Directory-Based Ontologies for Integrating XML Data

Lule Ahmedi*

Institut für Informatik, Universität Freiburg, Germany
ahmedi@informatik.uni-freiburg.de

Abstract. With the increase in popularity of XML on the Internet, the requirements of database management systems have shifted from traditional transaction-based databases towards the kind of characteristics provided, by design, by the Lightweight Directory Access Protocol. At the same time, the design and use of a middleware to provide a common querying interface to XML-based systems has become an increasingly relevant research problem, encouraged by the fact that XML has become the de facto standard for information interchange on the Internet. The purpose of this paper is to describe the formalisms for representing ontologies in `LGAccess`, our LDAP-based integration system, that is able to seamlessly integrate source data, schemata discrepancies, and semantic information under a common framework, thanks to the simplicity, coherence, and uniformity of the LDAP model. Due to a close resemblance between the XML model and LDAP, and to avoid "dirtying" sources or ontologies with semantic annotations necessary for integration, we introduce a middleware in our system to hold such annotations and support XML ontologies in their "native" representation without cumbersome transformations. Furthermore, instead of using the technologies that are still in the process of adoption by the community, the middleware is built on grounds of the standard and well-established LDAP technology.

1 Introduction

In recent years the proliferation of XML-based systems has dramatically increased to the point of requiring the redesign of existing software and even the creation of new paradigms to handle semistructured data. Furthermore, the structure and characteristics of the databases upon which this new software is based also needs to be revolutionized in order to efficiently process high volumes of transactions with fairly simplistic record management procedures.

At the same time, since the conception of the LDAP protocol version 3 in 1997 [WHK97], the use of lightweight directories to store a variety of information has been steadily gaining momentum. Today, many universities and research centers, like AT&T, use LDAP servers as a means to manage information about their members, organizations, networks, etc., and companies like Netscape or Microsoft offer LDAP support even in their Internet browsers.

* This work is supported by the Deutsche Forschungsgemeinschaft, Aktenzeichen La 598/4-1.

With the increase in popularity of LDAP servers, traditional problems in the field of information integration should be revisited to accommodate the new technology, especially, if this technology is so tied to network and distribution channels, as it is the case with LDAP technology, that, by design, offers distribution capabilities not found in more traditional database systems [WHK97]. Our approach has been developed in the context of `LGAccess`, an LDAP-based system to meet the needs of data integration in the Web.

Due to a close resemblance between the LDAP model and XML [BPSMM00], we use the LDAP middleware in our system to support XML ontologies in their "native" representation without the need to incur in some cumbersome transformations. Furthermore, we avoid "dirtying" sources or ontologies (as usual) with semantic annotations necessary for integration and use a middleware to hold such annotations. This way, the LDAP ontology in a middleware (internal ontology) consists of the extra primitives not included in the traditional models for ontologies, constituting an additional layer on the body of ontologies, the integration-specific body layer. Also, the development of XML technology, as well as the adoption of the still in process of development standards, like XLink [DMO00] or XPointer [DDM00], is not as well established as LDAP. Therefore, we have opted to base our ontology middleware on LDAP technology, and expect to leverage its strengths for the purpose of semistructured data integration.

In this paper, we present the design of a middleware with integration capabilities, targeted to the ontologies model, but adapted to LDAP technology by means of the representation formalisms for ontologies, both, in XML and in LDAP. In Sect. 2, we give a brief overview of the capabilities offered by LDAP servers, we have made use of in our model. Section 3 introduces the context in which the key idea of the paper came up. Then, Sect. 4 goes into the details of our approach by explaining the ontology representation formalisms, leaving the comparison to other systems for Sect. 5. Finally, Sect. 6 concludes this paper.

2 Overview of the LDAP Model

An LDAP server can be viewed as a semistructured database system, whose performance shines for the kind of operations performed on the Internet, that is, simple and fast read operations that occur much more frequently than writes, and support for the classical client/server architecture.

The LDAP specification [WHK97] defines the two fundamental models an LDAP directory service is based on: a data model and a functional model.
The data model consists of two components:
Directory Schema: Defines a finite set of classes, their content and organization in the schema hierarchy. The *class content* determines the required/allowed attributes that a given class instance must/may contain, and the type of allowed values for each attribute.
Directory Instance: Contains a finite set of entries organized in a forest, where each entry (1) has a non-empty set of (possibly) multi-valued attribute-value pairs $e(a, v)$ that conform to the schema definition; (2) belongs to at least one

class, i.e., the attribute *oc* that denotes what classes the entry belongs to is mandatory for each entry, as is the attribute *dn* that provides a unique distinguished name for each entry; and (3) is placed in the instance hierarchy based on its *dn*.

