Information Integration via an End-to-End Distributed Semantic Web System

Dimitre A. Dimitrov[1], Jeff Heflin[2], Abir Qasem[2], and Nanbor Wang[1]

[1] Tech-X Corporation, 5621 Arapahoe Avenue, Suite A, Boulder, CO 80303
{dad, nanbor}@txcorp.com
[2] Lehigh University, 19 Memorial Drive West, Bethlehem, PA 18015
{heflin, abir.qasem}@cse.lehigh.edu

Abstract. A distributed, end-to-end information integration system that is based on the Semantic Web architecture is of considerable interest to both commercial and government organizations. However, there are a number of challenges that have to be resolved to build such a system given the currently available Semantic Web technologies. We describe here the ISENS prototype system we designed, implemented, and tested (on a small scale) to address this problem. We discuss certain system limitations (some coming from underlying technologies used) and future ISENS development to resolve them and to enable an extended set of capabilities.

1 Introduction

Different groups or subdivisions of a large organization often develop data management solutions semi-independently from each other with their own data schemas. Moreover, the data is often semi-structured, contains large binary (e.g. images) entities, dynamically evolves, grows, and is distributed over a number of data servers on a network. The efficient extraction of relevant information from all available and diverse data sources require the solution of a number of problems related to data representation and information integration in a distributed environment [1]. We describe here our initial implementation of the ISENS system that we designed as a test case to address these problems. We integrated and extended emerging Semantic Web [2,3] technologies to handle information integration (based on logical views [4]) and querying of distributed metadata. We have developed the ISENS system prototype during a Phase I Small Business Inovation Research (SBIR) project.

At the user (top) level, the service that ISENS is to provide consists of answering queries over distributed metadata repositories that describe underlying semi-structured data. Moreover, ISENS is designed to address a number of challenges in information representation that cannot be solved with markup languages such as HTML and XML. ISENS is built on top of specific Semantic Web technologies that support relations over different concepts expressed in the metadata. This allows the use of logical reasoners to derive semantic information from the metadata and its representation (markup). Furthermore, a language is

needed to represent queries for computer programs to be able to parse, process, and extract relevant answers from the available metadata.

In a distributed environment of data centers that generate metadata on the same domain of knowledge, the syntax of the markup that each data center uses will generally be different from the syntax that each of the other centers are using to encode their data. The modeling of the data may be different too. For example, one data center modeler may see manufacturer as a property of a device, i.e. a device has a manufacturer, another modeler may see it as a property related to order, i.e. an order from a manufacturer. Then, the problem is how to provide information integration over metadata that each data center provides. Specifically, how can a user query the metadata from the distributed data centers using a uniform syntax even though the different data centers might be using their own syntax to markup the data they provide and how can the syntactic/semantic differences be resolved?

To encode metadata and address its representation we used the Resource Description Framework (RDF) and the OWL Web Ontology Language. This also includes the technologies that RDF and OWL extend and their associated schemas (XML, XMLS, and RDFS). For query representation, parsing, and processing, we selected and implemented a solution based on the SPARQL [5] query language for RDF. SPARQL is specifically designed to represent information for the Semantic Web. It provides to users and developers an extensible and flexible framework to construct queries over the extended set of RDF/RDFS capabilities. Finally, to address the third problem (uniform query over heterogeneous metadata from different and distributed data centers) we adopted the MiniCon algorithm [6]. This is a scalable algorithm for answering queries using views over different databases. We have enhanced it to work with a Semantic Web data model (as opposed to a relational data model in databases).

For testing of the implementation in a distributed, heterogeneous environment, and to simulate a plausible real-world scenario, we developed independently two different ontologies, one by the group at Tech-X Corp. and one by the Lehigh University group, on the same domain of knowledge. The two ontologies, while describing the same knowledge base, have different markup syntax. And furthermore, they model concepts differently from one another. To resolve syntactic and semantic heterogeneity we have used OWL axioms to describe a map between them. The axioms relate concepts from one ontology to the other. ISENS consults this map to retrieve information from sources that may have otherwise appeared to be providing differing information. In our framework, a map is an ontology that imports these ontologies whose terms it will align.

