ISSN 1392 – 1215 ELECTRONICS AND ELECTRICAL ENGINEERING 2011. No. 9(115)
ELEKTRONIKA IR ELEKTROTECHNIKA

T 125 AUTOMATION, ROBOTICS
AUTOMATIZAVIMAS, ROBOTECHNIKA

Hybrid Method for Storing and Querying Ontologies in Databases

E. Vysniauskas, L. Nemuraite, B. Paradauskas
Department of Information Systems, Kaunas University of Technology,
Studentu str. 50, Kaunas, Lithuania, phone: +37037453445, e-mails: vernest@email.lt, lina.nemuraite@ktu.lt, bronius.paradauskas@ktu.lt

crossref http://dx.doi.org/10.5755/j01.eee.115.9.752

Introduction

Ontology, previously having been an exclusive subject of research in philosophy, currently more and more often is investigated from a perspective of software and systems engineering. This turn was supported by emerged languages and technologies of Semantic Web: Resource Description Framework (RDF), RDF schema RDFS, and Web Ontology Language (OWL), currently OWL 2 [1]. Ontologies are increasingly used in Semantic Web as well as in everyday information systems for their semantic enrichment, intelligence and quality improvement [2, 3]. Such applications require scalability and performance, efficient storage and manipulation of large scale ontological data.

When ontology based systems were growing in scope and volume, backend storages of ontology reasoners began becoming unsuitable. In such circumstances, storing ontologies in relational databases became the relevant needs for Semantic Web and enterprises.

The goal of the paper is to present a method based on reversible, information preserving transformations and transformation algorithms between OWL 2 ontology and relational database (OWL2ToRDB). The advantages of such transformation are twofold. From the one side, it is desirable to store large ontologies in relational databases as these have ensured the best facilities for storing, updating and querying the information of problem domain. From the other side, the massive amounts of relational data need for exposing them on the Semantic Web along with mapping these data structures to an existing ontology or extracting the corresponding ontology from relational database.

The rest of the paper is structured as follows. Section “Related works” shortly presents the state of the art in the area of storing ontologies in databases. Two following sections present metamodels used in OWL2ToRDB method, its main transformations and principles of constructing transformation algorithms. “Method evaluation” section describes method implementation and experiments conducted for method approval. Finally, we present conclusions and future works.

Related works

In connection with emerging needs for storing ontologies, in 2005 the current research was initiated. In that time, methodology for storing ontology in relational databases was still in its infancy [4]. Similar methodologies for transforming Entity-relationship and UML conceptual models into relational database (RDB) structures, transformations between UML, RDB and XML schemas etc are well-established and implemented in CASE tools. OMG Ontology Definition Metamodel defines transformations between OWL, RDF, RDFS, UML, ER and other modelling languages [5]. One possible way to relate ontologies with relational schemas is to use standard methodologies for generating UML models from OWL ontology descriptions and then relational schemas from UML models. Another way is a direct transformation from OWL into relational schema.

Currently, there are several proposals for transforming ontology into relational databases or vice versa (e.g. [4, 6–10]); however, all of them have some drawbacks, or are intended for a certain purpose (e.g. ontology manipulation). These approaches fall into three categories: 1) methodologies using one or a few tables for storing ontology and its instances e.g. [4]; 2) storing ontology metamodel e.g. [9]; 3) using different schemas for storing ontology concepts and its instances e.g. [6]. First and second approaches do not lose ontology information but they are not suitable for efficient query processing that is one of the main advantages of relational databases. The third approach allows efficient querying but a part of ontological information is lost as existing approaches do not map all ontology concepts to relational schema e.g. [6].

Our research is directed towards connecting ontologies and databases for semantic querying or using in database/semantic applications. That means we aim at obtaining practised relational structures for empowering ontologies with advantages of relational databases; also, we aim at ensuring recovery of ontology and its instances for accessing them by semantic technologies. We propose the hybrid approach, when a part of ontology constructs is directly represented by relational database structures and another part having no direct correspondences in a relational database is stored in metadata tables [11].

