Transformation Framework for SBVR based Semantic Queries in Business Information Systems

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*—The paper presents transformation framework from questions in structured language based on Semantics of Business Vocabulary and Rules (SBVR) into SPARQL queries over ontologies defined in Web Ontology Language OWL 2 and, possibly, supplemented with Semantic Web rules SWRL. Such transformation depends on OWL 2 ontology related with corresponding SBVR vocabulary and rules. The current work considers a family of transformations and metamodels required for relating ontologies, rules, SPARQL queries and real business data supported by computerised information systems, as well as establishes requirements for harmonizing the coexistence and preserving semantics of these different representations.**

Keywords-SBVR; SBVR question; business vocabulary; business rule; ontology; SPARQL

I. INTRODUCTION

Semantics of Business Vocabulary and Business Rules (SBVR) is the OMG metamodel [1], which has gained a great attention both in commercial applications and academic world. It is streamlined towards raising an abstraction level one more step upwards in the field of modelling business software systems as makes it possible to conceptualize software artefacts in some kind of structured language understandable for business people and computing systems. SBVR could not serve as the last step in the vision of communicating with computers in natural languages but it can be seen as intermediate specification between linguistically processed text and computer supported knowledge models as e.g. Web Ontology Language OWL 2 [2] and Semantic Web Rules (SWRL). Currently, there are many SBVR related applications, for example, generation of software models and code [3], creation of terminological vocabularies for various fields [4], auditing compliance between governmental regulations and business actualities [5], etc. In such context, our work is related with SBVR support for analyzing business information – i.e., formulating SBVR questions and transforming them into ontology query language SPARQL [6] for immediate access to business data from various sources (databases, business process management and enterprise resource planning (ERP) systems, documents), supported with ontological descriptions.

For implementing such SBVR into SPARQL transformations in real world enterprises, the complex infrastructure is needed for keeping connections between business vocabularies, vocabularies of business rules, ontology storages, databases and software service systems. The growing complexity of dealing with business information demonstrates need for such infrastructures, especially for large organizations or public institutions. As a little attention is given for SBVR questions in research literature and SBVR specification itself [1], we have investigated the peculiarities of modelling SBVR questions and representing them as SPARQL queries in our previous works [7], [8]. In the current paper, we present a transformation framework among SBVR questions and OWL 2, SWRL, SPARQL representations of business data.

The rest of the paper is organized as follows. Section 2 presents related works. Section 3 presents the framework of metamodels and transformations for transforming SBVR questions into OWL 2 and SPARQL; section 4 – examples of SBVR question patterns in SPARQL. Section 5 draws conclusions and outlines the future work.

II. RELATED WORKS

The SPARQL is a Semantic Web query language for RDF and OWL. SPARQL, as well as RDF and OWL, are knowledge representation centric languages oriented towards computer processing, therefore, they are not suitable for supporting immediate access to business data for business people. SBVR metamodel, concentrated on specifying the meaning of knowledge in terms of concepts, propositions and questions, is devoted for filling this gap by proposing the abstract syntax, expressible in structured languages, understandable by human, and, contemporaneously, having formal ontological representation. SBVR may be treated as multilingual representation of meaning in structured language intermediate between natural languages and executable software specifications [9]. SBVR and OWL 2 were created to be compliant with Common Logics [10]; therefore, SBVR concepts can be mapped to OWL 2 concepts. Rough correspondences between SBVR and first OWL version were declared in SBVR specification, but OWL was overridden by more expressive OWL 2, and explicit SBVR into OWL 2 transformation is still under investigation [11]. Though some implementations of SBVR into OWL 2 transformation exist, they are embedded in commercial products as Collibra tool suite [12] or ONTORule project components [13] in the restricted manner directly unusable for other research projects or applications. As SPARQL queries are executed over RDF and OWL 2 ontologies, transformation of SBVR vocabularies and rules

 ISBN: 978-1-61208-223-3

into OWL 2 (SBVR2OWL2) is important for possibility of expressing SBVR questions in SPARQL.

OWL 2, the latest version of Web ontology language [2], is provided with two kinds of semantics – direct semantics [14], based on Functional style syntax, and RDF based semantics [15]. As OWL 2 metamodel has changed from the previous OWL version, standard RDF based metamodel of OWL 2 still does not exist though OMG is making efforts for finalizing a new version of ontology definition metamodel [9], which should harmonize OWL 2 Functional and RDF syntax. For practical purposes, the Manchester University has created converter between various syntaxes of OWL 2, so it is possible to convert OWL 2 Functional style ontologies into RDF documents; but the reverse is not always true. OWL 2 Functional style ontologies are limited to Description Logics whereas RDF format is capable representing OWL 2 FULL, the most expressive language specified by the W3C OWL 2 standard.

