2. It consists of two main layers. The lower layer is that of sources: each one incrementally joins the system in order to collaborate with others. The upper layer is that of the mediator, which includes the global schema of the system. One important part of this global schema is a set of semantic mappings that link the mediator to the sources involved. The mediator is also composed of a query resolution module which allows to locate relevant sources to be queried to answer a given user request. The results returned from the sources are combined and stored in a temporary repository in order to allow the user to analyze them.

2.1 Source Layer

The source layer of our mediator architecture is the set of sources that incrementally join the semantic mediation system. Each source is autonomous and stores its data, which can be very large, in relational databases (RDBs). Indeed, we consider that relational database technology is the best support for managing local data and it is also the most widely used. In particular, it offers an efficient access to data. In accordance with OBDA principles, we assume that the local RDBs are linked to a local ontology, which provides their conceptual view. This local ontology can be designed by experts to represent the data managed by the source, or it can be automatically generated from the database by using suitable tools such as RDBtoOnto [7]. Whatever the local ontology building method (automatic or manual), we compute from this local ontology the relevant parts to be shared by the source with respect to the domain of interest in which the mediation system is desired. We have proposed in [14, 13] an approach that uses a domain reference ontology[4] as a background knowledge support. With this reference ontology, our proposed approach selects in the source the knowledge fragments that must be shared in the semantic mediation system. The process, based on a combination of well established lexical-based algorithms for ontology alignment, is *supervised* by the local database administrator [14, 13], and its result consists in the *agreement ontology* depicted in the sources of Figure 2. This agreement ontology is composed of a relevant subset, for the application domain, of the source's ontology, plus a set of semantic mappings between concepts from this subset and those of the domain reference ontology. This agreement ontology represents the conceptual access point to the source's data, which is used during interactions between the mediator and the source.

In order to offer access to the data via the agreement ontology, the local database administrator must implement OBDA mappings [5, 3, 15]. These mappings establish links between the local ontology items and some SQL queries to select the actual values in the local relational databases for dynamically populating the ontology's instances at query time. The OBDA mappings can be automatically generated when the local ontology is automatically built from the database, otherwise they must be manually specified. The framework that we use for each local OBDA system is Ontop, used as a SPARQL endpoint through the Sesame server[5] [2]. We also use Sesame to give the mediator remote access to the agreement ontology of each source. To sum up, a source which would like to join the integration process must firstly have an ontology to represent its data. Next, it must use our agreement-ontology-building tool to generate its agreement ontology. Afterwards, it must use Ontop in order to implement the OBDA mappings. And finally, it has to register with the mediator (cf. sources directory in Figure 2) by giving its SPARQL endpoints: one for its agreement repository, one for its interrogation repository

[3] `http://www.openrdf.org/`

[4] Shared ontology usually developed by experts of the domain, which provides a robust conceptualization about a given generic domain such as medicine, tourism, agriculture, etc.

[5] `http://www.openrdf.org/`

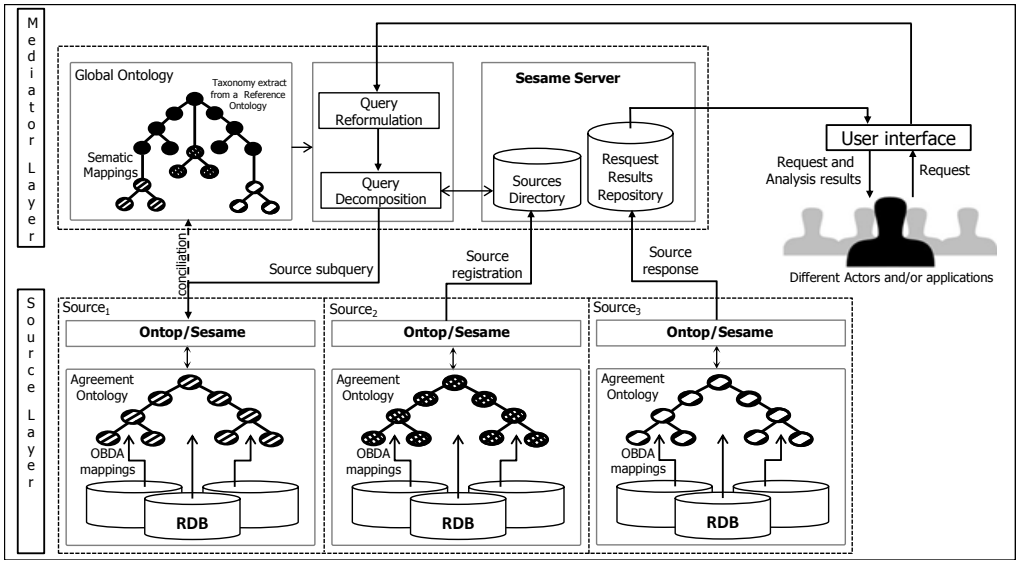

Figure 2: General mediator system architecture.

(containing the OBDA mappings) and one for its mapping repository used by the mediator to integrate the agreement in the global ontology.

2.2 Mediator layer

The mediator layer is composed of core components, through which the sources are integrated and queried. The first one is the sources directory, where all sources involved in the semantic mediation system are registered. To give the sources remote access to this directory, so that they register themselves, we again use a Sesame server. The integration process starts once the mediator layer is installed and when the publication of its sources directory endpoint is done. Figure 3 gives an idea of this directory's content: the shown RDF assertions indicate that the source identified by URI "*http://www.li.univ.tours.fr/personae#Chantres*" joined the mediation system and that its name is *Chantres*. Moreover its mapping, agreement and interrogation (i.e. OBDA) repositories are given, to be used by, respectively, the global ontology building and the query processing.

Another important component of the mediator layer is the global ontology. It must, indeed, provide a global consensual conceptual level of the application field and a structured vocabulary for querying relevant data sources. It is incrementally built by the mediator as new sources join the semantic mediation system. The building process is achieved through a conciliation step which links in the global ontology all agreements of registered sources, via a relevant subsumption hierarchy of concepts. As explained in [14, 13], this subsumption hierarchy is computed from the same reference domain ontology as for the agreement building.