For bijective transformations, Ehrig et al. presented a formal proof of the sufficient requirement for reversible bijective transformations [12]. This requirement states that the projection of transformation on the source model should cover all source constructs. However, OWL 2 transformation into RDB is not bijective as the most of practical transformations. Stevens has extended this requirement to non-bijective transformations pointing out that all transformations comprising the overall transformation process should be reversible [13]. She acknowledges the QVT Relation (QVT-R) language as the most suitable language for defining bidirectional transformations. However, QVT-R does not guarantee the lossless transformations per se, this has to be verified. Therefore, we have defined our transformations in QVT-R but also checked their correctness via an experiment.

On the base of analysis of existing methods for representing ontologies in relational databases, the following quality criteria were formulated: 1) the transformation should be fully automatic, realizable and have provable correctness; 2) it should not lose data, data types and structure of ontology; 3) the obtained relational database should be applicable not only for saving all ontology information, but also practically and flexibly usable in information systems.

Method description

The scenarios of using a method for transforming ontologies into databases and backward can be imagined in the following way. Information system designer, using some ontology tool (e.g. Protégé, Altova Semantic Works, etc), creates formal ontology for required domain. The ontology model is transformed into a relational database and filled with ontology data. RDB client applications may access database and render results to users. Also, OWL applications may access ontology and its instances from the relational database and apply semantic technologies for reasoning and querying ontology data stored in database tables. The input of the method is domain ontology with instances, the output – relational database with data. The direct OWL2ToRDB transformation is defined in such a way that the reverse transformation RDBToOWL2 also is realizable and lossless.

OWL2ToRDB transformation principles are shown in Fig. 1. OWL2ToRDB transformation consists of a set of transformations that were defined in QVT-R language as instances of QVT-R metamodel. They use OWL 2 model (instances of OWL 2 metamodel) as input and produce OWL 2 RDB model as output.

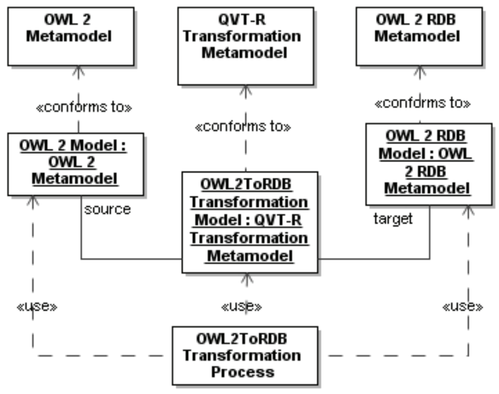
OWL 2 Metamodel
QVT-R Transformation Metamodel
OWL 2 RDB Metamodel
«conforms to»
«conforms to»
«conforms to»
OWL 2 Model : OWL 2 Metamodel
OWL2ToRDB Transformation Model : QVT-R Transformation Metamodel
OWL 2 RDB Model : OWL 2 RDB Metamodel
source
target
«use»
«use»
«use»
OWL2ToRDB Transformation Process

Fig. 1. OWL2ToRDB transformation principles

Top elements of OWL 2 metamodel are presented in Fig. 2.