Similarly, in Twouse project [16], SPARQL Abstract Syntax (SPARQLAS) metamodel is proposed, which is based on OWL 2 Functional Syntax and directed towards querying and two-way transforming of UML and OWL 2 models for software design, code generation and OWL 2 ontology engineering. SPARQLAS directly includes part of OWL 2 metamodel, so the advantage of using the SPARQLAS is a conceptual clarity perceived by developers, and decidability of reasoning and querying tasks; its shortcoming – the limitation to OWL 2 Description Logics (DL). There are other interesting works related with SPARQL, e.g. [17], which authors present solutions for transforming object oriented queries into SPARQL RDF query language and, vice versa, for translating SPARQL queries into object oriented ones. The approach allows to access RDF data from object models as well as to implement SPARQL endpoints for object oriented applications.

The mentioned works raise the question what SPARQL metamodel to choose for SBVR2SPARQL transformation. SPARQLAS has advantages for writing OWL 2 queries as users can do it faster; SPARQLAS queries can be automatically translated into SPARQL and executed using an OWL 2 reasoner and a SPARQL engine. SBVR to SPARQLAS transformations also would be much simpler but such transformations may be not suitable for all cases because the newest version, SPARQL 1.1, has dependencies on OWL 2 Full. Current ontology reasoners as Pellet or HermiT are based on OWL 2 DL, but there already are proposals as [18] for implementing OWL 2 Full reasoners on the base of first order logic formulas. As SBVR semantics also is not limited to OWL 2 DL, we have given the preference to RDF based SPARQL metamodel against OWL 2 Functional Syntax based one regarding the fact that the latter solution could be not applicable in a general case for SBVR questions that are considerably more expressive and should not be limited to OWL 2 DL capabilities.

RDF data model is based on subject-predicate-object expressions (triples) and is considered to be the most relevant standard for data representation and exchange on the Semantic Web. It gives a basis for other standards such as RDF Schema (RDFS) and OWL. OWL 2 ontologies can be mapped into RDF graphs and vice versa using special mapping rules [19].

Our transformation framework uses SPARQL RDF based metamodel capable to represent OWL 2 Full. Contemporaneously, we map SBVR concepts to ontology concepts with the requirement that SBVR transformation into OWL 2 (SBVR2OWL2) should preserve the same semantics as SBVR to SPARQL. For providing querying capabilities equal to SBVR questions, the SBVR transformation into to OWL 2 should represent in ontology SBVR related concepts as preferred, not-preferred and prohibited designations, synonyms and synonymous forms, predefined fact types etc. However, these terminological concepts should not make impact on behaviour of reasoners and other ontology processing means. Therefore, terminology and linguistics related SBVR elements should be separated from conceptual knowledge for efficient querying using synonyms or synonymous forms and getting correct answers.

III. SBVR BASED QUERYING FRAMEWORK

The process for accessing data and services of Business Information Systems by the means of SBVR questions over business vocabularies is presented in Figure 1. Business data and service software models could be generated from SBVR vocabularies and rules, and queried using SBVR questions.