The last, but not least important component of the mediator layer is the query processor, which contains three components: the query rewriting module (denoted Query Reformulation in Figure 2), the query distribution module (Query Decomposition), and the answer recomposition module. They, respectively, (i) reformulate a given user request, expressed in terms of the global ontology, into a set of queries expressed in terms of source ontologies, (ii) compute the sub-queries that must be sent to the sources involved in the original query and send each sub-query to the corresponding source, and (iii) retrieve the partial answers from sources and combine them into a temporary repository. This repository can be seen as a data warehouse of results, that can be graphically visualized, giving the users a comprehensive view about the global answer built from source answers. As we have already published the algorithms devised for building the agreement ontology and to integrate an agreement ontology in the global ontology, in this paper we focus on the query processing, which is presented in greater detail in the following section.

3. GLOBAL SCHEMA STRUCTURE AND QUERY ANSWERING

3.1 Structure Definition

We present in Figure 4 an example with two sources $S1$ and $S2$ which illustrates the structure of the global schema of our mediator system. At the top, we have the taxonomy of concepts which allows us to link the sources to each other. Remember that this taxonomy is extracted from a reference domain ontology. The middle part of Figure 4 contains subsumption relations between source concepts and those of the taxonomy: these are the mappings output of the agreement-ontology building performed by each source. Lastly, the lower part presents the source's agreement-ontologies.

The global schema is a TBox of $DL\text{-}Lite_{\mathcal{A}}$. Indeed, our query-rewriting algorithm relies on the same conceptual framework as the well-established OBDA systems of Calvanese et al., *i.e.*, (i) the $DL\text{-}Lite_{\mathcal{A}}$ [3, 15], which belongs to the family of Description Logics (DLs) [1], (ii) conjunctive queries and union of conjunctive queries, and (iii) *Consistent* and *PerfectRef* algorithms [3, 18]. Precisely, the global schema is specified in the following definition.

DEFINITION 1. - ***Global Schema*** $\mathcal{T}_g$

```
− <rdf:Description rdf:about="http://www.li.univ.tours.fr/personae#Chantres">
    <rdf:type rdf:resource="http://www.li.univ.tours.fr/personae#Source"/>
    <name>Chantres</name>
  − <hasMappingRepository>
      http://localhost:8080/openrdf-sesame/repositories/chantres_mapping
    </hasMappingRepository>
  − <hasAgreementRepository>
      http://localhost:8080/openrdf-sesame/repositories/chantres_agreement
    </hasAgreementRepository>
  − <hasInterrogationRepository>
      http://localhost:8080/openrdf-sesame/repositories/chantres_interrogation
    </hasInterrogationRepository>
    <IsConciliated>false</IsConciliated>
  </rdf:Description>
```

Figure 3: An excerpt of the sources directory.

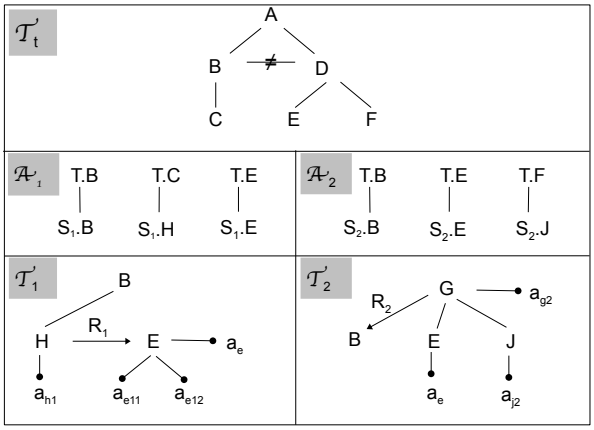

Figure 4: Global Schema Architecture.

The global schema $\mathcal{T}_g$ is a DL-Lite$_\mathcal{A}$ TBox that is composed of the following TBoxes:

- ***The TBox*** *$\mathcal{T}_t$, which (i) is a taxonomy of concepts of the considered application domain and (ii) serves to link the sources. $\mathcal{T}_t$ has two important properties:*
  Property *1: $\mathcal{T}_t$ is an atomic TBox, i.e. it contains positive inclusions $A_1 \sqsubseteq A_2$, or negative inclusions $A_1 \sqsubseteq \neg A_2$, where A_1 and A_2 are atomic concepts.*
  Property *2: if B is a concept of $\mathcal{T}_t$ then B subsumes (directly or transitively) at least one source concept.*

- ***The set $\mathcal{S}'$ of TBoxes*** *$\mathcal{T}_i$, one for each source S_i. $\mathcal{T}_i$ is the agreement ontology (the part of the source ontology that is involved in the mediation).*

- ***The set $\mathcal{A}$ of TBoxes*** *$\mathcal{A}_i$, one for each source S_i. $\mathcal{A}_i$ contains the mappings between $\mathcal{T}_i$ and the taxonomy, i.e. a set of assertions $B_i \sqsubseteq B$ where B_i is a concept of $\mathcal{T}_i$ and B is a concept of $\mathcal{T}_t$.*

This global schema has some important properties to notice. Firstly, it is possible to have a taxonomy $\mathcal{T}_t$ with several disconnected hierarchies if the reference ontology from which it is extracted contains disconnected hierarchies itself. As concepts of $\mathcal{T}_t$ do not come from a source, but are taken from the reference ontology of the mediation domain, by construction each concept of this taxonomy is a parent or an ancestor of a source concept. Thus, if a query is about a concept of $\mathcal{T}_t$ then there is always at least one source that can give an answer.

Secondly, the assertions of the mappings $\mathcal{A}_i$ associate to the taxonomic concepts a *union* of source concepts, without conjunctions. So, we can say that sources are "loosely coupled" by the taxonomy.