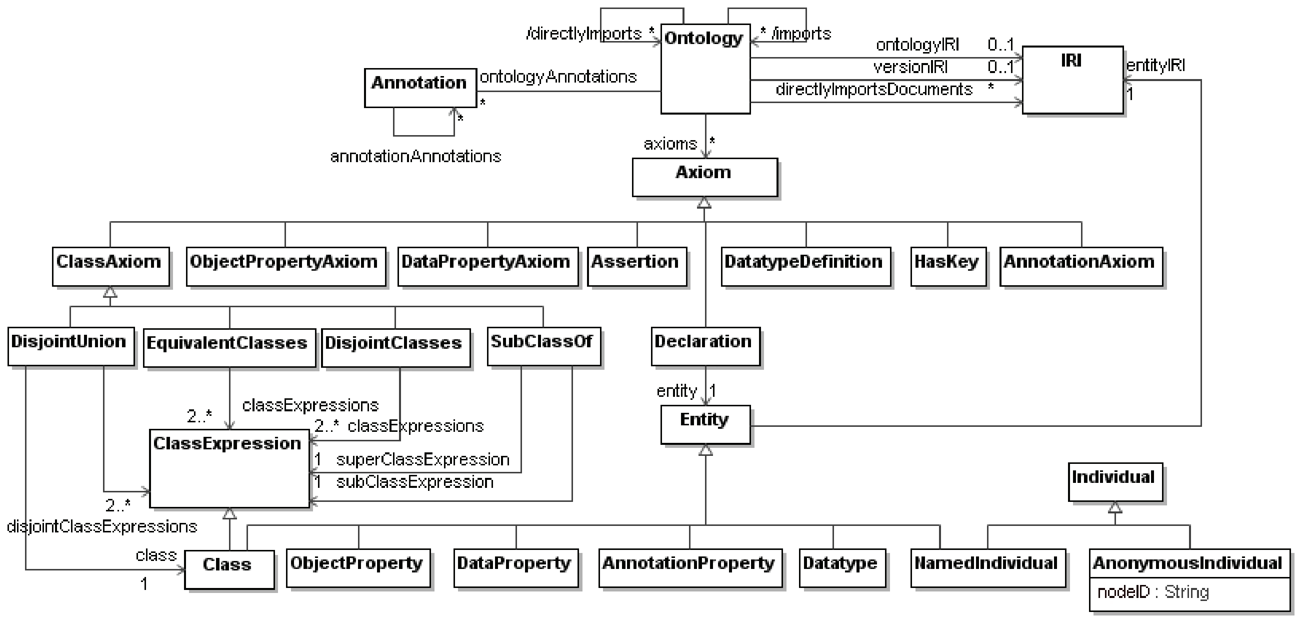

Fig. 2. Excerpt of OWL 2 metamodel [1]

Axiom is a main instrument to define OWL 2 semantic constructs. OWL 2 building blocks are entities (i.e. classes, object properties, annotation properties, data properties, named individuals, and datatypes) that comprise the vocabulary or signature of ontology. One can declare an entity by stating an axiom. Conversely, annotations have no semantics but serve as a powerful means for associating additional information with ontologies, entities, and axioms. Class axioms are defined by class expressions and can state that some class is a subclass of another class, some classes are equivalent or disjoint, or they comprise a disjoint union. In a similar way, you can define object property axioms etc.

Ontology RDB metamodel was obtained from CWM metamodel by eliminating its procedural components and adding metatables (here represented by a single class *Metatable* and three of 33 metatables as its subclasses) for preserving ontology elements having no corresponding constructs in the relational model.