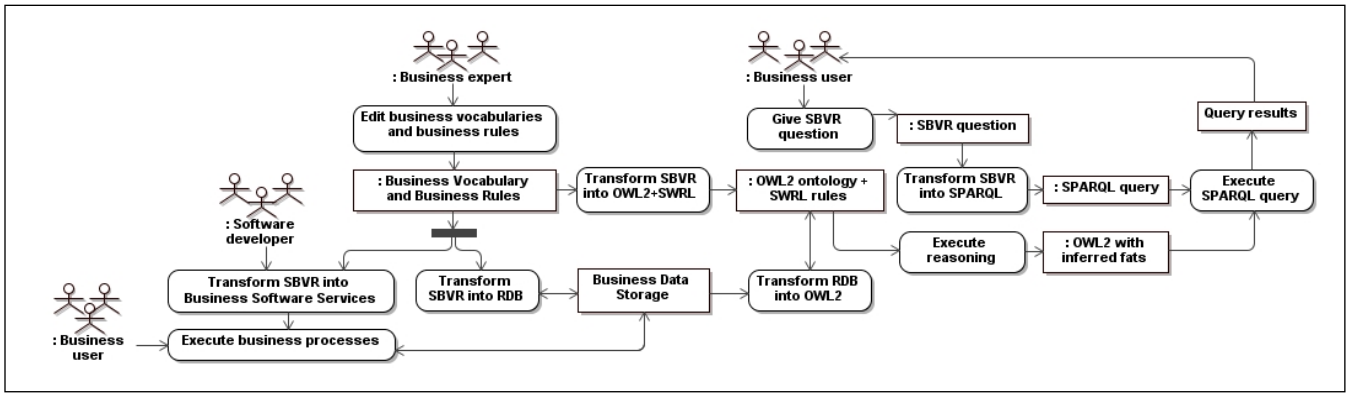

Figure 1. SBVR as a intermediate representation between data and services of Business Information Systems and user-understandable language

 ISBN: 978-1-61208-223-3

For the wholeness of the picture, we have included transformations among SBVR business vocabularies&rules and software services&databases. The latters are beyond the scope of the current paper but they are essential by giving a possibility for implementing business supporting software systems and querying about business state using the same business vocabulary and rules used for creating new (or conceptualizing existing) information technology assets. We suppose the usage of reasoners acknowledging OWL 2 axioms and SWRL rules that allow inferring additional facts and making SPARQL queries simpler – what should simplify translation from real natural languages into SBVR structured languages as well.

Metamodels and transformations required for implementing the querying process are represented in Figure 2. Two kinds of transformations are used: model–to–model transformations in ATL language for transforming SBVR models into OWL 2+SWRL (SBVR2OWLSWRL) and SBVR into SPARQL models (SBVR2SPARQL); and model–to–text transformations for transforming OWL 2, SWRL, SPARQL models into OWL 2, SWRL, SPARQL textual representations.

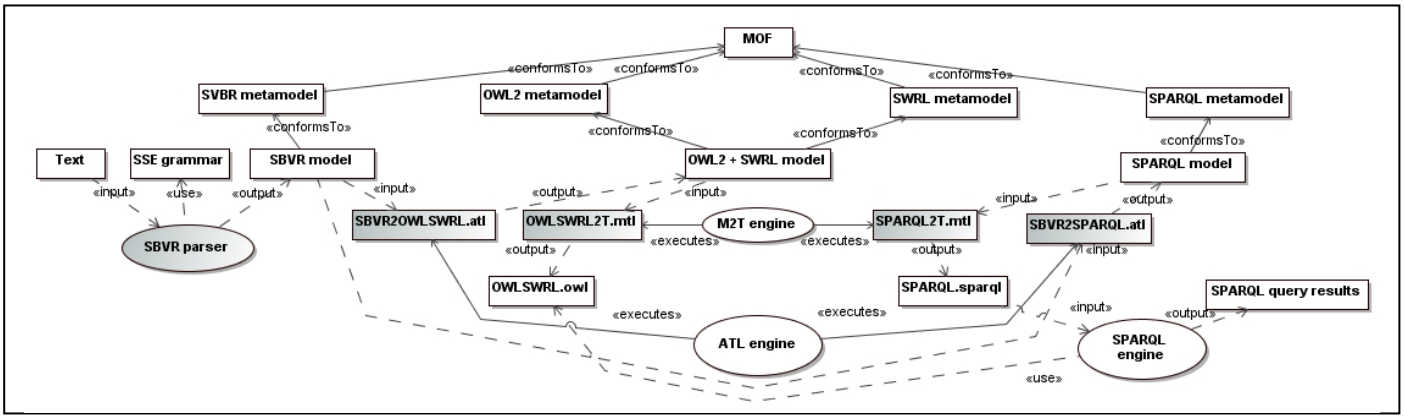

Figure 2. Metamodels and transformations of SBVR-based querying framework (shadowed components are implemented by authors and [11])

For alignment of SBVR, SPARQL, OWL 2, SWRL and RDF metamodels, we should establish mappings between relevant SBVR, OWL 2, SWRL, RDF, and SPARQL concepts.