Lastly, the TBoxes $\mathcal{T}_i$ contain concepts, roles and attributes that represent data in the sources S_i. Notice that $\mathcal{T}_i$ may be different from the complete TBox representing all the data in a source S_i, as $\mathcal{T}_i$ denotes the part that is involved in the mediation process. For instance, in the *Personae* mediator described in Section 4, the database *Bude* is an example of a source that does not provide all its content to the mediation process. $\mathcal{T}_i$ is the output of the agreement-ontology building process, together with the mappings $\mathcal{A}_i$.

All global schema's items may be queried using a global query. If we consider the more formal notation $\mathcal{J} = \langle \mathcal{G}, \mathcal{M}, \mathcal{S} \rangle$ used in Definition 2, each of its three components is illustrated in the right part of Figure 1, in the following way: $\mathcal{G}$ is denoted Global TBox in the figure, $\mathcal{M}$ is represented by the mappings and $\mathcal{S}$ is the set of source ontology parts that are involved in the mediation.

DEFINITION 2. ***Mediator System***
Our mediator system is a triplet $\mathcal{J} = \langle \mathcal{G}, \mathcal{M}, \mathcal{S} \rangle$, where

- ***The global schema*** *$\mathcal{G}$ is $\mathcal{T}_g$, defined in Definition 1.*

- ***The set of mappings*** *$\mathcal{M}$ is $\mathcal{M}_{GAV} \cup \mathcal{M}_{LAV}$, where:*
  - *$\mathcal{M}_{\mathcal{GAV}}$ is represented by $\mathcal{T}_t \cup \mathcal{A}$ (with $\mathcal{T}_t$ and $\mathcal{A} = \bigcup \mathcal{A}_i$ specified in Definition 1), i.e. the taxonomy that allows us to relate each query atom of $\mathcal{T}_t$'s alphabet to one or several atoms of one or several source alphabets defined by $\mathcal{T}_i$.*
  - *$\mathcal{M}_{\mathcal{LAV}}$ is represented by the $\mathcal{T}_i$, belonging to $\mathcal{T}_g$: each query atom using the alphabet of a source S_i exists in $\mathcal{T}_i$ and may eventually appear in other source TBoxes.*

- ***The set of sources*** *$\mathcal{S}$ is equal to $\mathcal{S}'$ defined in Definition 1.*

It can be noted that our global schema $\mathcal{T}_g$ is structured so that it contains all the relevant knowledge fragments of the mediator system, including the mappings and the sources schemas, which is an original feature, compared to classical data-integration frameworks. Our aim is to facilitate the query processing of the mediator through this global schema. Let us still consider the example in Figure 4: with mappings $\mathcal{M}_{\mathcal{GAV}}$ we can easily determine that, if a global query atom involves the concept A, then the concepts B, H, E of $\mathcal{T}_1$ and B, E, J of $\mathcal{T}_2$ should be queried. In the same way, thanks to mappings $\mathcal{M}_{LAV}$, a global query atom involving a_e will lead to query items of S_1 and S_2, if there is in both sides a concept that has this attribute and shares a common ancestor with the concept on the other side. This is verified by our adaptation of MiniCon, presented in Algorithm 1.

3.2 Query Algorithm

Algorithm 1 performs the Query Reformulation and Query Decomposition stages illustrated in Figure 2. The user, or the application developer, is given a view of the global schema where the different parts (taxonomy / sources) are clearly identified. For the global schema in Figure 4, we give in Figure 5 an abstract idea of such a view.

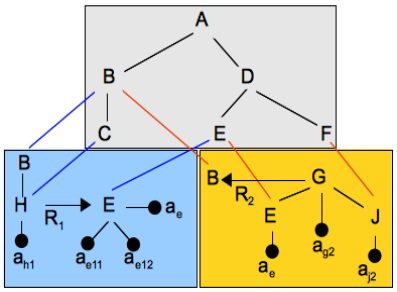

Figure 5: View of the Global Schema.

We should consider several scenarios for querying the global schema in order to query data sources:

(A) The user queries only concepts of the taxonomy.

(B) The user queries concepts, roles and attributes of a given source, and says that she wants only answers from this source.

(C) The user queries concepts, roles and attributes of a given source, without saying that she wants to limit answers to this source.

(D) The user queries concepts, roles and attributes that do not appear in the same source.

(E) The user queries concepts of the taxonomy, together with roles and attributes (that are in some sources).

Considering the example in Figure 5, concerning Scenario (A), if the user queries the concept A, with the query $q(x) \leftarrow A(x)$, then she should get answers given by concepts B, H, E of $\mathcal{T}_1$ and concepts B, E, J of $\mathcal{T}_2$. In the same way, if she queries the concept E then she must obtain answers from the concept E of $\mathcal{T}_1$ and the concept E of $\mathcal{T}_2$. With Scenario (B), the original query can be transmitted to the source without changes. However in Scenario (C), the user should receive answers corresponding to the queried concepts, roles and attributes, but she may also receive answers coming from other sources. For instance, if she queries H, R_1 and E, since E is both in $\mathcal{T}_1$ and $\mathcal{T}_2$ and also $\mathcal{T}_t$, then she will receive answers from $\mathcal{T}_2$. In Scenario (D), for instance if the user queries H, G and a_{h1} she will receive answers from S_1 (via $\mathcal{T}_1$) concerning H and a_{h1}, and also from S_2 (via $\mathcal{T}_2$) concerning G. Finally, with Scenario (E), for instance if the user queries D together with a_e and a_{j2}, then she will receive answers from S_1 (via $\mathcal{T}_1$) concerning instances of concept E with its attribute a_e and answers from S_2 (via $\mathcal{T}_2$) concerning its instances of concept E with its attribute a_e and its instances of concept J with its attribute a_{j2}. LAV mappings are important here.

