Towards a Semantic Web of Relational Databases: a Practical Semantic Toolkit and an In-Use Case from Traditional Chinese Medicine

Huajun Chen[1], Yimin Wang[2], Heng Wang[1], Yuxin Mao[1],
Jinmin Tang[1], Cunyin Zhou[1], Ainin Yin[3], and Zhaohui Wu[1]

[1] College of Computer Science, Zhejiang University, Hangzhou, 310027, China
{huajunsir,paulwang,maoyx,jmtang981,02rjgczcy,wzh}@zju.edu.cn
[2] Institute AIFB, University of Karlsruhe, D-76128, Germany
ywa@aifb.uni-karlsruhe.de
[3] China Academy of Traditional Chinese Medicine, Beijing, 100700, China
yinan@mail.cintcm.ac.cn
http://ccnt.zju.edu.cn/projects/dartgrid

Abstract. Integrating relational databases is recently acknowledged as an important vision of the Semantic Web research, however there are not many well-implemented tools and not many applications that are in large-scale real use either. This paper introduces the Dartgrid which is an application development framework together with a set of semantic tools to facilitate the integration of heterogenous relational databases using semantic web technologies. For examples, DartMapping is a visualized mapping tool to help DBA in defining semantic mappings from heterogeneous relational schemas to ontologies. DartQuery is an ontology-based query interface helping user to construct semantic queries, and capable of rewriting SPARQL semantic queries to a set of SQL queries. DartSearch is an ontology-based search engine enabling user to make full-text search over all databases and to navigate across the search results semantically. It is also enriched with a concept ranking mechanism to enable user to find more accurate and reliable results. This toolkit has been used to develop an currently in-use application for China Academy of Traditional Chinese Medicine (CATCM). In this application, over 70 legacy relational databases are semantically interconnected by an ontology with over 70 classes and 800 properties, providing integrated semantic-enriched query, search and navigation services to TCM communities.

1 Introduction

Up to date, many killer applications reported by the Semantic Web community often focus on processing the unstructured document data, using semantic annotation or various of learning, mining, and natural language processing techniques [1]. However, data in big organizations is normally stored in relational databases or other appropriately formatted documents. Over emphasizing on those applications, which handles unstructured document, may obscure the community from the fact that the essence of the Semantic Web comes from its similarity to a huge distributed database. To back up this

idea, consider the following statements made by Tim Berners-Lee in 2005 about his vision of the future Semantic Web[4].

> ...The Semantic Web is not about the meaning of documents. It's not about marking up existing HTML documents to let a computer understand what they say. It's not about the artificial intelligence areas of machine learning or natural language understanding... **It is about the data which currently is in relational databases, XML documents, spreadsheets, and proprietary format data files, and all of which would be useful to have access to as one huge database**...

From this point of view, one of the way to realize the vision of Semantic Web is (i) to interconnect distributed located legacy databases using richer semantics, (ii) to provide ontology-based query, search and navigation as one huge distributed database, and (iii) to add additional deductive capabilities on the top to increase the usability and reusability of data.

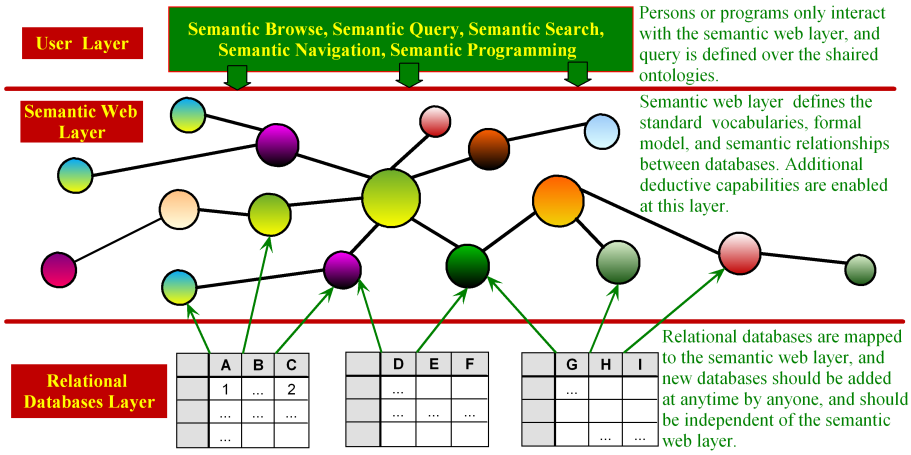

Fig. 1. Towards a semantic web of relational databases

Besides, since most of the data is currently stored in relational databases, for semantic web to be really useful and successful, great efforts are required to offer methods and tools to support integration of heterogeneous relational databases. Dartgrid is an application development framework together with a set of practical semantic tools to facilitate the integration of heterogenous relational databases using semantic web technologies. In specific, DartMapping is a visualized mapping tool to help DBA in defining semantic mappings from heterogeneous relational schemas to ontologies. DartQuery is an ontology-based query interface helping user to construct semantic queries, and capable of rewriting SPARQL semantic queries to a set of SQL queries. DartSearch is an

[4] `http://www.consortiuminfo.org/bulletins/semanticweb.php`

ontology-based search engine enabling user to make full-text search over all databases and to navigate across the search results semantically. It is also enriched with a concept ranking mechanism to enable user to find more accurate and reliable results.

