Ontology-based Dynamic Building of Relational Data Views

Yudit Ponce-Toste[1] , Maikel Pérez-Gort[1] , Félix Fernández-Peña[1] , Jyrki Nummenmaa[2]

[1]Politechnical University of Havana, Havana, Cuba.
[2]University of Tampere, Tampere, Finland.
yponce@hispavista.com, {mlazaro,felix}@ceis.cujae.edu.cu
jyrki.nummenmaa@uta.fi

Abstract

Traditionally, data views are seen as static, syntactically correct, data sets. The semantics of the data is not explicitly encoded in RDB (relational databases) but implicitly on the application level. However, the needs to create context-aware browsing methods in changing scenarios are demanding more flexible mechanisms. Data views need to be semantically enriched for capturing the real world changes. In this paper, we study how the semantics of RDB can be described and used in an ontology-based method for building dynamic data views. Our tests show that our approach increases the flexibility of information systems whilst decreases their maintenance costs.

Keywords: dynamic data views, semantic information layer, ontology-driven, semantic database schema, context-aware browsing.

1. Introduction

The corporate environment is increasingly dependent on the speed and accuracy of information retrieval. Reliable and up-to-date data, along with the necessary knowledge needed to interpret it, are probably the most important resources in this scenario. Operational data retrieval has a central role in corporate information systems as they act as an interface to data and information.

An efficient system for retrieving information must be flexible, being able to adapt to changes in the information system life cycle. New operational data, new data analysis and changing the understanding of data are common, and naturally accepted, as well as the fallibility of knowledge is.

However, adapting information systems to changing environments requires a lot of manual effort. The semantics of data may be implicit in the implementation of stored procedures and in the definition of integrity rules but RDB schemas do not use to contain a formal explicit definition of the meaning of data in the corresponding domain of discourse.

Thus, when the interpretation of stored data changes traditional information systems require the recoding of the application. We propose to reduce maintenance costs and to increase the flexibility of data retrieving in information systems by making explicit the semantics of RDB.

Since RDF (Resource Description Language) makes it possible to define the meaning of data in a machine readable form [1], it seems that semantic web technologies could be helpful in the alignment of RDB towards the semantic dimension of data manageability.

A lot of attention has been paid to the semantic enrichment of data stored in RDB [2] [3] [4]. The evolution of RDF into OWL (Web Ontology Language) allows a richer semantic description based on DL (description logic). The combination of RDF/OWL has been used in many specific scenarios for the construction of flexible data semantic models [5] [6] [7].

Interest in mapping relational data to RDF is increasing for the purpose of publishing linked data [8]. In this direction, SPARQL is the W3C recommended query language for information retrieving from RDF documents. In the semantic web vision, SPARQL is considered the theoretical equivalent to SQL in relational databases. However, as Hert et al. did in 2010 [9], we consider that converting relational data to RDF is often not feasible.

We propose not to transform databases into semantically aware data but to define a semantics-driven data retrieving method. The fundamental idea is to describe the data sources using an ontology based on both global and application area specific concepts and restrictions related to their instances.

In our design, we formalise the structure of a generic RDB based on the description of the interrelation of concepts and concepts' descriptors. Our aim is to define a RDB schema not in the syntactic but in the semantic dimension.

When updating the understanding of the domain of discourse there is no need to recode the data retrieving procedures but to transform the semantic representation of retrievable data. As a consequence, a new way for the dynamic generation of data views is achievable. Data are actually charted taking into account the relevance and the actual meaning of data in the relational schema.

We have constructed a prototype implementation of data views generator for evaluating capabilities of data presentation and browsing when using our method. Our comparison with a traditional implementation of RDBMS shows that our approach could be beneficial for data

retrieving in information systems when an accurate semantic description of the RDB exists.

