ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2006, Vol. 35, No. 3 A

TRANSFORMING ONTOLOGY REPRESENTATION FROM OWL TO RELATIONAL DATABASE

Ernestas Vysniauskas, Lina Nemuraite

Kaunas University of Technology, Department of Information Systems
Studentų st. 50-308, LT-51368 Kaunas, Lithuania

Abstract. The current work has arisen with respect to the growing importance of ontology modelling in Information Systems development. Due to emerging technologies of Semantic Web, it is desirable to use for this purpose the Web Ontology Language OWL. From the other side, the relational database technology has ensured the best facilities for storing, updating and manipulating the information of problem domain. The algorithms for transformation of domain ontology, described in OWL, to relational database are proposed. The methodology is illustrated with an example.

1. Introduction[1]

Ontology is a description of entities of a problem domain and their semantic relations. The short definition of the ontology is as the "explicit specification of conceptualization". The ontology clearly defines fundamental concepts and relationships used in particular problem domain. Ontology representations may vary:

- from simple taxonomy with minimal hierarchy of knowledge – only father/child relationships;
- to thesaurus (extended vocabulary), including words and synonyms;
- to conceptual model, having more complex relationships;
- and, finally, to logic theory, including very complex, expressive, and meaningful knowledge.

Well-formed ontology must have correct syntax and unambiguous machine-understandable interpretation adequate to its previous definition. Ontology descriptions are typically used in Semantic Web/Web2, but nowadays they find more and more adaptability in domains associated with everyday Information Systems. For ontology development, the Semantic Web languages and technologies are dedicated: Resource Description Framework RDF and schema RDFS; Web Ontology Language OWL that consists of three sublanguages – OWL Lite, OWL Description Logic (DL) and OWL Full. But, as [8] argue, the representation of ontology based on Semantic Web languages is insufficient to address semantic interoperability problems that arise in various concrete applications.

"Ontologies are attempts to more carefully define parts of the data world and to allow interactions between data held in different formats" [16]. In conceptual modelling, the Foundational Ontology is needed as domain-independent theoretical basis to guide and validate models of particular domains, as using of right modelling concepts and rules is making a great influence on the quality of Information Systems [7]. For such purpose, the transformations between conceptual models (expressed, for example, in UML) and ontological models, expressed in ontological languages (for example, OWL) are needed.

In this paper, another problem is considered that has arisen from practical needs: namely, possibilities for storing ontological information and processing this information by user applications. For this purpose, Relational Database (RDB) is a good candidate that have proven capabilities to cope with large amounts of data [18]. Methodologies for transforming Entity-relationship and Object-oriented (in nowadays, often expressed in UML) conceptual models to relational database structures, transformations between relational and XML schemas, UML models and XML schemas are well-established and implemented in CASE tools. Ontology Definition Metamodel, initiated by OMG [17], is seeking to define transformations between OWL, UML, ER and other modelling languages, where Simple Common Logic is chosen for definition of constraints. On the base of existing methodologies, there are some possible ways to relate ontological information described by ontological language, with relational schemas:

[1] The work is supported by Lithuanian State Science and Studies Foundation according to Eureka programme project „IT-Europe" (Reg. No 3473)

- Generation of standard ontology descriptions (e.g., OWL) and relational schemas from UML models maybe created using UML profile for ontology modelling [19];
- Generation of UML models from standard ontology descriptions (e.g., OWL) and relational schemas from UML models;
- Extracting or representing ontological descriptions from existing relational databases [9], [4] (unfortunately, this way does not ensure quality of ontological model).

Other scenarios for relating OWL and Relational Databases are possible, although, the direct transformation from OWL to relational schema is not defined anywhere. It is not only feasible way, but sure the fast and theoretically valid one to move ontological information to relational database and to make it accessible by different applications and systems. As OWL is built on XML, it seems worth to try transformations from XML to RDB. But OWL has more advanced features, than XML documents, namely, constraints and inference support, and these features must be preserved when mapping OWL to RDB. The only source, giving an outline of algorithm for going from OWL to relational data structures, preserving constraints (OWL2DB), is [2]. Our algorithms, presented in this paper, are created on the base of [2] ideas.

The rest of the paper is organized as follows. In Section 2, the overview of configuration domain is given, as this domain has become the subject of large amount of ontological research as well as a target of plenty practical applications. Section 3 presents the example of configuration ontology in OWL. Section 4 is devoted to explanation of transformation algorithms from OWL to RDB schema. In Section 5, relational database schema is presented for analyzed ontology example. Finally, Section 6 gives some conclusions and highlights the future work.

2. Ontology of a product configuration

One of domains often considered in association with ontology development is a product configuration system [14]. There are a lot of problems that may be solved from the configuration point of view. For example, customers often desire individual products and solutions, designed for their own needs. One of the ways to solve this problem is to let customer to design the conspicuous product by himself using virtual product configuration system. Product configuration systems or configurators are software tools that enable automation of order submissions by capturing customer requirements without human intermediaries, constructing new products and providing semantic search in supplier's database.

To do all these tasks, product configuration systems need knowledge about products, their components, various design rules and constraints. This information must be stored in a configuration system's knowledge base.

Configuration is an abstract description of system elements and their composition rules. One of the most important features of configuration management system is a semantic design of the product using the configuration knowledge base. The functioning of configuration system is based on formal configuration models. Entities of product configuration ontology are:

- `Product`
- `Product::product_atribute`
- `Product::product_component`
- `Product::necessary_component::product_component`
- `Product::product_component::inconsistent_component`
- `Product::similar_product`
- `Product::incompatible_product`, and others.

The configuration model is a strictly described (formalized) configuration ontology, or meta information about the certain class of products. In this paper, configuration models are considered on two levels – formal (logical) and relational. Formal model is described in OWL, and later transformed into relational schema, described using Data Definition Language (DDL) that may be executed for creating a relational meta database for storing relational descriptions of particular configuration model.