The functional model determines the operations that can be performed on the directory server. The search operation acts in a similar way a database query does. It is invoked by a search request defined as a combination of the following four components. The **base** denotes the distinguished name of the entry in the directory instance where the search will start. The **scope** can be base, if the search is to be restricted to just the first node, onelevel, if only the first level of nodes is to be searched, or subtree, if all nodes under the base should be considered by the filter expression. The **filter expression** is a boolean combination of atomic filters of the form $(a\ op\ t)$, where a is an attribute name, *op* is a comparison operator out of the set $\{=, \neq, <, \leq, >, \geq\}$, and t is an attribute value. The **projection** defines the set of attributes to be returned by the query.

For more detailed information about the LDAP protocol, see either [HSG99] for an informal description, or [JLM$^+$99] for a formal one. For the purpose of this paper, we are mostly interested in the schema definition capabilities, which we will use to model ontologies, as discussed next. Before, we sketch out the context in which these ontologies were defined.

3 Context Definition

3.1 Integrating Data with `LGAccess`

The issues discussed in this paper address just one aspect of the data integration platform we promote, the socalled `LGAccess` platform. `LGAccess` stands for **L**ightweight **G**lobal **Access** system, a non-intrusive extension of the traditional LDAP server modified to meet the needs of data integration in the Web. This section gives a brief description of the `LGAccess` system whose architecture is sketched in Figure 1.

An XML source that is subject of integration and its structure definition (DTD) are incorporated in our system such that, for each of them, the corresponding representation in LDAP is generated by the *xml2ldap* and *dtd2ldap* components, respectively, as described in [ML01b]. The *LDAP Integration Engine*, composed of the *Query Reformulation Module* and the *Query Answering Module*, is responsible to perform the data integration operations. The user formulates a query (in XPath) over the *external ontology* (Sect. 4.2) and poses it to the system. The *xpath2ldap* component transforms the query from XPath to LDAP. The *Query Reformulation Module* takes the LDAP query and rewrites it into the terms of the source level queries by using the knowledge derived from the query definition and the semantical definitions found in the *internal ontology* (Sect. 4.1). These queries are evaluated separately for each source by the LDAP server. Then, the results are combined by the *Query Answering Module* taking into account the rewriting previously formulated by the *Query Reformulation Module*. Finally, *ldap2xml* transforms the result from LDAP into XML.

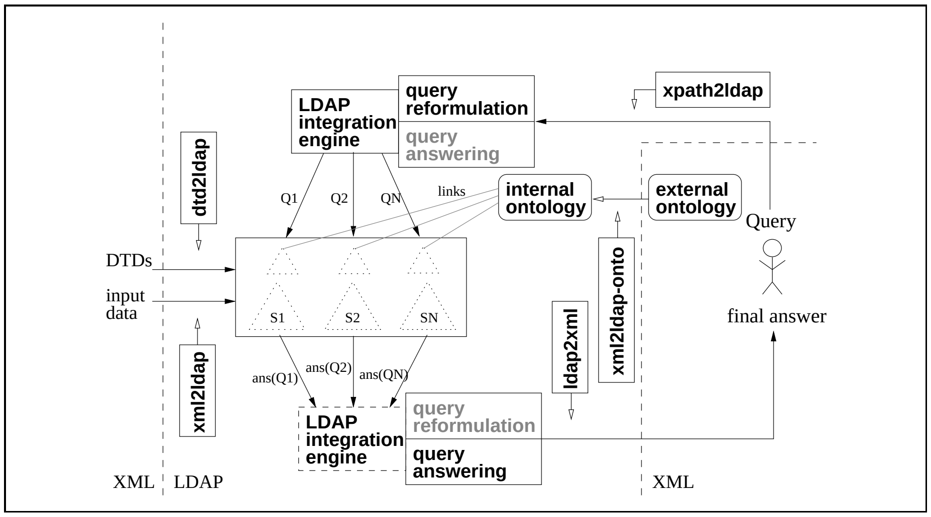

Fig. 1. The system architecture

This paper focuses on the *ontology* components of the system. The next section describes the functionality these components are intended to provide in `LGAccess`.

3.2 Role of Ontologies in `LGAccess`

According to Gruber [Gru93], an **ontology** is an explicit specification of an abstract, simplified view of the application domain, also called a conceptualization. Information integration systems often make use of ontologies: (i) to ease the problem of integration by limiting the domain of integration to a predefined collection of terms and their relationships; and (ii) to provide, via the ontology definition, a common query interface for all users of the data repository.

The LDAP model described in the previous section has support, not only for the integration process we just referred to, but also for the standard ontology design requirements [Usc96, UG96], namely:

Modeling capabilities: LDAP provides primitives for concepts and roles (by means of classes and attributes), key roles, subclass relationships, and general constraints.

Manipulation facilities: LDAP provides manipulation operators on top of which operators aimed at integration can be defined.

Schema querying: Since the LDAP data model is self-describing, concept and role names are allowed as part of the result of a query.