The previously described scenarios are concretely achieved through the query resolution steps that we present in Algorithm 1. Input is a conjunctive query $q(\bar{x}) \leftarrow conj(\bar{x}, \bar{y})$ on the global schema $\mathcal{T}_g$. Each query q on the global schema $\mathcal{T}_g$ as well as all queries from the reformulations of q w.r.t $\mathcal{T}_g$ are of the form $q(\bar{x}) \leftarrow c_t, c_s$, where: $c_t = conj(\bar{x}_t, \bar{y}_t)$ is a conjunction of atoms querying the taxonomy $\mathcal{T}_t$, and $c_s = conj(\bar{x}_s, \bar{y}_s)$ is a conjunction of atoms querying the sources. In other words, c_t is composed of *"taxonomic atoms"* (they are always unary), and c_s is composed of *"source atoms"* (they can be either unary or binary).

Input: the global schema $\mathcal{T}_g$, a conjunctive query q on $\mathcal{T}_g$
Output: the set $Q_{\mathcal{S}}$ of the queries that must be sent to involved sources

```
begin
    Q_S := {}
    if (checkConsistency(q, T_g)) then
        (c_t, c_s) := divide(T_t, S', q)
        Q_s := {}
        if (c_t ≠ ∅) then
            Q := PerfectRef(q(x̄) ← c_t, T_g ∪ {A})
            Q_s := select(S', Q)
            foreach (q_s ∈ Q_s) do
                complete(q_s, c_s);
            end
        else
            Q_s := Q_s ∪ {q(x̄) ← c_s}
        end
        foreach (q_s ∈ Q_s) do
            B := buckets(q_s, S')
            if (∄B_g ∈ B | B_g = ∅) then
                R := combine(B)
                foreach (r_i ∈ R) do
                    q_{S_i} := generate(r_i)
                    Q_S := Q_S ∪ q_{S_i}
                end
            end
        end
    end
    return Q_S
end
```

Algorithm 1: Query resolution algorithm

3.3 Analysis

Algorithm 1 runs as follows: for a given conjunctive query q expressed in terms of the global schema $\mathcal{T}_g$, the first step consists in applying the *Consistent* algorithm [3] on q considered as a canonical instance. The purpose is to verify the consistence of q with respect to constraints expressed in $\mathcal{T}_g$. In other words, this step makes it possible to avoid evaluating queries that could only lead to an empty result. For instance in our example given in Figure 5, since $B \sqsubseteq \neg D$, it is not useful to evaluate the query $q(x) \leftarrow B(x), D(x)$. The *Consistent* algorithm has a polynomial time complex-

ity on the size of the TBox $\mathcal{T}_g$ and on the size of the ABox, represented by q [3].

The second step starts by dividing the body of q into the two conjunctions c_t and c_s. We then apply the *PerfectRef* algorithm [3] only on taxonomic atoms of q, and using $\mathcal{T}_t \bigcup \mathcal{A}$ as TBox. The result of *PerfectRef* is a union of conjunctive queries (UCQ), among which we keep only those conjunctive queries that are entirely expressed on *source atoms* (function *select*). Then we complete the selected rewritings with the conjunction c_s of the initial query q. Still considering our example in Figure 5, this step allows us to evaluate queries of Scenario (A) and, partly, those of Scenarios (C), (D) and (E). The *PerfectRef* algorithm compiles the TBox knowledge into reformulations of the given query. In our mediator, this knowledge consists of the subsumption relations contained in $\mathcal{T}_t \cup \mathcal{A}$. Thanks to this knowledge, the atoms involving taxonomic concepts are rewritten into source atoms, *i.e.* concepts that are at the leaves of the taxonomy $\mathcal{T}_t \cup \mathcal{A}$. These leaves are inevitably source atoms, from the specification of $\mathcal{T}_t$ given in Definition 1. The *PerfectRef* algorithm has a polynomial time complexity on the size of $\mathcal{T}_t \cup \mathcal{A}$ and it is exponential on the size of q [3][6].

Let Q_s be the UCQ resulting from Step 2, the next step is to compute the queries to be sent to each source. This is performed by the **foreach** part in Algorithm 1, a *MiniCon* algorithm tailored to our semantic context. It addresses queries of Scenarios (C), (D) and (E). Step 2 outputs reformulations that involve only the source parts of the global schema. Then, we can use the LAV mappings (cf. Definition 2) in order to determine all the involved sources. For instance, the user can query a person's name. There may be several concepts in the global schema corresponding to a *Person* and having an attribute *name*. This knowledge is contained in the global schema and we have (*i*) to compute what are the sources that have this concept with this attribute and (*ii*) to build the queries that must be sent to these sources. To achieve this step, we have devised an adaptation of the *MiniCon* algorithm [16], which is one of the well known query rewriting algorithms for LAV mappings, that has been proven to be efficient and scalable in practice. In *MiniCon*, the information about a source S is given by a set of conjunctive views $v_i(x) \leftarrow conj(x, y)$, where $conj(x, y)$ is a conjunction of atoms expressed with S's alphabet. We adapted it to our context, where the information about a source S_i is given by the TBox $\mathcal{T}_i$ included in the global schema $\mathcal{T}_g$.

Our adaptation of *MiniCon* takes as input a query consisting exclusively of source atoms, that can exist in several sources. We adapted to our semantic context each of the three classical stages: (*i*) the building of *buckets*, (*ii*) the *combination* of these buckets, and (*iii*) the *generating* of valid subqueries to be sent to each involved source. To illustrate the process, we still consider our example given in Figure 5 and, for instance, the query:

$q(x, y, z) \leftarrow A(x), a_e(x, y), a_{g2}(x, z)$.

Before Step 3, Q_s contains the following set of queries:

$q_1(x, y, z) \leftarrow S_1.B(x), a_e(x, y), a_{g2}(x, z)$,
$q'_1(x, y, z) \leftarrow S_2.B(x), a_e(x, y), a_{g2}(x, z)$,
$q_2(x, y, z) \leftarrow S_1.E(x), a_e(x, y), a_{g2}(x, z)$,
$q'_2(x, y, z) \leftarrow S_2.E(x), a_e(x, y), a_{g2}(x, z)$,
$q_3(x, y, z) \leftarrow S_2.J(x), a_e(x, y), a_{g2}(x, z)$.

