SBVR Based Representation of SPARQL Queries and SWRL Rules for Analyzing Semantic Relations

Algirdas Sukys, Lina Nemuraite, Bronius Paradauskas, Edvinas Sinkevicius
Kaunas University of Technology
Kaunas, Lithuania
sukys.algirdas@gmail.com, lina.nemuraite@ktu.lt, bronius.paradauskas@ktu.lt, edvis.s@gmail.com

***Abstract*—Analyzing semantic relations is a cumbersome task because these relations often are distributed over different information sources and hidden in existing relational database structures. Even with Semantic Web ontologies describing semantic relations we need to explore them on the deep technological level that is not friendly for business users. Semantics of Business Vocabulary and Business Rules (SBVR) gives a possibility of representing OWL 2 ontologies, SWRL rules and SPARQL queries using concepts and semantic formulations expressed in SBVR Structured English language understandable for business users. We suggest formulating derivation rules and queries for analyzing semantic relations as SBVR rules and questions, and transforming them into SWRL and SPARQL.**

Keywords-semantic relations; SBVR; SPARQL; OWL 2; SWRL.

I. INTRODUCTION

More and more business information becomes available in form of ontologies that are accessible to users by dedicated query languages as SPARQL [24][32]. However, users prefer querying using sentences in natural language [10]. Query languages are limited, therefore complex parsing and validating means are needed for ensuring flexibility of queries that should allow using synonyms and synonymous forms, and asking in various ways. Ontology reasoners could help to query using expressions that are not directly defined in ontology but could be derived on the base of existing formulations.

Semantics of Business Vocabulary and Business Rules (SBVR) is the OMG metamodel that defines the vocabulary and rules for describing the business semantics – business concepts, business facts, and business rules using SBVR Structured English (SSE) or other Controlled Natural Language [21]. The SSE language cannot represent all possible constructions of natural English language, however, it is understandable to business and information system experts, and is interpretable by computers. SBVR directly models business meaning and can support many controlled languages; consequently, it can be seen as an interface between real natural languages and business software systems [11].

SBVR presents a high-powered metamodel but a lot of work should be done for introducing SBVR based business semantics management into the enterprise architecture. SBVR is capable to represent the human understandable semantics beyond Web Ontology Language (OWL 2), Semantic Web Rules (SWRL), OWL for Web Services (OWLS), Web Service Modeling Ontology (WSMO), and other ontological languages that were acknowledged as most expressive semantic representations before SBVR. SBVR directly models meaning and offers possibility to relate business semantics to software models and implementations; however, such a relation could be made through multiple transformations between various languages and architectural layers.

Despite the interest from the researchers and business side, until now only part of SBVR was used. SBVR questions provide a capability of querying business models and their implementations but they attained only little attention in the SBVR specification and related research. Kriechhammer in 2006 has noticed about possibilities of SBVR questions [12] for business people to query systems for business modeling without the support of programmers. To our knowledge, no further research in that direction was done. In [29], we applied the idea and presented the initial methodology for transforming SBVR questions into SPARQL.

In the current paper, we analyze further possibilities of querying business systems by supplementing previously described querying capabilities with the use of SWRL derivation rules together with OWL 2. We concentrate on querying some types of semantic relations: reflexive relations causing hierarchical links between individual concepts, and n-ary relations that cannot be directly represented in ontology and, consequently, are problematic for querying with SPARQL if we want to obtain SPARQL queries from SBVR questions acceptable for human.

The rest of the paper is organized as follows. Section 2 presents related works. Section 3 presents SBVR questions, corresponding SWRL rules and SPARQL queries for hierarchical relations, Section 4 – for relation properties allowing defining n-ary relations in OWL 2. Section 5 draws conclusions and outlines the future work.

II. RELATED WORK

SBVR metamodel and XMI schema may be used for developing software tools for managing business vocabularies as well as for automating development of software for managing business on the base of business semantics, i.e., in the way different of previously existed approaches, e.g., [18][22]. SBVR business vocabularies are

 ISBN: 978-1-61208-160-1

transformable into UML&OCL [3][17] and vice versa [2]; BPMN [1], RDB schemas [16], OWL [5], Web services [6], [9] etc. Besides automating the development of software models and code [14][15], SBVR Structured English may serve for creating semantic specifications of legacy information resources, integrating these resources, implementing contextualized and multilingual information systems, etc. The power of SBVR is disclosed by the fact that SBVR specification itself is formally written in SSE [13].

