Asian Journal of Information Technology 9 (3): 183-195, 2010
ISSN: 1682-3915
© Medwell Journals, 2010

Ontology-Based Query in Heterogeneous and Distributed Data Sources

Najood Al-Ghamdi, Mostafa Saleh and Fathy Eassa
Faculty of Computing and Information Technology,
King Abdulaziz University, Jeddah, Saudi Arabia

Abstract: Information sharing, exchanging and retrieving from heterogeneous data sources not only needs complete data accessibility but also it needs solving data heterogeneity between these data sources. To solve this rising problems of heterogeneity, a lot of recent study has been done towards solving this issue. This research aims to develop a software system based on ontology to semantically integrate heterogeneous data sources such as XML and RDF to solve some conflicts that occur in these sources. An agent framework is developed based on ontology to retrieve data from distributed heterogeneous data sources. With this technique, user will be able to send a mobile agent with classical input query to a data source. Then the mobile agent will carry the needed module of the global ontology prepared by the user stationary agents and transport itself from the user site to a remote XML or RDF data source. At remote XML data source, stationary agents will transform the heterogeneous XML source into temporary local RDF ontology. The stationary agents in all sources perform a mapping between the local and global ontologies, convert the query to XML or RDF query, execute it and set the results in a suitable form. Finally, the mobile agent will return back with the results. A partial implementation of this framework has been carried out using some modules and libraries of Java, Aglet, Jena and AltovaXML.

Key words: Ontology, ontology generation, schema mapping, multi-agent system, heterogeneous data sources, mobile agent

INTRODUCTION

The Web contains abundant repositories of information that make selecting just the needed information for an application a great challenge since computer applications understand only the Web pages structure and layout and have no access to their intended meaning. To enable users get information from the Web by querying these heterogeneous data sources, the new trend is by using the Semantic Web technologies (Saleh, 2008). The Semantic Web aims to enhance the existing Web with a layer of machine interpretable metadata. The American Heritage Dictionary defines semantics as the meaning or the interpretation of a word, sentence or other language form.

The emergence of the Semantic Web will simplify and improve knowledge reuse on the Web and will change the way people can access knowledge, agents will be a knowledge primary consumer.

By combining knowledge about their user and his needs with information collected from the Semantic Web, agents can perform tasks via Web services automatically. So agents can understand and reason about information and use it to meet user's needs. They can provide assistance using ontologies, axioms and languages such as DARPA Agent Markup Language which are cornerstones of the Semantic Web. The layers approach for Semantic Web is presented by Kumar *et al.* (2002) where it mainly includes eXtensible Markup Language (XML) and Resource Description Framework (RDF). XML allows users to add arbitrary structure to their documents but does not state anything about what the structures mean.

Meaning is expressed by RDF which encodes it in sets of triples, each triple being rather like the subject, verb and object of an elementary sentence. The third basic component of the Semantic Web consists of collections of information is the ontologies. Ontology is the backbone of the Semantic Web.

The Semantic Web's current focus is at the ontology and logic layers. However, study in the academic community on trust inference calculi across distributed information systems is ongoing and study on trust and the Semantic Web is beginning to appear (Warren, 2006).

The ontology consists of a specification of concepts to be used for expressing knowledge, including the types and classes of entities, attributes and properties, the relationships and functions and constraints. It provides a share and a common understanding of a domain as a theory not a data structured container. Actually, there are different kinds of ontologies from lightweight to heavyweight for different purposes. Formal ontology

Corresponding Author: Najood Al-Ghamdi, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

supports data standardization to support interoperability which requires explicating all different representations and interoperations of the data in a particular subject domain. Therefore, it is difficult to create due to complexity and enormous details but once created it is easy to deploy.

Now, ontologies are a trendy research topic in various areas such as information integration, knowledge engineering, cooperative information systems and natural language processing. In system integration, ontologies are playing an important role, mainly concerned with providing a set of mechanisms for resolving the semantic heterogeneity problems, resolving the queries, hiding the complexity of accessing data from different data sources and describing the contents of all data source as concepts in a global ontology.

In queries resolving, the query is defined in terms of global ontology and a set of its pre-defined concepts such as relations and operations. At a data source, answering a query involves rewriting the query in terms of a local ontology and related concepts or new concepts must be added to the global concepts by merging mechanisms to offer a standard integrated structure.

To take advantage of standards based integration, many companies and institutes have been moving to XML but XML doesn't hold the meaning or the semantics of the data. A need to standardize business vocabularies of the increasing number of standard XML dialects, illustrate the need for a semantic transformation.

Semantics is possibly the most important vision in driving the Web to its next phase. Challenges in developing semantic techniques are applied in many fields such as AI, database, information system, data modeling, query and transaction processing, knowledge representation, etc. Those techniques propose new approaches for data integration. Semantic integration is considered to be the best framework to deal with the heterogeneity, enormous scale and dynamic resources on the Web This study presents a semantic framework to provide a query interface to users with a flexible accessing to remote heterogeneous data sources. An agent platform was selected as a framework for building an ontology-based search engine based on some available Semantic Web technologies. Using a global ontology attached with common vocabulary dictionary to solve the semantic heterogeneity, hide the complexity of retrieving information of heterogeneous data sources from the user and make the computing environment very flexible.

BACKGROUND AND LITERATURE REVIEW

One of the difficulties in the Web information integration applications is the heterogeneity of the distributed data sources. These heterogeneities can be classified as syntactic, schematic and semantic.

Syntactic: Logical data models and their representations (relational, object-oriented) in the underlying DBMS (Sotnykova, 2001). For instance, XML and RDF provide two completely different paradigms for representing Web data. There are currently many attempts to use a conceptual-level schema (ontology) or a conceptual-level query language to integrate heterogeneous data sources (Xiao *et al*., 2004).

Schematic: Schematically the same real world phenomena can be represented with different abstractions in heterogeneous data sources. Within this heterogeneity range, different representations for equivalent data include:

- Entity vs. relation
- Entity vs. attribute
- Relation vs. attribute representations (Sotnykova, 2001)

Semantic: Semantics in linguistic is defined as the relationship between words and the things to that these words refer. Computer model's semantics is defined as the relationship among the computer representations and the corresponding real-world features within a certain context (Sotnykova, 2001). The semantic heterogeneity can by found in different ways, such the semantic problem with naming when two terms represent same concept or one term represents two different concepts. Another problem is cognitive heterogeneity when a fact or a real word object serves different purposes or have different views.