For each of these queries we have to compute the set of buckets. Formally, let $q(x) \leftarrow g_1(z_1), \ldots, g_n(z_n)$ be the input query, for each atom g_i of q, the bucket b_i contains the sources owning answers for g_i. With the same aim as for *MiniCon*'s optimization with respect to the bucket algorithm of [12], we introduce here some tests whose purpose is to put in the bucket of an atom g only those sources whose instances of g could actually be joined with instances of *at least one* of the other atoms of the same query. Consider for example the reformulated query:

$q_1(x, y, z) \leftarrow S_1.B(x), a_e(x, y), a_{g2}(x, z)$

No bucket will be created because B has neither attribute a_e, nor attribute a_{g2}. In fact, among the 5 queries in Q_s, only q_2 and q'_2 have at least one source in each of their buckets.

In the next stages, the buckets are combined in order to get valid subqueries to be sent to the involved sources. Here again, we adapted the *validity* property to our context, where each source is represented by a TBox. We then verify all constraints contained in the involved $\mathcal{T}_i$ while building its corresponding subquery. In our example, the subqueries computed for q_2 and, respectively, q'_2 are:

$q_{2.S_1}(x, y) \leftarrow E(x), a_e(x, y)$ (sent to S_1),
$q_{2.S_2}(x, y, z) \leftarrow a_e(x, y), a_{g2}(x, z)$ (sent to S_2),
$q_{2'.S_1}(x, y) \leftarrow a_e(x, y)$ (sent to S_1),
$q_{2'.S_2}(x, y, z) \leftarrow E(x), a_e(x, y), a_{g2}(x, z)$ (sent to S_2).

Concerning complexity, creating the buckets for each query $q_{\mathcal{S}}$ in Q_s is in (worst case) $O(n \times l \times k \times t)$, where n is the number of queries in Q_s, l is the average number of atoms in queries $q_{\mathcal{S}}$, k is the number of sources and t is the average size of the TBoxes $\mathcal{T}_i$. Combining the buckets is in $O(n' \times b^l)$, where n' is the number of queries $q_{\mathcal{S}}$ which have at least one source in each of their buckets ($n' \leq n$), l is the average number of atoms in these queries and b is the average size of the buckets. Our query resolution algorithm is based on scalable algorithms, but it is clear that optimizing Step 2 in order to get *fewer rewritings* with *shorter length* is the way to optimize the whole resolution process.

After a run of Algorithm 1, the computed subqueries are sent to the sources, by using the Sesame architecture presented in Section 2. Then, each of these queries is evaluated on its source's relational database, through the OBDA system (i.e. Ontop) and the answer is sent back to the mediator. Each answer feeds the global result relation, whose attributes are the distinguished variables of the initial global query q, plus an extra attribute that represents the source that has given the answer. Notice that there are NULL values in tuples of this global answer set, for the variables not instantiated in the corresponding source. These principles and algorithms have been implemented for the application presented in Section 4.

4. APPLICATION TO PERSONAE

The Personae project, led by the History Institute CESR[7], has been an interesting application for our semantic mediation system because in this project, the partners wanted to share their data but each of them had excellent reasons for keeping their own data schema. Indeed in these fields

[6]Notice that other reformulation algorithms exist, such as *Presto* [18], that optimize and greatly reduce the complexity in practical cases, by avoiding many reformulations that are contained in each others.

[7]Centre d'Etudes Supérieures de la Renaissance `http://umr6576.cesr.univ-tours.fr/`

(humanities), diversity and thus heterogeneity seems to be an asset rather than a liability, and it should be preserved. This is the reason why our proposals are welcomed in this context, because they allow partners to keep their databases independent of the mediator structure.

4.1 Personae Project

The Personae project associates several groups of historians who have built databases containing descriptions of people and their relationships during the Middle Ages and the Renaissance, in order to perform prosopographical studies. Prosopography [22] is a traditional historical practice which aims to interconnect biographies for series of social groups, more or less large and more or less precisely identified (for instance the Renaissance humanists, publishers, teachers and / or students of a faculty or university, artists, etc.). It uses natural modes of expression such as local dictionaries, professional dictionaries or thematic dictionaries. The Personae project aims to develop new modes of expression, publication and processing of this kind of data.

To start, Personae is focused on the Center West France, which was a high place of knowledge in the late Middle Ages with its three great universities founded in the 14th and 15th centuries (Orléans in 1306, Poitiers in 1431, Bourges in 1464). Several identified databases exist to form the basis of the project. The ultimate goal is to attract into the planned "central portal" other sources, built and maintained by other research teams. One of the existing databases, the *Bude*[8] database, covers humanists who conserve and transmit texts (about 12000 persons). Another source, the database *Chantres*, comes from the project *Prosopographie des Chantres de la Renaissance* (PCR[9]). It gathers biographies of professional singers of the 15e and 16e centuries (about 5000 persons). There is also the database *Lesellier*, that contains information about French people that were in relation with the papacy, information gathered from registers of letters to the Pope (about 20000 persons). Other data sources are involved, for instance the students and professors that were at the University of Poitiers in the late Middle Ages and in the Renaissance (hundreds of persons) and medieval doctors, astronomers and astrologists of the late Middle Ages (about 300 persons). Moreover, it is planned to progressively add people involved in other activity fields (naturalists, painters, jurists, etc.).

The Personae project is not restricted to the integration of a number of prosopographic data sources, which is in fact a first stage towards the design of new modes of publication and exploitation of these data. Visualisation tools based on maps will provide renewed representations of this kind of data, which can assist historians to discover places of scholarly circles, and of knowledge production and transmission. The integration phase is a cornerstone for the design of new modes of expression, publication and processing of prosopographic data.