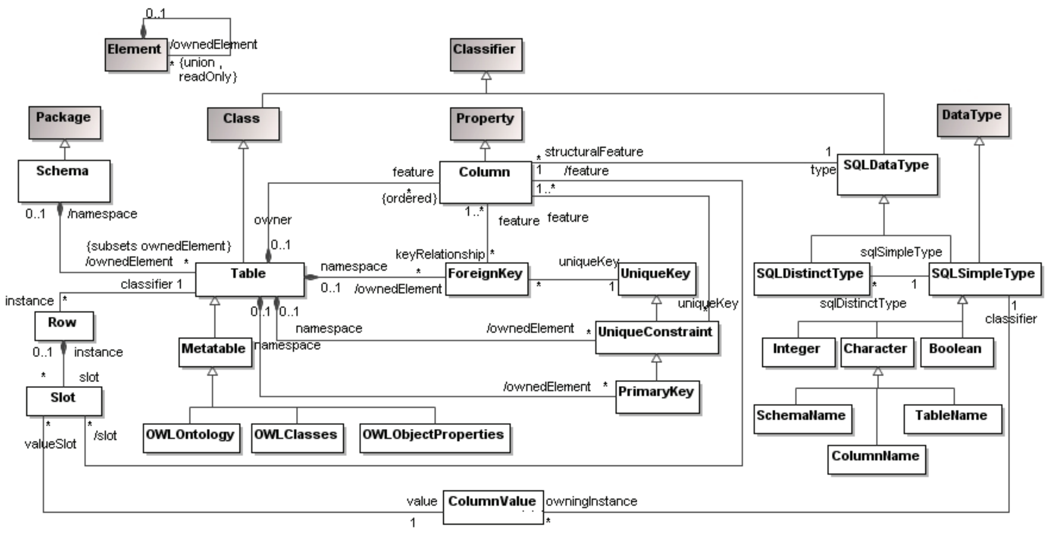

Fig. 3. Simplified Relational metamodel (adopted from [13])

Mappings of main OWL 2 constructs to RDB schema and metaschema are presented in Table 1 (only a little part of mappings for axioms and restrictions to metatables is shown due to space limits).

Table 1. Mappings of main OWL 2 constructs to relational schema and metaschema

OWL Ontology	RDB schema	RDB metaschema
Class	*Table*	Row of *OWLClasses* metatable
SubClassOf	*Foreign Key*	Value of *foreignKey* column *superClass* in *OWLClasses* metatable
HasKey	*UniqueKey*	-
ObjectProperty	*ForeignKey* or *Table*	Row of *OWLObjectProperties*
ObjectPropertyDomain	Points to *Table* owning *ForeignKey* of *ObjectProperty*	Value of *foreignKey* column *objectPropertyDomain* in *OWLObjectProperties* metatable
ObjectPropertyRange	Points to *Table* owning *PrimaryKey*, which is *ForeignKey* of *ObjectProperty*	Value of *ForeignKey* column *objectPropertyRange* in *OWLObjectProperties* metatable
FunctionalObjectProperty	*ForeignKey* of *ObjectProperty* domain *Table*	Value of column *functionalObjectProperty* in *OWLObjectProperties* metatable
InverseFunctionalObjectProperty	*ForeignKey* of *ObjectProperty* range *Table*	Value of column *inverseFunctionalObjectProperty* in *OWLObjectProperties* metatable
EquivalentClasses	-	Row of *OWLEquivalentClasses* metatable
DataProperty	*Column* or *Table* and 3 *Columns* for *DataProperty* domain identifier, range identifier, and value	Row of *OWLDataProperties*
DataPropertyDomain	If *DataProperty* maps to *Column*, points to *Table* owning that *Column*; If *DataProperty* maps to *Table*, points to *DataProperty* domain *Table*	Value of *foreignKey* column *dataPropertyDomain* in *OWLDataProperties* metatable
DataPropertyRange	Points to type (*SQLDataType*) of *DataProperty Column*	Value of *foreignKey* column *dataPropertyRange* in *OWLDataProperties* metatable
FunctionalDataProperty	*Column*	Value of column *functionalDataProperty* in *OWLDataProperties* metatable
DataRange	Points to *Column*'s type (*SQLDataType*) together with *SQL Check* functions	
DataType	*SQLDataType*	
Annotation	*Foreign Key* in every *Table* of corresponding class from *OWLAnnotations* metatable	Value of *Column annotationValue* in *OWLAnnotation* meatatable
EquivalentClasses	-	Row of *OWLEquivalentClasses* metatable
DisjointClasses	-	Row of *OWLDisjointClasses* metatable
DisjointUnion	-	Row of *OWLDisjointUnion* metatable

The overall Metaschema includes 33 metatables; a subset of metatables for OWL 2 classes is shown in Fig. 4.