Building upon Dartgrid, we have developed and deployed a semantic web application for China Academy of Traditional Chinese Medicine (CATCM)[5][6]. It semantically interconnects over 70 legacy TCM databases by a formal TCM ontology with over 70 classes and 800 properties. The TCM ontology acts as a separate semantic layer to fill up the gaps among legacy databases with heterogeneous structures, which might be semantically interconnected. Users and machines only need to interact with the semantic layer, and the semantic interconnections allow them to start in one database, and then move around an extendable set of databases. The semantic layer also enables the system to answer semantic queries across several databases such as "What diseases does this drug treat? " or "What kind of drugs can treat this disease?", not like the keyword-based searching mechanism provide by conventional search engines.

The paper is organized as follows. Section 2 talks about the system architecture and technical features. Section 3 elaborates on the implementation of the semantic mediator and the visualized semantic mapping tool. Section 4 introduces the TCM semantic portals which provides semantic query and search services. Section 5 reports the user evaluation and lessons learned from this developing life-cycle. Section 6 mentions some related works. Section 7 gives the summary and our future directions.

Please also note due to the special character of TCM research, in which the Chinese terminologies and definitions are not always interpretable, some figures in this paper contain Chinese search results and web interface. We have annotated all the necessary parts of the figures in English, and we would expect it would be sufficient to understand the functionalities of this application.

2 System Architecture and Technical Features

2.1 System Architecture

As Fig. 2 depicted, there are four key components in the core of DartGrid.

1. **Ontology Service** is used to expose the shared ontologies that are defined using web ontology languages. Typically, the ontology is specified by a domain expert who is also in charge of the publishing, revision, extension of the ontology.
2. **Semantic Registration Service** maintains the semantic mapping information. Typically, database providers define the mappings from relational schema to domain ontology, and submit the registration entry to this service.
3. **Semantic Query Service** is used to process SPARQL semantic queries. Firstly, it gets mapping information from semantic registration service. Afterward, it translates the semantic queries into a set of SQL queries and dispatch them into specific databases. Finally, the results of SQL queries will be merged and transformed back to semantically-enriched format.

[5] `http://ccnt.zju.edu.cn/projects/dartgrid/tcmgrid.html.`
[6] Demo videos `http://ccnt.zju.edu.cn/projects/dartgrid/demo`

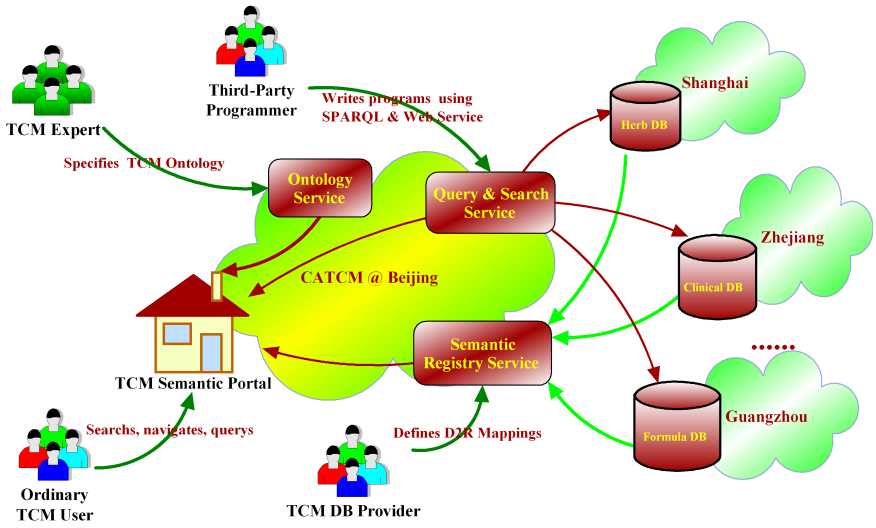

Fig. 2. System Architecture and Usage Senario

4. **Search Service** supports full-text search in all databases. The search results will be statistically calculated to yield a *concepts ranking*, which help user to get more appropriate and accurate results.