Ontology: An ontology is a conceptualization of an application domain in a human-understandable and machine-readable form and typically comprises the classes of entities, relations between entities and the axioms which apply to the entities which exist in that domain (Gibbins *et al*., 2003). A survey of Web tools (Laender *et al*., 2002) presented that extraction ontologies provide resiliently and scalability natively where in other approaches for information extraction, the problem of resiliently and scalability still remains.

One serious difficulty is creating the ontology manually is the need for a lot of time, effort and might contains errors. Also, requires a high degree of knowledge in both database theory and Perl regular-expression syntax. Professional groups are building metadata vocabularies or the ontologies. Large hand-built ontologies exist for example medical and geographic terminology. Researchers are rapidly working to built systems to automate extracting them from huge volumes of text.

The US Defense Advanced Research Projects Agency (DARPA) released a draft language known as the DARPA Agent Markup Language (DAML).

The DAML language is an extension to XML and the Resource Description Framework (RDF). The language provides a rich set of constructs with which to create ontologies and to markup information so that it is machine readable and understandable. It leverages and extends the express-ability of RDF and RDF-Schema (RDFS) (Gibbins *et al*., 2003). RDFS is a simple ontology language written in RDF that allows the creation of vocabularies with classes, properties and subclass/super class hierarchies (Heß and Kushmerick, 2003). DAML + OIL (DARPA Agent Markup Language + Ontology Inference Layer) is an extension of RDFS that allows finer-grained control of classes and properties with features such as cardinality constraints and inverses of properties (Sotnykova, 2001).

Agent: An Agent is a computer program that acts autonomously on behalf of a person or organization. Agents come in several different types, usually depending on the nature of their environment. There are many ways to classify existing software agents e.g., by their mobility, by several attributes they have such as autonomy, learning and cooperation, by their roles or hybrid agents.

Agents on the semantic web: Web services is one of the main forces driving the internet from its infant, a vast collection of text and images, to today's huge growing marketplace of service providers and consumers. Agent technology plays an important role in this evolution. One of the fundamental problems in building open Web based applications is how to find information providers and how to integrate information agents in such a dynamic environment. There is an obvious need for a standardized meaningful communication among agents to enable them to know each other and perform collaborative task execution. One approach relies on ontology based language for agent services description. This language exploits ontology of service domain and provides the flexibility for developers to plug in a suitable language to describe the constraints (Nyunt and Thein, 2005). Ontology is a specification of the agent knowledge about the world.

It specifies an agent's knowledge of the world including the agent's knowledge about its domain of existence and communication ability (both protocol the content). Communication is basis of knowledge sharing, without which we would be unaware of the abilities, knowledge and use of the agents around us. Intelligent agents operating in isolation are of little use in the world as they are unable to share or acquire knowledge. Agents need to interact with other applications, human, agent or information source (e.g., database). For agents to share knowledge with one another they must communicate. Previously, several agent communication languages (KQML, FIPA ACL to name a few) have required a predetermined agreed upon ontology prior to commencing communication. Static ontologies result in the two agents that speak the same languages being unable to communicate due to the fact that their lexicons do not overlap or they are not equivalent. The use of such static ontologies results in communication between only homogeneous agents. By allowing agents to merge and update their ontologies, the agents become able to interoperate in a heterogeneous multi agent system (March, 2004). This research aims to enable the user to query data resides in heterogeneous data sources in a specific domain as if it is in a single schema. The proposed system can be viewed as a merged solution of the two previous approaches that will have a common global ontology as a federated metadata model covering all concepts in the databases with different schemas. However, there is no federated database; so databases will remain without any change. Beside there will be a query translation to be particularly assigned for the data resource. Amann *et al*. (2002) proposed ontology mediator architecture for the querying and integration of XML data sources.

Cruz *et al*. (2004) proposed mediator to providing data interoperability among different databases. Also, Philipi and Kohler (2004) introduced architecture for ontology-driven data integration based on XML technology. Others presented some solutions to enhance the metadata representation as by Hunter and Lagoze (2001) by combining RDF and XML schemas and by using metadata dictionary as for solving some semantic heterogeneity.

In solving some problems in the query processing, Yan and MacGregor (2003) presented a query translation approach, Corby *et al*. (2006) addressed the problem of a dedicated ontology-based query language, Saleh (2008) presented a semantic framework that addresses the query mapping approach. In Mena *et al*. (2000) OBSERVER is an approach for query processing in global information system. Wang and Shakshuki (2005) presented a SDMS system which utilizes software agent and Semantic Web technologies; they addressed the problem of improving the efficiency of information management across weak data. A data warehousing approach with ontology based query facility presented by Munir *et al*. (2007).

ARCHITECTURE

System scenario: The architecture of the proposed framework is shown in Fig. 1. The System senario is going as follows:

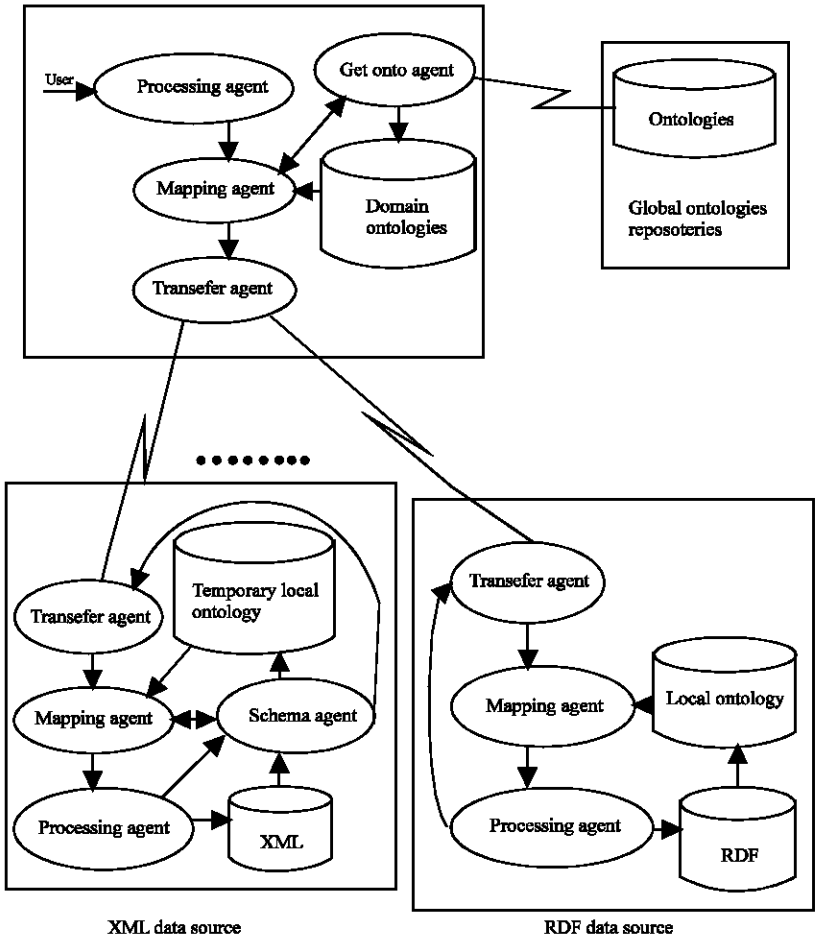

Fig. 1: The proposed system architecture

Step 1: The user enters a query by choosing suitable terms and a Processing agent constructs the query using a classic query language.

Step 2: Processing agent passes the user query to Mapping agent which connects to chosen standard domain ontology and maps the query metadata to the domain ontology metadata.

Note: if the chosen ontology cannot be found, the Mapping agent invokes GetOnto agent which brings and updates ontologies from the global ontologies repositories. In this way, the system keeps up with ontologies extensibility.

Step 3: Mapping agent then passes the query with a module holds the necessary mapping and merging instances and tables of mapping and common vocabulary of the metadata of the query to a mobile agent called Transfer Agent. Then, the Transfer agent transfers to the appropriate data sources.

Step 4: On RDF data sources:

- The transfer agent passes the query and the global ontology module to a stationary Mapping agent
- The mapping agent maps the local ontology to the global ontology module, performs the merging mechanism to construct the necessary instance if required and then it maps the query metadata to the mapped metadata
- The mapping agent passes the query to a Processing agent who transforms the query to the local RDF query language and retrieves data corresponding to the query over the RDF sources
- The Processing agent passes the results to the Transfer agent and then it returns back to the user site

Step 5: On XML data sources:

- The transfer agent passes the query and global ontology module to a stationary Mapping agent
- The mapping agent invokes a Schema agent to transform the data source to be interpreted as RDF data

- The schema agent first provides structural prescriptions for XML documents by derive XML Schema then constructing RDFS from it as a temporary local ontology
- The schema agent invokes the Mapping agent which merges the temporary local ontology to the global ontology module; it maps the temporary local ontology to the global ontology module, as well as the query metadata to the temporary ontology mapped metadata
- The mapping agent passes the query to a Processing agent which translates the query to local XML query language. After that the agent retrieves data corresponding to the translated query over the XML sources
- The processing agent invokes the Schema agent for result transformation from XML data to RDF data
- The schema agent passes the results to the Transfer agent and it returns back

System data components: The system is equipped with basic data components the ontologies and the data sources. The global ontology provides the definitions of the terms and all concepts of the domain ontology and it is updated by linking to global ontologies repositories which contain developed and integrated domain-specific ontologies, each one describing a content of all data sources on its domain. This updating of existed ontology or retrieving new domain ontology is done by agent technology.

The idea of establishing global ontologies repositories is to achieve the benefits of standardization references while at the same time meeting the different requirements of the users. As a result, the global ontology solves the semantic problem of cognitive and by associating each global ontology with a dictionary of common vocabularies of its domain which can be used within local ontologies and to solve the semantic problem with naming.

A local ontology is a set of terms in a particular information domain. This ontology is expressed in a semantic metadata to describe the local data. There is an important point here that needs a clarification, that is how does the local ontology relate to the global ontology. Well, there are some possibilities:

- The local ontology created from the pre-existent global ontology, so the local ontology seen as a subset of the global ontology and the mapping here will be direct. This implies that the global ontology must be formal and as what said before, this will be difficult but easy to deploy
- The local ontology created from scratch or from different sources. In these conditions, the global ontology must have a merging mechanism to construct and add new instances from the different local ontologies sources before starting mapping. The merging should be done one time only unless the local ontology has been updated which must then provide an updating indicator by its agents to inform the global ontology to remerge

A local XML schema that describes the local data is transferred into a temporary local ontology and it merged to the global ontology. The merging should be done one time only unless the XML source has been updated which must then provide an updating indicator by its agents to inform the global ontology to remerge.

On the other hand, a data source is a facility for storing data. It can be as complicated as a complex database for a large corporation or as simple as a file with rows and columns.

Several providers provide RDF data sources interfaces to create, read and modify data sources. A local RDF data source is a single RDF document or a relational database that has an RDF interface. As we mentioned earlier about the relation between local and global ontologies if the local ontology of the data source typifies the second possibility this would lead to a semantic heterogeneous RDF data sources.

Most web applications are connected to XML database. A local XML data source is a single XML document or a relational database that has an XML interface. Usually, different data sources are heterogeneous and can have more than one of heterogeneity classes.

At user site, a Mapping agent contains mechanism to map the user data input to the global ontology to determine the corresponding global metadata. At remote site, a stationary Mapping agent should provides mechanisms for mapping metadata and mapping concepts, knowledge and relations between the permanent or temporary local ontology and the global ontology module and passes the outcome to a Processing agent.

Query rewriting: The proposed system should provide a query rewriting algorithm. The integration design is rely on a local view for query translation. This known as source-centric approach, also known as Local as View (AV). Algorithms for query rewriting are presented in (Convey *et al*., 2001) that are the Bucket Algorithm, MiniCon Algorithm, Shared-Variable-Bucket Algorithm and the CoreCover Algorithm.