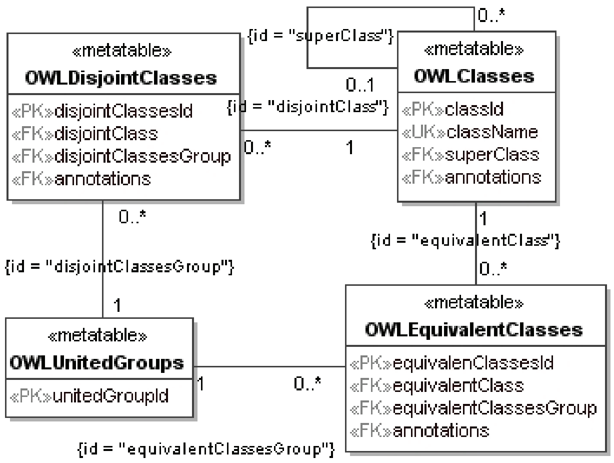

Fig. 4. Subset of OWL 2 class metatables

Transformation algorithms

Algorithm for the overall OWL2ToRDB transformation is presented in Fig. 5.

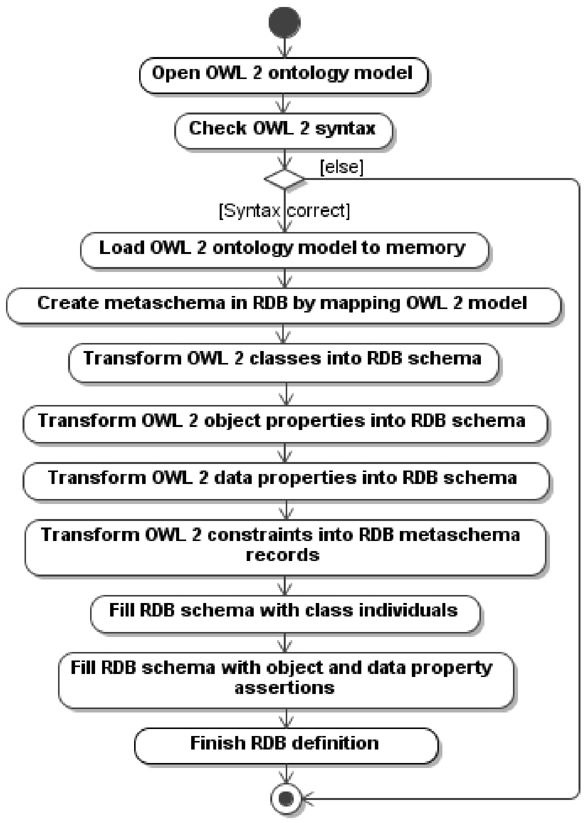
Open OWL 2 ontology model
Check OWL 2 syntax
[else]
[Syntax correct]
Load OWL 2 ontology model to memory
Create metaschema in RDB by mapping OWL 2 model
Transform OWL 2 classes into RDB schema
Transform OWL 2 object properties into RDB schema
Transform OWL 2 data properties into RDB schema
Transform OWL 2 constraints into RDB metaschema records
Fill RDB schema with class individuals
Fill RDB schema with object and data property assertions
Finish RDB definition

Fig. 5. Algorithm for transforming OWL 2 ontology into relational database

The algorithm parses an OWL 2 file and creates object-oriented model containing classes, object and data properties, axioms, restrictions and all other ontology concepts except individuals and assertions. In the next step, it creates relational OWL 2 metaschema corresponding to object-oriented ontology model. The metaschema always has the same structure that must be filled with information of the particular ontology. The detailed steps of the algorithm may be illustrated by algorithm for transforming classes (Fig. 6).