Flexibility and Maintainability: Changes to an LDAP-based ontology are easily made, allowing for the upgrade of concepts and roles in a flexible manner, as opposed to fixed ontology systems.

In addition, the modeling and technical grounds of LDAP, their tight connection to network and distribution channels offer capabilities not present in prevailing data description and integration approaches, as will be detailed in the next sections.

On the other side, to keep pace with the current trends on data exchange standards in the Web, the compatibility with XML presents an imperative.

Lead by these desiderata, we take a slightly different approach on constructing ontologies and choose to partition the ontology component into an *internal* ontology to be represented in the LDAP middleware in `LGAccess`, and an *external* ontology which resides as an XML source at a given, accessible location on the Web. Also, the *xml2ldap-onto* component is introduced in the system to provide a mapping between the conceptual entities of the internal LDAP ontology and those marked up externally in XML.

The role of ontologies in `LGAccess` is manifold: (i) they represent a common conceptual model for all users of a domain, (ii) they provide information that enables integration of data from distinct sources, and (iii) they allow to specify information that makes it possible to expand to other related domain ontologies if the received answer set does not suffice.

To incorporate an (external) ontology of the domain of interest available in the Web in our system, it is taken in its original representation, namely XML, and transformed into the corresponding LDAP-modeled ontology by the *xml2ldap-onto* component, as depicted in Fig. 1. Later, as part of the application design, the correspondences between the entries in the ontology and those in the source schemata are established by means of the link attributes to enable integration.

In the next section we show how ontologies can be represented, both, internally in LDAP and externally in XML.

4 Ontology Representation

Independent of the data model (LDAP, XML, DL, F-Logic, etc.), when developing a representation formalism for ontologies, there are two different parts in the ontology to be distinguished[HFB$^+$00]: The **ontology header** contains meta-information of an ontology, i.e., title, creator, subject, date, version, etc. In turn, the **ontology body** which distinguishes between two layers, the *generic body layer* and the *application-specific body layer*. The generic layer presents the main part of an ontology. It defines the vocabulary to be used in the ontology, as well as the structural behavior (i.e., in database terms, the signature) of these vocabulary entities within the ontology definition. The application-specific layer is thought to serve the needs that are characteristical for an application domain. Here we are concerned with the problem of data integration as a specific application that employs ontologies, and will at next talk about the *integration-specific layer* as an application-specific body layer in the ontology definition.

Following this design framework, we define the formalisms for representing ontologies in our system, both at the internal (LDAP) and at the external level (XML).

As a running example to illustrate our representation, we use an excerpt of the Geography ontology extracted (partially) from the Geography subontology in CYC [LG90]. Figure 2 provides a textual representation that uses indentation to indicate the subconcept relationship and parenthesis to indicate properties.

```
Geography(Continent,GeopoliticalEntity,Mountain,BodyOfWater,
          Desert,Island)
Continent(name,area)
GeopoliticalEntity(name,population)
  Country(area,car_code,has_city,has_capital,has_state,
          population_growth,infant_mortality,gdp_total,
          gdp_agri,gdp_ind,gdp_serv,inflation,government,
          encompassed,ethnicgroup,religion,language,border)
    IndependentCountry(indep_date)
  StateGeopolitical(area,abbrev,has_city,has_capital,is_state_of)
  City(longitude,latitude,is_city_of)
```

Fig. 2. Geography ontology

4.1 Internal Representation

According to the previously discussed way of differentiating the ontology content, it makes sense to start designing the LDAP-based ontology with introducing the class ontology as a subclass of the special top class with the following structure:

```
ontology OBJECT-CLASS ::= {
   SUBCLASS OF {top}
   MUST CONTAIN {oc,name}        // required attributes
   TYPE oc OBJECT-CLASS
   TYPE name STRING }
```

and the classes header and body as subclasses of the ontology class in the traditional LDAP data model. The structure for the header and bodyDefinition classes is given below.

To gain a better insight into the structure of LDAP ontologies, Figure 3 shows the (partial) LDAP-based representation of the Geography ontology sketched in Figure 2, which has been generated following the formalism described in this section.

The Ontology Header. Modeling the header layer in LDAP, we consider that OIL[HFB+00] makes the most comprehensive coverage of the features important for describing an ontology, and adopt the same structure as was there defined for this purpose under the *ontology-container* meta-level, with some minor modifications, inferring the same semantics, too.

The Ontology Body. To model the body layer of an ontology, we define the bodyDefinition class with the subclasses concept, role, roleConstraint, and conceptExpr.