The paper is organised as follows. First the related work is briefly reviewed. After that, we give essential background information needed to understand the rest of the paper. In section 3, we explain how our method works and in the next section we describe our RDF/OWL model of RDB schema. Then, in section 5, we analyse some usability issues of information gathering in our ontology-based procedure. A scenario of proof and validation of the proposal is described in section 6. The discussion focuses on data retrieving using our approach and traditional RDBMS (Relational Database Management Systems).

2. Related Work

2.1. View-Based semantics browsing

Defining and using views in information seeking tasks has been the focus of several researchers. Dichev and Dicheva proposed a view-based semantic search and browsing model [10]. Their proposal is based on the semantic description of view descriptors derived from users' tasks or goals.

The customization capabilities of the user interface improve since information about the user profile is made explicit in context. However, implicit relevance of the kind of data stored in the RDB keeps hidden to the application. The importance of the context-awareness in the application layer of view-based initiatives is also mentioned by Hong et al. in their survey of context-aware systems [11] and, more recently, by Namiot [12].

2.2. Ontology-based RDB Schemas

The transformation of the vast quantities of data, currently residing in Relational Databases, into semantically aware data has been previously identified as a necessity [2] [3] [5] [13] [14] [15] [4]. Ontology-based RDB schemas have been proposed for specific scenarios [2] and distributed environments [3] [4].

Barsalou et al. have worked in a semantic data model to enhance relational databases (in the area of immunogenetics). With structuring and manipulating tools for data retrieving, Barsalou et al. obtained an appropriate level of abstraction in the studied scenario [2]. Almost twenty years later, Sun and Fan concluded that semantic extraction is essential for semantic interoperability in multi-enterprise business collaboration environments [3]. Sun and Fan proposed a unified syntax-independent conceptual model that showed to be extensible and flexible in multi-enterprise business collaboration environments [3]. Further, Guido and Paiano proposed to take the integration of information systems as a whole to a semantic dimension [16].

More recently, Song et al. proposed SIL (a semantic information layer) as mediation media among heterogeneous database systems [4]. A dynamic multi-strategies ontology alignment with automatic matcher selection and dynamic similarity aggregation allowed them mapping data sources for retrieving distributed data. All these have been important contributions in data retrieving. However, these studies in the semantic dimension of RDB are not fully focused on the influence of semantics over the generation of the data views but on overcoming the gap of conceptual heterogeneity. To our knowledge, the semantic web technologies have not been widely used in the generation of data views and it seems there is no consensus yet on how to fully take RDB into the semantic dimension of data management.

2.3. RDB Semantic Mapping

Many mapping languages and approaches were explored leading to the ongoing standardization effort of the World Wide Web Consortium (W3C) carried out in the RDB2RDF Working Group [13]. *RDOTE* is a recent proposal for the automatic and custom mapping and transportation of data residing in RDB into RDF [15].

Agus-Santoso et al. incorporate concept hierarchy as background knowledge for the OWL ontologies extraction on top of RDB [5]. The RDB2OWL language [6] reuses the OWL ontology structure as a backbone for mapping specification by placing the database link information into the annotations for ontology classes and properties.

Hert et al. have worked in updating data stored in RDB from the semantic dimension of data management. They emphasized in the importance of formalising a semantics-based RDB management towards flexibility [9].

Wu et al. have been working in semantic query, search and navigation services by dynamically mapping SPARQL queries (the W3C recommended query language for information retrieving from RDF documents) to SQL queries. Their proposal considers using a concepts-ranking mechanism to provide more accurate and reliable search results for the users [14].

We consider that 1) RDB mapping into the semantic web should not be syntactically limited to one-to-one relations between database tables and ontology concepts and that 2) the implicit semantics enrichment of RDB should be taken into account and made explicit. Further, the translation of SPARQL queries into SQL is far from trivial when the use of grouping or mathematic functions is needed to calculate actual values for data fields defined in the logical design of RDB.

This limitation could be related to the fact that mathematic functions and aggregates have not been actually included in the SPARQL language until a few months ago [17]. Anyway, two independent studies have shown that RDB to RDF mapping systems executing SPARQL on relational databases are several orders of magnitude slower than executing the semantically equivalent SQL query directly on the RDBMS [18].