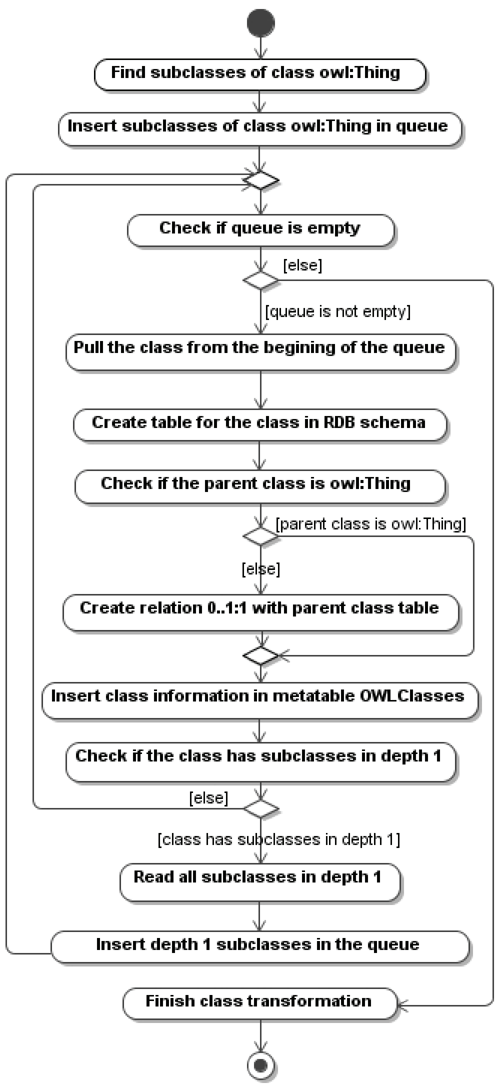
Find subclasses of class owl:Thing
Insert subclasses of class owl:Thing in queue
Check if queue is empty
[else]
[queue is not empty]
Pull the class from the begining of the queue
Create table for the class in RDB schema
Check if the parent class is owl:Thing
[parent class is owl:Thing]
[else]
Create relation 0..1:1 with parent class table
Insert class information in metatable OWLClasses
Check if the class has subclasses in depth 1
[else]
[class has subclasses in depth 1]
Read all subclasses in depth 1
Insert depth 1 subclasses in the queue
Finish class transformation

Fig. 6. Transforming ontology classes into RDB tables

The particular class information is inserted in a metaschema table *OWLClasses*. The mapping point of RDB schema and metaschema is a unique name of a particular class. In relational schema it is a name of a transformed table of a corresponding class, and in metaschema it is a value of a column *className* in a table *OWLClasses*. All other steps of OWL2ToRDB algorithm are implemented in a similar way.

The evaluation of the method

For evaluating the method, the OWL2ToRDB tool prototype was implemented with the use of Protégé OWL plug-in for OWL2 ontology management and Jena API for storing and manipulating ontology in memory. The experiment for evaluating the pursued quality criteria was performed.

For evaluating if the OWL2ToRDB transformation is fully automatic, correct and does not lose data, data types and structure of ontology, vehicle ontology was created having 10 classes, 12 object properties, 15 data properties, 46 individuals, 43 object property and 63 data property assertions, 20 annotations, 90 axioms and restrictions. After performing transformation, all these constructs were saved in RDB schema.

Methods of storing ontologies are evaluated by exploring their querying capabilities [14]. For ensuring the reversibility of OWL2ToRDB transformation and applicability of obtained RDB schema for practical applications, the prototype tool was created for extracting ontology from database and querying ontology data.

Usually, querying ontologies is performed in the following way: ontology reasoner (e.g. Pellet) reads ontology, including individuals, from a XML file, and performs queries in a computer memory. In our case, only ontology classes, their hierarchies, object and data properties, axioms and restrictions are extracted into a memory. Individuals are accessed by SQL queries obtained by converting fragments of SPARQL to SQL. Our prototype RDBToOWL2 tool creates ontology model for the reasoner, rewrites SPARQL queries and executes SQL for obtaining results. The algorithm of transforming the database into ontology is based on the previously described OWL2ToRDB transformations. Several types of SPARQL queries were performed with different amounts of OWL 2 individuals.