The main constructs of an ontology, namely *concepts* and *roles*, together known as *terms*, are introduced in our LDAP-based ontology as subclasses of the bodyDefinition class with the following structure:

```
concept OBJECT-CLASS ::= {
   SUBCLASS OF {bodyDefinition}
   MAY CONTAIN {documentation,conceptOf,subconceptOf,
        roleConstraint,link,synonym,hypernymOf,hyponymOf}
   TYPE documentation STRING
   TYPE conceptOf DN(concept|conceptExpr)
   TYPE subconceptOf DN(concept|conceptExpr)
   TYPE roleConstraint DN(roleConstraint)
   TYPE link DN(source concept)
   TYPE synonym,hypernymOf,hyponymOf DN(concept) }
role OBJECT-CLASS ::= {
   SUBCLASS OF {bodyDefinition}
   MAY CONTAIN {documentation,subroleOf,domain,range,
                inverse,properties}
   TYPE documentation STRING
   TYPE subroleOf,inverse DN(role)
   TYPE domain DN(concept|conceptExpr)
   TYPE range DN(concept|conceptExpr), STRING
   TYPE properties {TRANSITIVE,SYMMETRIC,OTHER} }
```

The two classes roleConstraint and conceptExpr referred above are defined to have the following structure:

```
roleConstraint OBJECT-CLASS ::= {
   SUBCLASS OF {bodyDefinition}
   MUST CONTAIN {role}
   TYPE role DN(role)
   MAY CONTAIN {key,hasValue,valueType,filter,cardinality,
                maxCardinality,minCardinality,
                link,synonym,hypernymOf,hyponymOf}
   TYPE key {YES}
   TYPE hasValue,valueType DN(conceptExpr)
   TYPE filter LDAP-Filter
   TYPE cardinality,maxCardinality,minCardinality NUMBER
   TYPE link DN(source roleConstraint)
   TYPE synonym,hypernymOf,hyponymOf DN(roleConstraint) }
conceptExpr OBJECT-CLASS ::= {
   SUBCLASS OF {bodyDefinition}
   MAY CONTAIN {concept,roleConstraint,AND,OR,NOT}
   TYPE concept,NOT DN(concept|conceptExpr)
   TYPE roleConstraint DN(roleConstraint)
   TYPE AND,OR (DN(concept|conceptExpr),
                DN(concept|conceptExpr)+) }
```

The roleConstraint entries are used to define more constraints on the already defined roles in the ontology, this time in relation with the concept to which they belong. The conceptExpr entries are a kind of *defined terms*, i.e., they apply further restrictions on already defined terms (constraints on roles, or boolean combination on concepts), as the structure definition of the conceptExpr class illustrates.

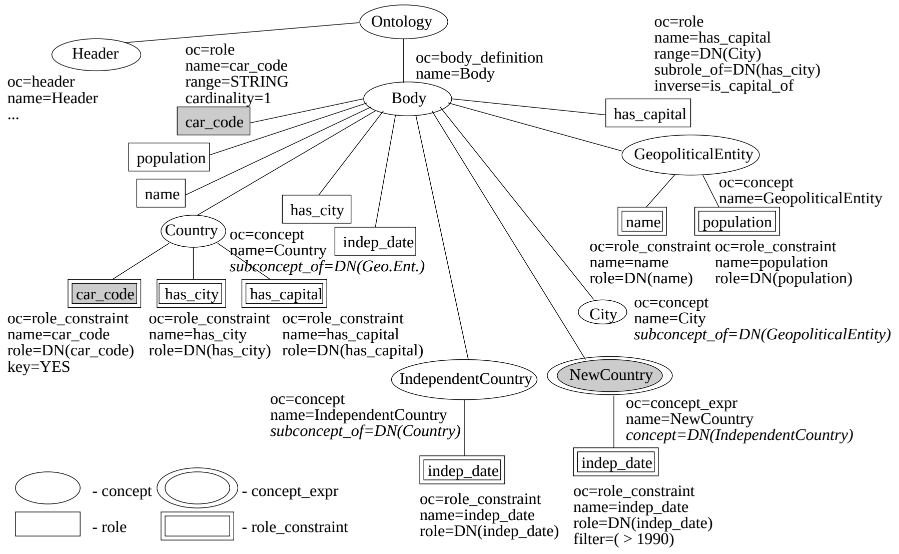

Fig. 3. Geography ontology in LDAP; shaded components are referred in the text

Each class introduced here contains a set of attributes, which can be categorized as either *generic constructs* or *integration-specific constructs* according to which layer of the body in the ontology definition they belong. The purpose of these attributes in the ontology definition is explained in the sequel.

Generic Constructs. The generic constructs create the generic body layer of an ontology, i.e., they define its vocabulary and its signature. Due to the obvious semantics of the attributes also included in other ontology modeling frameworks, we restrict our discussion to those attributes that are specific in our definition.