Instead of migrating available legacy data in relational database into ontologies, it seems an option to keep using

SQL for what it has proved to be good at and to explicitly link concepts from the semantic dimension of data manageability to relational tables.

3. Outline of Method

The motivation behind our work is to support data retrieving procedures in information systems in terms of the semantics (meaning) of stored data in the corresponding RDB. Our work is inspired on the design of ontologies for data integration for OLAP, previously developed in collaboration with Niemi et al. [19]. Similarly, the semantic RDB Schema we propose is structured in three layers with different levels of abstraction.

The top layer of our framework is a generic ontology. The so called *General Relational Database Ontology* (*GROD*) defines common relevant concepts for information gathering in any RDB. The middle layer, Domain Specific Relational Database Ontology (DROD), is a domain ontology that extends GROD in the context of a specific application area. Further, the bottom layer, *DROD Instances* refers to the actual data in the RDB. Figure 1 illustrates the structure of our framework.

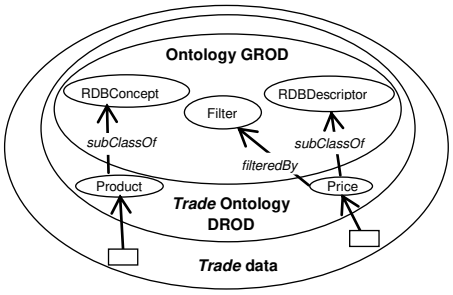

Fig. 1. Structure of the framework

To get full benefit from the power of the Semantic Web, any DROD ontology should be defined based on common concepts of GROD. This means that the common concepts have global definitions shared among all domain specific ontologies related to data retrieving. Together, the three layers define what we call a G/DROD schema.

It is used world trade data to illustrate our method. This data contain auto-generated import/export figures. Figure 2 shows the Syntactic RDB Schema. The example aims to demonstrate the different relationships among the semantic and syntactic layers of language that G/DROD formalises for RDB. The implementation of a prototype for the defined ontology-based procedure to retrieve information was tested using this data. Results are analised in section 6.

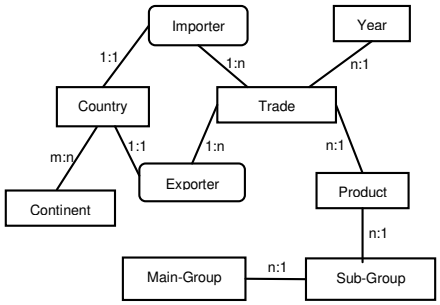

Fig. 2: RDB Model of world trade data.

In our example, the data stored in the RDB characterise products, importers, exporters and trades among them. As a whole, our data gathering procedure works as follows:

- Row data is available in the RDB through specific SQL queries.
- A conceptual definition of the RDB in the semantic dimension is created by extending the GROD ontology in a specific DROD ontology. SQL queries are mapped to relevant concepts of the study case.
- A user interface for browsing, filtering and charting information is generated. –our prototype is a generic implementation of the user interface for data retrieving (its implementation is described in section 5).

4. RDF/OWL Model of RDB Schema

As already indicated, we classify the information into three different abstraction levels:

- The GROD ontology,
- The DROD ontology (which reuses concepts from the GROD ontology), and
- DROD Instances. In this level, the SQL queries mapping the semantic layer (in the GROD ontology) to the syntactic layer (in the actual data of the RDB) are defined.

4.1. GROD Ontology

The semantic RDB-Schema called GROD is an abstract definition of relationships among concepts and descriptors. The graphical presentation of the GROD ontology is seen in figure 3 and its XML representation in figure 4.

Every concept and descriptor is formalised as an OWL class *RDBConcept* and *RDBDescriptor*, respectively. Moreover, *RDBConcept* is defined and described by a set of descriptors that semantically embody the definition of the concept.