Main concepts of SBVR used in formulating questions are object types, individual concepts, fact types, facts, variables and various kinds of logical formulations. Following recommendations for correspondence SBVR and OWL concepts [1], we suppose mapping SBVR object types to OWL 2 classes; SBVR binary fact types to OWL 2 object properties; SBVR *is_property_of* fact types to OWL 2 data properties; SBVR concept generalization to OWL 2 `SubclassOf`, `SubObjectProperty` or `SubDataProperty` with corresponding disjointness axioms; SBVR individual concepts to OWL 2 individuals, and some of SBVR logical formulations to OWL 2 axioms and restrictions. Additionally, we consider mapping of n-ary relations, roles and categorization schemes according [11], and the wider set of logical formulations. Only conceptual elements, i.e. SBVR meanings represented by preferred designations are transformed into OWL 2. SBVR synonyms and synonymous forms, non-preferred or prohibited designations should not be lost but may be included in terminological ontologies or annotations in such a manner that they would not make impact or require additional processing from reasoners acting on domain ontology.

In our chosen RDF based SPARQL metamodel (based on Eclipse EMFTEXT project), OWL 2 ontologies are represented by RDF triples. It means that all complexity of OWL 2 axioms is expressed using a few concepts as *VarOrTerm*, *Verb*, *Object*, representing OWL 2 `Class`, `DataProperty`, `ObjectProperty`, `Individual`. It hides conceptual clarity of OWL 2 but gives a great flexibility e.g. by allowing to access OWL 2 classes and axioms in the same manner as individuals and their property assertions.

SPARQL query language uses the power and flexibility of RDF. The conditions of queries are composed using sets of graph patterns, which are written in a same form as RDF triples, but can have variables in position of subject, predicate or object. SPARQL is not very user friendly especially when one needs to express complex OWL 2 statements in RDF to compose query condition. As our approach is based on generating SPARQL automatically from SBVR questions, written in structured natural language, the complexity of SPARQL makes no impact to end users.

SPARQL query consists of two main parts – at least one variable and conditional element – `WHERE` clause, containing graph pattern elements – *TriplesSameSublectLeft* (Figure 3). Graph pattern element of SPARQL consists of two *VarOrTerm* elements, representing subject and object, and *VarOrIRIRef* element, representing predicate of graph pattern.

OWL 2 ontologies are comprised of axioms that can be expressed with RDF. Having OWL 2 ontology in RDF format one can query it using SPARQL. World Wide Web Consortium defines mappings between structural specification of OWL 2 and RDF graphs. Using this mapping any OWL 2 ontology can be transformed into RDF graph without changing formal meaning of the ontology [19]. Some OWL axioms (for instance, `SubClassOf`, `SubObjectProperty`, `SubDataPropertyOf`) that can be represented using binary relation are transformed into single

triples. Other axioms (e.g., `EquivalentClasses`, `EquivalentObjectProperties`) are transformed into several triples. Annotations of axioms are also transformed into sets of triples.

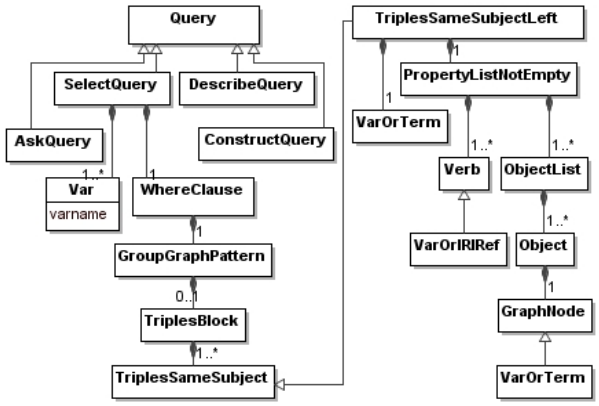

Figure 3. SPARQL metamodel fragment

IV. EXAMPLES OF APPLYING SBVR, OWL 2 AND SPARQL CONCEPTS FOR FORMULATING QUERIES

Transforming SBVR questions into SPARQL, it is essential to consider SBVR mappings to OWL 2 concepts for preserving the intended semantics during executing queries. It must be known how SBVR concepts are represented in OWL 2 to compose proper SPARQL queries. For example, SBVR uses instantiation formulations to classify things, while in OWL 2 there is an `rdf:type` property for this purpose. Only preferred designations of SBVR concepts are transformed for representing domain ontology in OWL 2; consequently, SBVR question concepts also are transformed into SPARQL using preferred designations. In the following we will show how SBVR concepts and formulations should be treated to compose SPARQL queries for OWL 2 ontology corresponding to SBVR questions in SBVR Structured English (SSE).