The oc and name attributes are required to be declared for every entry in the ontology since they are (a MUST) part of each class specification in the ontology schema starting with the ontology class. They represent the class (or classes) the entry belongs to, and the name of the entry, respectively. The conceptOf attribute expresses the meta-class relationships between concepts similar to the instanceOf element in the XOL ontology definition [KCT99]. It *may* be declared for a concept entry if it contains a meta-class which is referred by using its dn. The key attribute indicates whether or not a given *roleConstraint* can uniquely identify a concept for which it is defined. The filter attribute specifies *general constraints* imposed on a particular roleConstraint to define the socalled *defined terms*.

For example, in Figure 3, the car_code node is first defined globally in the body as a role by stating that it belongs to the class oc=role, its range is string, and its cardinality, 1, to be then used as a roleConstraint for a Country by stating that it belongs to the class oc=roleConstraint, and it inherits the car_code role role=DN(car_code) with all its definitions, restricting it, in addition, by stating that it is a key of Country. As an example of a **defined term** definition, the NewCountry node has a constraint of the form filter=(> 1990) on the indep_date role that indicates that only independent countries whose independence date is more recent than 1990 are considered to be NewCountries.

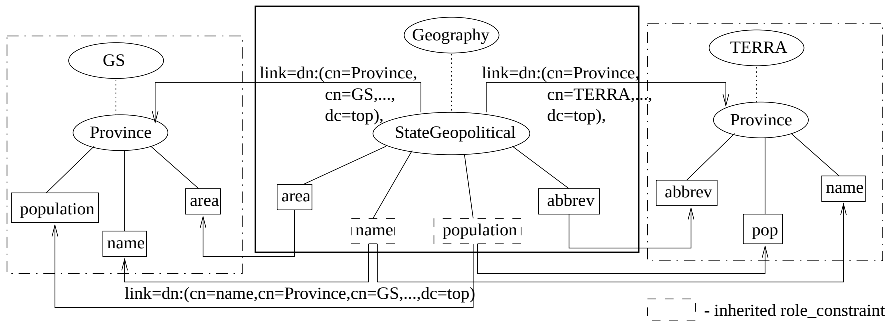
GS
Province
population
name
area
Geography
link=dn:(cn=Province, cn=GS,..., dc=top),
StateGeopolitical
area
name
population
abbrev
link=dn:(cn=Province, cn=TERRA,..., dc=top),
TERRA
Province
abbrev
pop
name
link=dn:(cn=name,cn=Province,cn=GS,...,dc=top)
- inherited role_constraint

Fig. 4. The GS and TERRA source schemata embedded in the Geography ontology

Integration-Specific Constructs. The link attribute is used to enable integration. It indicates that a given entry in the ontology is terminologically equivalent with the term in the LDAP source schema it (its value) references to. Variants of a link attribute enable more complex semantic relationships to be established between entries at the ontology level and entries at the source schema level. By using the link attributes, our system is able to resolve most of the conflicts existing between distinct sources, allowing to relate any queries performed over the ontology with their corresponding source level queries in a relatively easy fashion. Figure 4 provides a graphical representation of the way the link attributes are specified to relate our ontology and the schemata of the sources that are subject of integration, say GS and Terra sources [May00].

In order to capture the semantic subtleties, differences, and similarities among ontologies, we use interontology relationship constructs, namely, synonym, hypernymOf and hyponymOf attributes. The synonym attribute indicates that two concepts with different names are equivalent, and should be treated as such, whereas the hypernymOf and hyponymOf specify a subsumption relationship that defines whether a term is more general than another, as defined by the former, or more specific, as defined by the latter.

What makes LDAP particularly attractive in the middleware is its hierarchical namespace, where each entry is uniquely identified with the dn of the entry plus the name of the LDAP host where the entry resides. This allows the conceptual objects to be unambiguously referenced by the integration-specific constructs, be it on the same host or elsewhere in the network.

4.2 External Representation

There are two parties that can make use of ontology definitions in the LGAccess system. One are information searchers who do not substantially care about the semantic descriptions of entities as described by the ontology and would rather be satisfied with a kind of a **view** containing only structural elements of the domain and keeping the internal layout of the view across several different sources and/or ontologies transparent to them. The other party includes the users/applications who are interested about the meta-data definitions in the **ontology**. In the next two sections we try to consolidate both of these important deployments of LDAP ontologies for the usage in practice adapting them to the XML-based trading technologies.

XML Ontology. The Extensible Markup Language (XML) [BPSMM00] is on the way to prevail over all other existing data representation formats in the Web. As a consequence, expressing ontologies in XML has become crucial for a worldwide knowledge-to-knowledge communication. Recently, as a result of a joint effort of researchers to provide a standard ontology description language in XML, several proposals for the socalled XML ontologies [KCT99, HFB$^+$00, DvHB$^+$00] appeared. In this line, we provide a model for describing ontologies in XML to serve as the external representation of ontologies in our system.

We illustrate how the description of ontologies according to our definition is done by an example. Following this formalism, the Geography ontology (Fig. 2) looks as in Fig. 5.