For practical usage, SBVR suffers from various limitations and the lack of tools as editors, validation mechanisms and inference engines. It would be desirable having the larger collection of data types and patterns for constructs needed for expressing arithmetic operations, data and time, past and future events, and similar. Spreeuwenberg and Anderson notice more deficiencies of SBVR: lack of inference; lack of references (rules should be stated in one sentence); necessity to introduce concepts before referencing to them; impossibility to express directives, etc. [26][27]. Individual researchers and their groups have proposed various SBVR extensions, but these extensions have not resulted in a new version of SBVR specification yet.

Realizing the idea of querying business in SBVR requires a creation of a whole infrastructure including tools for authoring SBVR business vocabularies and rules, transforming them into various software models and code, including OWL, SQL, Web Services, business process execution languages, and so on. Several EU projects are devoted for this purpose: OPAALS (2006–2010, generating Web services and data models from SBVR specifications [16][25]), ONTORULE (2009–2012, aiming at integrating knowledge and technologies needed for extracting ontologies and business rules from various documents, including natural language texts; managing them and implementing in software systems). The commercial tool suite for Business Semantics Management Collibra presents capabilities for authoring SBVR vocabularies and rules, generate ontologies and various models of information systems.

In our previous work, we focused on generating UML&OCL models from SBVR specifications. We proposed the methodology for specifying information system requirements on the base of SBVR business vocabularies and business rules related with business process models, and implemented the prototype of tool VeTIS supporting that methodology [3][17]. VeTIS tool recognizes SBVR concepts (object types, roles, fact types, fact type roles, individual concepts) and business rules (various kinds of semantic formulations) that make foundations for conceptualizing business and correspond to knowledge and metaknowledge level [19]. For representing SBVR questions, the extension of the VeTIS tool was needed for including business facts – instances of fact types (ground facts) and propositions (instances of complex semantic formulations based on several fact types) comprising the bottom knowledge level – fact level [20]). Representing business facts is crucial for implementing the semantic enterprise where you are able not only tracing business requirements to their implementations in software, but also accessing business data using a single language and terminology. In [29], we considered the way of transforming SBVR questions into SPARQL queries that can be executed against OWL 2 ontologies obtained from SBVR vocabularies and business rules.

SBVR questions are based on logical projections and are much more comfortable for business people than various query languages that are platform-specific and intended for IT specialists. We can apply several slightly different methodologies for querying: to derive relations using SWRL rules and OWL 2 reasoning tools, and querying using asserted as well as inferred individuals and properties, or formulating direct SPARQL queries. Also, we can store ontology instances in a relational database and apply gradual querying methodology by formulating part of a query in SPARQL and retrieve instances from a relational database using SQL [30].

Here, we will analyze how to translate SBVR questions and business rules into corresponding SWRL and SPARQL expressions. We will concentrate on two aspects: querying facts based on hierarchical and n-ary relations. Taxonomy of semantic relations analyzed by Chaffin [4] and Storey [28] includes seven main relation types: inclusion (class inclusion, meronymic, spatial), possession, attachment, attribution, antonyms, synonyms, and cases. Some meronymic, possession, and attachment relations can arise between objects of the same type. Typical reflexive relations are kinship relations that can comprise complex trees and forests; their analysis requires recursive queries.

N-ary relations have no problems for representing them in UML or SBVR, but they cannot be directly represented in ontology. There are solutions proposed by the W3C for representing n-ary relations in OWL [31] but they are based on the creation of new object types that often seem unnatural for formulating related queries close to natural language. We agree with Hoekstra who argues that the disallowance of n-ary relations is rather an advantage than the drawback of the OWL 2 because n-ary relations can be expressed by binary ones [8]; the presence of n-ary relations demonstrates that the meaning of the corresponding associations is not well-understood. Furthermore, if n-ary relations are un-ordered the semantics of relation can be lost because we may miss the subject of the sentence.