4.2 A Semantic Mediator for Personae

Figure 6 represents the intended architecture of the Personae semantic mediator. At the top, map-based visualization and navigation tools will use the mediator to exploit the

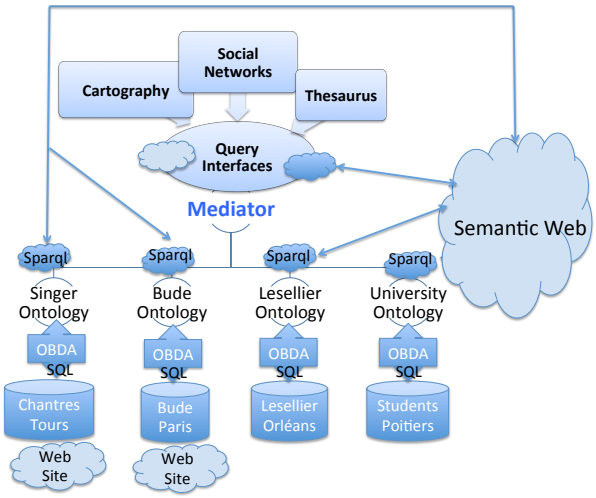

Figure 6: Personae Mediator

different sources. They will work with RDF and OWL formats. Moreover, several resources are currently built by our historian colleagues such as a thesaurus of first names (with equivalences in Latin and other European languages of the Renaissance), dictionaries of functions, clerical or military positions, etc. in order to enrich the information retrieval potential offered by the portal.

At the lower level, each local database has an OBDA system installed, *i.e.*, the local administrator has built the TBox $\mathcal{T}_i$ and the OBDA mappings between $\mathcal{T}_i$ and the local relational database. In this way, any authorized application can query the source in SPARQL via its ontology. The source can publish this ontology in order to allow applications to access its data, alternatively it can give the ontology and grant access to some identified partners. Irregardless of this access to applications and/or the mediator, the source administrator can continue to publish her data as she wants, via a classical web site. Indeed, both *Bude* and *Chantres* databases have been funded to provide their data in a specific web site, via query forms that are used by several teams of researchers, not necessarily involved in Personae. Moreover a source may cover a wider field than that covered by the integration system, it is the case for the database *Bude* with respect to Personae. By applying the principles presented in this article, only the conceptual portion of *Bude* that is relevant for Personae is selected and stored in the agreement ontology built for joining Personae.

This is out of the scope of this paper, but we precise that we had to design and develop new extensions to our agreement-ontology building algorithm, because there exists no reference ontology for the prosopography field. It would be of interest to design such a reference ontology in collaboration with our historian colleagues (as an extension of the well established CIDOC CRM[10]). To apply right now our proposals to the project, we used taxonomies extracted from YAGO2[11], which has the advantage of offering vocabu-

[8] `http://bude.irht.cnrs.fr/`, see also: `http://heloise.hypotheses.org/85`

[9] `http://ricercar.cesr.univ-tours.fr/3-programmes/PCR/`

[10] `http://www.cidoc-crm.org/`

[11] `http://www.mpi-inf.mpg.de/yago-naga/yago/`

mappingId M:67 target :Personne-{pers_id} a :Personne . source select pers_id from Personne mappingId M:68 target :Personne-{pers_id} :designation {designation} . source select pers_id, designation from Personne mappingId M:69 target :Personne-{pers_id} :nom {nom} . source select pers_id, nom from Personne	mappingId M:10 target :Personnages-{id} a : Personnages. source select id from Personnages mappingId M:11 target :Personnages-{id} :nom {nom} . source select id, nom from Personnages mappingId M:12 target :Personnages-{id} : date_mort {date_mort} . source select id, date_mort from Personnages
Chantres OBDA mappings	*Bude OBDA mappings*

Figure 7: Excerpts of Bude and Chantres OBDA mappings.

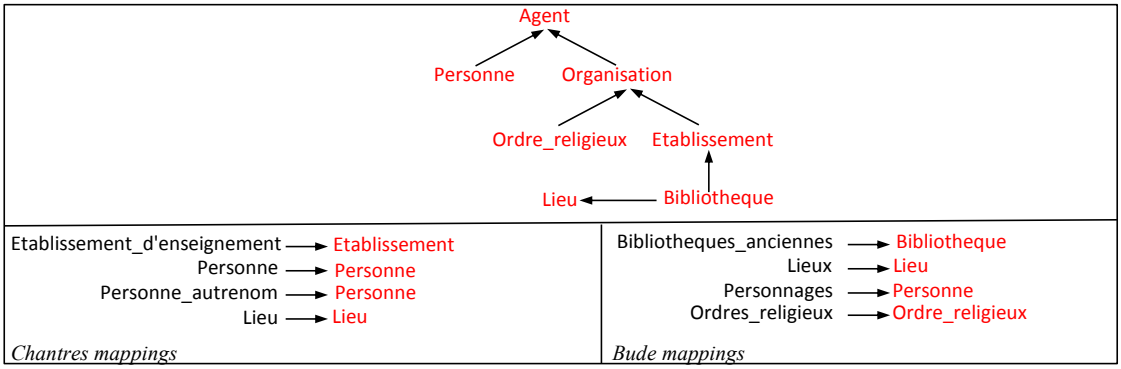

Figure 8: Excerpt of Personae's global schema.

laries in several languages, including French. The drawback of such a general resource, when used as the reference ontology in our system, is the output of many candidate mappings due to the high degree of ambiguity, induced by the resource's generality. To finally get accurate mappings, we devised extensions to assist the database administrator to choose among the candidate mappings.

We focus on the two sources *Bude* and *Chantres* to describe the query-processing. Figure 7 shows excerpts of their OBDA mappings. If the source ontology is automatically built from its database with RDBtoOnto, then our agreement-ontology-building tool can build the corresponding OBDA mappings, otherwise they must be manually defined. According to Ontop, each mapping axiom is a pair of source and target. The source is a SQL query over the database and the target is a graph pattern in RDF Turtle[12]. The target's triples have some subjects or objects that contain place holders which reference column names mentioned in the SQL query. Thus, mapping axioms allow the system to create RDF triples, by replacing the place holders in the target with the data values returned by SQL evaluation.