2.2 Technical Features

The following four features that distinguish this application from other similar semantic data integration tools, which will be introduced in detail in Section 6.

Semantic View and Visualized Semantic Mapping Tool. In our system, an ontology acts as the semantic mediator for heterogenous databases. Relational database schemas are mapped into corresponding classes or properties, and related by semantic relationship defined in this ontology. To be specific, the mappings are defined as *semantic views*, that is, each relational table is defined as a *view* over this shared ontology. Defining mappings is a labor-intensive and error-prone task. In our system, new database could be added into the system by using a visualized mapping tool. It provides many easy-of-use functionalities such as drag-and-drop mapping, mapping visualization, data source annotation and so on.

SPARQL Query Rewriting with Additional Inference Capabilities. A view-based query rewriting algorithm is implemented to rewrite the SPARQL queries into a set of SQL queries. This algorithm extends earlier relational and XML techniques for rewriting queries using views, with consideration of the features of web ontology languages. Otherwise, this algorithm is also enriched by additional inference capabilities on predicates such as *subClassOf* and *subPropertyOf*.

Ontology-based Semantic Query User Interface. A form-based query interface is offered to construct semantic queries over shared ontologies. It is automatically gen-

erated at runtime according to property definitions of classes, and will finally generate a SPARQL query.

Intuitive Search Interface with Concepts Ranking and Semantic Navigation. This Google-like search interface accepts one or more keywords and makes a complete full-text search in all databases. Users could semantically navigate in the search results, and move around an extendable set of databases based on the semantic relationships defined in the semantic layer. Meanwhile, the search system could generate a suggested list of concepts which are ranked based on their relevance to the keywords. Thereafter, users could explore into the semantic query interface of those concepts, and specify a semantic query on them to get more accurate and appropriate information.

3 Semantic Mediation

3.1 Semantic View and View-based Mapping

In our system, databases are mediated and related by a shared ontology, and each relational table is mapped into one or more classes. For example, the mapping scenario in Fig. 3 illustrates relational schemas from two sources(W3C and ZJU), and a shared ontology (a part of the foaf ontology).

Mappings are typically defined as views in conventional data integration systems in the form of GAV (global-as-view), LAV (local-as-view) [2]. Considering the case in this paper, GAV is to define each class or property as a view over relational tables, and LAV is to define each relational table as a view (or query) over the shared ontology. The experiences from conventional data integration systems tell us that LAV provides greater extensibility than GAV: the addition of new sources is less likely to require a change to the mediated schema [2]. In our TCM case, new databases are regularly added so total number of databases is increasing gradually. Therefore, the LAV approach is employed in our system, that is, each relational table is defined as a view over the ontologies. We call such kind of views as *Semantic View*.

The lower part of Fig. 3 showcases how to represent the mappings as *semantic views* in a Datalog-like syntax. Like in conventional data integration, a typical *semantic view* consists of two parts. The left part is called the view head, and is a relational predicate. The right part is called the view body, and is a set of RDF triples. There are two kinds of variables in the view definitions. Those variables such as “?en,?em,?eh,?pn,?ph” are called distinguished variables, which will be assigned by an data or instance values from the database. Those variables such as “?y1, ?y2” are called existential variables.

In general, the body can be viewed as a query over the ontology, and it defines the semantics of the relational predicate from the perspective of the ontology. The meaning of semantic view would be more clear if we construct a *Target Instance* based on the semantic mapping specified by these views. For example, given a relational tuple as below, applying the View-4 in Fig. 3 on this tuple will yield a set of RDF triples.

```
Relational Tuple:
     w3c:emp("DanBrickley","danbri@w3.org",
                "SWAD","http://swad.org","EU");
```