In SBVR and UML, fact type roles and association ends are ordered. However, for keeping the required order of fact type roles in SBVR, one should respectively formulate fact types; in UML, the order of n-ary association ends is set by the order of introducing these ends into a model – this is not always understood be modelers. Our proposed way of representing n-ary relations by binary ones in OWL 2 would be possible if we could define properties for relations (properties of OWL 2 object properties). Then we could distinguish fact type roles of subject and object as roles of a binary relation, and the remaining fact type roles as properties of that relation. In Section IV, we show how to define the SBVR fact types with synonymous forms that allow formulating SSE queries, acceptable for humans, and how to represent these queries in SPARQL by using OWL 2

 ISBN: 978-1-61208-160-1

punning for defining classes and object properties having same IRIs.

III. ANALYZING HIERARCHICAL RELATIONS BETWEEN INDIVIDUAL CONCEPTS

Reflexive relations arise between objects of the same type playing different roles in the relation. For example, in SBVR terminology, fact type "father *has* son" represents associative fact type where fact type roles are played by different persons that are instances of object type "person" (here and in the following, we will use SBVR style "term" for terms, representing noun concepts; *verbs*, representing fact type symbols; "Names" for individuals, and "keywords" for keywords, articles etc making SSE sentences more understandable). We take a genealogy tree as an example of reflexive relations. OWL 2 ontology of a genealogy tree is presented in Figure 1 in UML notation where the association ends correspond to OWL 2 object properties and SBVR fact types e.g., has_son (not fact type roles e.g., "son" as it should be in actual UML models).

In the tree ontology, the asserted classes are "Person", "Marriage" and "Sex"; they have asserted data properties and object properties "has_parent", "has_sex", and "has_marriage". Remaining classes and object_properties can be inferred from OWL 2 axioms and SWRL rules, e.g., "Man" can be defined as "Person" who has sex of male:

```
EquivalentClasses( Man Person
(ObjectHasValue(has_sex value male_sex)))
```

A couple is defined as an object property and can be derived by SWRL rule:

```
has_child(?x,?y),has_child(?z,?y),
  DifferentFrom(?x,?z)→couple(?x,?z)
```

We also use OWL 2 punning and create a class "couple" that is understood as a pair of male and female that have at least one child and, optionally, may be married. Figure 2 presents individuals that are analyzed in the following.

For defining an object property "has_kin" we can use the classical SWRL rules:

```
has_parent(?x,?y)→has_antecedent(?x,?y)
has_parent(?x,?y),has_antecedent(?y,?z)→
has_antecedent(?x,?z)
has_parent(?x,?y)→has_descendant(?y,?x)
has_parent(?x,?y),has_descendant(?y,?z)→
has_descendant(?x,?z)
has_antecedent(?x,?y)→has_kin(?x,?y)
has_descendant(?x,?y)→has_kin(?x,?y)
has_antecedent(?x,?y),has_descendant(?y,?z),
DifferentFrom(?x,?z)→has_kin(?x,?z)
```

That means kin of a person are her antecedents and descendants, as well as descendants of her antecedents except the person herself. We can define rules for all object properties of the genealogy tree in a similar way.