A. SBVR Facts and Binary Fact Types

SBVR binary fact types are expressed using OWL 2 object properties or RDF properties. Instances of fact types (facts) are expressed using triples with individuals in positions of subject or object and a property in a position of predicate.

For searching facts by SPARQL, graph pattern with variables in positions of subject or object (depending on what individuals we are searching for) should be used. Example of representing SBVR fact type and fact in SBVR SSE and RDF Turtle format is presented in TABLE I; query example – in TABLE II.

TABLE I. EXAMPLE OF REPRESENTING SBVR FACT TYPES AND FACTS

Language	Representation expressions
SBVR SSE	person1 *has_parent* person2 Ina_Griniute *has_parent* Jurgis_Grinius
RDF	:*has_parent* rdf:type owl:ObjectProperty rdfs:range :Person rdfs:domain :Person :Ina_Griniute :*has_parent* :Jurgis_Grinius

TABLE II. EXAMPLE OF QUERYING SBVR FACTS

Language	Query expressions
SBVR SSE	Who *is_parent_of* person Ina_Griniute?
SPARQL	SELECT ?person { :Ina_Griniute :has_parent ?person }

B. SBVR assortment fact type

SBVR assortment fact type is expressed using `rdf:type` property. Example of assortment fact type is presented in TABLE III, querying instances of a particular class – in TABLE IV.

TABLE III. EXAMPLE OF SBVR ASSORTMENT FACT TYPE

Language	Representation expressions
SBVR SSE	Ina_Griniute *is_a* person
RDF	:Ina_Griniute rdf:type :person

TABLE IV. EXAMPLE OF QUERYING ASSORTMENT FACT TYPE

Language	Query expressions
SBVR SSE	What subjects *are_of_type* person?
SPARQL	SELECT ?subjects { ?subjects rdf:type :person }

C. SBVR Specializations

For expressing that a particular concept is a category of more general concept, SBVR uses concept specialization, which corresponds to OWL 2 classes related by property `rdfs:subClassOf` stating that one class is a subclass of another class. We can use specializations for querying more general concepts. Also, we can access OWL 2 metaconcepts (classes, object and data properties, etc.) in the same manner as individuals and properties of individuals. For example, we can find subclasses of a particular class (TABLES V and VI). Similarly, specialization may be defined for fact types what corresponds to OWL 2 `SubObjectPropertyOf` or `SubDataPropertyOf`.

TABLE V. SBVR SPECIALIZATION FACT TYPE IN RDF

Language	Representation expressions
SBVR SSE	book *is_a* publication
RDF	:book rdfs:subClassOf :publication

TABLE VI. EXAMPLE OF QUERYING SBVR FACTS WITH SPARQL

Language	Query expressions
SBVR SSE	What class *is_sublass_of* publication?
SPARQL	SELECT ?class { ?class rdfs:subClassOf :publication }

D. SBVR Implication Formulations

SBVR implication formulations can be expressed using SWRL rule language. SWRL rules can infer new facts in OWL ontology. SPARQL queries can access all inferred

 ISBN: 978-1-61208-223-3

facts in a same way as asserted ones, using OWL 2 object properties or RDF properties.

E. SBVR Conjunction

Conjunction is a logical operation, which is true if each of its logical operands is true. It can be used to link SBVR fact types, propositions or logical formulations. In SBVR SSE language keyword "and" is used to indicate conjunction. In OWL 2, conjunction is represented by intersection of triples expressing axioms or restrictions for defining classes. Such definitions are used for inferring classes that could be directly accessed by SPARQL. In SPARQL queries, conjunction is used for finding facts satisfying conjunction condition. E.g. facts presented in TABLE VII are used in SPARQL query conjunction expression (TABLE VIII).

TABLE VII. EXAMPLE OF SBVR FACTS USED IN CONJUNCTION

Language	Representation expressions
SBVR SSE	Ina_Griniute *has_friend* Tadas_Gudas Ina_Griniute *has_kin* Jurgis_Titas
RDF	:Ina_Griniute :has_friend :Tadas_Gudas :Ina_Griniute :has_kin :Jurgis_Titas

TABLE VIII. EXAMPLE OF QUERYING SBVR FACTS USING CONJUNCTION

Language	Query expressions
SBVR SSE	What person *is_friend_of* Tadas_Gudas and *is_kin_of* Jurgis_Titas?
SPARQL	SELECT ?person { ?person :is_friend_of :Tadas_Gudas ; :is_kin_of :Jurgis_Titas }

F. Disjunction of Facts

Disjunction in SBVR is a logical operation, which is true if at least one of its logical operands is true. In SBVR SSE, it is indicated using keyword "or". Disjunction is useful when we want to find individuals having at least one of certain properties. For expressing disjunction in SPARQL, alternative graph patterns specified by UNION keyword can be used. As a result, all matching solutions are returned. Example of facts, used in alternative graph pattern, is presented in TABLE IX, example of query – in Table X.