In the rest of the paper, we discuss first the architecture of the ISENS prototype we designed and implemented. Then, we describe the approach we developed for information integration with mapping ontologies and illustrate it with a number of example test queries. Finally, we summarize our experience from the implementation and testing of ISENS, discuss current limitations (some inherited from underlying technologies we used) and future development to extend the system.

2 Prototype System Architecture

The ISENS system prototype consists of three main components: the Web Interface Component (WIC), the Distributed Querying System (DQS), and the Distributed Enabling Component (DEC). For the purpose of driving the development of the prototype system, we developed two ontologies on a specific domain of knowledge. However, the issues addressed by the ontologies are representative of the general case about managing business related information faced by commercial organizations, e.g. for tracking on-going performance and/or forecasting future developments. Additionally, we created data individuals based on the two ontologies, configured two remotely located Sesame [7,8] RDF/OWL metadata repositories, and stored in them the ontology individuals. The overall architecture of ISENS is represented in Fig. 1.

The WIC provides a basic user interface to ISENS that is accessible via a Web browser. This user interface allows submission of SPARQL queries and displays the returned results. The WIC takes the user input in the form of a SPARQL query and passes it to the DQS for processing. The DQS returns the answers it finds in response to the query. The answers are encoded in XML and passed back to the WIC. The WIC then parses the returned XML and presents it to the user in a certain format.

The DQS component processes each SPARQL query and enables data integration over different, distributedly located, ontologies. The DQS contains a set of source descriptions. A source description specifies the attributes that can be found in the data source and the constraints on the contents of the source. One of the approaches for specifying source descriptions is to represent the contents of a data source as a view over a mediated schema [4]. This approach is known as Local as a View (LAV) in the database literature.

A mediated schema is a set of virtual relations that provide a uniform interface to the user. The LAV approach facilitates the addition of new data sources since existing views do not have to change when a source is added to the system. Although we use LAV as a foundation, in the DQS system the ontology of the query plays the role of the mediated schema. Thus, either of our two ontologies can serve as the mediated schema. In this sense, we are more general than LAV-based information integration (there is no need to construct a mediated schema in our system). The DQS relies on KAON2 [9] for its description logic reasoning functionality. The mapping ontologies that the DQS needs to do its information integration tasks are written in a subset of OWL DL.

In order to answer a query, a data integration system needs to translate a query in terms of the mediated schema into one that refers directly to the schemas in the data sources. Since the contents of the data sources in LAV are described as views, the translation problem amounts to finding a way to answer a query using a set of views. MiniCon [6] is one of the more scalable LAV algorithms.

The basic idea of the MiniCon algorithm is to examine each query sub goal against each view and check if the view can be used to "answer" the query sub goal and if so, in what "form"? It also treats "shared" variables carefully for certain optimizations. Finally, it combines views to answer all query sub goals.