Identifying and descriptive relationships between *RDBConcept* classes and *RDBDescriptor* classes are defined. Thus, *identifiedBy* and *describedBy* properties are defined in the domain of *RDBConcept* and the range of *RDBDescriptor*.

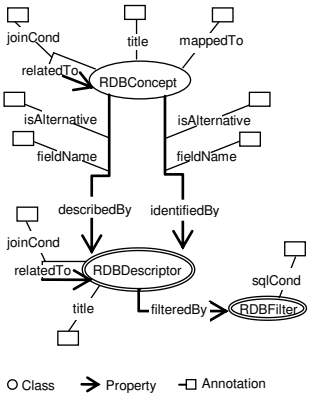

Fig. 3. View of Ontology GROD

The set of defining and describing descriptors of *RDBConcept* are disjointed (a defining descriptor is never a describing descriptor for a specific *RDBConcept* and vice versa).

As in [6], annotations are used in GROD when the value they take does not change, no matter the actual data in the RDB. Our OWL/RDF model includes the declaration of annotations, properties and classes in the linkage of the semantic and syntactic layers as follows:

- *Equivalent SQL query.* Every *RDBConcept* is mapped to a syntactically equivalent SQL sentence using the annotation *mappedTo.*
- *Equivalent fieldName*. The definition of any (identifying/descriptive) relationship between a concept and a descriptor requires the declaration of an annotation *fieldname*. This string value matches the appropriate name or alias name of the descriptor in the corresponding *mappedTo* annotation of the related concept. As the annotation *mappedTo* is declared in *RDBConcept*, the matching between the annotations *fieldName* and *mappedTo* should be verifiable for avoiding ambiguity.
- *Valid filters.* Filters for general data types (numbers, strings and dates) are declared as subclasses of *RDBFilter*. Their declaration includes setting the corresponding annotation *sqlCond*. This annotation sets the string pattern of the *WHERE* condition involved in setting the filter on. Meanwhile, the subproperties of *filteredBy* sets which are the valid filters for each descriptor.
- *Data type of descriptors*. More specific descriptors for specific data types (string, int, float, date, etc.) are declared in GROD as subclasses of *RDBDescriptor*. Any descriptor in DROD should be defined as *subClassOf* the appropriate datatype-specific descriptor. This declaration is used together with the *mappedTo* annotation and *filteredBy* properties in the automatic generation of SQL queries for applying filters in user views.
- *Conditionals of inter-related data items.* The annotation *joinCond* declares the "join condition" of the corresponding relation. The "join condition" sets the correspondence among identifying fieldnames involved in the relationship.

All these five categories of constructions actually link the semantic layer to the syntactic layer of language in the RDB Schema.

However, other annotations were necessary. Thus, the annotation *title* refers to the string value that describes the meaning of a concept or descriptor in natural language. Likewise, the annotation *isAlternative* sets whether the identifying and descriptive relationships are included or not (by default) in the tabular data report of the *RDBConcept* for which the relationship applies.

In the *Trade* study case, the *Trade* concept may be defined by the trade value (the amount of money involved) and, alternatively, by the trade count (the number of trade operations involved). The decision could be including the trade value in the tabular data report of data and including the trade count as related data –not present in the view of trade figures by default.

RDBDescriptor and *RDBFilter* are double-circled in figure 3 emphasizing that more specific (based on data types) descriptor classes and filters respectively, are used in DROD. Some of these more specific descriptor and filter types are defined in GROD (they are not included in the representation for saving space). Nevertheless, if any further specific descriptor or filter is necessary, it is defined in the DROD ontology for the corresponding domain of discourse.