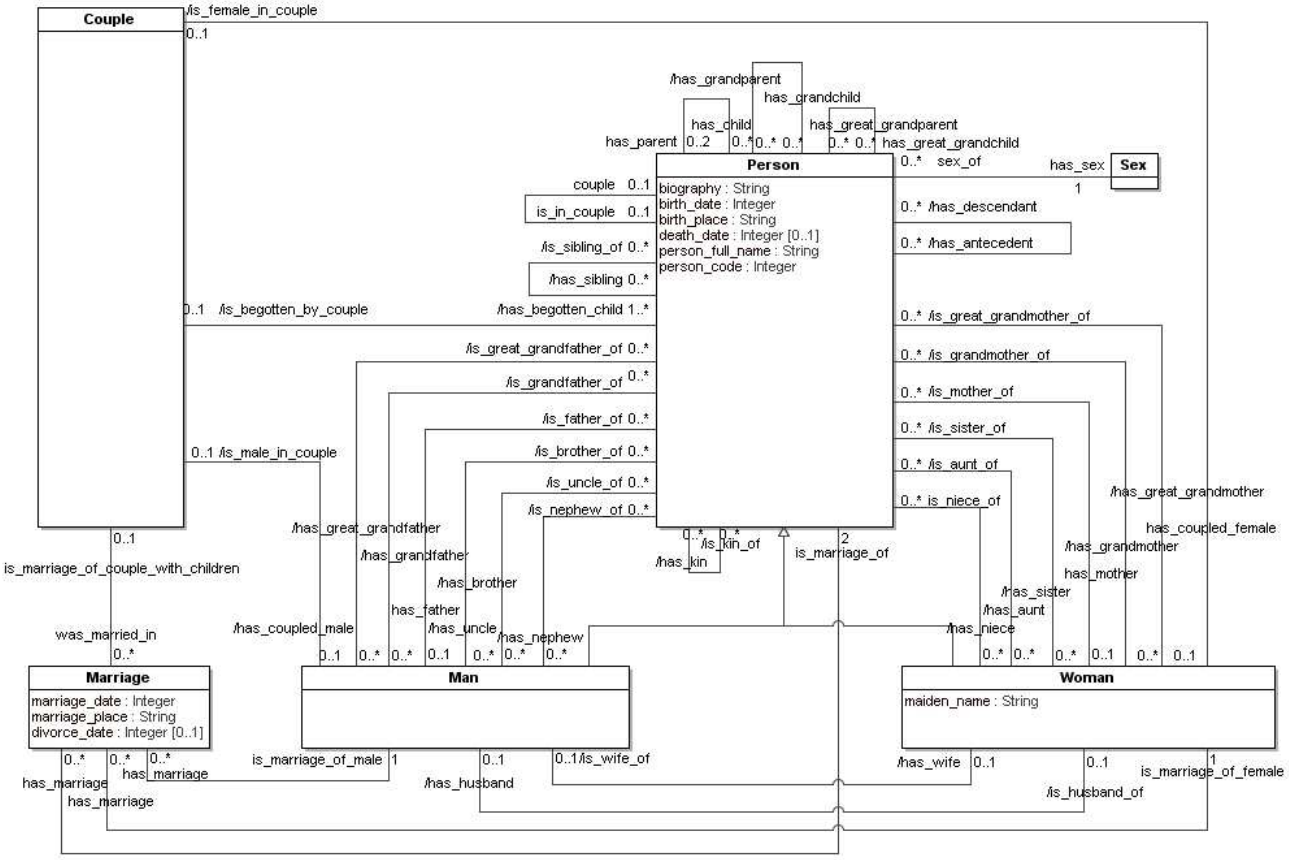

Figure 1. OWL 2 ontology of the genealogy tree in UML notation

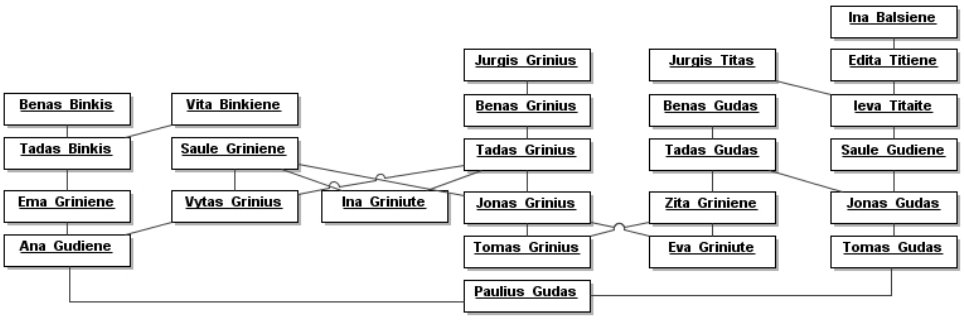

Figure 2. Individuals with "has_parent" links of a genealogy tree ontology in UML notation

Representing SWRL rules in SBVR is straightforward with the use of implication formulation. On the contrary to SWRL, SBVR rules can use disjunctive logical formulations that should be transformed into separate SWRL rules:

It is necessary that person1 has_antecedent person2 if person1 has_parent person2.