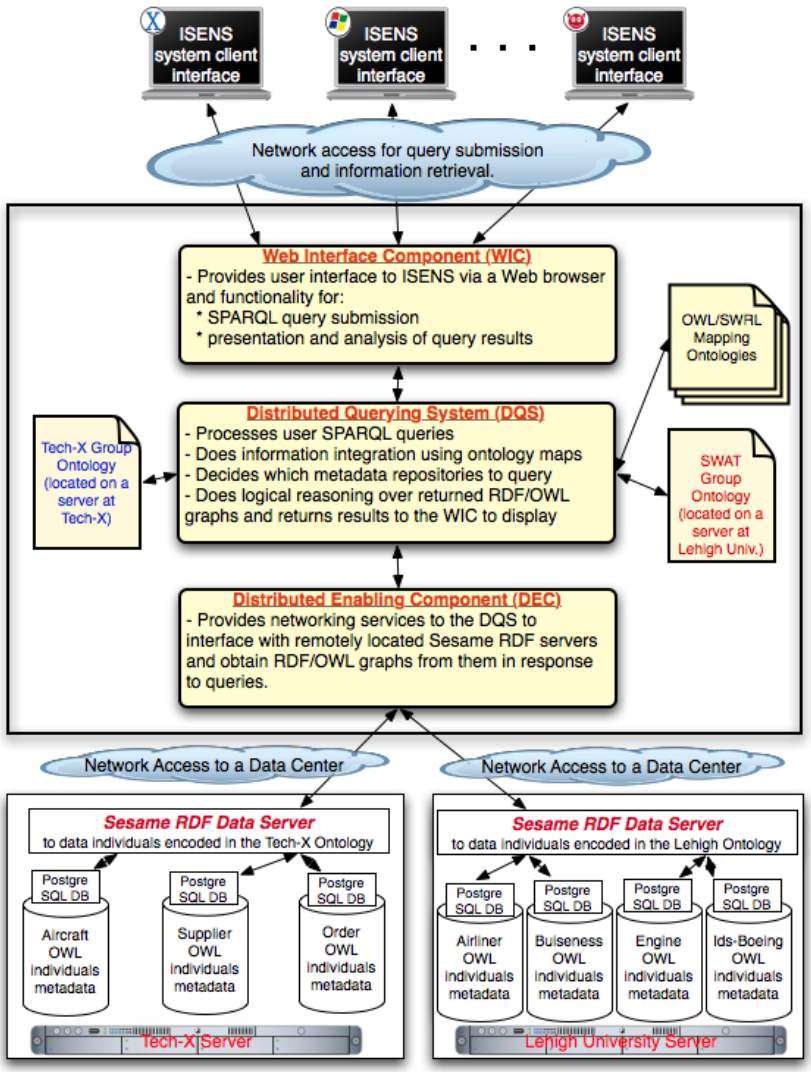

Fig. 1. The architecture of the ISENS prototype system is currently configured for querying of distributed RDF/OWL metadata at two data centers: one at Tech-X Corporation and the other at Lehigh University. The OWL individuals data is stored in multiple PostgreSQL databases that are managed by Sesame RDF servers. The Sesame servers provide remote access to the OWL individuals data. This set up allows testing of ISENS in a distributed and heterogeneous environment.

The DQS uses network services that are provided by the DEC to retrieve RDF/OWL graphs from distributedly located Sesame RDF servers. In the current configuration, the DEC can remotely query Sesame RDF servers. The DEC is also being designed to provide authentication and security services in ISENS per user/customer requirements.

For the ISENS prototype testing, we separated the OWL data individuals into seven different parts. These were then loaded into two Sesame RDF servers configured to use PostgreSQL databases. The Sesame RDF server at Tech-X Corporation stores the data individuals encoded in the Tech-X group ontology into three PostgreSQL databases. The data individuals encoded in the Lehigh University group ontology syntax are stored on a Sesame server at Lehigh University into four PostgreSQL databases.

3 Information Integration with Mapping Ontologies

In this Section, we use a top-down approach to describe an information integration solution that incorporates Semantic Web technologies. First, we consider a general way to address the problem. Then, we report on our current implementation in the ISENS system.

3.1 General Approach

In the general case, an operational environment consists of a set of ontologies and a set of data sources.

Each data source commits to one or more ontologies. The ontologies may include axioms which augment the data sources that commit to them with additional semantic information. Each ontology may extend other ontologies in order to refine the semantic definitions of the concepts in them. This model can be applied to legacy data sources if we assume that each data source has an ontology associated with it. Minimally, the ontology can just be a database schema. Integration is enabled by special ontologies called mapping ontologies. This model is represented in Fig. 2.

The key problem here is that a query may be formulated using the language of one ontology but the relevant data may be in a data source that commits to a different ontology. The DQS will solve this problem by using the mapping ontologies to decompose the query into a set of queries that can be directly sent to some subset of the data sources. If there is no direct mapping, it will traverse through a semantic connection of maps to determine relevant ontologies and data sources. Consider our example described above. There is a direct map M12 between the O1 and O2 ontologies. Therefore, a query q using terms from O1 can access data in the S2 and S3 data sources using the map. There is, however, no direct map between O1 and O3 but by combining M12 and M23 we can now retrieve answers from S1, S2, S3, S4 and S5.

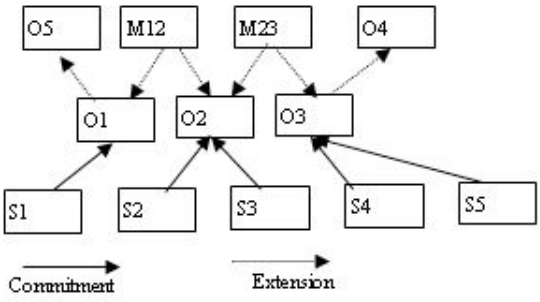

Fig. 2. Information integration using Semantic Web ontologies. Here, O1-O5 represent ontologies, S1-S5 represent data sources, and M12 and M23 are mapping ontologies.