We present in Figure 8 some excerpts of the top and middle parts of Personae's global schema, or, in the words of Section 3, some excerpts of $\mathcal{T}_t$, $\mathcal{A}_1$ and $\mathcal{A}_2$.

The global query is expressed either in SPARQL, or via a form. It is then translated in a conjunctive query, input of our query resolution algorithm described in section 3. We give in Table 1, first row, an example of a SPARQL query on the Personae's global schema illustrated in Figure 8, that asks for the persons whose name starts by "ac" ("iu" is the flag value for function *regex*).

Query on the Global Schema:
`PREFIX : <http://www.li.univ.tours.fr/personae#>`
`PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>`
`SELECT ?x ?n WHERE {`
`?x rdf:type :Personne.`
`?x :nom ?n.`
`FILTER(regex(str(?n), '^ac', 'iu')) }`

Query on Chantres:
`PREFIX : <http://www.li.univ.tours.fr/personae#>`
`PREFIX r:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>`
`CONSTRUCT ?x :nom ?n. ?x :hasSource 'Chantres' WHERE {`
`{ ?x r:type :Personne. ?x :nom ?n.} UNION`
`{?x r:type :Personne_autrenom. ?x :nom ?n.}`
`FILTER(regex(str(?n), '^ac', 'iu')) }`

Query on Bude:
`PREFIX : <http://www.li.univ.tours.fr/personae#>`
`PREFIX r:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>`
`CONSTRUCT ?x :nom ?n. ?x :hasSource 'Bude' WHERE {`
`?x r:type :Personnages.`
`FILTER(regex(str(?n), '^ac', 'iu')) }`

Table 1: Example of a global query and the corresponding generated local queries.

The corresponding conjunctive query for the global query

[12] `http://www.w3.org/TR/2014/REC-turtle-20140225/`

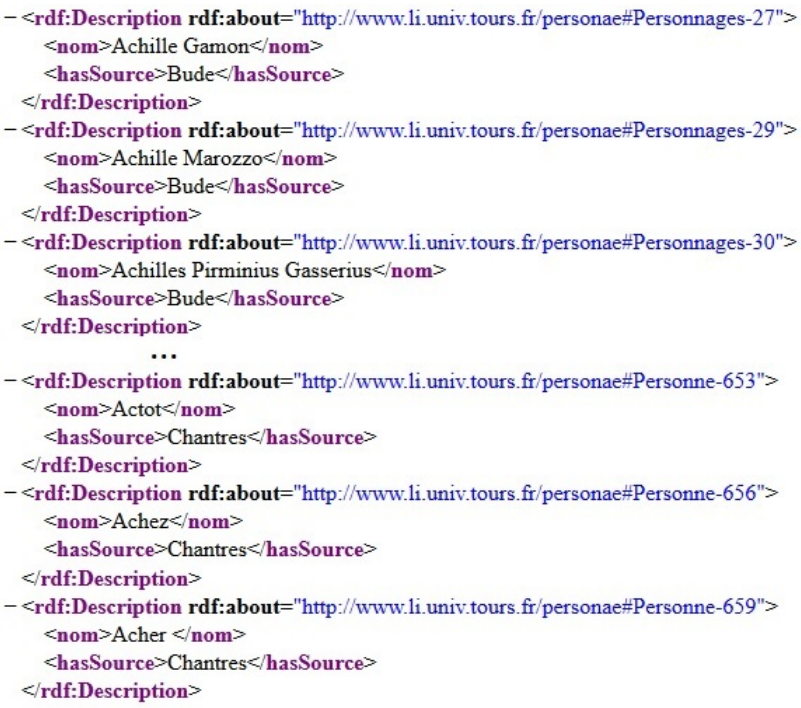

Figure 9: Temporary repository results.

in Table 1 is: $q(x, n) \leftarrow Personne(x), nom(x, n)$.

The conjunctive query is generated without the FILTER constraint, which is however added to the SPARQL subquery that is sent to the selected sources after the query rewriting process. Algorithm 1 takes as input the query q and the global schema and outputs the following UCQs:
Chantres: $Q(x, n) \leftarrow Personne(x), nom(x, n) \cup Q(x, n) \leftarrow Personne\_autrenom(x), nom(x, n)$.
Bude : $Q(x, n) \leftarrow Personnages(x), nom(x, n)$.

Each UCQ is then translated to a SPARQL query for each involved source. We use the CONTRUCT form for storing each answer as a set of RDF graphs, to simplify the merge of all answers. The generated queries for sources *Chantres* and *Bude* are given in Table 1, rows 2 and 3.

Then the generated queries are sent to the involved source. More precisely, as described in Section 2, the mediator gets the answers from a source by invoking its interrogation repository service through the SPARQL endpoint declared by this source in the sources directory of the mediator. The query is locally evaluated by the source on its relational database through its OBDA mappings. Each answer from a source is a set of RDF graphs which is saved by the mediator in its temporary repository results. Each of these RDF graphs keeps the source it comes from, thanks to the *hasSource* property. Figure 9 shows an excerpt of the content of the temporary repository results after the interrogation process, for our example of global query, given in Table 1.

With this repository of results, we can present to the user the answers from both *Chantres* and *Bude* sources, we can also allow the user to see only those from a chosen source, or perform an ordering, etc., *i.e.*, running more specific queries on the result repository, without querying the sources again.

5. CONCLUSION

We propose a mediator to integrate OBDA systems, which takes the form of a web portal that provides access to several resources. It may also be queried by applications, through a SPARQL endpoint. Each resource preserves its autonomous running, maintaining its proper web site or any other type of access, independently from the mediator. To join the mediation system, a resource must install an OBDA access (i) to inform about its conceptual model (i.e. a lightweight ontology that represents content it wants to share) and (ii) to output the data queried using this model, through a SPARQL endpoint. As explained in the introduction, our proposal keeps the advantages of OBDA systems while limiting the efforts to build the global schema and while relying on a truly distributed query-answering process.