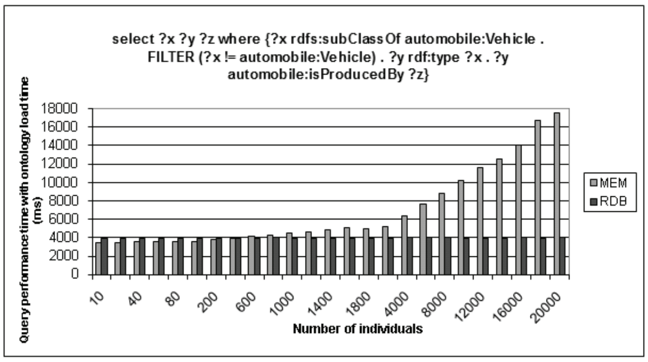

Fig. 7. Dependency of a sample query performance time from the number of individuals (with ontology load time) when all ontology in kept in memory (MEM) and database (RDB)

For all analyzed queries results were similar to presented in Fig. 7. As can be seen from this figure, query performance time is similar while the number of individuals is moderate. When this number is growing, RDB based querying performance time remains stable but memory based methods are slackening and even failing after reaching some bound that depends on a size of computer memory. For executed experiments with RDBToOWL2 tool, the query performance time in average was decreased by about 40% in comparison with memory based method, but the benefits of developed method become obvious when the number of individuals is growing.

Conclusions and future works

1. The investigation of scientific literature and practices of ontology engineering has shown that it is desirable to store large ontologies in relational databases but current methods and tools are insufficient for this purpose because they do not preserve all ontological information or do not use all advantages of relational databases.
2. Such a method should have the following features: the transformation should not lose data, data types and structure of ontology; the transformation should be fully automatic, realizable and have provable correctness; the obtained relational database should be suitable for practical database and semantic applications.
3. The desirable features may be achieved by creating a hybrid method combining direct mapping of ontology concepts to relational schema structures with using metaschema for storing ontology axioms and restrictions having no corresponding elements in relational schema.
4. Experimental investigation of the implemented OWL2ToRDB tool with the vehicle ontology has shown that the created method is capable for transforming OWL 2 concepts into RDB structures without a loss of ontology information.
5. The experiments with query processing technique that supplements OWL2ToRDB method have shown the better query performance times than memory-based ontology processing methods because it allows to extract into memory only ontology concepts and access individuals by SQL queries obtained by converting fragments of SPARQL into SQL.
6. Future works are directed for providing exhaustive experiments and industrial case studies for investigating method quality and possible improvements.

References

1. **Motik B., Patel–Schneider P. F., Parsia B.** OWL 2 Web Ontology Language Structural Specification and Functional–Style Syntax. – W3C Proposed Recommendation, 2009. – P. 134.
2. **Nenortaitė J., Butleris R.** Improving Business Rules Management through the Application of Adaptive Business Intelligence Technique // Information Technology And Control. – Kaunas: Technologija, 2009. – No. 38(1). – P. 21–28.
3. **Kalnius R., Eidukas, D.** Probability Model Quality of Informations Systems // Electronics and Electrical Engineering. – Kaunas: Technologija, 2009. – No. 4(92). – P. 13–18.
4. **Lee J., Goodwin R.** Ontology management for large–scale enterprise systems // Electronic Commerce Research and Applications, 2006. – No. 5(1). – P. 2–15.
5. **OMG**. Ontology Definition Metamodel. Version 1.0. OMG Document Number: formal/2009–05–01. – P. 334
6. **Astrova I., Korda N., Kalja A.** Storing OWL Ontologies in SQL Relational Databases // Engineering and Technology, 2007. – No. 23. – P. 167–172.
7. **Lu J., Ma L., Zhang L., Brunner J. S., Wang C., Pan, Y., Yu Y.** SOR: a practical system for ontology storage, reasoning and search // Proceedings of the 33rd International Conference on Very Large Data Bases. – Vienna, Austria, 2007. – P. 1402–1405.