Product configuration systems or configurators are considered as being among the most successful applications of artificial intelligence. They facilitate entering of information about products and provide semantic search in the particular product class by any characteristic proper to that class. In general, configurator implements an interface between a supplier and his customer over the internet. Its main task is to support customers in the self-configuration of their products according to particular individual requirements [12]. For example, customers may be provided with the possibility to alter a basic product and also to graphically visualize the effects of these changes.

Configurators support the configuration process. This process may by handled by an agent right understanding the customer needs in order to create a complete description of the product variant that suits his or her individual requirements. Given a set of customer requirements and the description of a family of products, the task of configuration is to find a valid and completely specified product instance among all of the alternatives that the generic structure describes.

In the technical literature, there are many definitions of product configurators. The artificial intelligence community generally addresses them as the software tools. For example, in [15] product configurator is defined as a "…software with logic capabilities to create, maintain and use electronic product models that allow to complete definition of all possible product options and variation combinations,

with a minimum of data entries and maintenance". The main technical component of the configurator is the knowledge base, which consists of two subcomponents, namely, the database, and the configuration logic.

Whereas the database contains the total set of component types and their instances, the configuration logic specifies the constraints existing between the different components to allow only valid and completely structured product variants.

Within a product portfolio, there is a lot of knowledge and rules about the products and their relationships to one another, yet this is hidden in the data without which it would be hardly possible to extract and use product configurations in supporting tools. Hence typical applications upon product catalogue data (supply chain, marketing, research & development, B2C e-commerce) are implemented in a costly hard-coded approach in which catalogue data are imported and manipulated as they suit the particular context of use.

Due to the structure of product catalogue data, in which characteristics and relationships are not explicitly expressed by some sort of standardized terminology, there is a high cost involved in developing and maintaining this data, reinterpreting it for different contexts of use, or for creating new catalogue structures, working with them in collaborative environments, and sharing them between different participants, each of which may have a different understanding of their purpose and meaning [20].

As a result, an ontology usage is a viable approach to improving the development of product catalogues and their maintenance over the entire product lifecycle, as they offer the consistent terminology and the possibility to generate different views for different contexts for the same products [13], [21].

3. Modelling the configuration ontology

Ontological model simplifies the development and support of the knowledge base of business domain [6], [10], [3], [11] as it defines well-established terminology, and enables the generation of different views of the same object according to existing context. The preferred language for ontology modelling is OWL, as it has the most possibilities for expressing semantics in comparison with RDF and RDFS (Table 1). The exclusive feature of ontological languages is the capability to express constraints and individuals (objects) of problem domain.

In order to develop the ontology of some domain, one must define the fundamental concepts, relationships, constraints and individuals of that domain. [15] defines the configuration ontology in such a way:

- A set of components (products or services), such, that these components may be described by a set of properties and ports connecting them to other components.
- Constraints for each port describing the components that may be connected to that port, and other structural constraints.
- User requirements with the description of the desired configuration; and, possibly, some criteria for optimizing the solution.

Table 1. The capabilities of ontological languages

Concepts	RDF(S)	OWL
Bounded lists	X	X
Cardinality constraints		X
Class definitions		X
Data types	X	X
Defined classes		X
Enumerations		X
Equivalence		X
Extensions	X	X
Formal semantics	X	X
Inheritance	X	X
Inference		X
Constraints		X
Reification	X	X

Therefore, the main concepts that seem to be the candidates to appear in the configuration ontology are components, ports, connections, properties or attributes, constraints, and user requirements.

In Figure 1, the configuration ontology schema is presented using these concepts.

4. The process of development of a knowledge base

There are some alternatives, how to create a knowledge base. For example, Felferning et al. [1] has proposed the following process for a configuration ontology creation: firstly, a model of the configuration object is developed using a modelling language, e.g. UML with OCL constraints. After testing the syntax of concepts, model is unambiguously transformed to logical expressions, which are used by configuration engine generating domain configurations. Finally, knowledge base must be checked by expert of business domain using testing examples.

The way, considered in our work, is presented in Figure 2 [5]. Business analyst gives specifications of configuration domain, expressed in natural language, to knowledge engineer (designer of knowledge-based Information System). Designer, using some modelling tool (for example, Protege, Altova Semantic Works, or other), creates formal ontology for required domain. After that, the ontology model is transformed into DDL script with all constraints using special transforming tool. The DDL script is used to save the knowledge descriptions into a relational database. Client applications may access this relational database and render results to users (Figure 2).