The implications of GROD annotations into the user interface for data retrieving are analysed in section 5.

```
...
<owl:ObjectProperty rdf:about="#identifiedBy">
        <isAlternative></isAlternative>
        <fieldName></fieldName>
        <rdfs:domain rdf:resource="#RDBConcept"/>
        <rdfs:range rdf:resource="#RDBDescriptor"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#describedBy">
        <isAlternative></isAlternative>
        <fieldName></fieldName>
        <rdfs:domain rdf:resource="#RDBConcept"/>
        <rdfs:range rdf:resource="#RDBDescriptor"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#filteredBy">
        <rdfs:range rdf:resource="#Filter"/>
        <rdfs:domain rdf:resource="#RDBDescriptor"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#relatedTo">
        <rdfs:domain rdf:resource="#RDBConcept"/>
        <rdfs:domain rdf:resource="#RDBDescriptor"/>
        <rdfs:range rdf:resource="#RDBConcept"/>
        <rdfs:range rdf:resource="#RDBDescriptor"/>
</owl:ObjectProperty>
...
```

Fig. 4. Partial view of GROD in OWL

4.2. The DROD Ontology

Above we have given a general/abstract definition of our semantic RDB-Schema. The semantic schema of a particular RDB is created by defining subclasses of GROD classes and sub-properties of GROD properties.

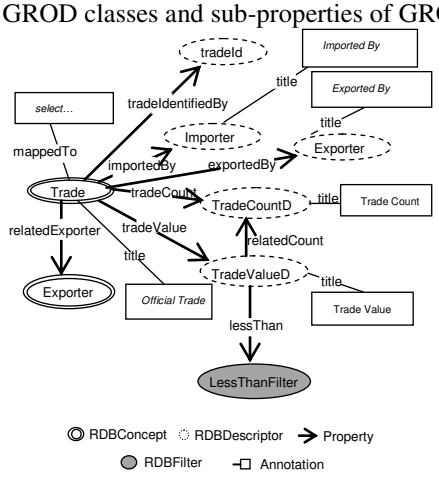

Fig. 5. A partial view of the ontology DROD

The DROD ontology for our example *Trade*, partially shown in figure 5, can be defined as follows:

- The concept *Trade* is defined as a subclass of *RDBConcept*.
- *Trade* is identified by the descriptor *tradeId* (subclass of *RDBDescriptor*). The relationship between *Trade* and *tradeId* is named *tradeIdentifiedBy* (subproperty of *identifiedBy*).
- *Trade* is described by the descriptors *importer* and *exporter* (of string data type), *tradeCountD* (of integer data type) and *tradeValueD* (of float data type).
- The properties that embody *Trade* as a concept are sub-properties of *describedBy* named *importedBy, exportedBy, tradeCount and tradeValue*.
- The relationship *relatedExporter* states that trading is *conceptually* related to exporters. Thus, *relatedExporter* is a sub-property of *relatedTo*.
- The relationship *relatedCount* states that a user interested in the trades value may be interested in the related number of trade operations as well. Correlation among these figures could be significant for decision making.

Figure 6 shows the hierarchy of properties defined in the DROD ontology for *Trade*. These properties are visually related to the generic properties defined in the GROD ontology.

One *RDBConcept* may be related with one or more tables, and vice versa. Thus, *Sub-Group* and *Main-Group* (tables in the RDB) are considered in the definition of only one concept *Group*. Likewise, *Exporter*, *Importer* and *Product* were semantically defined and described but no further concepts were defined. However, if anything else makes sense for the end users, the definition is easily updated by making changes to the OWL file.

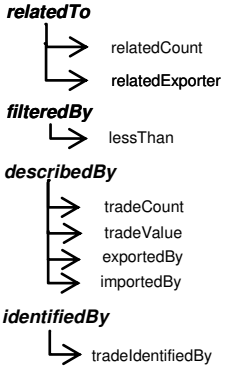

Fig. 6. Hierarchy of properties in DROD

Each *RDBDescriptor* takes an atomic value in the semantic view of the RDB. However, this value is determined in the syntactic layer by combining one or more fieldnames, grouping functions and calculated fields in SQL syntax. I.e., *tradeValueD* and *tradeCount* are calculated in the *mappingTo* annotation of *Trade* by using the SQL functions *sum()* and *count()*, respectively.