8. **Konstantinou N., Spanos D. M., Nikolas M.** Ontology and database mapping: a survey of current implementations and future directions // Journal of Web Engineering, 2008. – No. 7(1). – 2008. –P. 001–024.
9. **Khalid A., Shah A. H., Qadir M. A.** OntRel: An Ontology Indexer to store OWL–DL Ontologies and its Instances // Proc. of 2009 International Conference of Soft Computing and Pattern Recognition, 2009. –P. 478–483.
10. **Ghawi R., Cullot N.** Database–to–Ontology Mapping Generation for Semantic Interoperability // VDBL'07 conference. – VLDB Endowment ACM, 2007. – P. 1–8.
11. **Vyšniauskas E., Nemuraitė L., Šukys A**. A hybrid approach for relating OWL 2 ontologies and relational databases // Perspectives in Business Informatics Research. Proceedings of the 9th international conference (BIR'2010). – Rostock, Germany, Berlin–Heidelberg–New York, Springer, 2010. – P. 86–101.
12. **Ehrig H., Ehrig K., Ermel C., Hermann F., Taentzer G**. Information preserving bidirectional model transformations // Proceedings of Fundamental Approaches to Software Engineering (FASE'2007). – Springer, Heidelberg, 2007. – P. 72–86.
13. **Stevens P**. Bidirectional model transformations in QVT: semantic issues and open questions // Software and Systems Modeling, 2010. – No. 9. – P. 7–20.
14. **OMG**. Common Warehouse Metamodel Specification. Object Management Group. OMG Document Number: pas/06–04–02, 2006. – P. 496.
15. **Bizer C., Schultz A**. The Berlin SPARQL Benchmark // International Journal on Semantic Web and Information Systems (IJSWIS), 2009. – No. 5(2). – P. 1–24.

Received 2011 05 15
Accepted after revision 2011 08 25

E. Vysniauskas, L. Nemuraite, B. Paradauskas. Hybrid Method for Storing and Querying Ontologies in Databases // Electronics and Electrical Engineering. – Kaunas: Technologija, 2011. – No. 9(115). – P. 67–72.

The goal of the paper is to present a method based on reversible, information preserving transformations and transformation algorithms between OWL 2 ontology and relational database (OWL2ToRDB). The authors propose the hybrid approach, when a part of ontology constructs is directly represented by relational database structures and another part having no direct correspondences in a relational database is stored in metadata tables. Experimental investigation of the implemented OWL2ToRDB tool has shown that the created method is capable for lossless transformations and obtained database schemas are practically and flexibly usable in information systems. When a number of ontology instances is growing, the proposed method shows better query performance time in comparison with traditional methods when all instances are processed in computer memory. Ill. 7, bibl. 15, tabl. 1 (in English; abstracts in English and Lithuanian).

E. Vyšniauskas, L. Nemuraitė, B. Paradauskas. Hibridinis metodas ontologijoms saugoti ir užklausoms vykdyti duomenų bazėse // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2011. – Nr. 9(115). – P. 67–72.

Straipsnio tikslas – pateikti metodą, apimantį abipuses, informacijos neprarandančias transformacijas ir transformavimo algoritmus, skirtus OWL 2 ontologijoms saugoti reliacinėse duomenų bazėse (OWL2ToRDB). Autoriai sukūrė hibridinį metodą, kuris dalį ontologijos konceptų tiesiogiai vaizduoja reliacinių duomenų bazių struktūromis, o kitus darinius, neturinčius tiesioginių atitikmenų reliacinėje bazėje, saugo metaduomenų lentelėse. Eksperimentinis OWL2ToRDB metodą realizuojančio įrankio tyrimas parodė, kad sukurtas metodas nepraranda ontologinės informacijos, o gautos duomenų bazės schemos yra tinkamos praktiškai taikyti informacinėse sistemose. Kai ontologijos egzempliorių skaičius didėja, šiuo metodu gaunama geresnė užklausų vykdymo trukmė nei metodais, apdorojančiais ontologijos egzempliorius kompiuterio atmintyje. Il. 7, bibl. 15, lent. 1 (anglų kalba; santraukos anglų ir lietuvių k.).