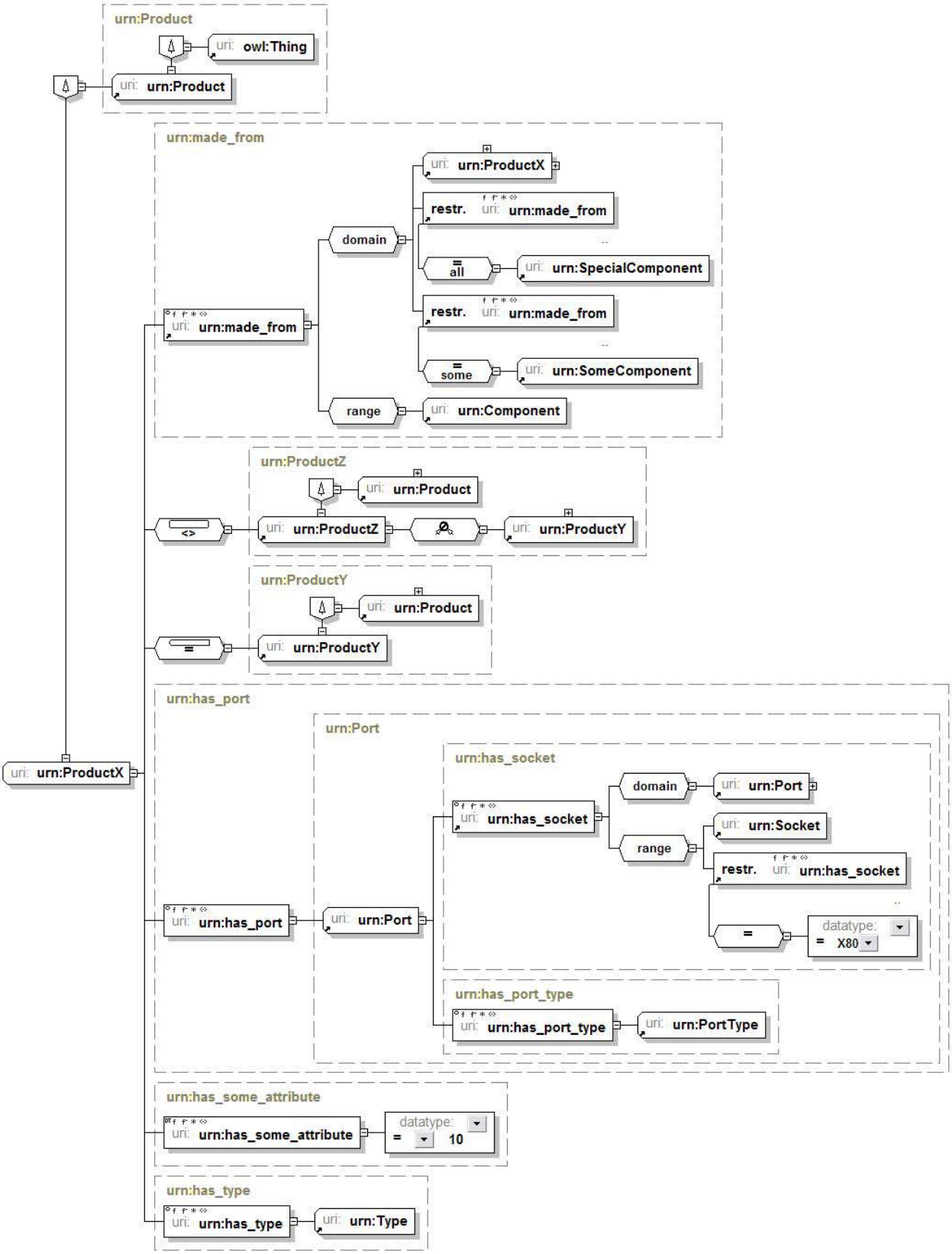

Figure 1. An example of configuration ontology

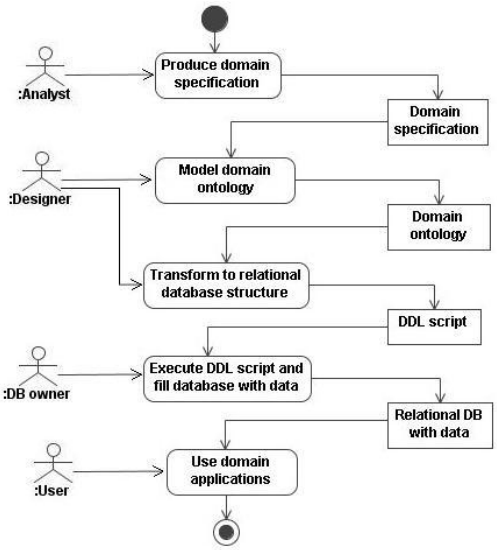

Figure 2. The proposed process for developing and transforming domain ontology to relational database

5. Transformation of ontology to relational knowledge base

One of the steps of knowledge base constructing process is to transform domain ontology into relational database. For this purpose, the transformation algorithm was created, which parses OWL documents and generates DDL scripts, containing database descriptions with included domain ontology constraints.

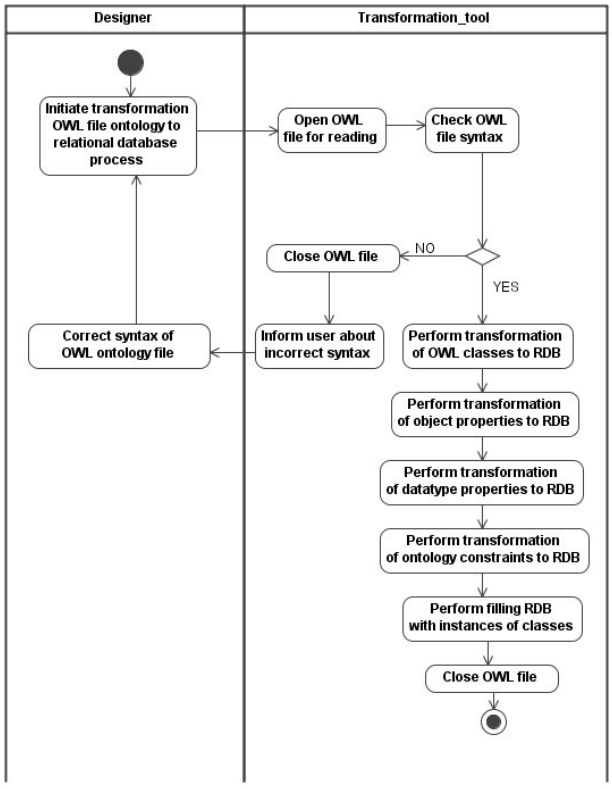

Figure 3. Algorithm for transformation of ontology into relational database

The process of transforming ontology into relational database is shown in Figure 3.

Designer initiates transformation of domain ontology described in OWL file into relational database. Transformation tool opens OWL file for reading and checks the correctness of OWL syntax. If the file syntax does not match OWL notation, system closes it and informs designer about errors. If syntax of the file matches OWL notation, transformation tool executes steps of transforming ontology to relational database. At first, system transforms ontology classes, the next steps are transformations of object and data type properties, constraints and, finally, database is filled with instances of the classes. At the end of successful transformation system closes the file.

The more detailed steps of the algorithm are presented in Figures 4, 5, and 6.