The incrementally built global schema is composed of the parts of the source's ontology that are relevant for the mediator. These parts are interlinked via a taxonomy. This taxonomy models the mediation domain, it may be manually built, but in our system it is incrementally built from a reference ontology and from each new source. Our system may be considered as a simplified case of those described for distributed description logics (DDL) [21, 8], because all links between sources are through the taxonomy and not peer-to-peer, with a focus on query processing, as in the framework for ontology integration described in [6]. In this article, we presented the complete query answering functionality, in particular the query-rewriting algorithm.

We have also described an application of our proposal, within the context of the Personae project. This project is to enable historians to share their data on Middle Ages and Renaissance prosopography. Our mediator architecture is a way for them to achieve this goal while keeping their data

under control and without changing their internal formats.

This experiment has led us to suppose that our mediator architecture could be used to integrate resources of the semantic web, in particular linked open data. Provided that a conceptual definition of the resource is available, it could be used by the mediation system. This is our most interesting future study to formalize this new direction, because the current trends in linked data integration miss some means of assisting data consumers, who have the heavy charge to discover links when they do not exist [10]. Links do not exist in general, because data owners do not know how to link their data to other graphs. Clearly, linked data integration is not just a matter of graphs that can be easily extended, because edges in those graphs do not share a common *semantics*: some of them associate instances to their class, some of them are instances of concepts properties, some of them are defining concepts and properties, and so on. Our proposal, aiming to integrate the conceptual description parts, may be a way of alleviating the charge of data consumers without increasing the charge of data providers in the LOD context [20].

Another important future work raised by the Personae project is to study how entity-resolution may be supported by the mediator system, *i.e.* if some similarity measures could be inferred to assist historians to detect, for instance, people described in several different sources. Apart from Personae, we are working on experiments on run-time performance of the query evaluation implementation against different data sources, which may suggest some optimizations of the whole query resolution process.

6. REFERENCES

[1] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[2] J. Broekstra, A. Kampman, and F. van Harmelen. Sesame: An architecture for storing and querying RDF data and schema information. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*, 2003.

[3] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable Reasoning and Efficient Query Answering in Description Logics: The *DL-Lite* Family. *J. Autom. Reasoning*, 39(3):385–429, 2007.

[4] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Conceptual modeling for data integration. In *Conceptual Modeling: Foundations and Applications - Essays in Honor of John Mylopoulos, volume 5600 of Lecture Notes in Computer Science*, pages 173–197. Springer, 2009.

[5] D. Calvanese, D. Lembo, M. Lenzerini, A. Poggi, and R. Rosati. Linking data to ontologies: The description logic *DL-Lite*$_{\mathcal{A}}$. In *In Proc. of OWLED 2006*, 2006.

[6] D. Calvanese and M. Lenzerini. A framework for ontology integration. pages 303–316. IOS Press, 2001.

[7] F. Cerbah. Learning highly structured semantic repositories from relational databases. In *The Semantic Web: Research and Applications*, pages 777–781. Springer, 2008.

[8] C. Ghidini, L. Serafini, and S. Tessaris. On relating heterogeneous elements from different ontologies. In *Proceedings of the 6th international and interdisciplinary conference on Modeling and using context*, CONTEXT'07, pages 234–247, Berlin, Heidelberg, 2007. Springer-Verlag.

[9] G. D. Giacomo, D. Lembo, M. Lenzerini, A. Poggi, R. Rosati, M. Ruzzi, and D. F. Savo. Mastro: A reasoner for effective ontology-based data access. In *Proc. of the OWL Reasoner Evaluation Workshop (ORE 2012), volume 858 of CEUR Electronic Workshop Proceedings, http://ceur-ws.org*, 2012.

[10] R. Isele, A. Jentzsch, C. Bizer, and C. Becker. Silk Server - Adding missing Links while consuming Linked Data. In *COLD 2010*, 2010.

[11] M. Lenzerini. Data integration: A theoretical perspective. In *PODS*, pages 233–246, 2002.

[12] A. Y. Levy, A. Rajaraman, and J. Ordille. Querying heterogeneous information sources using source descriptions. In *VLDB*, 1996.

[13] C. Niang, B. Bouchou, M. Lo, and Y. Sam. Appropriate global ontology construction: a domain-reference-ontology based approach. In *iiWAS*, pages 166–173, 2011.

[14] C. Niang, B. Bouchou, M. Lo, and Y. Sam. Automatic building of an appropriate global ontology. In *ADBIS*, pages 429–443, 2011.

[15] A. Poggi, D. Lembo, D. Calvanese, G. D. Giacomo, M. Lenzerini, and R. Rosati. Linking Data to Ontologies. *J. Data Semantics*, 10:133–173, 2008.

[16] R. Pottinger and A. Y. Halevy. Minicon: A scalable algorithm for answering queries using views. *VLDB J.*, 10(2-3):182–198, 2001.

[17] M. Rodriguez-Muro and D. Calvanese. Quest, an OWL 2 QL Reasoner for Ontology-based Data Access. In *OWLED*, 2012.

[18] R. Rosati and A. Almatelli. Improving Query Answering over DL-Lite Ontologies. In *KR*, 2010.

[19] D. F. Savo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, V. Romagnoli, M. Ruzzi, and G. Stella. Mastro at work: Experiences on ontology-based data access. In *Proceedings of the 23th International Workshop on Description Logics, CEUR-WS 573, Waterloo, Canada*, DL 2010, pages 20–31, 2010.

[20] A. Schultz, A. Matteini, R. Isele, P. N. Mendes, C. Bizer, and C. Becker. LDIF - A Framework for Large-Scale Linked Data Integration. In *WWW2012 Developer Track*, 2012.

[21] L. Serafini and A. Tamilin. Drago: Distributed reasoning architecture for the semantic web. In *ESWC*, pages 361–376. Springer, 2005.

[22] L. Stone. Prosopography. *Daedalus*, 100(1):46–71, 1971.

[23] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. Ontology-Based Integration of Information - A Survey of Existing Approaches. In *IJCAI'01 Workshop on Ontologies and Informations Sharing*, pages 108–117, 2001.