…

It is necessary that person1 *has_kin* person2 if
person1 *has_antecedent* person2 or
person1 *has_descendant* person2 or
person1 *has_antecedent* person3 and
person3 *has_descendant* person2 and not
person1 *equals* person2.

Using OWL 2 reasoner (Pellet or Hermit), we can infer kin of all persons and formulate simple SBVR questions as:

"What are *kin_of* person Vita_Binkiene?"

"Are *kin* person Ema_Griniene and Vytas_Grinius?"

that are translated into simple SPARQL queries:

```
SELECT DISTINCT ?kin
{gen:Vita_Binkiene gen:has_kin ?kin},
```

which gives the results presented in Figure 3. For executing queries, we have used the ARQ 2.8.4 query engine that supports SPARQL 1.1 extensions, and Pellet 2.2.2 OWL Reasoner.

```
kin
<http://Data.Genealogy_tree/Tadas_Binkis>
<http://Data.Genealogy_tree/Ema_Griniene>
<http://Data.Genealogy_tree/Ana_Gudiene>
<http://Data.Genealogy_tree/Paulius_Gudas>.
```

Figure 3. Results of the query "What are *kin_of* person Vita_Binkiene?"

The second query is of ASK type (the "kin" is the synonymous noun form of "has_kin"):

```
ASK{gen:Ema_Griniene
gen:has_kin gen:Vytas_Grinius}
```

and gives the result `false`.

Without the use of reasoner, formulating SPARQL queries from SBVR questions can become complicated as SPARQL recursive queries are problematic to identify from SBVR rules:

```
select distinct ?kin{{select ?antecedent{
gen:Vita_Binkiene gen:has_parent* ? antecedent .
not exists {? antecedent gen:has_parent ?x}}}
?kin gen:has_parent* ? antecedent .
filter(?kin != gen:Vita_Binkiene)}
```

Another way of transforming SBVR questions into SPARQL is to formulate derivation rules in CONSTRUCT query form [23]. However, CONSTRUCT queries inefficiently work with large rule sets. Therefore we transform SBVR rules into separate CONSTRUCT queries that give results in separate triple graphs, and apply Jena Union function that dynamically relates result graphs through their common elements. Finally, we execute SELECT query in the united graph:

```
CONSTRUCT {?x gen:has_antecedent ?y .}
WHERE{?x gen:has_parent ?y .}
...
CONSTRUCT{?x gen:has_kin ?y .}
WHERE{?x gen:has_antecedent ?y .}
CONSTRUCT{?x gen:has_kin ?y .}
WHERE{?x gen:has_descendant ?y .}
CONSTRUCT{?x gen:has_kin ?z .}
WHERE{?x gen:has_antecedent ?y .?y
gen:has_descendant ?z .filter(?x != ?z)}
SELECT DISTINCT ?kin{gen:Vita_Binkiene
gen:has_kin ?kin}
```

The SELECT gives the same results as in Figure 3.

IV. DEFINING N-ARY RELATIONS IN OWL 2

SBVR metamodel allows formulating fact types with more than two fact type roles and preserves the ordering of these roles. However, for further retaining the meaning of relations it is desirable to express them as binary relations because in the fact type there always are two main fact type roles – subject and object; the remaining fact type roles mean either properties of the fact type relating subject and object, either properties of object. We should formulate fact types in such a way that it would be possible to identify what fact type roles mean properties of fact type, and what ones mean properties of object. That is, instead of defining SBVR fact type as

"person *works_in* organization in position from date till_date",

it is desirable to represent it in such a way:

"person *works_in* organization from_date till_date",

Synonymous form: position,

"person *occupies_position_of* position",

"position *performs_office_of* office".

Here the last fact type is introduced for representing object type "office" that is a role type of the role "position". Figure 4 presents example ontology for modeling properties of relation in OWL 2. Role "Position" is explicitly presented as a class. Dependency with stereotype <<EquivalentClasses>> means that EquivalentClasses axiom

is defined for classes "works_in" and "Position". We cannot define properties of relations (i.e., object properties) in OWL 2, but we can use punning and define a class with the same name as the object property "works_in". For flexibility of formulating propositions and questions, we create the additional class "Position" equivalent to class "works_in", and attribute properties of the relation for that class.