3.2 Implementation in the DQS

The basic structure of the implemented DQS architecture is shown in Figure 3. It involves two separate processes. The Map Processor reads the source descriptions and the mapping ontologies in order to load a meta-information knowledge base. Source descriptions identify sources that have relevant information with respect to classes and/or properties. If a source can express that it has relevant information we can choose to query it as opposed to other sources that do not express this information. In this way we can locate the desired information without querying every possible source. Having relevant information for a query, however, does not mean that the source is capable of answering the query completely. It only indicates that the source has some useful information on the query.

To implement source descriptions in OWL, we introduce a `isRelevantFor` predicate in our framework. We use a new namespace `meta` for this predicate. For example, a data source may have the following OWL statement to assert that it has relevant information on individuals of the class `Aircraft`.

```
<meta:isRelevantFor>
  <owl:Class rdf:ID="Aircraft"/>
</meta:isRelevantFor>
```

The load is done asynchronously from the query process. It can check for updates periodically or receive updates directly at a users request. The meta-information is augmented with ontological information in order to ease the reformulation process.

When the DQS receives a query, it will be processed by the Reformulator. Using the meta-information knowledge base, the Reformulator determines which data sources should be queried and what queries should be sent to them. This is currently based on the MiniCon algorithm for relational data sources. In a

future development, we will extend the DQS with a peer data management algorithm [10] that generalizes the two common mapping schemes from information integration: local-as-view (LAV), where the local source schemas are defined as views over a global mediated schema, and global-as-view where the global mediated schema is defined as a view over the local source schemas. Note, this will reuse much of the MiniCon algorithm we already implemented.

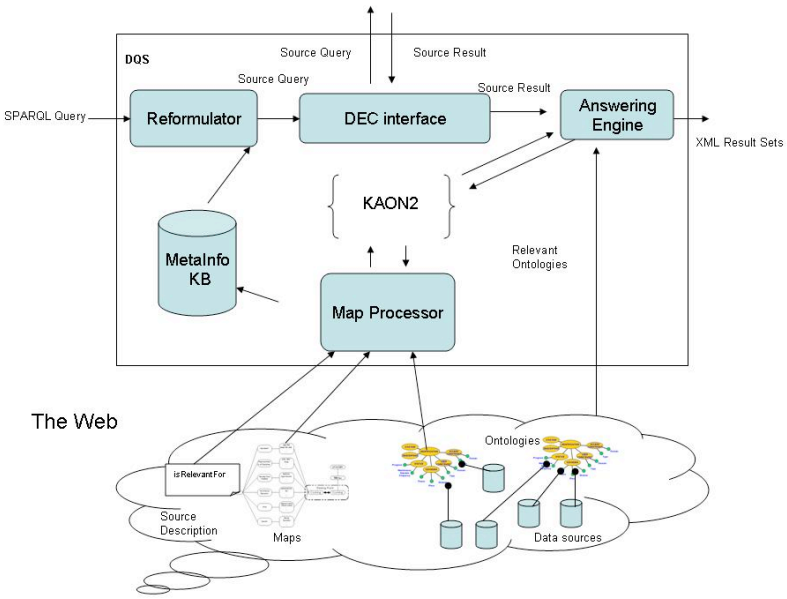

Fig. 3. Architecture of the DQS.

The Reformulator produces a set of $\langle source\,id, query \rangle$ pairs that are given to the DEC. Using this information, the DEC queries the data sources using their native interfaces and returns the results in a standard format.

The answers returned by the DEC are processed by the Answering Engine. The answering engine loads these answers into the KAON2 knowledge base system. It also loads all relevant ontologies, so that the KAON2 knowledge base represents the subset of data that is relevant to the query. The Answering Engine issues the original query to the knowledge base and outputs the answers in XML format. *Note, by using a knowledge base system here, we are able to benefit from the inferences that KAON2 can derive.* For example, KAON2 can determine the sub classes of a given class. If some of the data is described in terms of a sub class (as opposed to the class mentioned in the query), we will still obtain them

in our results. We implemented the DQS (the code for the MiniCon algorithm is part of it) in Java.