A complete syntax for XML ontology specification can be easily inferred by carefully examining the ontology definitions in LDAP we presented before. The key attribute, although not originally defined for XML ontologies, has been included into the ontology definition. It may either not be defined at all, or get the YES value. The filter attribute extends the value definitions of OIL and XOL. In OIL, only some of the previously defined <class> or <class-expr> elements (types) in the ontology are allowed to be assigned to the <value-type> and <has-value> elements. XOL extends OIL in this aspect, recognizing basic data types, i.e., strings, integers, etc., as well as (numeric-minimum,numeric-maximum) range restrictions in the facet (roleConstraint) definition. For example, the above filter definition in XOL is specified by the constraint numeric-minimum=1990 within the indep_date facet. But, constraints on basic or defined types, like filter=(= *), filter=(="*nia"), filter=(~="Baden"), etc., and other expressions supported by LDAP filters, enrich the signature specification functionality for ontologies. The link attribute is transparent to the XML users and is specified only in the LDAP middleware.

XML View. The XML ontology tree cannot directly be used to derive the syntax of a query against an XML instance. Instead, we propose to use an *XML view* of the ontology which represents an higher level abstraction of the domain of interest released from accurate semantic descriptions. In this way, to access the data in LGAccess, the information searcher formulates

```
<ontology><header> ... </header>
  <body-definition>
    <role-def><role name="name"/><range>STRING</range></>
    <role-def><role name="population"/><range>STRING</range></>
    <role-def><role name="car_code"/><range>STRING</range>
              <cardinality>1</cardinality></>
    <role-def><role name="has_city"/>
              <range><concept name="City"/></range>
              <inverse>is_city_of</inverse></>
    <role-def><role name="has_capital"/>
              <subrole-of><role name="has_city"/></subrole-of>
              <range><concept name="City"/></range>
              <inverse>is_capital_of</inverse></>
    <role-def><role name="indep_date"/>
              <range>NUMBER</range></>
    <concept-def><concept name="GeopoliticalEntity"/>
      <role-constraint><role name="name"/></role-constraint>
      <role-constraint><role name="population"/></role-constraint></>
    <concept-def><concept name="Country"/>
      <subconcept-of><concept name="GeopoliticalEntity"/></subconcept-of>
      <role-constraint><role name="car_code"/><key>YES</key>
                       </role-constraint>
      <role-constraint><role name="has_city"/></role-constraint>
      <role-constraint><role name="has_capital"/></role-constraint></>
    <concept-def><concept name="City"/>
      <subconcept-of><concept name="GeopoliticalEntity"/></subconcept-of>
      </>
    <concept-def><concept name="IndependentCountry"/>
      <subconcept-of><concept name="Country"/></subconcept-of>
      <role-constraint><role name="indep_date"/></role-constraint></>
    <concept-def><concept name="NewCountry"/>
      <subconcept-of><concept name="IndependentCountry"/></subconcept-of>
      <role-constraint><role name="indep_date"/>
                       <filter>( > 1990)</filter></role-constraint></>
  </body-definition></ontology>
```

Fig. 5. Geography ontology in XML

a query by using the view as a pattern. The XML view is obtained from the XML ontology following these principal criteria:

- in the case of data querying, the focus shifts from elements as semantical entities to elements as data entities;
- neither descriptions about the semantics of terms in the domain, nor descriptions about their structural relationships (documentation, class membership, domain, key, cardinality, general constraints, etc.) are given;
- defined relationships in the concept hierarchy (if any) need to be evaluated, resulting, e.g., in additional inherited roleConstraints from super-concepts (structural inheritance);

– value type descriptions denote that the property contains a reference to an element of a given type.

The result is a kind of a generic XML instance. The XML view derived from the XML Geography ontology shown in Figure 5 looks as follows:

```
<Geography>
 <GeopoliticalEntity oc=concept>
   <name oc=roleConstraint type=STRING/>
   <population oc=roleConstraint type=STRING/>
   </GeopoliticalEntity>
 <Country oc=concept subconceptOf="GeopoliticalEntity">
   <name oc=roleConstraint type=STRING/>
   <population oc=roleConstraint type=STRING/>
   <area oc=roleConstraint type=STRING/>
   <car_code oc=roleConstraint type=STRING key=YES/>
   <has_city oc=roleConstraint type=City/></Country>
   ...
 <StateGeopolitical oc=concept subconceptOf="GeopoliticalEntity">
   <name oc=roleConstraint type=STRING/>
   <population oc=roleConstraint type=STRING/>
   <area oc=roleConstraint type=STRING/>
   <abbrev oc=roleConstraint type=STRING/></StateGeopolitical>
 <City oc=concept subconceptOf="GeopoliticalEntity">
   <name oc=roleConstraint type=STRING/>
   <population oc=roleConstraint type=STRING/></City>
</Geography>
```

When formulating an XPath query over this view, say:

$$Geography//StateGeopolitical[population > 10000000]$$

the result is computed by the queries:

$$GS//Province[population > 10000000] \quad \text{in the source gs.xml and}$$

$$TERRA//Province[pop > 10000000] \quad \text{in the source terra.xml,}$$

leaving the user away from the fact that this result is obtained as a combination of data from two distinct sources (see Figure 4).