At RDF site, a stationary Processing agent will get the mapped needed data from the Mapping agent and transform the classic query to RDF query. Then, the agent will connect to the RDF local source and execute the

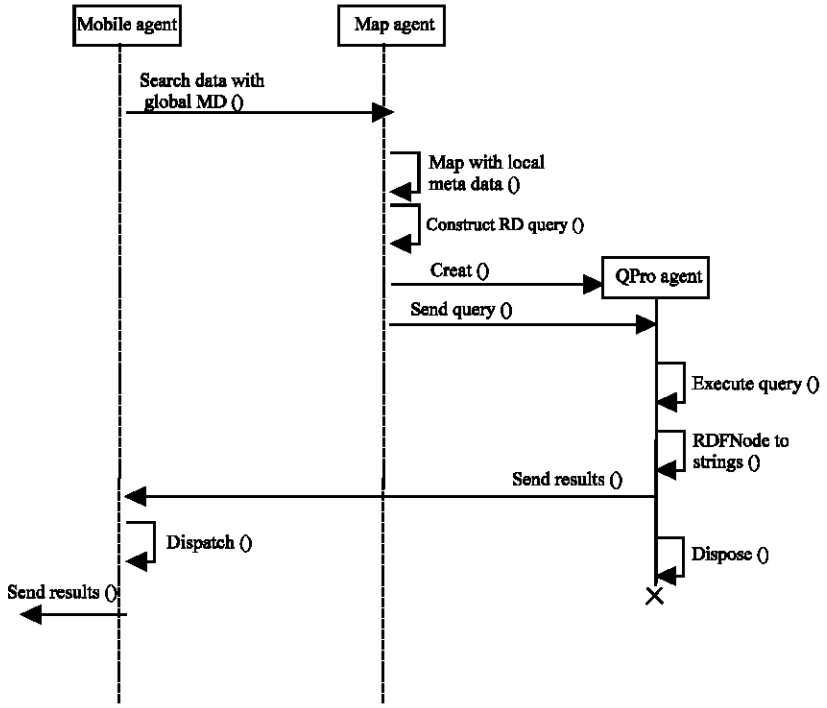

Fig. 2: Sequence diagram for RDF data source site

query to get the final result. At another side when the mobile agent is sent to XML site, a stationary Processing agent get the mapped needed data from the Mapping agent and transform the classic query to XML query. Then, the agent will connect to the XML local source and execute the query to get XML result such as DOM tree. This result will be send to a stationary Schema agent to get the final result.

To make RDF data and XML data work together, a Schema agent will transform the data between the two schema languages directly. At XML site, a stationary Schema agent will take the data of XML data source and transform it to temporary local ontology, so the Mapping agent can connect it with the global ontology module.

In addition, when Processing agent gets XML query result it will pass it to the Schema agent to transform the result to an appropriate RDF result.

Sequence diagram: The design is based in the view of agent technology and current web standards. It provides a simple mechanism for the use of ontology in querying over the web. As mentioned in the system scenario, the following sequence diagrams (Fig. 2 and 3) were designed for each node or site in the system (Fig. 1).

SYSTEM IMPLEMENTATION ENVIRONMENT

A prototype has been implemented using Java. Sun Microsystems provide Java Development Kits (JSDK) for many platforms, a standard edition and enterprise edition (Microsystems, 2010). Concerning RDF there are many RDF libraries now, but the most complete library is probably Jena from HP Labs. Jena was developed to provide an API that was designed specifically for the Java programming. Jena provides support for serialization, relational database storage and querying. Also it provides an ontology API and comes with rule engines for basic inference on RDF Schemas. It also has limited support for DAML+OIL and OWL. We used Jena to provide the RDF data sources and querying. Additionally, RDQL, RDF Data Query Language, first released in Jena but currently SPARQL is supported in Jena via a module called ARQ. It is a powerful language and simpler than the earlier RDF query languages which make it the best choice.

AltovaXML is a free XML application package that can be used to validate XML documents, transform XML documents using XSLT style sheets and execute XQuery documents. Also, it can be used from the command line via a COM interface in Java programs and in NET applications. The Altova XQuery 1.0 Engine conforms to the W3C. This Engine Executes an XQuery document using well-formed XML documents. However, they do not need to be valid according to an XML Schema since the invalid file is loaded without schema information. If the XML file is associated with an external schema and is valid according to it then post-schema validation information is generated for the XML data and will be used for query evaluation. Numerous agent platforms

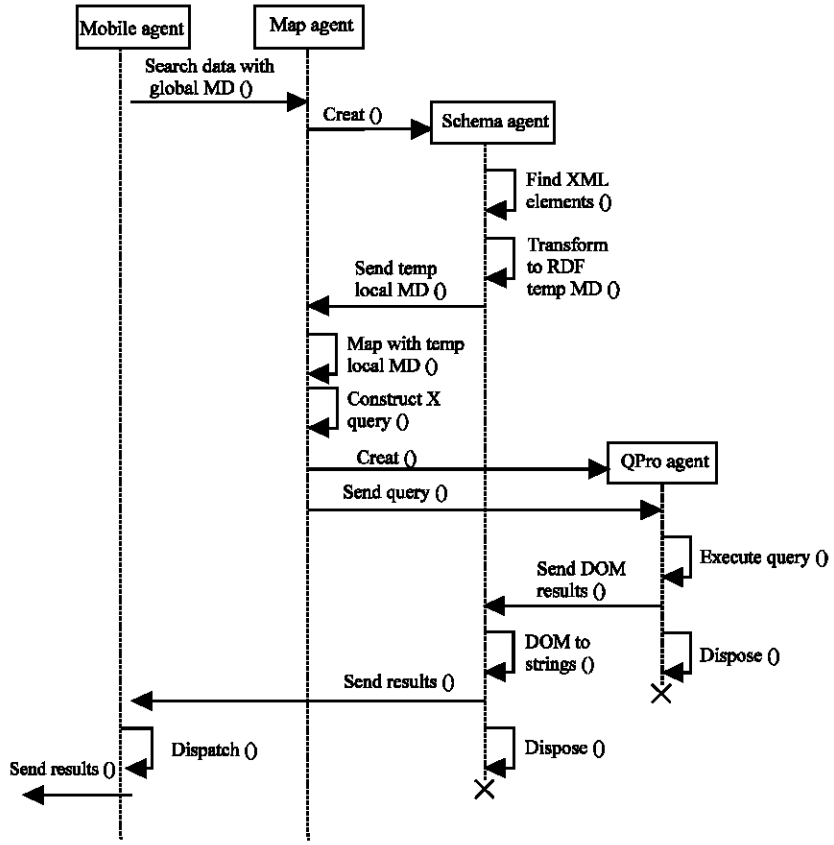

Fig. 3: Sequence diagram for XML data source site

have been implemented or are currently under development. One of them that implement many important features of agent is Aglet. Aglet originally developed at IBM Tokyo Research Laboratory and currently the project is hosted at source forge (Luca Ferrari, 2004). Aglets are a mobile-agent technology built on top of Java. They are similar to applets in that they run as threads inside the context of a host Java application. Aglets need a host Java application to be running on or they want to immigrate.