3.3 Aircraft Ontologies and Data Individuals

The two ontologies were created independently (one by the Tech-X group and the other by the Lehigh University group) to drive the development and testing of the ISENS system. They consist of about 30/35 classes each, four levels deep, and of about 30/35 object and data properties each. We developed the two ontologies on aircraft and related manufacturers. The rationale for this decision is that information for this domain of knowledge is very easy to obtain off the Internet. Moreover, issues for integrating information relevant to the aircraft industry will, in general, be similar to other enterprise-related domains of knowledge. We show a graph structure for one of the ontologies in Fig. 4.

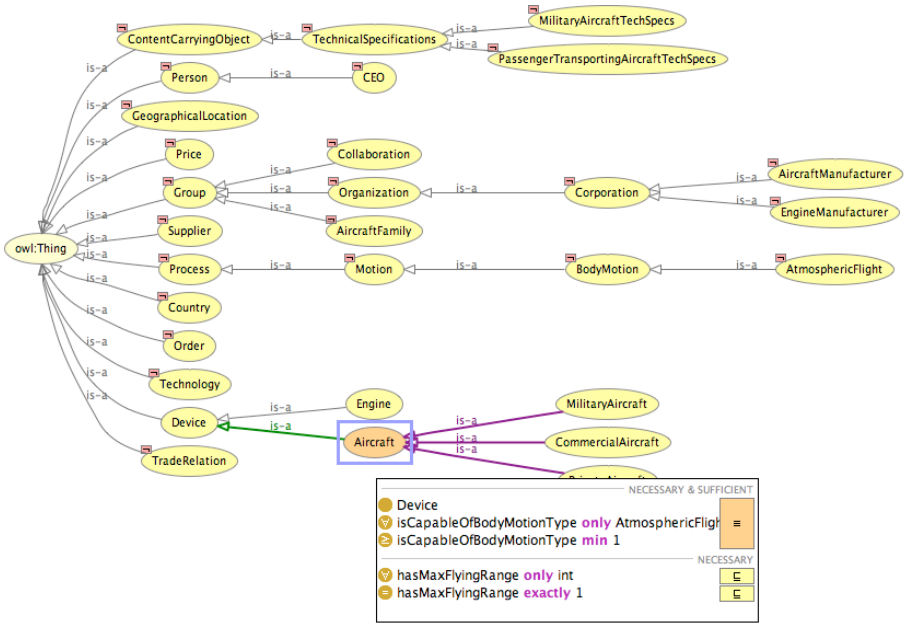

Fig. 4. The asserted model for the Tech-X group ontology and its hierarchy shows how we have modeled the domain of knowledge on aircraft manufacturers and related conceptual entities.

We developed the ontologies independently to simulate a real-world scenario where different data centers will generally provide different encoding syntax to

the metadata they provide. The integrated access to the data from such heterogeneous sources will then require a system to enable translation among the different encodings. The translation will have to be smart enough to support not only the equivalent object/data types, but also the relations among them.

For the current ISENS set up, the Tech-X ontology[3] (without its related data individuals) and the corresponding Lehigh University group ontology[4] are available on-line.

After a model for the concepts in the ontology has been developed, one can start creating data individuals of the different types supported. Moreover, once we have developed the ontologies to model the domains of knowledge that are of interest, we can design and implement computer programs to automate the generation of RDF/OWL data individuals that describe existing and/or dynamically evolving raw data. We created seven data sets and stored them in seven different PostgreSQL databases that were accessible via our two Sesame RDF servers (see Fig. 1). For each data set, we created one source description file that summarizes the kind of data that can be found in it.

3.4 ISENS Map Ontologies

Since the two ontologies were developed separately, many similar concepts have been modeled using different terms. For example, the Tech-X group defined the concept of commercial aircraft as the class `CommercialAircraft`, whereas the Lehigh University group defined it as `Airliner`. To integrate data sources that commit to these ontologies one needs to formally specify the fact that `CommercialAircraft` is equivalent to `Airliner`. We achieve this in our solution as follows:

```
<owl:Class rdf:about="&txcorp;CommercialAircraft">
  <owl:equivalentClass rdf:resource="&swat;Airliner"/>
</owl:Class>
```

Object and datatype property equivalence is expressed with similar syntax in the map ontology, e.g.:

```
<owl:ObjectProperty rdf:about="&txcorp;wasOrderedBy">
  <owl:equivalentProperty rdf:resource="&swat;soldTo"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:about="&txcorp;wasOrderedOn">
  <owl:equivalentProperty rdf:resource="&swat;sellDate"/>
</owl:DatatypeProperty>
```

An ontology that is a collection of this type of axioms is herein after called a map ontology and each of these axioms is referred to as a map. The complete source code of our map ontology is available[5] on-line.