In order to process such a query in our system, i.e., to decompose it into source-level queries, the set of the link attributes found in both `concepts` and `role constraints` are used. The process of evaluating queries in our system is an issue for itself and is tackled elsewhere (see [AML]) in detail.

4.3 Mapping Ontologies from XML to LDAP

In the previous section we have shown two separate representations of ontologies in XML and in LDAP, and enumerated some of the reasons of keeping them both in the system. There is a canonical transformation between these two formalisms which is performed by the *xml2ldap-onto* component in `LGAccess`. The LDAP ontology representation in Figure 3 is obtained from the XML Geography ontology shown in Figure 5 as a result of the mapping algorithm.

5 Related Work

Although extensive work has been done in the area of modeling ontologies for data integration, most researchers use logic-based languages like description logics [Bor95], or F-Logic [KLW95] to describe them and benefit from the reasoning abilities of logic systems. Our approach, on the other hand, uses a relatively simple model with a clear syntax that allows us to provide a unified formal framework to be used for both, ontology definition and information integration using the standard and well-established LDAP technology in the middleware.

Recently, a number of consecutive models adapting to XML (RDF [LS99] and RDF-Schema [BG00], XOL [KCT99], OIL [HFB$^+$00]) appeared, leading to the socalled XML ontologies. To keep pace with current trends on data, we also support the use of ontologies in XML syntax at the external (user-side) level.

Furthermore, due to the hierarchical and flexible structure of LDAP, we can very easily integrate not only structured data, but also semistructured data, as opposed to systems like Sims [AK93] or Observer [MKSI00] that are limited to the expressive power of their ontology description languages and are only able to deal with relational and flat file databases. Not even systems like Florid [HKL$^+$98], Ontobroker [DEFS99], or Momis [BCV99], which combine the reasoning capabilities of logic systems with the expressive power of an object-oriented model are able to describe XML data, as naturally as we do in our system. The lack of a tree-like modeling structure forces them to map XML into an artificial structure not particularly well suited for such graphs, whereas our model is based on a tree structure that resembles that of XML. Tsimmis [GMPQ$^+$97] addresses the above deficiencies by taking a similar approach to ours, namely using the OEM model to describe mediated views and sources, as opposed to our LDAP-based data model, which offers, by design, additional benefits derived by the nature of the LDAP directory services.

Some other approaches, like [BGL$^+$99, CCS00] use XML as their common model for data exchange and integration, but fail to consider the use of ontologies as an integral part of the system. Our LDAP-based model, on the other hand, provides seamless integration with ontologies and DTDs, which allows us to scale our system both, at the source level and at the user level. These advanced features, combined with the fact that the hierarchical LDAP namespace allows us to implicitly distinguish nodes belonging to different source trees without incurring in any additional work, makes LDAP an ideal candidate for the field of integration.

6 Conclusion

In this paper, we have presented an approach of combining XML and ontologies into a unified LDAP-based framework in the middleware in our integration system that provides a common querying interface to XML data sources by means of a representation formalism for ontologies. Given the ubiquity of XML on the Web, our system will be of crucial importance for applications that need to interact with semistructured databases. It provides obvious advantages with respect to more classical integration approaches because of the simplicity, coherence and uniformity of the LDAP model.

Furthermore, our system is, to the best of our knowledge, the only system that combines the advantages of a hierarchical data model, used among other things to map XML documents, and those of ontologies as basis for its integration capabilities. A preliminary version of the system is already in the works and providing very promising results, as detailed in [ML01a].

References

[AK93] Y. Arens and C. Knoblock. SIMS: retrieving and integrating information from multiple sources. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 22(2):562–563, June 1993.

[AML] L. Ahmedi, P. J. Marrón, and G. Lausen. Ontology-based Access to Heterogeneous XML Data. In Proceedings of the Workshop on Web Databases at the Annual Conference of the German and Austrian Computer Societies (GI'2001), Vienna, Austria, Sept. 2001.

[BCV99] S. Bergamaschi, S. Castano, and M. Vincini. Semantic Integration of Semistructured and Structured Data Sources. *SIGMOD Record*, 28(1):54–59, 1999.

[BG00] D. Brickley and R. Guha. Resource Description Framework (RDF) Schema Specification 1.0 - W3C Recommendation. `http://www.w3.org/TR/2000/CR-rdf-schema-20000327/`, March 2000.