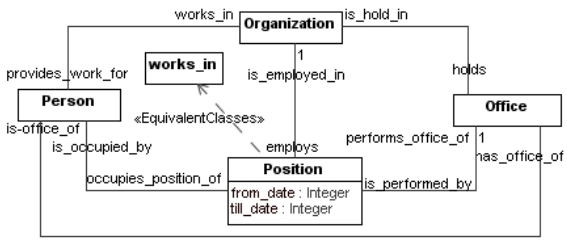

Figure 4. Example of modeling properties of relations in UML

Guizardi and Wagner suggested the similar solution [7] (without the role type) for representing properties of relations in UML despite that UML has the association class for this purpose. They have criticized the ambiguity of UML association class and proposed to create an association for representing a formal relation, and a separate class for materializing that relation.

Having such ontology, we can simply ask

"What are organizations that person Jonas_Grinius *works_in* from_date 2000-01-01 till_date 2003-01-01?"

Here we cannot use current reasoners (Pellet, Hermit) because they do not derive relation properties on the base of punning but we can attain these properties using SPARQL. For this, we should supplement the question with a business rule

"It is necessary that person *works_in* organization from_date till_date if the person *occupies_position_of* position from_date till_date and the position *is_employed in* the organization."

Then we instantiate the rule for person "Jonas_Grinius" and transform the question and the rule into the SPARQL query (!bound is added to optional variables):

```
select distinct ?org{
org:Jonas_Grinius org:works_in ?org.
org:Jonas_Grinius
org:occupies_position_of ?position .
?position org:is_employed_in ?org .
?position org:from_date ?from .
optional {?position org:till_date ?till} .
filter (?from <= "2003-01-01"^^xsd:date ) .
filter (?till >= "2000-01-01"^^xsd:date ||
!bound(?till))}
```

For the given ontology example of organizations and persons we also can define the OWL 2 object property chain:

```
SubObjectPropertyOf(ObjectPropertyChain
(occupies_position_of performs_office_of)
has_office_of)
```

that is represented as SBVR structural business rule:

It is necessary that person *has_office_of* office if the person *occupies_position_of* position and the position *performs_office_of* the office.

Current OWL 2 reasoners understand object property chains. As previously, we will consider two cases: 1) when we use a reasoner and formulate a simple question; 2) when we do not use a reasoner and formulate a question together with derivation rule.

Using reasoner, we can give the SBVR question

"What person *has_office_of* Professor?"

that transforms into the SPARQL query:

```
SELECT ?persons
{?persons org:has_office_of gen:Professor}
```

SBVR question without reasoning

"What person *occupies_position*

that *performs_office_of* Professor?"

transforms into SPARQL query:

```
SELECT ?persons{?persons
org:occupies_position_of ?position .
?position org:performs_office_of
org:Professor}
```

V. CONCLUSIONS ANF FURTHER WORKS

The paper presented some formulations of SBVR fact types and questions for recursive and n-ary relations, together with representing them in OWL 2, SWRL and SPARQL, and executing obtained queries in current ontology reasoning tools and SPARQL engines. For analyzing recursive relations, we represent them by SBVR derivation rules and propose two scenarios for transforming them: 1) using SWRL rules, when we obtain simple SPARQL queries but should apply OWL 2 reasoner in addition to SPARQL engine; 2) using sequences of SPARQL CONSTRUCT queries. Both cases are practicable in real life applications. For analyzing n-ary relations, we propose to represent them by using SBVR synonymous forms. Transformation of these forms into OWL 2 results in introducing a new class and a binary relation with the same name (allowed by OWL 2 punning) for defining the main relation between subject and object; and object properties of that class representing remaining roles of n-ary relation. Such representation allows avoiding unnatural names that appear in traditional solution.