This host application is called an aglet host. The aglet host installs a security manager to implement restrictions on any istrusted aglets. Several versions of Aglet have been released and we obtained the latest version which is Aglet 2.0.2 from the web site.

Usage: Admittedly, the implementation is a prototype that lacks some fundamental features. As mentioned in limitations section it does not provide a query transformation and nesting XML data sources. But it can be considered as a source for further implementations.

Actions take place sequentially agents are carrying out their designated tasks explained previously in the system scenario. All the framework communications and operations are done in view of Aglet agents.

The heterogeneous data source is annolated in XML and RDF and stored them in files. Transforming XML data into RDF data directly does not consider the XML structure and need RDFS specifications. The transformation is made at element-level on the data source on DTD files where the complex-type element will be mapped as RDF class, simple-type elements and attributes will be mapped as RDF properties for the class. The mapped data is the metadata of the temporary ontology.

Since the purpose of using ontologies in this framework is to define the metadata, modeled them as a metadata dictionary documents and therefore, there is no need to build real ontologies. In mapping metadata, followed a simple mechanism through sequentially searching of the dictionary is followed.

Query processing is done at the data sources sites. At RDF site, after mapping the local metadata to the global metadata, an agent executes an interaction with Jena and constructs the query using predefined SPARQL query templates. Then, the agent with Jena executes the query on the RDF data. Similarly, at XML site, after mapping the temporary local metadata to the global metadata, an agent executes an interaction with Altova XQuery engine and constructs the query by using predefined XQuery query templates. Then the agent with Altova executes the query,

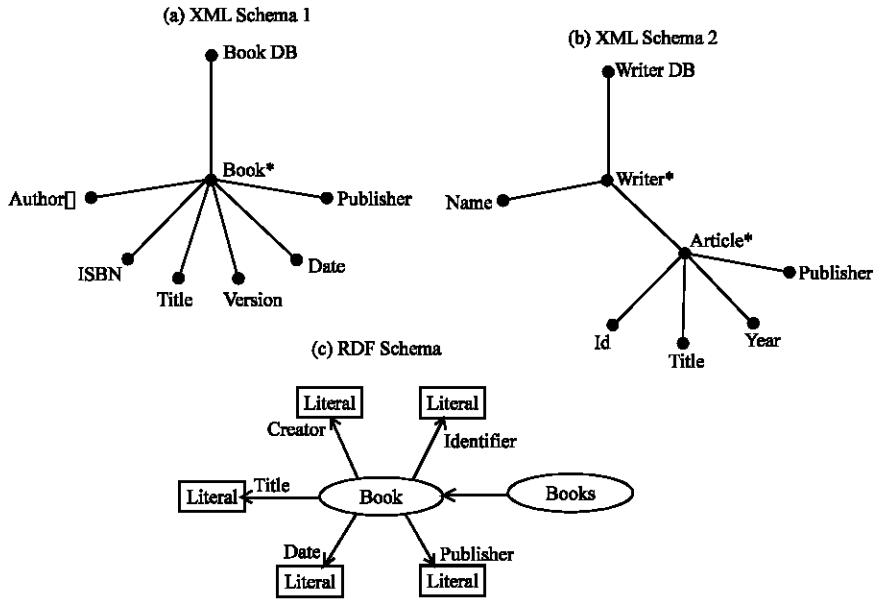

Fig. 4: Remote data sources schemas

on the XML data. It was not achievable to develop a real application using different modules that most of them were designed for academic purposes. we faced some problems in using those modules, particulary when the RDF Query Agent connects to Jena module to execute a query the Jena module results appear on the form of RDF data which called RDFNode. RDFNode is an interface on data type representation for RDF triples. However, RDFNode is undefined for Aglet agent. The Aglet agent is unable to send message and unable to dispatch with undefined data. But the Jena RDFNode can be converted to different implementations and that what we actually did. the results to string data typeis converted that can be carried by the agent. Therefore, the need of converting XML query results (DOM) to RDF data results is useless within the system utilities. A Schema Agent pareses DOM to determine the data as strings. Also, the AltovaXML Java interface connects to the AltovaXML COM Server object; providing access to XML Validator and to the XQuery engine. When XML Query Agent connects to Altova COM Server object, then to the XQuery engine object to execute the query on the XML data, the engine object executes and gets the results. After the execution, the engine object connection to the COM Server should be released. COM Server object will shut down automatically and the problem that has emerged will required the Tahiti Aglet Server to be rebooted for the execution of another query.

System testing: The prototype developed in this study, enables the user to retrieve data by ontology-based querying on distributed heterogeneous data sources through the delegation of specific tasks to agents. Mobile agent migrates to remote site and interacts with the resource through the residing agents while ontologies provide supporting in querying and solving conflicts. Books-Publishing domain is used as a global ontology associated with a vocabulary dictionary consisting of common vocabularies to capture the metadata of data sources. The distributed heterogeneous data sources is supposed for books in XML and RDF format. Since the concern is about semantic heterogeneity, we used schemas shown in Fig. 4 as remote sites data sources.

The prototype allows the user to issue a question in a form such as, select writer, title, isbn. After accepting the user inputs, User agent invokes the Mapping agent to map the inputs data to the global ontology metadata. To simplify the development of the test application, two terms in will track selection query such as Author and date. The Processing agent prepares the global ontology module, creates a Transfer mobile agent and sends it with the module to a remote data source server. In the prototype framework the module consists of the global metadata to be searched associated with its common vocabularies as mentioned before if we track author and date the module is select author name, date, common vocabularies of author name and common vocabularies of date. The Transfer agent arrives and passes the module to the residing agents of the data source and waits for the results. If the remote server was the first site or the second site that mentioned in Fig. 4, a Mapping agent can't maps on XML Schema, it invokes Schema agent to start data transformation from XML Schema to RDF Schema. Data transformation is done at element level and it is shown in Fig. 4.