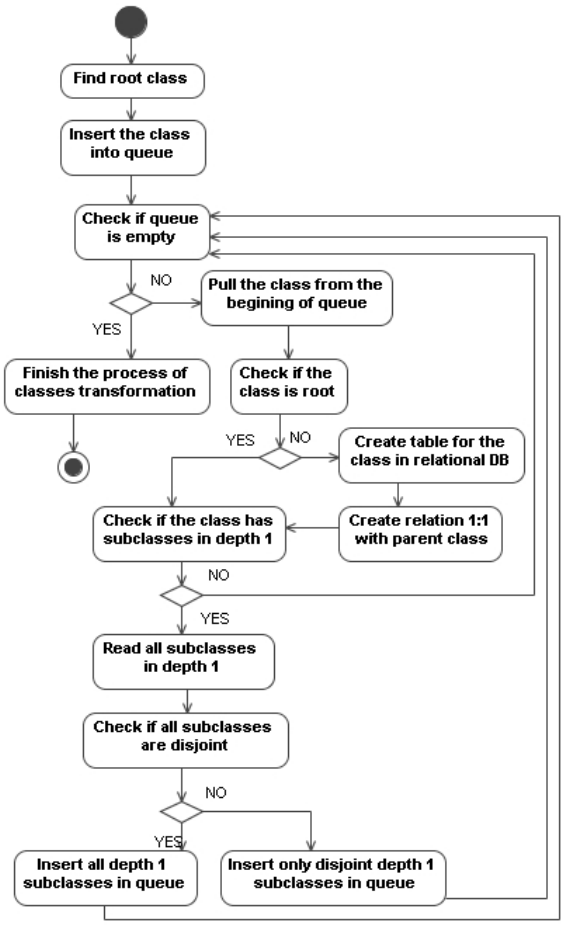

Figure 4. Transformation of ontology classes into relational database tables

5.1. Transformation of ontology classes into RDB tables

During the process of transforming domain ontology into relational database, at the first step the ontology classes are transformed into relational database tables.

A class is the most basic concept in ontology. Class in OWL syntax is defined in this way:

```
<owl:Class rdf:ID="Winery"/>
<owl:Class rdf:ID="Region"/>
<owl:Class rdf:ID="ConsumableThing"/>
```

The fundamental taxonomic constructor for classes is "`rdfs:subClassOf`". It relates a more specific class to a more generic class. If "X" is a subclass of "Y", then every instance of "X" is also an instance of "Y". The "`rdfs:subClassOf`" relation is transitive. If "X" is a subclass of "Y" and "Y" a subclass of "Z", then "X" is a subclass of "Z". Example of OWL syntax, defining class hierarchical relations:

```
<owl:Class rdf:ID="PotableLiquid">
  <rdfs:subClassOf
rdf:resource="#ConsumableThing" />
  ...
</owl:Class>
```

Every ontology class is implicitly a subclass of "`owl:Thing`" class.

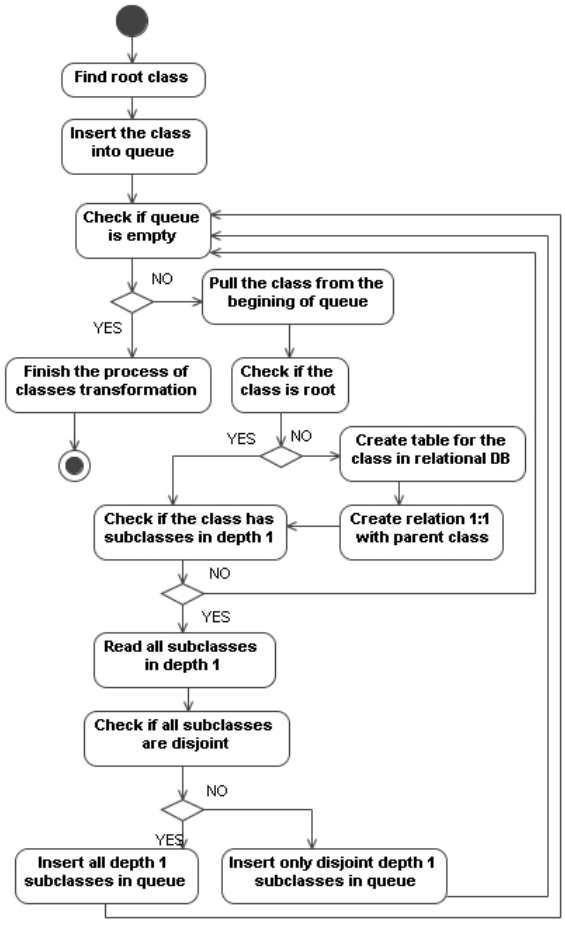

Figure 4. Transformation of ontology classes into relational database tables

An algorithm, transforming ontology classes into relational database tables, is shown in Figure 4. It uses breadth-first search. Breadth-first search algorithm goes across one hierarchical level of ontology class tree. Thus, root classes are parsed first, then their subclasses, and so on. The one table in relational database is created for every class in ontology with one-to-one relations between classes and their subclasses. Because algorithm uses breadth-first search, it is guaranteed that, when some subclass is being created, its parent class in hierarchy has already been created.

5.2. Transformation of ontology object-properties into RDB algorithm

When OWL classes are mapped to tables object-properties may be transformed into RDB relations.

Object property is a relation between instances of two classes. When we define a property, there is a number of ways to restrict the relation: the domain and range can be specified; the property can be defined to be a specialization (sub-property) of an existing property, and so on.

Example of OWL syntax, defining object property:

```
<owl:ObjectProperty rdf:ID="madeFromGrape">
  <rdfs:domain rdf:resource="#Wine"/>
  <rdfs:range rdf:resource="#WineGrape"/>
</owl:ObjectProperty>
```

Example of OWL syntax defining specialization of object-properties:

```
<owl:Class rdf:ID="WineDescriptor" />
<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf              rdf:resource=
      "#WineDescriptor" />
  ...
</owl:Class>
<owl:ObjectProperty                   rdf:ID=
      "hasWineDescriptor">
  <rdfs:domain rdf:resource="#Wine" />
  <rdfs:range                   rdf:resource=
      "#WineDescriptor" />
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasColor">
<rdfs:subPropertyOf             rdf:resource=
      "#hasWineDescriptor" />
  <rdfs:range rdf:resource="#WineColor" />
  ...
  </owl:ObjectProperty>
```

An algorithm, transforming ontology object-properties into relational database relations between tables is shown in Figure 5. This algorithm also uses breadth-first search. Firstly, it parses properties that do not have properties of the higher hierarchical level; in the next, it parses their sub-properties, and so on. Depending on the local cardinality of some class property, one-to-many or many-to-many relations between tables of classes are created. In a case of many-to-many relation, an intermediate table is created.