The limitations of the approach are in the fact that OWL 2, even supplemented with SWRL, is not enough to model the complete semantics of SBVR vocabularies, and, consequently, SPARQL is not enough to model SBVR questions. From the other side, it still is not clear how to represent some OWL or SPARQL constructs in SBVR in a natural way. Our future work will focus on analysis of more patterns for improved flexibility of SBVR questions translatable to SPARQL queries as well as providing experiments for integrating the proposed solutions into the realistic enterprise context. Also, we have some initial prototype of SBVR editor for Lithuanian language and are willing to work for embodying the SBVR Structured Lithuanian.

REFERENCES

[1] L. Bodenstaff, P. Ceravolo, R. E. Damiani, C. Fugazza, K. Reed, and A. Wombacher, "Representing and Validating Digital Business Processes", in Web Information Systems and Technologies, LNBIP, vol. 8(1), 2008, pp. 19–32.

[2] J. Cabot, R. Pau, and R. Raventos, “From UML/OCL to SBVR specifications: A challenging transformation”. Information Systems, 2008, pp. 1–24.

[3] L. Ceponiene, L. Nemuraite, and G. Vedrickas, “Semantic business rules in service oriented development of information systems”, in Information Technologies' 2009: proceedings of the 15th International Conference on Information and Software Technologies, IT 2009, Kaunas, Lithuania, April 23-24, 2009, pp. 404–416.

[4] R. Chaffin and D. J. Herrman, “The Nature of Semantic Relations: A Comparison of Two Approaches”, in Relational Models of the Lexicon: Representing Knowledge in Semantic Networks. M. Evens, Ed. New York: Cambridge University Press, 1988, pp. 289–334.

[5] B. Demuth and H. B. Liebau, “An Approach for Bridging the Gap Between Business Rules and the Semantic Web”, in Advances in Rule Interchange and Applications, LNCS, vol. 4824, 2009, pp. 119–133.

[6] S. Goedertier and J. A. Vanthienen, “A Vocabulary and Execution Model for Declarative Service Orchestration”, in Business Process Management Workshops, LNCS, vol. 4928, 2008, pp. 496–501.

[7] G. Guizzardi and G. Wagner. „What’s in a Relationship: An Ontological Analysis“, in Conceptual Modeling – ER 2008. LNCS, vol. 5231, 2008, pp. 83–97.

[8] R. Hoekstra. Ontology Representation Design Patterns and Ontologies that Make Sense. SIKS Dissertation Series No. 2009-15, SIKS, Dutch, 2009.

[9] A. Kamada and M. Mendes, “Business Rules in a Service Development and Execution Environment”, in Eleventh International IEEE EDOC Conference Workshop (EDOCW'07), 2007, pp. 1366–1371.

[10] E. Kaufmann and A. Bernstein, “Evaluating the Usability of Natural Language Query Languages and Interfaces to Semantic Web Knowledge Bases”. Journal of Web Semantics: Science, Services and Agents on the World Wide Web, vol. 8, 2010, pp. 1–23.

[11] M. Kleiner, P. Albert, and J. Bézivin, “Parsing SBVR-Based Controlled Languages”, in Model Driven Engineering Languages and Systems, LNCS, vol. 5795, 2009, pp. 122–136.

[12] M. Kriechhammer, “Querying Systems for business models” 2006. Available from: http://www.kriechhammer.com /?English_Portfolio:my_Documents:Finals.[Accessed 06 May 2011].

[13] I. Lemmens, M. Nijssen, and S. Nijssen, “A NIAM2007 Conceptual Analysis of the ISO and OMG MOF Four Layer Metadata Architectures”, in On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, LNCS, vol. 4805, Germany: Springer Berlin/Heidelberg, 2007, pp. 613–623.

[14] M. H. Linehan, “Ontologies and Rules in Business Models”, in Proc. 11th Int. EDOC Conference Workshop (EDOC '07), 2007, pp. 149–156.

[15] M. H. Linehan, “Semantics in Model-Driven Business Design”, in Proc. 2nd Int. Semantic Web Policy Workshop (SWPW'06), 2006.

[16] A. Marinos and P. Krause, “An SBVR Framework for RESTful Web Applications”, in Rule Interchange and Applications, LNCS, vol. 5858, Germany: Springer Berlin/Heidelberg, 2009, pp. 144–158.