Now, the local ontology in an XML site is ready and the Mapping agent can maps the temporary local ontology to the global module. But if the remote server

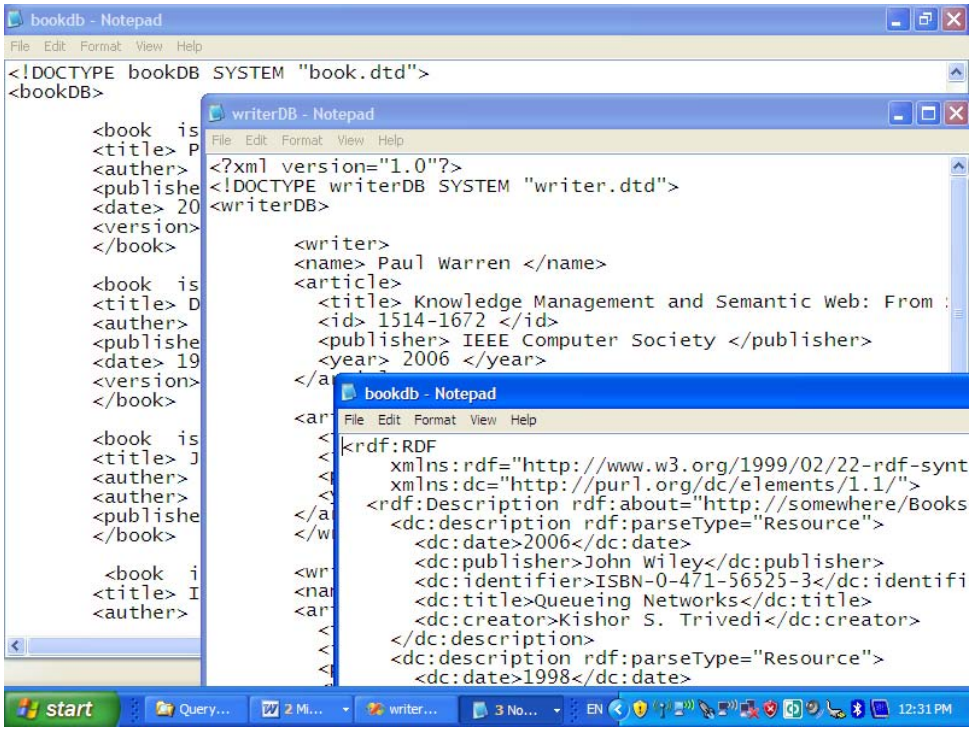

Fig. 5: Screen shot from the three data sources

was the third site, Mapping agent of the server maps the local ontology to the global module directly. The mapping between local ontology schema and global ontology is realized according to the common vocabularies. Table 1 shows a sample relationship between the global ontology and local ontologies and schemas. So, we can issue different queries as follows:

- SELECT author, date based on the first XMLSchema
- SELECT name, year based on the second XMLSchema
- SELECT creator, date based on the RDFSchema

Next, the Processing agent receives the result of mapping and constructs the suitable query, XQuery in the first and the second XML sites or SPARQL in the third RDF site. Then it executes the query and sends the results to the waiting Transfer agent directly in the third RDF site or through Schema agent in the first and the second XML sites, for parsing DOM result of XQuery to suitable form. The Transfer agent returns back to the user site and the User agent displays the result received.

Running example: Three distributed data sources are used; two contain heterogeneous XML data based on schemas in Fig. 4a, b and one RDF data based on schema in Fig. 4c. Figure 5 shows screen shot from the three data sources. When the user issues a query like, Select author, title, date for any site compatible with the first schema (Fig. 4a), the User Query Interface agent calls the Map agent to map the local schema into the global ontology metadata based on Table 1, like Table 2. The tasks are done. The input and the output of the Schema agent to construct temporary local ontology are shown in Fig. 6. When the local metadata is ready it mapped to the global metadata (Table 2). The constructed XQuery for the data is in Fig. 7. The result that obtained by the result

Table 1: Mapping keywords between global and local schemas and ontologies

Global ontology	Related mapping keywords
Auther_name	Writer, name, Auther, auther, creator
Book_title	Title, Tit, art_title
ISBN	ISBN, book_ISBN, identifier, id
Year	Date, year, Pub_date
Publisher	Pub, Publisher, publisher, Pub

Table 2: Ontology mapping for data source 1

Author	Author_name
User input to global metadata mapping	
Title	Book_title
Date	Year
Global to local metadata mapping	
Title	Book_title
Date	year

Table 3: Ontology mapping for data source 2

Author	Author_name
User input to global metadata mapping	
Title	Book title
Date	Year
Global to local metadata mapping	
Article_title	Book title
Article_date	Year Date

```
<!-- file: book.dtd -->
<!ELEMENT bookDB (book+)>
<!ELEMENT book (title,auther+,puplisher,date?,version?)>
<!ATTLIST book isbn ID #REQUIRED>
<!ELEMENT title (#PCDATA)>
<!ELEMENT auther (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT version (#PCDATA)>
```

```
rdf:class :  bookDB
rdf:class :  book
rdf:property:  isbn
rdf:property:  title
rdf:property:  auther
rdf:property:  publisher
rdf:property:  date
rdf:property:  version
```

Fig. 6: XML metadata to RDF metadata

```
<authlist>{for $a in fn:distinct-values(/bookDB/book/auther)
order by $a return <auther>{$a} <books> { for $b in /bookDB/book[auther = $a]
return <option>{$b/title}  {$b/date} </option> } </books></auther>} </authlist>
```

Fig. 7: XQuery for data source 1

RESULTS

creator	title	publisher
==========	==========	==========
Peter Galvin	Operating System Conccepts	Addison Wesley Longman
Abraham Siberchatz	Operating System Conccepts	Addison Wesley Longman
Thomas C. Bartee	Computer Architecture and Logic Design	McGraw-Hill
Ethel Tiersky	The Language Of Medicine In English	Prentice Hall
Martin Tiersky	The Language Of Medicine In English	Prentice Hall
Edward Norminton	Maple Computer Guide	John Wiley and Sons
Mortone Sternheim	Physics	John Wiley and Sons
Erwin Kreyszig	Maple Computer Guide	John Wiley and Sons
Roger Jennings	Database Developer's Guide With Visual Basic 6	Sams Puplishing
Tony L. Hansen	The C++ Answer Book	Addison-Wesley Puplishing Company
Joseph Kane	Physics	John Wiley and Sons