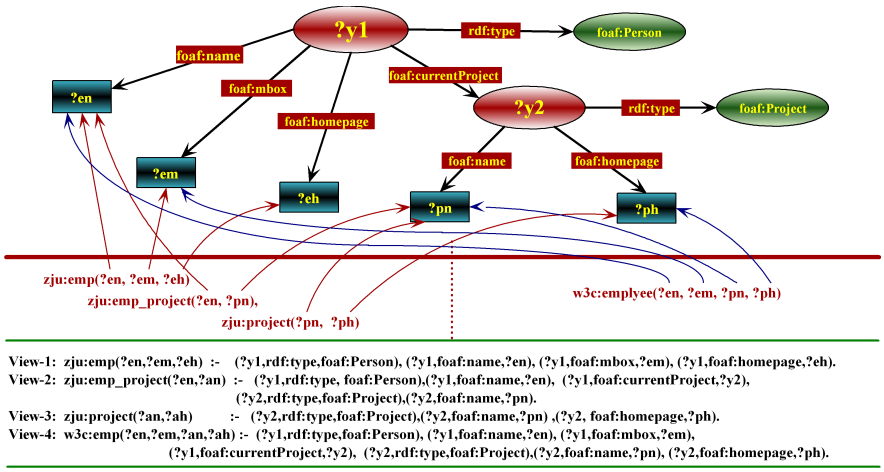

Fig. 3. Mappings from two relational databases with different structures to an ontology. "?en,?em,?eh,?pn,?ph" are variables and represent "employee name", "employee email", "employee homepage", "project name", "project homepage", respectively.

```
Yielded RDF triples by Applying View-4:
     _:bn1 rdf:type foaf:Person;
           foaf:name "Dan Brickley";
           foaf:mbox "danbri@w3.org";
           foaf:currentProject _:bn2.
     _:bn2 rdf:type  foaf:Project;
           foaf:name "SWAD";
           foaf:homepage "http://swad.org".
```

One of the key notion is the newly generated blank node ID. As illustrated, corresponding to each existential variable $?y$ in the view, a new blank node ID is generated. For examples, *_:bn1*,*_:bn2* are both newly generated blank node IDs corresponding to the variables $?y1, ?y2$ in View-4 respectively. This treatment of existential variable is in accordance with the RDF semantics, since blank nodes can be viewed as existential variables. We give the formal definition of the semantic view as below. More detailed Foudermental aspects about *semantic view* could be found in another paper [3].

Definition 1. Semantic View. Let Var be a set of variable names . A typical semantic view is like the form :$R(\bar{X}) : -G(\bar{X}, \bar{Y}), where$:
1. $R(\bar{X})$is called the head of the view, and R is a relational predicate ;
2. $G(\bar{X}, \bar{Y})$ is called the body of the view, and G is a RDF graph with some nodes replaced by variables in Var;
3. The $\bar{X}, \bar{Y}$ contain either variables or constants.The variables in $\bar{X}$ are called distinguished variables , and the variables in $\bar{Y}$ are called existential variables.

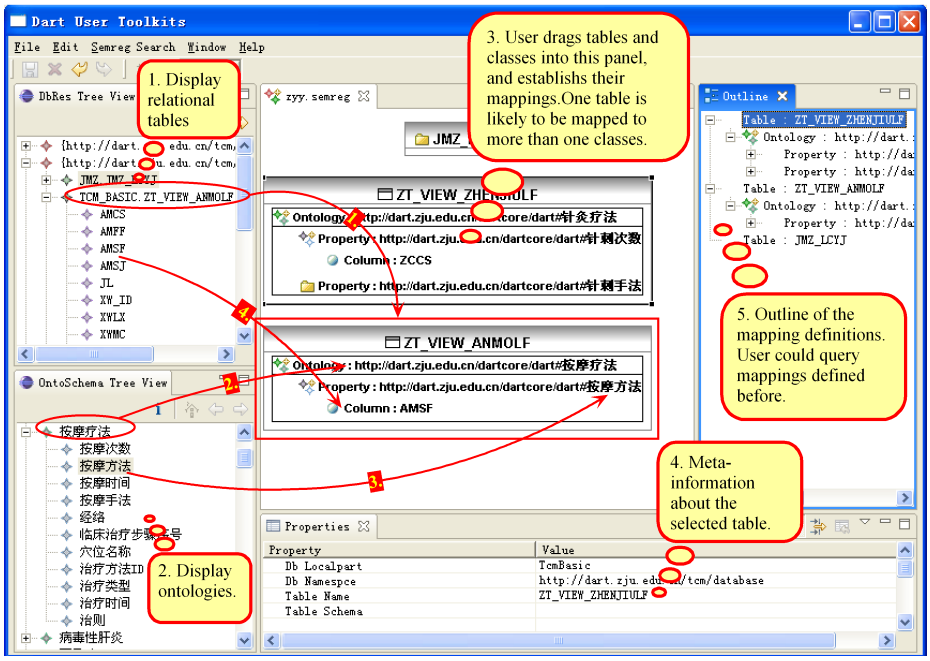

Fig. 4. Visualized Semantic Mapping Tool

3.2 Visualized Semantic Mapping Tool

The task of defining semantic mappings from relational schema to ontologies is burdensome and erroneous. Although we could develop tools to automate this process, it still can not be fully automated and requires humans involvement, especially for integration of databases with different schema structures.