When the designer initiates transformation of domain ontology described in OWL file into relational database, transformation tool opens OWL file for reading and checks the correctness of OWL syntax. If the file syntax does not match OWL notation, system closes it and informs designer about errors.

If syntax of the file matches OWL notation, transformation tool executes steps of transforming

ontology to relational database. Primarily, system transforms ontology classes, next steps are transformation of object and data type properties, constraints and finally database is filled with instances of the classes. At the end of successful transformation system closes the file, replacing one many-to-many relation with two one-to-many relations.

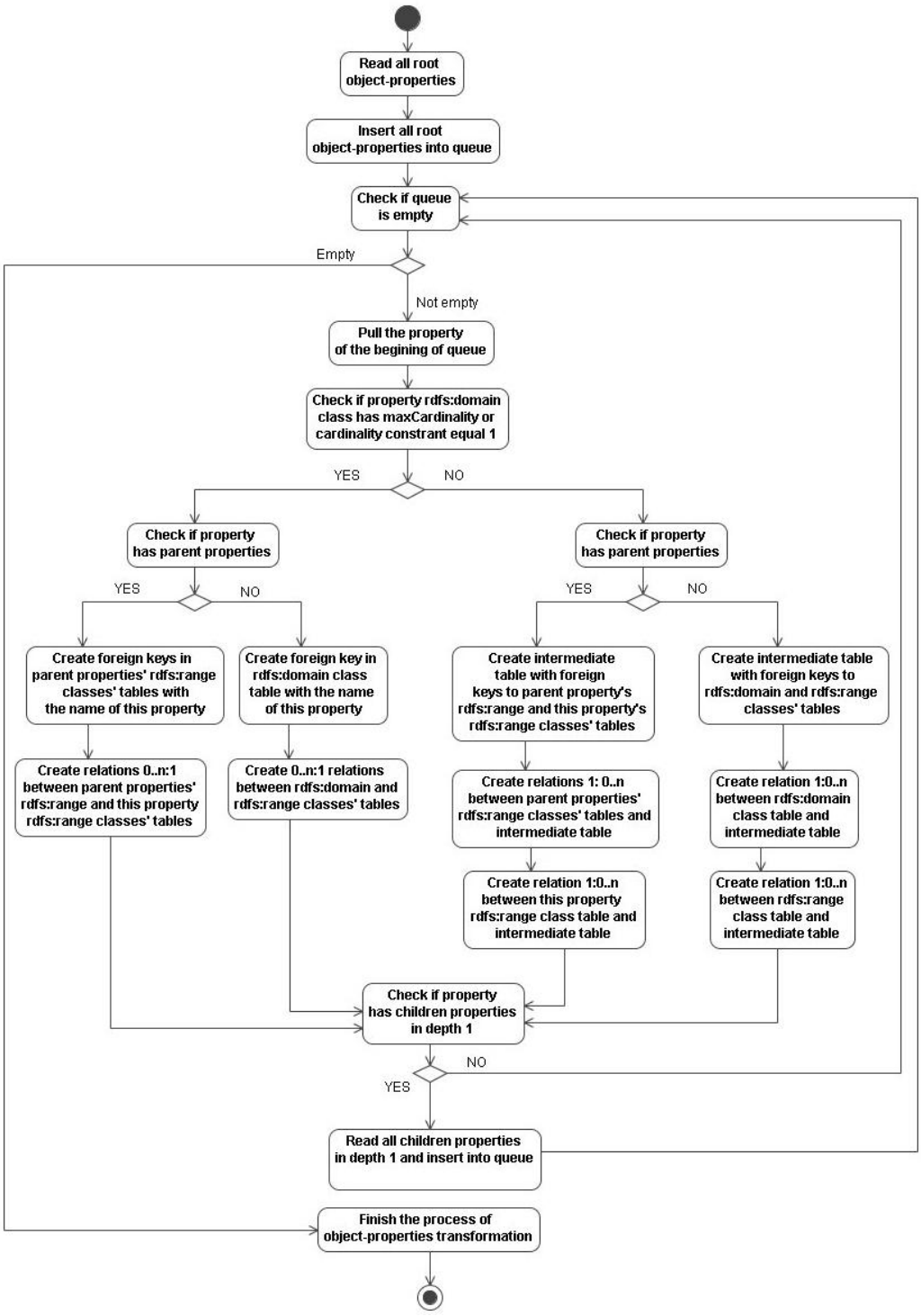

Figure 5. Transformation of object properties into relational database relations

5.3. Transformation of ontology data type properties into RDB algorithm

In the process of transforming domain ontology into relational database, after transformation of object-properties into RDB relations between class tables, data type-properties are transformed into RDB data columns.

Data type properties are relations between instances of classes and RDF literals, and XML Schema data types. Example of OWL syntax, defining data type property:

```
<owl:Class rdf:ID="VintageYear" />

<owl:DatatypeProperty rdf:ID="yearValue">
  <rdfs:domain rdf:resource="#VintageYear" />
  <rdfs:range                    rdf:resource=
      "&xsd;positiveInteger"/>
</owl:DatatypeProperty>
```

An algorithm, transforming ontology data type properties into relational database data columns of tables is shown in Figure 6.