TABLE IX. EXAMPLE OF SBVR FACTS, USED IN DISJUNCTION

Language	Representation expressions
SBVR SSE	Publication1 *has_author* Tomas_Gudas Publication2 *has_editor* Tomas_Gudas
RDF	:Publication1 :has_author :Tomas_Gudas :Publication2 :has_editor :Tomas_Gudas

TABLE X. EXAMPLE OF QUERYING SBVR FACTS USING DISJUNCTION

Language	Query expressions
SBVR SSE	What publications *has_author* Tomas_Gudas or *has_editor* Tomas_Gudas?
SPARQL	SELECT ?publications { {?publications :has_author :Tomas_Gudas} UNION {?publications :has_editor :Tomas_Gudas } }

G. Negation

Logical negation is a logical operation in SBVR having one logical operand. Negation, introduced in SPARQL 1.1, filters query solution by checking if query graph pattern does not match ontology. It is specified using keyword NOT EXISTS (TABLE XI).

TABLE XI. EXAMPLE OF QUERYING USING NEGATION

Language	Query expressions
SBVR SSE	What persons *are_author_of* publications that *are_not_cited_by* any publication?
SPARQL	SELECT ?persons { ?persons :is_author_of ?publication . FILTER NOT EXISTS {?publication :is_cited_by ?o}}

H. Synonyms and Synonymous Forms

SBVR synonyms and synonymous forms have the same meaning but different representation. They allow more flexible formulation of questions as synonyms are treated as equivalent classes or data properties, and synonymous forms are treated as equivalent object properties. Synonyms and synonymous forms in SBVR questions are transformed into SPARQL using preferred designations (i.e. in the same way as in SBVR2OWL2 transformation). Example of synonyms is presented in TABLE XII, example of querying using synonyms – in TABLE XIII (the query will find individuals of type "dictionary").

TABLE XII. EXAMPLE OF SBVR SYNONYM

Language	Representation expressions
SBVR SSE	vocabulary Synonym: dictionary Book1 *is_a* dictionary
RDF	:Book1 rdf:type :vocabulary

TABLE XIII. EXAMPLE OF QUERYING USING SYNONYM

Language	Query expressions
SBVR SSE	What books *are_of_type* dictionary?
SPARQL	SELECT ?books { ?books rdf:type :vocabulary }

Similarly, SBVR synonymous forms allow saying "person *is_author_of* publication" or "publication *is_published* by person". Example of using synonymous forms is presented in TABLE XIV and TABLE XV.

TABLE XIV. EXAMPLE OF SBVR SYNONYMOUS FORM

Language	Representation expressions
SBVR SSE	publication *is_published* by person Synonymous_form: person *is_author_of* publication Tadas_Grinius *is_author_of* Book1
RDF	:Book1 is_published_by :Tadas_Grinius

TABLE XV. EXAMPLE OF QUERYING USING SYNONYMOUS FORM

Language	Query expressions
SBVR SE	What person *is_author_of* Book1?
SPARQL	SELECT ?person { :Book1 :is_published_by ?person }

I. Predefined SBVR fact types

Some concepts and fact types having dedicated meaning are predefined in SBVR metamodel [1] (e.g. number1 *is_greater_than* number2, number1 *is_less_than* number2), SBVR extensions [4] (e.g. time interval1 *is_before* time interval2) or may be introduced into business domain vocabularies by users. Extensibility of SBVR allows conceptualizing terminology of specific domains for the way people think and communicate using natural language, and support computer understandable reasoning. E.g., we can introduce a predefined meaning for SBVR fact type *is_number_of* to express counting operation in SPARQL (Table XVI).