Fig. 4 displays the visualized mapping tool we developed to facilitate the task of defining semantic views. It has five panels. The *DBRes* panel displays the relational schemas, and the *OntoSchem* panel displays the shared ontology. The *Mapping Panel* visually displays the mappings from relational schemas to ontologies. Typically, user drag tables or columns from DBRes panel, and drag classes or properties from OntoSchem panel, then drop them into the mapping panel to establish the mappings. By simple drag-and-drop operations, users could easily specify which classes should be mapped into a table and which property should be mapped into a table column. After these operations, the tool automatically generates a registration entry, which is submit to the semantic registration service. Besides, user could use the *Outline* panel to browse and query previously defined mapping information, and use the *Properties* panel to specify some global information, such as namespace, or view the meta-information about the table.

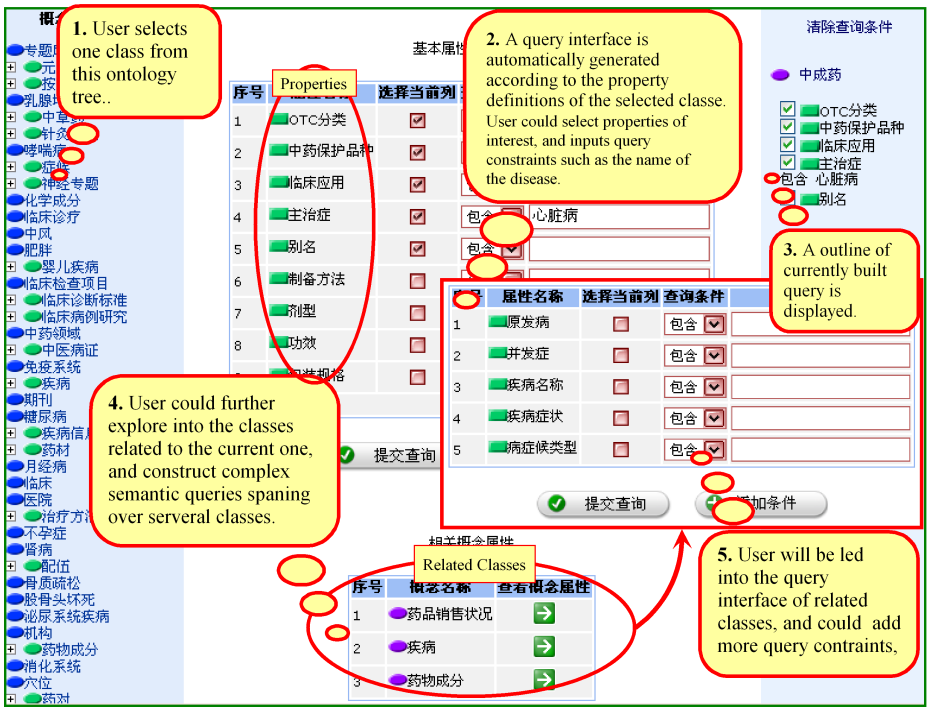

Fig. 5. Dynamic Semantic Query Portal. Please note: because many Chinese medical terminologies are only available in Chinese language and they are not always interpretable, we have annotated all the necessary parts of the figures in English.

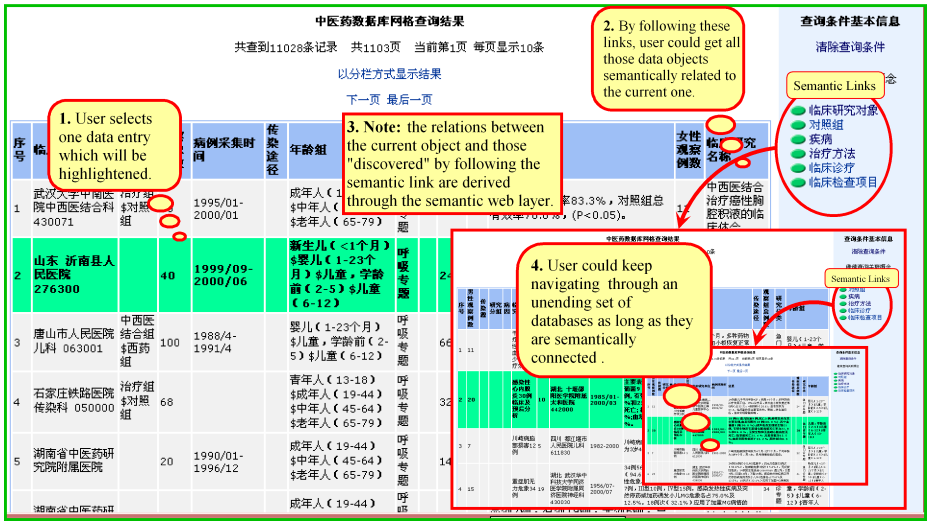

Fig. 6. Semantic navigation through the query results.