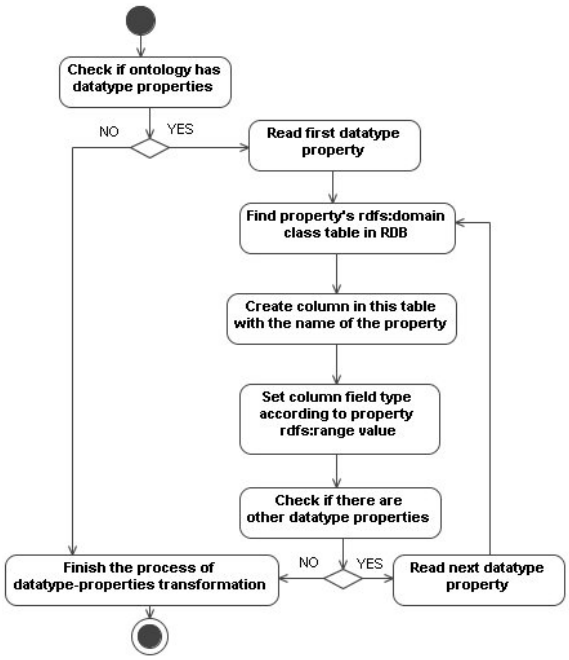

Figure 6. Transformation of data type properties into relational database table columns

The algorithm, transforming ontology data type properties into relational database columns of tables, searches and parses all data type properties in series. According to `rdfs:domain` value it finds database table and creates data column with the name of the property. Column data type is a set according to property `rdfs:range` value. When all data type properties are parsed, the process of data type properties transformation is finished.

5.4. Transformation of ontology constraints into RDB algorithm

In the process of transforming domain ontology into relational database, after transformation of data type-properties into RDB data columns, ontology constraints are transformed into RDB metadata tables.

In addition to designating property characteristics, it is possible to further constrain the range of a property in specific contexts in a variety of ways. It is done with property restrictions. The various forms of restrictions can only be used within the context of an `owl:Restriction`. The `owl:onProperty` element indicates the restricted property. Example of OWL syntax, defining a restriction of class property:

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource=
     "&food; PotableLiquid"/>
  <rdfs:subClassOf>
<owl:Restriction>
  <owl:onProperty rdf:resource=
     "#madeFromGrapes"/>
  <owl:minCardinality
     rdf:datatype=
     "&xsd;nonNegativeInteger">1
  </owl:minCardinality>
 </owl:Restriction>
</rdfs:subClassOf>
...
</owl:Class>
```

An algorithm, transforming ontology constraints into relational database metadata tables is shown in Figure 7.

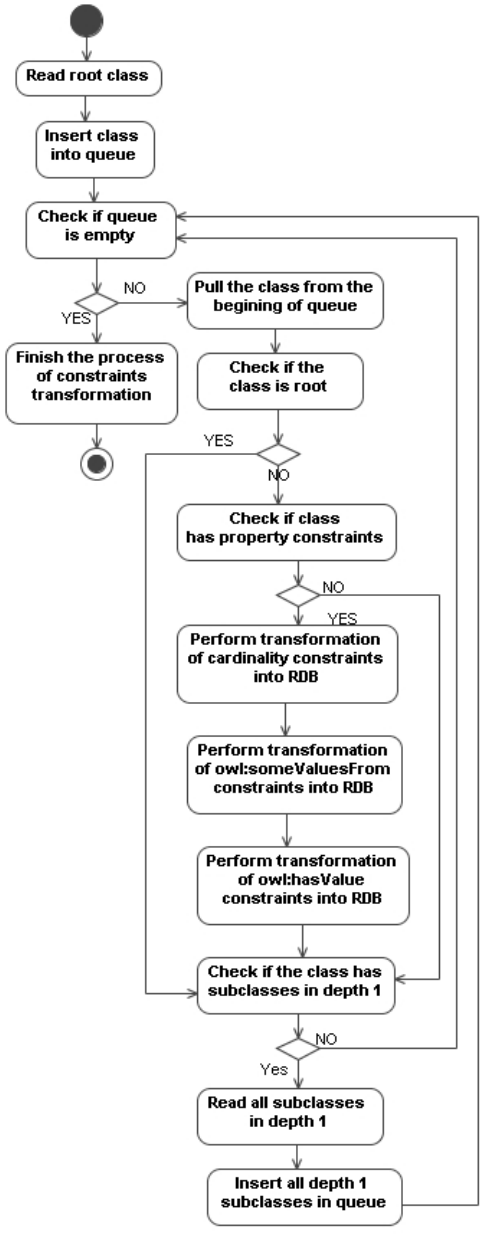

Figure 7. Transformation of ontology constraints into relational database metadata tables

Algorithm, transforming ontology constraints into relational database metadata tables performs breadth-first search. Firstly it parses constraints of root class properties, then constraints on properties of its sub-classes, and so on. If a class has constraints of the property, the algorithm performs transformation of constraints into metadata tables. All constraints of particular class are parsed in series. Every type of constraint has its own table with the name of the constraint type.

At the last stage of transforming domain ontology into relational database, transformation tool inserts all instances of classes into created database.

5.5. An example of transforming ontology into RDB

We have standard wine ontology shown in Figure 8. Wine ontology is often used as an example when we talk about ontology, because it is simple and understandable for everybody. A wine itself, as a potable liquid, is not a configurable thing; however the wine as a product or a description of the product is fully configurable, because it has many different features, such as wine grapes or vintage year, restrictions, such as sugary and other properties. We can create new wine descriptions or perform the semantic search in the existing ones, so wine is the fully configurable product from this viewpoint. Using OWL2DB algorithm to transform the wine ontology presented on Figure 8 into relational database we obtain the resulting schema presented on Figure 9.

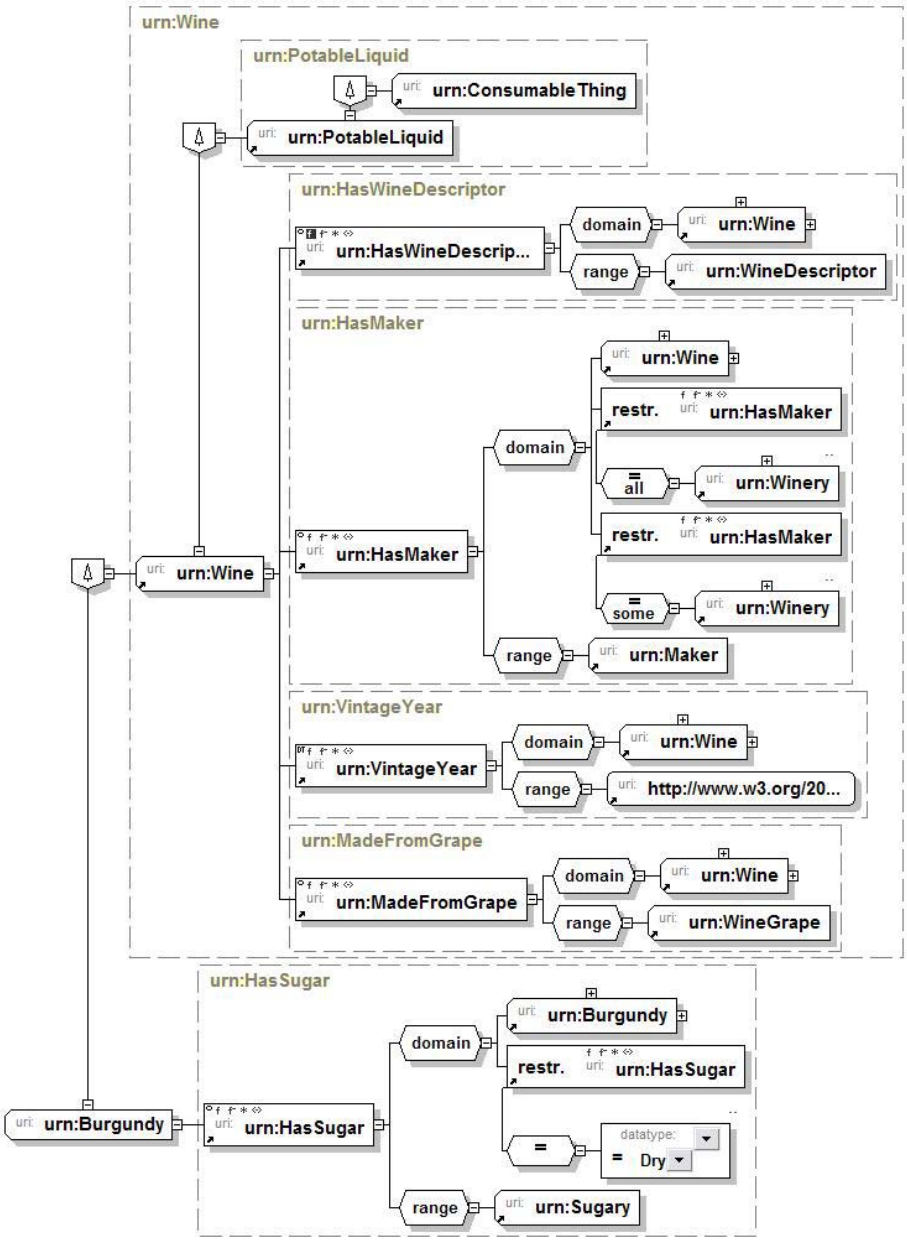

Figure 8. Example of wine ontology

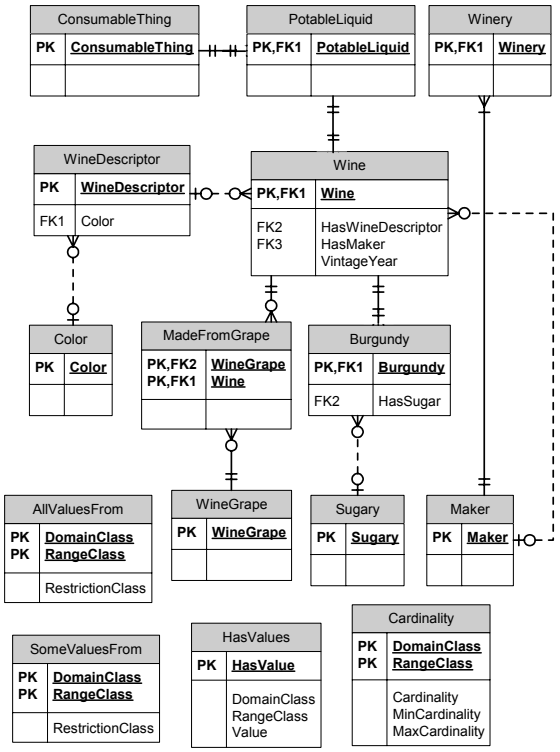

Figure 9. Example of wine ontology transformed into relational database

Meta data tables have been filled with this metadata:

Table „AllValuesFrom“

DomainClass	RangeClass	RestrictionClass
Wine	Maker	Winery

Table „SomeValuesFrom“

DomainClass	RangeClass	RestrictionClass
Wine	Maker	Winery

Table „HasValue“

HasValue	DomainClass	RangeClass	Value
1	Burgundy	Sugary	Dry

Table „Cardinality“

Domain Class	Range Class	Cardi nality	Min Cardinality	Max Cardinality
Wine	Wine Descriptor	1	Null	Null

After transformation process we have got eleven data tables and four metadata tables (Figure 9). Properties, such as “HasWineDescriptor” are transformed into foreign keys indicating to parent tables. Special OWL constraints, such as “AllValuesFrom” are kept in metadata tables. Metadata tables are filled with data according to the algorithm, described previously in the article. This minimal database structure can be used by applications, performing semantic search of special wines in general set of wines.

6. Conclusions

Ontological descriptions are gaining more and more popularity as a perspective way to enhance Information Systems in different problem domains, especially for such complex ones as configuration systems are. In this paper, we have analyzed the process how ontology of a particular domain described in OWL may be transformed and stored in a relational database. Summarizing we can make some conclusions:

- For large ontological descriptions, it is desirable to store ontological information in relational databases, although there is a lack of algorithms suitable to transform ontology concepts to RDB schema.
- The algorithm was created for transforming ontology, represented in OWL, to RDB schema. According this algorithm, ontology classes are mapped to relational tables, properties to relations and attributes, and constraints – to metadata. Such an algorithm was not created before, although it is partially based on the OWL2DB approach.
- The proposed algorithm is capable to transform all OWL Lite and part of OWL DL syntax. The further expansion of the algorithm to cover more capabilities of OWL should be based on the same principles.
- The algorithm was tested for transforming ontology examples from product configuration domain. The current version of algorithm works without a loss of information though possibilities to represent more advanced OWL features require for thorough investigation.
- A tool, performing transformations, may be implemented as add-in for ontology development tool (e.g. Protege), or as independent software, capable to import OWL documents, and to export generated DDL scripts.