TABLE XVI. EXAMPLE OF QUERYING USING COUNT AGGREGATION

Language	Query expressions
SBVR SE	What *is_number_of* persons who is_child_of person Tadas_Grinius?
SPARQL	SELECT count(?persons) { ?persons :is_child_of :Tadas_Grinius}

V. CONCLUSIONS AND FURTHER WORKS

Transforming SBVR questions into SPARQL queries has shown that it is very important to synchronize SBVR to OWL 2 and SBVR to SPARQL mappings because it makes impact on correctness of SPARQL queries. Transforming SBVR vocabularies and rules into ontologies brings specific features into OWL 2 ontologies that could be rationally managed for obtaining more semantics from domain concepts.

Currently, main transformations from SBVR into SPARQL were realized and analysed; however, the efficiency of such transformations could be reached only by implementing the overall framework relating business vocabularies, rules, questions and executable business models. Our future work is streamlined towards elaborating the framework components, applying them in the real business environment, and, possibly, relating with linguistic models for connecting a really natural language and business software systems.

REFERENCES

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

[2] B. Motik, P. F. Patel-Schneider, B. Parsia, OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax. W3C Proposed Recommendation 22 September 2009.

[3] A. Marinos, P. Krause, "An SBVR Framework for RESTful Web Applications", Rule Interchange and Applications, Springer Berlin/Heidelberg, Germany, LNCS, Vol. 5858, 2009, pp. 144–158.

[4] OMG. Date-Time Vocabulary (Date-Time). OMG Document Number: bmi/2011-08-01.

[5] K. R. Bouzidi, C. Faron-Zucker, B. Fies, and N. L. Thanh, "An Ontological Approach for Modeling Technical Standards for Compliance Checking", Web Reasoning and Rule Systems, LNCS, Vol. 6902, Springer-Verlag Berlin, Heidelberg, 2011, pp. 244–249.

[6] W3C. SPARQL 1.1 Query Language. W3C Working Draft 14 October 2010.

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

[8] A. Sukys, L. Nemuraite, B. Paradauskas, and E. Sinkevičius, "SBVR based representation of SPARQL queries and SWRL rules for analyzing semantic relations", the First International Conference on Business Intelligence and Technology, (Bustech 2011), Sep 25–30, Rome, Italy, IARIA, 2011, pp. 1–6.

[9] M. Kleiner, P. Albert, J. Bézivin, "Parsing SBVR-Based Controlled Languages", Model Driven Engineering Languages and Systems, LNCS, Vol. 5795, 2009, pp. 122–136.

[10] OMG. Ontology definition metamodel. OMG Document Number: ptc/2008-09-07.

[11] J. Karpovic, L. Nemuraite, "Transforming SBVR business semantics into Web ontology language OWL2: main concepts", Information Technologies' 2011 : proceedings of the 17th international conference on Information and Software Technologies, IT 2011, Kaunas, Lithuania, April 27–29, 2011 pp. 231–238.

[12] Collibra. Data governance software. http://www.collibra.com/ [Accessed 10 Mar 2012]

[13] Ontorule project. Ontologies meet business rules. http://ontorule-project.eu/ [Accessed 10 Mar 2012]

[14] I. Horrocks, B. Parsia, U. Sattle, OWL 2 Web Ontology Language Direct Semantics. W3C Recommendation 27 October 2009.

[15] J. Carrol, I. Herman, P. F. Patel-Schneider, OWL 2 Web Ontology Language RDF-Based Semantics. W3C Recommendation 27 October 2009.

[16] M. Schneider, "SPARQLAS – Implementing SPARQL Queries with OWL Syntax", Proceedings of the 3rd Workshop on Transforming and Weaving Ontologies in Model Driven Engineering, Málaga, Spain, Jun 30, 2010, pp. 1–7.

[17] G. Hillairet, F. Bertrand, and J. Y. Lafaye, "Rewriting Queries by Means of Model Transformations from SPARQL to OQL and Vice–Versa", ICMT '09 Proceedings of the 2nd International Conference on Theory and Practice of Model Transformations", Springer-Verlag Berlin, Heidelberg, 2009, pp. 116–131.

[18] M. Schneider, "Reasoning in expressive extensions of the RDF semantics", ESWC'11 Proceedings of the 8th Extended Semantic Web Conference on the Semanic Web: Research and Applications", Springer-Verlag Berlin, Heidelberg, LNCS, Vol. 6644, 2011, pp. 487–491.

[19] W3C. OWL 2 Web Ontology Language Mapping to RDF Graphs. W3C Recommendation 27 October 2009.

 ISBN: 978-1-61208-223-3