4 TCM Semantic Portals

The semantic mediator is designed to separate data providers and data consumers so that they only need to interact with the semantic layer. For example, developers could write applications using the shared ontology without the need of any knowledge about databases. Besides that, our system also offer two different kinds of user interfaces to support query and search services.

4.1 Dynamic Semantic Query Interface

This form-like query interface is intended to facilitate users in constructing semantic queries. The query form is automatically generated according to class definitions. This design provides the extensibility of the whole system – when ontology is updated with the changes of database schema, the interface could dynamically adapt to the updated shared ontology.

Fig. 5 shows the situation how a TCM user constructs a semantic query. Starting from the *ontology view panel* on the left, user can browse the ontology tree and select the classes of interest. A query form corresponding to the property definitions of the selected class will be automatically generated and displayed in the middle. Then user can check and select the properties of interests or input query constraints into the text boxes. Accordingly, a SPARQL query is constructed and could be submit to the semantic query service, where the query will be rewritten into a set of SQL queries using mapping views contained in the semantic registration service. The query rewriting is a somewhat complicated process, and [3] gives the detailed introduction on the rewriting algorithm. In addition, user could define more complex queries. For example, depicted in the lower-middle part of Fig. 5, user could follow the links leading to related classes of the current class, and select more properties or input new query constraints.

Fig. 6 shows the situation in which a TCM user is navigating the query results. Starting from selecting one result highlighted, the user can find out all of the related data entries by following the semantic links. Please note that in this example, the relations between the search results and those “discovered” by following the semantic links, are derived from the semantic layer.

4.2 Intuitive Search Interface with Concepts Ranking and Semantic Navigation

Unlike the semantic query interface, this Google-like search interface just accepts one or more keywords and makes a complete full-text search in all databases. Fig. 7 shows the situation where a TCM user performs some search operations. Starting from inputting a keyword, the user can retrieve all of those data entries containing one or more hits of that keyword. Being similar to the case of the query interface, user could also semantically navigate the search results by following the semantic links listed with each entries.

Meanwhile, the search system generates a list of suggested concepts which are displayed on the right part of the portal. They are ranked based on their relevance to the keywords. These concept links will lead the users to the dynamic query interface introduced in previous section. Thereafter, users could specify a semantic query on them to get more accurate and appropriate information. We call it as intuitive search because it could generate a list of concept suggestions to help user improve the search results.

Fig. 7. Intuitive Search Portal with Concept Ranking and Semantic Navigation

5 User Evaluation and Lesson Learned

5.1 Feedbacks from CATCM

The first proof-of-concepts prototype was deployed during fall 2004. By using that prototype, we convinced CATCM partner to take the semantic web technologies to help them in managing their fast increasing TCM databases. After a thorough requirements analysis and with a careful redesign and re-engineering of the entire system, a more stable and user-friendly version was released in September 2005, and deployed at CATCM for open evaluation and real use.

Currently, the system deployed at CATCM provides access to over 70 databases including TCM herbal medicine databases, TCM compound formula databases, clinical symptom databases, traditional Chinese drug database, traditional Tibetan drug database, TCM product and enterprise databases, and so on. The TCM shared ontology includes over 70 classes, 800 data or object properties.

In general, users from CATCM reacted positively to the entire semantic web approach and our system. They indicated that the system provided an amazing solution for the semantic heterogeneity problem which had been troubling them for a long time. In particular, they gave high praise to the visualized semantic registration tool, and indicated that the features of semantic registration of new database considerably save them a lot of time when new database were developed and needed to be integrated.

They also gave positive comments to the semantic portals as well, especially the semantic navigation functionality. They indicated that semantic interconnections among

different databases was indeed what they wanted. Nevertheless, we found most of the users prefer Google-like search to semantic query interface. Some of them complained that the learning cycle of using the semantic query interface was too long, although it could return more accurate results. They also said they would very like to use the concepts ranking functionality to get more accurate result by constructing further queries when the entries returned from search was overwhelming.

5.2 A Survey on the Usage of RDF/OWL Predicates

RDF/OWL has offered us a range of predicates, but not all of them are useful for relational data integration. We made a survey on the usage of RDF/OWL predicates for relational database integration, and the results are indicated in table 1.

In this survey, we invited ten developers who are familiar with both semantic web technologies and our system. They are asked with the same questions: "From a practical view, what are those most important constructs do you think for relational data integration in semantic web", and are requested to write down some explanation for the reason of their choice. We summarize their comments and the score result as follows.