References

[1] **A. Felfernig, G. Friedrich, D. Jannach**. UML as domain specific language for the construction of knowledge-based configuration systems. *International Journal of Software Engineering and Knowledge Engineering. Vol*.10, *No*.4. 2000.

[2] **A. Gali, C.X. Chen, K.T. Claypool, R. Uceda-Sosa.** From Ontology to Relational Databases. *Shan Wang et all (Eds.): Conceptual Modeling for Advanced Application Domains, LNCS Vol*.3289, 2005, 278-289.

[3] **B. Parsia, E. Sirin, A. Kalyanpur.** Debugging OWL ontologies. *The 14th international world wide web conference* (www2005), *Chiba, Japan, May* 2005. *http://www.mindswap.org/papers/ debuggingOWL.pdf*.

[4] **C. Perez de Laborda, S. Conrad.** Relational.OWL – a data and schema representation format based on OWL. *Hartmann, Sven and Stumptner, Markus (Eds.): Second Asia-Pacific Conference on Conceptual Modelling* (APCCM2005), *ACS, Australia, Vol*.43, 2004, 89-96.

[5] **E. Vysniauskas, L. Nemuraite.** The ontological modeling of product configuration knowledge base. *Tarpuniversitetine doktorantu magistrantu konferencija "Informacines technologijos* 2006"IT *Kaunas*, 2006, 133-138.

[6] **G. Antoniou, F. van Harmelen.** Web Ontology Language: OWL. *S. Staab and R. Studer (Eds.): Handbook on Ontologies in Information Systems, Springer-Verlag*, 2003.

[7] **G. Guizzardi, G. Wagner, N. Guarino, M. Sinderen.** An Ontologically Well-Founded Profile for UML Conceptual Models. *A .Person and J. Stirna* (*Eds*.): CAISE 2004, LNCS 3084, 2004, 112-126.

[8] **G. Guizzardi.** The role of Foundational Ontologies for Conceptual Modelling and Domain Ontology Representation. *O. Vasilecas et all (Eds.): Proceedings of* 2006 *Seventh International Baltic Conference on Databases and Information systems* (*Baltic* DB&IS 2006), *July* 3-6, *Vilnius, Lithuania*, 17-25, 2006.

[9] **I. Astrova.** Towards the Semantic Web – An Approach to Reverse Engineering of Relational Databases to Ontologies. *Advances in Databases and Information Systems: proceedings of the* 9*th East-European Conference, ADBIS* 2005, *Tallin, September* 12-15, 2005. - ISBN 9985-59-545-9. - *Tallin*, 2005, 111-122.

[10] **I. Horrocks, P.F. Patel-Schneider, F. van Harmelen**. From SHIQ and RDF to OWL: the making of a web ontology language. *Journal of Web Semantics*, 1(1), 2003, 7–26.

[11] **J.Z. Pan, I. Horrocks.** OWL-Eu: Adding Customised Datatypes into OWL. *Journal of Web Semantics, Vol*. 4(1), 2005, 1-20.

[12] **L. Hotz, T. Krebs.** Configuration – State of the art and New Challenges. *L. Hots, T. Krebs (Eds.) Proc. of* 17*th Workshop Plannen, Scheduling and Configurieren, Entwerfen*, (PUK2003) – KI 2003 *Workshop*, 2003, 145-157.

[13] **L. Hotz, T. Krebs**. Supporting the product derivation process with a knowledge-based approach. *L. Hots, T. Krebs (Eds.) Proc. of* 17*th Workshop Plannen, Scheduling and Configurieren, Entwerfen*, (PUK2003) – KI 2003 *Workshop*, 2003, 24-29.

[14] **L. Hvam.** A Multi-perspective Approach for the Design of Product Configuration Systems – an Evaluation of Industry Applications. *International Conference of Economic, Technical and organizational aspects of Product Configuration Systems, Technical University of Denmark, June* 28-29*th*, 2004, 1-12.

[15] **N. Pena, E. Garcia, J.M. Lazaro.** Configuration Ontology & Multi-product Configuration Tool (I). *Ontology-Based ELectronic Integration of CompleX Products and Value Chains IST Project* IST-2001-33144 OBELIX, 2003.

[16] **N. Shadbolt, T. Berners-Lee, W. Hall.** The Semantic Web revisited. *IEEE Intelligent Systems*, 2006, 96-101.

[17] Ontology Definition Metamodel. *Preliminary Revised Submission to OMG RFP ad*/2003-03-40, 18 *August*, 2004.

[18] **R. Goodwin, J. Lee, G.A. Stanoi, M.I. Leveraging.** Relational Database Systems for Large-Scale Ontology Management. *CIDR Conference*, 2005. *http://www.alphaworks. ibm.com /topics/semantics*.

[19] **S. Brockmans, P. Haase, P. Hitzler.** A Metamodel and UML Profile for Rule-extended OWL DL Ontologies. *Y. Sure, J. Domingue (Eds): The Semantic Web: Research and Applications*: 3*rd European Semantic Web Conference, ESWC* 2006 *Budva, Montenegro, June* 11-14, 2006 *Proceedings, LNCS, Vol*. 4011, 2006, 303-316.

[20] **S. Decker.** Semantic Web and Databases: Relationships and some Open Problems. *Proceedings of the NSF-EU Workshop on Database and Information Systems: Research for Semantic Web and Enterprises, April* 3 - 5, *Amicalola Falls and State Park, Georgia*, 6-14

[21] **T. Krebs, L. Hotz, C. Ranze, G. Vehring**. Towards evolving configuration models. *Hots, L., Krebs, T. (Eds.) Proc. of* 17*th Workshop Plannen, Scheduling and Configurieren, Entwerfen*, (PUK2003) – KI 2003 *Workshop*, 2003, 123-134.

Received August 2006.