[BGL+99] C. Baru, A. Gupta, B. Ludaescher, R. Marciano, Y. Papakonstantinou, and P. Velikhov. XML-based Information Mediation with MIX. In Demo Session, ACM-SIGMOD'99, Philadelphia, PA, 1999.

[Bor95] A. Borgida. Description Logics in Data Management. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):671–682, 1995.

[BPSMM00] T. Bray, J. Paoli, C. Sperberg-McQueen, and E. Maler. Extensible Markup Language 1.0. `http://www.w3.org/TR/2000/REC-xml-20001006`, 2000.

[CCS00] V. Christophides, S. Cluet, and J. Siméon. On Wrapping Query Languages and Efficient XML Integration. Proceedings of ACM SIGMOD Conference on Management of Data, Dallas, Texas, May 2000.

[DDM00] J. R. Daniel, S. DeRose, and E. Maler. XML Pointer Language 1.0. `http://www.w3.org/TR/2000/CR-xptr-20000607`, June 2000.

[DEFS99] S. Decker, M. Erdmann, D. Fensel, and R. Studer. Ontobroker: Ontology based access to distributed and semi-structured information. In DS-8: Semantic Issues in Multimedia Systems. Kluwer Academic Publisher, 1999.

[DMO00] S. DeRose, E. Maler, and D. Orchard. XML Linking Language (XLink) Version 1.0 - W3C Proposed Recommendation. `http://www.w3.org/TR/xlink/`, December 2000.

[DvHB+00] S. Decker, F. van Harmelen, J. Broekstra, M. Erdmann, D. Fensel, I. Horrocks, M. Klein, and S. Melnik. The Semantic Web: The Roles of XML and RDF. *IEEE Internet Computing*, 4(5), 2000.

[GMPQ$^+$97] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. D. Ullman, V. Vassalos, and J. Widom. The TSIMMIS Approach to Mediation: Data Models and Languages. *Journal of Intelligent Information Systems*, 8(2):117–132, 1997.

[Gru93] T. R. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowledge Aquisition*, 5(2):199–220, 1993.

[HFB$^+$00] I. Horrocks, D. Fensel, J. Broekstra, S. Decker, M. Erdmann, C. Goble, F. van Harmelen, M. Klein, S. Staab, R. Studer, and E. Motta. The Ontology Interchange Language OIL. Technical Report, Free University of Amsterdam, http://www.ontoknowledge.org/oil/, 2000.

[HKL$^+$98] R. Himmeröder, P. Kandzia, B. Ludäscher, W. May, and G. Lausen. Search, Analysis, and Integration of Web Documents: A Case Study with FLORID. In *Proc. Intl. Workshop on Deductive Databases and Logic Programming (DDLP'98)*, pp. 47–57, Manchester, UK, 1998.

[HSG99] T. A. Howes, M. C. Smith, and G. S. Good. *Undestanding and Deploying LDAP Directory Services.* Macmillan Technical Publishing U.S.A., 1999.

[JLM$^+$99] H. V. Jagadish, L. V. S. Lakshmanan, T. Milo, D. Srivastava, and D. Vista. Querying Network Directories. In *Proceedings of ACM SIGMOD International Conference on Management of Data, Philadephia, Pennsylvania*, 1999.

[KCT99] P. D. Karp, V. K. Chaudhri, and J. Thomere. XOL: An XML-based ontology exchange language. Technical report, Version 0.3, 1999.

[KLW95] M. Kifer, G. Lausen, and J. Wu. Logical Foundations of Object-Oriented and Frame-Based Languages. *Journal of the ACM*, 42(4):741–843, 1995.

[LG90] D. B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the CYC Project.* Addison-Wesley, 1990.

[LS99] O. Lassila and R. Swick. Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation. `http://www.w3.org/TR/REC-rdf-syntax/`, February 1999.

[May00] W. May. Mondial DB. `http://www.informatik.uni-freiburg.de/~may/Mondial`, 2000.

[MKSI00] E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. *Distributed and Parallel Databases*, 8(2):223–271, 2000.

[ML01a] P. J. Marrón and G. Lausen. HLCaches: An LDAP-based Distributed Cache Technology for XML. Tech. Rp. 147, Uni. Freiburg, Jan. 2001.

[ML01b] P. J. Marrón and G. Lausen. On processing XML in LDAP. In *Proc. of 27th Intl. Conf. on Very Large Databases, Roma, Italy*, 2001. (to appear).

[UG96] M. Uschold and M. Gruninger. Ontologies: principles, methods, and applications. Knowledge Engineering Review, 11(2), 93–155, 1996.

[Usc96] M. Uschold. Building Ontologies: Towards a Unified Methodology. In *16th Annual Conf. of the British Computer Society Specialist Group on Expert Systems*, Cambridge, UK, 1996.

[WHK97] M. Wahl, T. Howes, and S. Kille. Lightweight Directory Access Protocol (v3). RFC 2251, December 1997.