Predicate	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	AVG
rdf:datatype	9	10	8	9	10	10	9	7	10	9	9.1
rdfs:subClassOf	8	8	7	9	9	8	8	9	10	7	8.3
rdfs:subPropertyOf	8	8	8	7	8	8	9	9	9	8	8.2
owl:inverseOf	8	8	7	8	7	9	8	9	7	9	8.0
owl:cardinality	7	8	7	7	6	7	9	7	7	9	7.4

Table 1. The results for the survey of predicates usage.

Data type support was considered to be important, because most commercial RDBMS has well-defined and unique data type system. RDFS predicates *rdfs:subClassOf* and *rdfs:subPropertyOf* have higher scores because they could enhance the query processing with additional inference capabilities. OWL predicate *owl:inverseOf* is useful when defining relations in both directions which is a usual case in relation database integration. One of the developer indicated that predicate *owl:inverseOf* could help to find more efficient rewritings in some cases. Predicate *owl:cardinality* is useful in adding more constraints to ensure the data integrity.

Some other predicates are considered as useful include: *owl:TransitiveProperty, owl:SymmetricProperty, owl:DatatypeProperty, owl:ObjectProperty*. Some of them thought both *owl:TransitiveProperty* and *owl:SymmetricProperty* could add additional deductive capabilities on top to yield more query results. *owl:DatatypeProperty* and *owl:Object Property* could be used to distinguish simple data value column and foreign key column.

6 Related Works

6.1 Semantic Web Context

In the Semantic Web community, semantic data integration has been always a noticeable research topic. In particular, there have been a number of works dealing with how to make contents of existing or legacy database available for semantic web applications. A typical one is D2RQ[7]. D2RQ is a declarative language to describe mappings between relational database schemata and OWL/RDFS ontologies, and is implemented as a Jena plugin that rewrites RDQL queries into SQL queries. The result sets of these SQL queries are transformed into RDF triples that are passed up to the higher layers of the Jena framework. RDF Gateway[8] is a commercial software having similar functionalities. It connects legacy database resources to the Semantic Web via its *SQL Data Service Interface*. The *SQL Data Service* translates a RDF based query to a SQL query and returns the results as RDF data. Our system is different from D2RQ and RDF Gateway. We take the view-based mapping approach which has sound theoretical foundation, and we have visualized mapping tool and ontology-based query and search tool which are not offered by these two systems.

Some other works propose direct manipulation of relational data to RDF/OWL format, and then the data could be processed by OWL reasoners or be integrated by ontological mapping tool. D2RMap, KAON REVERSE[9] and many other toolkits offer such kind of reverse engineering functionality. Cristian Perez de Laborda and colleagues [4] propose an ontology called “Relation OWL” to describe the relational schema as OWL, and then use this OWL-representation to transform relational data items into RDF/OWL and provide query service by RDQL. The shortcoming of this kind of approaches is that they have to dump all the relational data into RDF/OWL format before querying, which would be impractical if the RDBMS contains huge volume of data. Moreover, they did not consider the issue of integrating heterogeneous databases using formal ontologies, which is one of the focuses of our solution.

Yuan An and colleagues [5] present an interesting paper concerning about defining semantic mappings between relational tables and ontologies within semantic web context. They introduce a tool which could automatically infer the LAV mapping formulas from simple predicate correspondences between relational schema and formal ontologies. Although completely automatic approach to define semantic mapping is difficult, it would be great enhancement to our visualized tool if some candidate mapping suggestions could be provided beforehand. That will be one of our future work.

The DOPE project (Drug Ontology Project for Elsevier) [6] explores ways to provide access to multiple life science information sources through a single semantic interface called DOPE browser. However, it is still a document management system, mainly concerning on thesaurus-based search, RDF-based querying, and concept-based visualization of large online document repositories. It can not answer semantic queries such as “What diseases does this drug treat?” or ”What kind of drugs can treat this disease?”. We’ve seen the authors of DOPE are considering it as one of their future work.

[7] `http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/`
[8] `http://www.intellidimension.com`
[9] `http://kaon.semanticweb.org/alphaworld/reverse/view`

Piazza [7] is an interesting P2P-based data integration system with consideration of semantic web vision. But the current system has been implemented with the XML data model for its mapping language and query answering. However, we think P2P architecture would be a promising direction, and we are considering to extend our system to support P2P working mode and test its scalability and usability.

For other related works, Dejing Dou and colleagues [8] propose an ontology-based framework called OntoGrate. It can automatically transform relational schema into ontological representation, and users can define the mappings at the ontological level using bridge-axioms. Francois [9] considers theoretic aspect of answering query using views for semantic web and Peter Haase and Boris Motik introduces a mapping system for OWL-DL ontology integration [10].