Fig. 8: Query results

```
<!-- file: writer.dtd -->
<!ELEMENT writerDB (writer+)>
<!ELEMENT writer (name,article+)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT article (title,id,publisher,year?)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT id (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT year (#PCDATA)>
```

```
rdf:class :  writerDB
rdf:class :  write
rdf:property:  name
rdf:class :  article
rdf:property:  title
rdf:property:  id
rdf:property:  publisher
rdf:property:  year
```

Fig. 9: XML metadata to RDF metadata

agent is shown in Fig. 8. Also For any site compatible with the 2nd schema (Fig. 4b), the User Query Interface agent calls the Map agent to map the local schema into the global ontology metadata based on Table 1, like Table 3. The tasks are done. The input and the output of the Schema agent to construct temporary local ontology are shown in Fig. 9. The constructed query is written in Fig. 10 and the result is shown in Fig. 11. Finally, for any site compatible with the third RDF schema (Fig. 4c), the User Query Interface agent calls the Map agent to map the local schema into the global ontology metadata based on Table 1, like Table 4. The tasks are done.

Table 4: Ontology mapping for data source 1

Author	Author_name
(a) User input to global metadata mapping	
Title	Book_title
Date	Year
(b) Global to local metadata mapping	
Title	Book_title
Date	Year

```
for $a in  /writerDB/writer/article/title, $b in /writerDB/writer/article/publisher
order by $a, $b
 return   if (fn:exists(/writerDB/writer/article[title eq $a  and publisher eq $b]))
 then <option>{$a}{$b}</option>  else ()
```

Fig. 10: Constructed query

Query Results

Author	Title	Date
Howard Anton	Elementary Linear Algebra	1987
Kishor S. Trivedi	Probability and Statistics with Reliability	2002
Paul Warren	Knowledge Management and Semantic Web: From Sce...	2006
Paul Warren	The Next Steps for The WWW	2003
Thomas B. Passin	Explorer's Guide To The Semantic Web	2004

Fig. 11: Query results

Author_name	Creator
Book_title	Title
Publisher	Publisher

Fig. 12: Global Metadata to Local Metadata

```
PREFIX  dc: <http://purl.org/dc/elements/1.1/>

SELECT  ?creator ?title ?publisher
WHERE
 { ?x  dc:creator    ?creator ;
       dc:title      ?title ;
       dc:publisher  ?publisher .
 }
```

Fig. 13: Constructed SPARQL

RESULTS

creator	title	publisher
Peter Galvin	Operating System Conccepts	Addison Wesley Longman
Abraham Siberchatz	Operating System Conccepts	Addison Wesley Longman
Thomas C. Bartee	Computer Architecture and Logic Design	McGraw-Hill
Ethel Tiersky	The Language Of Medicine In English	Prentice Hall
Martin Tiersky	The Language Of Medicine In English	Prentice Hall
Edward Norminton	Maple Computer Guide	John Wiley and Sons
Mortone Sternheim	Physics	John Wiley and Sons
Erwin Kreyszig	Maple Computer Guide	John Wiley and Sons
Roger Jennings	Database Developer's Guide With Visual Basic 6	Sams Puplishing
Tony L. Hansen	The C++ Answer Book	Addison-Wesley Puplishing Company
Joseph Kane	Physics	John Wiley and Sons

Fig. 14: Query result

The input and the output of the Schema agent to construct temporary local ontology are shown in Fig. 12. And when the local metadata is ready it mapped to the global metadata (Table 4). The constructed SPRQL query for the data is in Fig. 13 and the result is shown in Fig. 14.

DISCUSSION

The main focus in this study is to propose integrated access to heterogeneous and distributed data sources using ontology. The ontology is desirable not merely because it provides support in querying but also sharable and reusable representations, consistency and hiding complexity. Querying based on can overcomes semantic defects and lead to professional accurate searching.

In the running examples, the data sources are about books and writers. There are difference in terms exist between each data source. Therefore, if a user inputs a query to select author's name and book's title an agent without ontology will be dispatched with a query look like this: SELECT, title. The agents of a site can rewrite the query using XML query language in XML data source and RDF query language in RDF data source, but only a translated query can be answered.

In first data source shown in Fig. 4 and 5, (A) the terms of the metadata author and title exist but in the second site in Fig. 5, (A) the term author does not exist, instead the term name exists and also in the third site in Fig. 4, (A) the term author does not exist, instead the term creator exists. The agent will not be able to tell that the terms author, name and creator are for one metadata, i.e., it will not be able to distinguish the naming conflict. As a result, the query can only be answered in the first data source and the user will get incomplete information.

On the other hand an agent used in this approach with ontology will be dispatched with a query that contains metadata terms of the global ontology which will look like this: SELECT author-name, book-title in the supposed global ontology metadata example. This is done by mapping the user inputs to the global domain ontology metadata as shown in the Fig. 6, 9 and 12.

At remote sites, the global ontology metadata will be mapped to the local ontologies metadata using the common vocabularies associated with the global. The terms author, creator and name are associated to the term author name of the global ontology. The agents can get all the mapped local metadata, rewrite and translate query in all sites and the user will get all the available information at any chosen site which means that the naming conflicts are solved.

In fact, the ontology provides query support not only by defining common vocabularies to solve terms or naming conflicts but also by defining knowledge, concepts and taxonomy. This provides strong benefit for query processing and other ontology purposes. Although, this implementation is a prototype provides querying with SELECT clause it does not provide other querying clauses that require the existence of ontology concepts.

To illustrate, let's go back to the framework architecture shown in Fig. 1; as a complete framework, the ontologies should offer their concepts, knowledge and taxonomy. These features of ontologies can overcome all the rest of semantic defects. For example, Saleh (2008) showed how associating common vocabularies with the ontology concepts can solve not only the synonymy but also the polysemy naming problems e.g., associating the common vocabularies of the concept property write with the inverse concept written-by to solve such heterogeneity of local ontologies having different concepts: Book, written-by, author and author, write, Book.

CONCLUSION

The design of a system that can research on the distributed heterogeneous data sources is a difficult task because it involves managing many subjects such as data heterogeneity, query languages support and different access mechanisms. In this thesis I have proposed ontology-based framework using agents to query data on distributed heterogeneous sources.