4.3. DROD Instances

An instance of a descriptor is formalised as the atomic value for a concept specific property in the specific domain for which this property is defined. Thus, an instance of a concept is formalised as the set of atomic values corresponding to its properties. Moreover, a user view is defined as the matrix derived from a set of instances of a concept (filtered or not) and a semantically relevant set of properties of this concept.

Likewise, the instance level of a particular Semantic RDB-Schema is defined by creating instances for OWL classes with the actual data from the RDB. This layer materialises a knowledge base for making inference about the stored data from a data retrieving perspective.

However, querying an RDF document is considerably slower than using SQL statements to gather information from a RDB, as proved on experimentation [18]. In our use case, the DROD ontology is not populated but used as semantic schema for deciding which SQL statement to execute.

This way, the efficiency of data retrieving is not affected. However, creating instances of the model is a straightforward process. Once the relevance of our approach for data retrieving is proved, offline inference processes seems to be a natural line of future work.

5. An Outline of the Implementation

Our current prototype implementation basically consists of the following steps:

- A DROD ontology is loaded into the prototype.
- Every *RDBConcept* defined in the DROD ontology appears in the interface and becomes into an access point to information.
- The annotation *mappingTo* is used in order to request actual data to the RDB. The user interface is rearranged in each case and the retrieved data is shown.
- The user makes use of related concepts, related descriptors and defined filters for expanding the search spectrum.

Figure 7 illustrates the matching of data into the user view of the concept *Trade* for the study case. The user view combines three aspects, as depicted by the squared hints: 1) ´*Official Trade*´, the *title* annotation of the *Trade* concept, heads the user view, 2) the value of the *title* annotation of each *Trade* descriptor (with *isAlternative* annotation in ´*false*´) heads each column of the tabular data report, and 3) each row contains the corresponding data of each *Trade* instance.

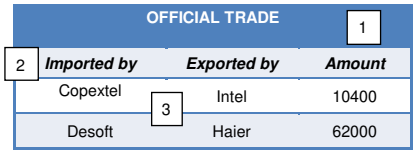

Fig. 7. Trade Concept's User View

Data retrieving and browsing is context aware in the prototype. When a user hovers over a descriptor value and activates the contextual menu two different options appear in the user view, as shown in figure 8. The user then may ask for related data and the user may filter the data in the tabular data report.

Figure 8 illustrates the steps for browsing data (from *Trade* to *Exporter* and then to *Product*) whilst the user goes asking for related data. Data items of each instance of these concepts are charted in terms of relevant descriptors (with *isAlternative* annotation in ´false´). Moreover, it is important to note that not only related concepts but related descriptors as well are available when activating the ´related to´ contextual submenu. These descriptors are not shown by default in the tabular data report because they are marked with the *isAlternative* annotation in ´true´. This is the case of the descriptor labeled ´Trade Count´ in its *title* annotation, related to the *TradeValueD* descriptor of the concept *Trade*.

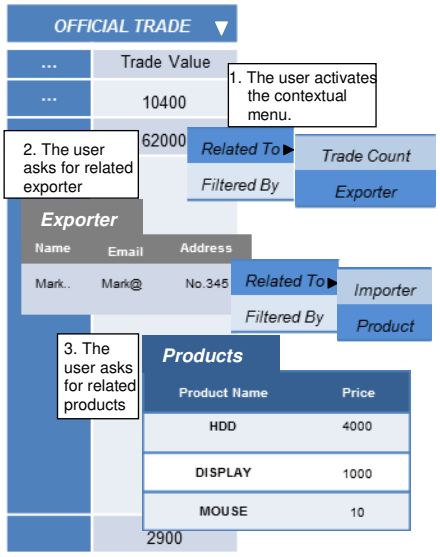

Fig. 8. Semantic navigation in data retrieving

Figure 9 shows the steps for filtering data in the tabular data report of the user view whilst using the example of the *Trade* DROD Ontology. The options of the contextual menu are automatically generated based on the defined *isFilteredBy* relationships with classes *filter* defined in the domain ontology.