6.2 Conventional Data Integration Context

Without considering the semantic web technologies, our solution can be categorized to the topic ”answering query using view”, which has been extensively studied in database community [2] [11]. Most previous works has been focused on the relational case [2], and XML case [12].

On the one hand, we believe it would be valuable for the semantic web community to take more consideration of the techniques that have been well studied in the database community such as answering query using view. On the other hand, we think that the semantic web research does raise a lot of new issues and challenges for database researchers. From our experiences, the challenges include: From our experiences, the challenges include: how to rank the data object just like the page rank of google? how to maintain highly evolvable and changeable schema mappings among an great number of and open-ended set of databases with no centralized control?

Moreover, a lot of works have been done in the area of ontology-based data integration [13]. Many of them took some ontological formalism such as DL to mediate heterogenous databases, and used the view-based mapping approach. In comparison with them, our implementation is the case of RDF/OWL-based relational data integration with a *semantic web vision in mind*.

7 Summary and Future Work

In this paper, we presented an in-use application of Traditional Chinese Medicine enhanced by a range of semantic web technologies, including RDF/OWL semantics and reasoning tools. The ultimate goal of this system is to realize the ”web of structured data” vision by semantically interconnecting legacy databases, that allows a person, or a machine, to start in one database, and then move around an unending set of databases which are connected by rich semantics. To achieve this demanding goal, a set of convenient tools were developed, such as visualized semantic mapping tool, dynamic semantic query tool, and intuitive search tool with concepts ranking. Domain users from CATCM indicated that the system provided an amazing solution for the semantic heterogeneity problem troubling them for a long time.

Currently, although this project is complete, several updated functionalities are still in our consideration. To be specific, we are going to enhance the mapping tools with some heuristic rules to automate the mapping task as far as possible, just like the approach proposed by Yuan An and colleagues [5]. Otherwise, we will develop a more sophisticated mechanism to rank the data objects just like the page rank technology provided by popular search engines.

Acknowledgements

The authors' research is supported by China 973 subprogram "Semantic Grid for Traditional Chinese Medicine" (NO.2003CB316906), China NSF program (NO. NSFC60503018) and the EU-IST-027595 NeOn project. We would thank the fruitful discussion and first hand evaluation from our colleagues and partners.

References

1. Buitelaar, P., Olejnik, D., Sintek, M.: OntoLT: A protégé plug-in for ontology extraction from text. In: Proceedings of the International Semantic Web Conference (ISWC). (2003)
2. Halevy, A.Y.: Answering queries using views: A survey. The VLDB Journal. **10** (2001) 270–294
3. Chen, H., Wu, Z., Wang, H., Mao, Y.: Rdf/rdfs-based relational database integration. In: ICDE. (2006) 94
4. de Laborda, C.P., Conrad, S.: Bringing relational data into the semantic web using sparql and relational owl. In: International Workshop on Semantic Web and Database at ICDE 2006. (2006) 55–60
5. An, Y., Borgida, A., Mylopoulos, J.: Inferring complex semantic mappings between relational tables and ontologies from simple correspondences. In: International Semantic Web Conference. (2005) 6–20
6. Stuckenschmidt, H., van Harmelen, F., de Waard et al, A.: Exploring large document repositories with rdf technology: The dope project. IEEE Intelligent Systems. **19** (2004) 34–40
7. Halevy, A.Y., Ives, Z.G., Madhavan, J., Mork, P., Suciu, D., Tatarinov, I.: The piazza peer data management system. IEEE Trans. Knowl. Data Eng. **16-7** (2004) 787–798
8. Dou, D., LePendu, P., Kim, S., Qi, P.: Integrating databases into the semantic web through an ontology-based framework. In: International Workshop on Semantic Web and Database at ICDE 2006. (2006) 33–50
9. Goasdoue, F.: Answering queries using views: a krdb perspective for the semantic web. ACM Transaction on Internet Technology. (2003) 1–22
10. Haase, P., Motik, B.: A mapping system for the integration of owl-dl ontologies. In: IHIS '05: Proceedings of the first international workshop on Interoperability of heterogeneous information systems. (2005) 9–16
11. Abiteboul, S., Duschka, O.M.: Complexity of answering queries using materialized views,. In: The Seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems,. (1998) 254–263
12. Yu, C., Popa, L.: Constraint-based xml query rewriting for data integration. In: 2004 ACM SIGMOD international conference on Management of data. (2004) 371–382
13. Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hubner, S.: Ontology-based integration of information - a survey of existing approaches. In Stuckenschmidt, H., ed.: IJCAI01 Workshop: Ontologies and Information Sharing. (2001) 108–117