Then, I represented the architecture components of the framework and the tasks that agents should handle. Also a prototype implementation has been developed through limited processes of user inputting, schema matching, metadata mapping, query construction and execution.

The user query can be correctly transmitted to XML or RDF data sources while the different access mechanisms within the distributed data are opaque to the user.

The ontology proved that it is the best choice to solve the semantic heterogeneity problems and hide the complexity of the distributing computing environment from the user in a flexible way. The well-know advantages of the agent technology have been usefully in this framework.

Under such architecture, adding more agents, developing and improving ontologies are possible without change the existing components. I hope that agent vendors such as Aglet vendor will add new features to their platforms such as RDFS module so the agent technology and the ontology can fit together. With this it will be available to implement the suggestion that mentioned in my proposed framework that the agent carries a subset of the global domain ontology.

RECOMMENDATION

For future directions, intend to improve this reserch to support nesting structure of XML data sources. Also the user choices will expand to input query with more clauses by developing a robust query rewriting scheme. Finally, We will improve the metadata dictionary by supporting it with an advanced ontology language such as RDFS, OWL.

LIMITATIONS

There is a number of important limitations which are due to the complexity and immensity of the framework. The first limitation is that there is no mechanism for query composition and decomposition in our developed system. It means that no support for query translation, the query to retrieve data is cited only. The user will type the data to be retrieved. Therefore, we only need to send the mapped global metadata and the query will be constructed by the Processing agent at the remote site using templates.

Second, we didn't concern the nesting structure of XML data source. So, the transformation from XML data to RDF data is done directly at element-level to define classes and properties from the XML elements and attributes. Consequently, all what we need from the ontologies is the metadata. So, we modeled the ontologies as metadata dictionary.

REFERENCES

Amann, B., C. Beeri, I. Fundulaki and M. Scholl, 2002. Querying XML Sources Using an Ontology-Based Mediator. Proceeding of On the Move to Meaningful Internet Systems, Confederated International Conference DOA, CoopIS and ODBASE 2002, October 30 November 01, Springer-Verlag London, UK, 429-448.

Convey, C., O. Karpenko, N. Tatbul and J. Yan, 2001. Data Integration Services. 1, Brown University, New York.

Corby, O., R.D. Kuntz and F. Gandon, 2006. Approximate query processing based on ontologies. IEEE Intelligent Systems, 21: 20-27.

Cruz, I.F., H. Xiao and F. Hsu, 2004. An ontology-based framework for xml semantic integration. Proceedings of the International Database Engineering and Applications Symposium, July 07-09, IEEE Computer Society,IEEE Computer Society, Washington, DC, USA, 217-226.

Gibbins, N., S. Harris and N. Shadbolt, 2003. Agent based semantic web services. Proceedings of the 12th International Conference on World Wide Web, May 20-24, ACM New York, NY USA, 710-717.

Heß, A. and N. Kushmerick, 2003. Learning to Attach Semantic Metadata to Web Services. In: The SemanticWeb-ISWC 2003, Fensel, D. *et al*. (Eds.). Springer, Berlin, Heidelberg, ISBN-13: 978-3-540-20362-9, pp: 258-273.

Hunter, J. and C. Lagoze, 2001. Combining RDF and XML Schemas to Enhance Interoperability Between Metadata Application Profiles. Proceedings of the 10th International Conference on World Wide Web May 01 - 05, ACM, New York, 457-466.

Munir, M. Odeh, R. McClatchey, S. Khan and I. Habib, 2007. Semantic information retrieval from distributed heterogeneous data sources. CCS Research Centre, University of West of England http://www.scribd.com/doc/19942537/Semantic-Information-Retrieval-From-Distributed-Heterogeneous-Data-Sources.

Kumar, S., A. Kunjithapatham, M. Sheshagiri, T. Finin, A. Joshi, Y. Peng and R.S. Cost, 2002. A personal agent application for the semantic web. Proceedings of AAAI Fall Symposium on Personalized Agents, November 2002. http://ebiquity.umbc.edu/paper/html/ id/109/A-Personal-Agent-Application-for-the-Semantic-Web.

Laender, A.H.F., B.A. Ribeiro-Neto, A.S. da Silva and J.S. Teixeira, 2002. A brief survey of web data extraction tools. ACM SIGMOD Record, 31: 84-93.

Luca Ferrari, 2004. The Aglets 2.0.2 User's Manual. October, 2004.

March, O., 2004. Resolving Agent Ontology Dynamically Through Communication. The University of Melboune, Australia, pp: 1-16.

Mena, E., V. Kashyap, A. Sheth and A. Illarramendi, 2000. OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. Int. J. Distributed Parallel Databases (DAPD), 8: 223-271.

Nyunt, P.P. and N.L. Thein, 2005. Software agent oriented information integration system in semantic web. Proceedings of the 6th Asia-Pacific Symposium on Information and Telecommunication Technologies APSITT, November 10, Computer Studies Univ., Yangon, pp: 266-271.

Philipi, S. and J. Kohler, 2004. Using XML technology for the ontology-based semantic integration of life science databases. IEEE Trans. Inform. Technol. Biomed., 8: 154-160.

Saleh, M.E., 2008. Semantic query in heterogeneous web data sources. Int. J. Comput. Their Appl., 15: 11-20.

Sotnykova, A., 2001. Design and implementation of federation of spatio-temporal databases: Methods and tools. Centre de Recherche Public-Henri Tudor and Laboratoire de Bases de Donnees Database Laboratory.

Wang, Y.A. and E. Shakshuki, 2005. An agent-based semantic web department content management system. Proceedings of the 6th International Conference on Information Technology Based Higher Education and Training, July 7-9, IEEE., F3C/1-F3C/6.

Warren, P., 2006. Knowledge management and the semantic web: From scenario to technology. IEEE Comp. Soc., pp: 53-59.

Xiao, H., I. Cruz and F. Hsu, 2004. Semantic mappings for the integration of XML and RDF sources. Proceeding of Workshop on Information Integration on the Web, August 2004, Toronto, Canada, 1-6.

Yan, B. and R. MacGregor, 2003. Translating naive user queries on the semantic web. Proceedings in Semantic Integration Workshop, ISWC 2003. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.5.4442&rep=rep1&type=pdf.