[17] L, Nemuraite, T. Skersys, A. Sukys, E. Sinkevičius, and L. Ablonskis, “VETIS tool for editing and transforming SBVR business vocabularies and business rules into UML&OCL models”, in Information Technologies' 2010: proceedings of the 16th International Conference on Information and Software Technologies, IT 2010, Kaunas, Lithuania, April 21-23, 2010, pp. 377–384.

[18] J. Nenortaite and R. Butleris, “Improving Business Rules Management through the Aplication of Adaptive Business Intelligence Technique”. Information Technology and Control, vol. 38(1), 2009, pp. 21–28.

[19] S. Nijssen, **“**SBVR: Semantics for Business”. Business Rules Journal, vol. 8(10), Oct. 2007, available from: http://www.BRCommunity.com/a2007/b367.html [Accessed 06 May 2011].

[20] S. Nijssen, “SBVR ~ Ground Facts and Fact Types in First-Order Logic”, Business Rules Journal, vol. 9(1), Jan. 2008, Available from: http://www.BRCommunity.com/a2008/b386.html [Accessed 06 May 2011].

[21] OMG, 2008. Semantics of Business Vocabulary and Business Rules (SBVR). Version 1.0. December, 2008, OMG Document Number: formal/2008-01-02.

[22] B. Paradauskas and A. Laurikaitis, “Business Knowlegde extraction using program understanding and data analysis techniques”, in Information Technologies' 2009: proceedings of the 15th International Conference on Information and Software Technologies, IT 2009, Kaunas, Lithuania, April 23-24, 2009, 337–354.

[23] A. Polleres, F. Scharffe, and R. Schindlauer. “SPARQL++ for Mapping between RDF Vocabularies”, in OTM'07 Proceedings of the 2007 OTM Confederated international conference on On the move to meaningful internet systems: CoopIS, DOA, ODBASE, GADA, and IS – Volume Part I, Springer-Verlag Berlin, Heidelberg, 2007.

[24] E. Prud‘hommeaux and A. Seaborne, A. SPARQL Query Language for RDF. W3C Recommendation. 15 January 2008. Available from: http://www.w3.org/TR/rdf-sparql-query/ [Accessed 06 May 2011].

[25] A. Razavi, A. Marinos, S. Moschoyiannis, and P. Krause, “RESTful Transactions Supported by the Isolation Theorems”, in Web Engineering, LNCS, vol. 5858, Germany: Springer Berlin/Heidelberg, 2009, pp. 394–409.

[26] S. Spreeuwenberg and H. K. Anderson, “SBVR's approach to controlled natural languages”, in Workshop on Controlled Natural Language (CNL 2009), 2009.

[27] S. Spreeuwenberg and R. Gerrits, “Business Rules in the Semantic Web, are there any or are they different?”, in Reasoning Web. Springer Berlin/Heidelberg, Germany, LNCS, vol. 4126, 2006, pp. 152–163.

[28] C. Storey, “Understanding Semantic Relationships”. VLDB Journal, vol. 2, 1993, pp. 455–488.

[29] A. Sukys, L. Nemuraite, B. Paradauskas, and E. Sinkevicius “Querying ontologies on the base of semantics of business vocabularies and business rules”, in Information Technologies' 2011: proceedings of the 17th international conference on Information and Software Technologies, IT 2011, Kaunas, Lithuania, April 27-29, 2011. Kaunas, 2011, pp. 247–254.

[30] E. Vysniauskas, L. Nemuraite, and A. Sukys, “A hybrid approach for relating OWL 2 ontologies and relational databases”, in Perspectives in Business Informatics Research: proceedings of the 9th international conference, BIR 2010, Rostock, Germany, September 29 – October 1, 2010, Berlin-Heidelberg-New York, Springer, 2010, pp. 86–101.

[31] W3C. Defining N-ary Relations on the Semantic Web. W3C Working Group Note, Apr. 2006. Available from: http://www.w3.org/ TR/swbp-n-aryRelations [Accessed 06 May 2011].

[32] W3C. SPARQL 1.1 Query Language. W3C Working Draft 14 October 2010. Available from: http://www.w3.org/TR/sparql11-query/ [Accessed 06 May 2011].

 ISBN: 978-1-61208-160-1