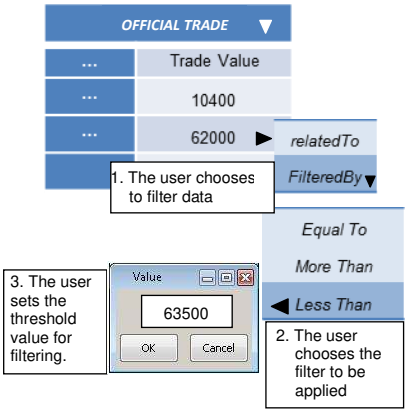

Fig. 9. Applying filters to data

Data retrieval can be done as close to the data source as possible. After the local computation, the data is transmitted to the user's client node where it is integrated. An overview of the implementation is shown in figure 10.

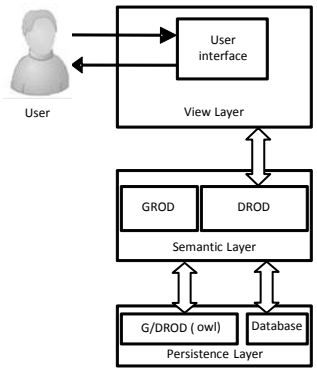

Fig. 10. Implementation architecture

6. Discussion

An experiment was designed to compare data retrieving using our proposal and a traditional approach based just on SQL. A group of ten undergraduate students of Software Engineering participated in the evaluation. Each of them received the same assignment of ten data retrieving operations to be implemented, with progressive complexity in the required SQL Syntax. The last four tasks demanded changes in the semantic interpretation of data.

All the students have experience in database design and RDBMS implementation in Java. A computer with an i3 processor, 2 GB RAM and enough hard drive space was assigned to every student. Each student received the code of a RDBMS implemented in Java and our G/DROD-based data retrieving prototype, both of them working with the RDB ´Trade´.

Five students were asked to complete the tasks by updating the RDBMS and then they were asked to do the same but using the prototype of our approach instead. The other five students were asked to work with our prototype and then with the RDBMS. Time consuming was computed for each task / each student. The amount of computing resources used in each case was computed as well.

Our preliminary tests indicate that our approach facilitates data retrieving and the maintenance of information systems when an accurate semantic description of the RDB exists. In 92% of cases, the student obtained the solution faster when using our approach. The table below illustrates the mean value of time measurements for each task.

Further, the capacities of computing required for adapting our prototype to changing data retrieving needs are irrelevant compared to those required when Eclipse was used to update the equivalent functionality in the RDBMS.

A last relevant result to recall is that the five students who were asked to perform tasks with our approach first, improved the time in which the other five students obtained the equivalent solution with the RDBMS. Students who worked with the RDBMS first, did not improve the results of the other five students when working with G/DROD. So, it seems that our proposal actually can facilitate the understanding of the meaning of data in the context it is used.

Task	RDBMS (*)	G/DROD (*)
T1	1.52	3.83
T2	1.41	3.82
T3	1.35	3.79
T4	1.42	4.2
T5	1.36	3.9
T6	1.35	3.88
T7	1.67	3.72
T8	1.65	3.69
T9	1.62	6.8
T10	1.42	3.82

*Values are expressed in minutes.

Table 1. Experiment results.

7. Conclusions and Future Work

In our approach, data are stored in RDB with their proven track record of scalability, efficient storage, optimised query execution, and reliability.

However, compared to the relational data model, RDF/OWL demonstrates to be more expressive. By mapping RDB to RDF in a specific way, our proposal allows users to focus on data retrieving.

A prototype of a general tool for data retrieving was implemented and used for testing proposes. Our preliminary testing indicated that G/DROD facilitates the maintenance of data retrieving tools of information systems in changing environments.

Data represented in RDF can be interpreted, processed and reasoned over by software agents. Our future work will focus on the improvement of data retrieving processes based on the inference of new information.