[3] `http://fusion.txcorp.com/~dad/isens/onto/txcorp/aircraft.owl` (the URLs cited here were last accessed in May 2006).

[4] `http://swat.cse.lehigh.edu/onto/airdata.owl`

[5] Map ontology URL: `http://swat.cse.lehigh.edu/onto/txcorp-swat-map.owl`

Mapping ontologies are published using the same OWL syntax that is used to define ontologies that are being mapped. The maps are then available for use by anyone authorized to use the system. *As such, the mapping work performed by one organization can be easily shared with others.* This is in stark contrast to many contemporary integration efforts in which custom code is written for each integration effort.

The use of OWL as a mapping language combined with a suitable description logic reasoner KAON2, allowed us to express more complex maps than mere equivalences. In addition to constructing taxonomical correspondences where we map classes and properties to their sub classes and sub properties, we could also use OWL restrictions to map a defined class with an inferred class. For example:

```
<owl:Class rdf:about="&swat;IDS">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="&txcorp;Aircraft"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="txcorp;hasApplication"/>
      <owl:someValueFrom rdf:resource="txcorp;Military"/>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

While OWL allowed us to achieve an acceptable amount of integration, it should be noted that there were some concepts in the two ontologies that, although similar, were impossible to map using OWL. Consider the property "maximum thrust" of an engine. In the Lehigh group (`swat`) ontology it has been defined as a property of the class `Engine`. However, in the txcorp ontology, an `Engine` is related to its maximum thrust via the `TechnicalSpecifications` class (it models a technical specification document). In First Order Logic, the relationship can be specified as:

$$\begin{aligned} \texttt{txcorp} : \texttt{Engine}(X) \wedge \texttt{txcorp} : \texttt{hasEngineTechSpec}(X, S) \wedge \\ \texttt{txcorp} : \texttt{hasEngineMaxThrustTechSpec}(S, Z) \Leftrightarrow \\ \texttt{swat} : \texttt{Engine}(X) \wedge \texttt{swat} : \texttt{maxThrust}(X, S) \end{aligned}$$

However, this axiom cannot be expressed in OWL (and most description logics, for that matter) because it requires role composition. Note, this kind of mapping can be supported in certain OWL-compatible rule-based Semantic Web languages such as SWRL, and is likely to be supported in the Rule Interchange Format (RIF) currently being developed by a W3C working group.

3.5 Information Integration Query Test Cases

Here, we consider how information integration over two separately developed ontologies on the same domain of knowledge and data individuals that are distributedly located over multiple databases is handled by the ISENS system prototype. Specifically, we can compare SPARQL queries that can be executed only

on one of the separate ontologies to extract the relevant answers from each of them to queries executed by the ISENS system to obtain the answers from both sources at the same time. We used such comparisons to test the data integration achieved by the ISENS system to retrieve data encoded with different ontologies from remotely located data servers.

Moreover, users can query (using specific syntax) distributed RDF/OWL metadata repositories that are encoded with different syntax (from each of the two different ontologies in this case). The ISENS system uses its map ontology in the process of integrating information from the different ontologies. We start with a "simple" mapping example to show this capability. Consider, querying ISENS to find the commercial aircraft it has information about.

The SPARQL query[6]:

```
SELECT ?Aircraft
WHERE { ?Aircraft rdf:type tx:CommercialAircraft .}
```

is in the Tech-X ontology syntax. The DQS uses its ontology maps to determine which data repositories contain relevant data to solve the query from both the Tech-X and the Lehigh University data individuals, thus achieving the desired information integration. The results from executing this query[7] contain the commercial aircraft from both ontologies.

The DQS does the needed logical reasoning to decide to which of the two Sesame RDF servers (and their specific PostgreSQL databases) to issue network calls in order to obtain relevant data for a query from all available RDF/OWL data repositories. The commercial aircraft encoded in the Tech-X ontology syntax are extracted from the Sesame RDF server at Tech-X that stores such data in its "Aircraft" data individuals PostgreSQL database. The commercial aircraft encoded in the Lehigh University group ontology are extracted from their Sesame RDF server and its "Airliner" PostgreSQL database.

Since the DQS has ontology maps to translate concepts between the two ontologies, we can also use the syntax from the Lehigh University group ontology to search for the same information with the following SPARQL query:

```
SELECT ?Aircraft
WHERE { ?Aircraft rdf:type ac:Airliner .}
```

[6] We present the example SPARQL queries here in a concise form. For each query, we have omitted the appropriate explicit prefix definitions from the set:
`PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>`
`PREFIX ac: <http://swat.cse.lehigh.edu/onto/airdata.owl#>`
`PREFIX tx:`
`<http://fusion.txcorp.com/~dad/isens/onto/txcorp/aircraft.owl#>.`

[7] We have set up a test version of the ISENS system prototype that is available online. We provide here URLs to PHP scirpts that can directly execute some of the queries here and display the results ISENS found together with the submitted query (to enable further interaction with the system if necessary). The URL for the above query is:
`http://fusion.txcorp.com/~dad/isens/wic/run-dqs-q6.php`

Running this query[8] returns the same results (only in different order) as the ones from the query that used the Tech-X group ontology syntax. These two queries demonstrate that one can publish the ontologies separately from their RDF/OWL data individuals.

The two SPARQL queries above demonstrated the ability of the ISENS system prototype to handle information integration over an OWL object type. In the next example, we show that ISENS provides integration over RDF/OWL object predicates (properties) as well. If one executes[9] the SPARQL query:

```
SELECT ?Aircraft ?Manufacturer
WHERE {
  ?Aircraft rdf:type ac:Airliner .
  ?Aircraft ac:manufacturedBy ?Manufacturer .
}
```

using the Lehigh group ontology syntax, then the data individuals that correspond to aircraft and their manufacturers are properly extracted from the data repository encoded with the Tech-X group ontology syntax (there are currently no such RDF/OWL data individuals in the Lehigh University group ontology). Again, the DQS reasons using its map ontologies to do the data integration/translation and decides from which RDF/OWL data repositories to obtain the relevant answers. For this case, ISENS supports data integration over the `tx:isMadeBy` predicate.

The next SPARQL query:

```
SELECT ?sellOrder ?soldProduct
WHERE {
  ?sellOrder rdf:type ac:Sell .
  ?sellOrder ac:soldProduct ?soldProduct .
  ?sellOrder ac:soldTo
    <http://.../isens/onto/txcorp/aircraft.owl#DeltaAirlines>.
}
```

also uses the Lehigh University group ontology syntax to search for all sell orders to the `DeltaAirlines` data individual. Notice that this individual is in the Tech-X ontology namespace. Only one of the Tech-X RDF/OWL repositories has relevant data for this query. The ISENS prototype again relies on the information integration algorithm in the DQS to find[10] the proper answer.

We have also used KAON2's reasoning capability over numerical datatypes (e.g. ordering) to implement filtering in our queries. When a query has a filtering request, we remove the filtering part of the query and decompose the rest. After we generate the reformulation and load the KAON2 knowledge base, we reinsert the filtering part and issue the whole query. In the future, we plan to implement

[8] `http://fusion.txcorp.com/~dad/isens/wic/run-dqs-q6-reversed.php`
[9] `http://fusion.txcorp.com/~dad/isens/wic/run-dqs-q5.php`
[10] `http://fusion.txcorp.com/~dad/isens/wic/run-dqs-q1-4m.php`

a variation of MiniCon that supports comparison predicates, which should allow the system to scale to larger datasets.

The final query we consider here:

```
SELECT ?sellOrderID ?sellPriceInUSDollars
WHERE {
  ?sellOrderID rdf:type ac:Sell .
  ?sellOrderID ac:sellAmount ?sellPriceInUSDollars .
  FILTER (?sellPriceInUSDollars > 10 &&
    ?sellPriceInUSDollars < 40)
}
```

shows support for filters over XMLS integer type data properties together with information integration in ISENS. Running[11] this query shows the data individuals that match the imposed constraints. There are, however, current limitations with the support for XMLS data types that we discuss in Section 4.

All of the above queries completed in under two seconds, including network latency. Although this was only a prototype and extensive evaluation is yet to be conducted, we consider these results to be promising.

4 Summary and Future Development

We developed a prototype application, ISENS, based on Semantic Web (including RDF/OWL/SPARQL/SWRL) technologies. The system demonstrates how information integration can be accomplished over two remotely located RDF/OWL data repositories that are encoded with syntax from different ontologies. This was implemented in the DQS component of the ISENS system. It is important to note that for its reasoning tasks the DQS uses the KAON2 logical reasoner [9]. We used the LUBM [11] to evaluate three candidate reasoners (Sesame, OWLIM and KAON2) for use in the DQS. Of these, KAON2 is the only reasoner that is complete. Moreover, KAON2's ability to handle both description logic and rules was one of the key factors that led us to select it. In addition KAON2 is light weight and is available for free if used for research purposes.

The current limitations of the ISENS system prototype are:

1. KAON2 is unable to handle XML schema date data type correctly. As a consequence, the DQS fails to use FILTER queries on the date data type. We have been informed by KAON2's developers that a future release will be able to handle such cases.
2. The system relies on a compilation of the map ontologies and the source descriptions into an intermediate form. In order, to get the prototype working with limited resources, this compilation was performed manually. A full system will need an algorithm to perform this translation automatically.

[11] `http://fusion.txcorp.com/~dad/isens/wic/run-dqs-q2.php`

3. The prototype is designed to work with a two-ontology architecture. However, in the general setting there may be many ontologies, and sometimes multiple maps will need to be composed in order to achieve complete integration.

In future development, we will generalize the DQS to address the current two-ontology architecture limitation. We will use the schema mediation in peer data management systems (PDMS) idea by Halevy *et al.* [10] to design and implement such a system. This is a natural extension to our system because a key component of the PDMS rewriting algorithm is the MiniCon algorithm. However, the PDMS algorithms will have to be extended to be used in a Semantic Web context.

Our experience in implementing the ISENS system prototype and the results from its testing show that this is a feasible approach to build a system for information integration, managing, and sharing of RDF/OWL data that could provide important, new, Semantic Web capabilities.

5 Acknowledgements

We are grateful to the Department of Energy for supporting this work under a Phase I SBIR grant.

References

1. Halevy, A.Y., Ashish, N., Bitton, D., Carey, M., Draper, D., Pollock, J., Rosenthal, A., Sikka, V.: Enterprise information integration: successes, challenges and controversies. In: SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, New York, NY, USA, ACM Press (2005) 778–787
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Sci. Am. (2001)
3. Heflin, J.: Towards the Semantic Web: Knowledge Representation in a Dynamic, Distributed Environment. PhD thesis, University of Maryland (2001)
4. Ullman, J.D.: Information integration using logical views. In: Proc. of ICDT, Delphi, Greece (1997) 19–40
5. SPARQL: Query Language for RDF, W3C Working Draft. ("`http://www.w3.org/TR/rdf-sparql-query/`")
6. Pottinger, R., Halevy, A.: MiniCon: A scalable algorithm for answering queries using views. VLDB Journal: Very Large Data Bases **10**(2–3) (2001) 182–198
7. Broekstra, J., Kampman, A.: Sesame: A generic architecture for storing and querying rdf and rdf schema. In: Proceedings of the 2nd International Semantic Web Conference (IAWC2003), Sanibel Island, Florida. (2002)
8. Aduna: Sesame RDF Framework. ("`http://www.openrdf.org`")
9. Motik, B.: KAON2: infrastructure for managing OWL-DL and SWRL ontologies. ("`http://kaon2.semanticweb.org/`")
10. Halevy, A., Ives, Z., Suciu, D., Tatarinov, I.: Schema mediation in peer data management systems. In: Proc. of ICDE. (2003)
11. Guo, Y., Pan, Z., Heflin, J.: An evaluation of knowledge base systems for large owl datasets. In: Proceedings of the 3rd International Semantic Web Conference (IAWC2004), Hiroshima, Japan. (2004) 274–288