8. References

[1] T. Berners-Lee. “Relational databases on the Semantic Web”. www.w3.org/DesignIssues/RDB-RDF.html. 1998.

[2] T. Barsalou, W. Sujansky, L.A. Herzenberg, G. Wiederhold. "Management of complex immunogenetics information using an enhanced relational model". *Journal of Computers and Biomedical Research*, Vol. 24, No. 5, pp. 476-498. October, 1991.

[3] H. Sun, Y. Fan. “Semantic Extraction for Multi-Enterprise Business Collaboration”. *Journal Tsinghua Science & Technology*, Vol. 14, No. 2, pp. 196-205. April, 2009.

[4] F. Song, G. Zacharewicz, D. Chen. “An ontology-driven framework towards building enterprise semantic information layer”. *Journal of Advanced Engineering Informatics*, Vol. 27, No. 1, pp. 38-50. January, 2013.

[5] H. Agus-Santoso, S. Cheng-Haw, Z. Abdul-Mehdi. “Ontology extraction from relational database: Concept hierarchy as background knowledge”. *Journal of Knowledge-Based Systems*, Vol. 24, No. 3, pp. 457-464. April, 2011.

[6] K. Čerāns, G. Būmans. "RDB2OWL: a RDB-to-RDF/OWL Mapping Specification Language". *Proceeding of the 2011 conference on Databases and Information Systems*. 2010.

[7] K. Munir, M. Odeh, R. McClatchey. “Ontology-driven relational query formulation using the semantic and assertional capabilities of OWL-DL”. *Journal Knowledge-Based Systems*, Vol. 35, pp. 144-159. 2012.

[8] M. Al-Sudairy, T. Vasista."Semantic data integration approaches for e-governances". *International Journal of Web & Semantic Technology (IJWesT),* Vol.2, No.1. January, 2011.

[9] M. Hert, G. Reif, H.C. Gall. “Updating relational data via SPARQL/update”. *EDBT '10: Proceedings of the EDBT/ICDT Workshops. 2010.*

[10] Ch. Dichev, D. Dicheva. “View-Based Semantic Search and Browsing”. *WI '06: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*. 2006.

[11] J. Hong, E. Suh, S. Kim. “Context-aware systems: A literature review and classification”. *Journal of Expert Systems with Applications*, Vol. 36, No. 4, pp. 8509-8522. May 2009.

[12] D. Namiot. “Context-Aware Browsing -- A Practical Approach”. *Proceedings of the Sixth International Conference on Next Generation Mobile Applications, Services and Technologies.* 2012.

[13] M. Hert, G. Reif, H.Gall. "A comparison of RDB-to-RDF mapping languages". *Proceedings of the 7th International Conference on Semantic Systems*, 2011.

[14] Z. Wu, H. Chen, X. Jiang. “Semantic-Based Database Integration for Traditional Chinese Medicine”. *Modern Computational Approaches to Traditional Chinese Medicine*, pp. 199-212. 2012.

[15] K. Vavliakis, T. Grollios, P. Mitkas. “RDOTE – Publishing Relational Databases into the Semantic Web”. *Journal of Systems and Software*, *Vol. 86, No. 1*, *Pages 89-99. January 2013.*

[16] A. Guido, R. Paiano. “Semantic integration of information systems”. *International Journal of Computer Networks & Communications*, Vol. 2, No. 1. January, 2010.

[17] S. Harris, A. Seaborne. “SPARQL 1.1 Query Language”. *W3C Recommendation.* 21 March, 2013.

[18] G. Antoniou, O. Corcho, K. Aberer, et al. “Semantic Data Management”. *Report from Dagstuhl Seminar*. 2012.

[19] T. Niemi, S. Toivonen, M. Niinimäki, J. Nummenmaa. “Ontologies with Semantic Web/grid in data integration for OLAP”. *International Journal on Semantic Web and Information Systems, Special Issue on Semantic Web and Data Warehousing*, Vol.3, No. 4. 2007.