SBVR to OWL 2 Mappings: An Automatable and Structural-Rooted Approach

Emiliano Reynares
UTN FRSF, CIDISI Research Center,
Santa Fe, Argentina, S3004EWB,
ereynares@frsf.utn.edu.ar

and

Ma. Laura Caliusco
UTN FRSF, CIDISI Research Center,
Santa Fe, Argentina, S3004EWB,
mcaliusc@frsf.utn.edu.ar

and

Ma. Rosa Galli
INGAR-UTN-CONICET,
Santa Fe, Argentina, S3002GJC,
mrgalli@santafe-conicet.gov.ar

Abstract

Wide applicability of mapping business rules expressions to ontology statements have already been recognized. Some of the most important applications are: (1) using of ontology reasoners to prove the consistency of business domain information, (2) generation of an ontology intended to be used in the analysis stage of a software development process, and (3) the possibility of encapsulate the declarative specification of business knowledge into information software systems by means of an implemented ontology. The Semantics of Business Vocabulary and Business Rules (SBVR) supports that approach by providing business people with a linguistic way to semantically describe business concepts and specify business rules in an independent way of any information system design. Previous work have presented some proposals but an exhaustive and automatable approach for them still is lacking. This work presents a broad and detailed set of transformations that allows the automatable generation of an OWL 2 ontology from the SBVR specifications of a business domain. Such transformations are rooted on the structural specification of both standards and are depicted through a case study. A real case validation example was performed, approaching the feasibility of the mappings by the quality assessment of the developed ontology.

Keywords: business rule, ontology, mapping, SBVR, OWL 2

1 Introduction

The Semantics of Business Vocabulary and Business Rules (SBVR) provides business people a linguistic way to semantically describe business concepts and specify business rules [13]. The linguistic approach adopted by the proposal enables the expression of business knowledge through statements rather than diagrams. That is based in the insight that diagrams are helpful for depicting structural organization of concepts but they are impractical as a primary means of defining vocabularies and expressing business rules.
SBVR is rooted in first-order predicate logic with some restricted extensions into higher-order logics and some limited extensions into modal logic. Despite such sound theoretical foundation on formal logic is a key

feature in automated reasoning contexts, SBVR has been conceptualized for business people and designed to be used for business purposes independent of information systems designs.
Several works have already recognized the benefits of having a mean of mapping SBVR expressions to ontology statements. Important applications of such mappings can be mentioned. For example, ontology reasoners could be used to automatically prove the consistency of business models [4][10][1][7]. Ontologies intended to be used in the analysis stage of a software development process could be generated from main business knowledge sources [3]. The mappings could also be used to generate ontologies that encapsulate business knowledge into information software systems, enabling unambiguous representation of knowledge and efficient management of highly dynamic environments [5][17][15]. Use of semantic technologies for creating more intelligent and effective enterprise information systems has increased considerably in the last years. Before depicted examples highlight the strong benefits and wide applicability of ontologies in such kind of systems.
Although previous proposals have presented some SBVR to OWL 2 transformations, an exhaustive and automatable approach for them still is lacking. This work presents a broad and detailed set of transformations that allows the automatable generation of an OWL 2 ontology from the SBVR specifications of a business domain. Transformations are rooted on the structural specification of both standards and are depicted along the paper through a case study. OWL 2 Web Ontology Language (OWL 2) has been selected as the receipt language of the transformations because it has evolved as a de-facto standard for a broad spectrum of applications [21].
The rest of this paper is organized as follow. Section 2 analyses the differences with previous works. Section 3 and Section 4 provides an overview of the SBVR and OWL 2 specifications, respectively. Section 5 presents the SBVR to OWL 2 mappings and illustrate them through a case study. Finally, some conclusions and future research directions are presented in Section 6.

2 Related Work

Several works have already recognized that having a mean of mapping SBVR expressions to ontology statements will have strong benefits and will gain wide applicability in the future vision of enterprise information systems [4][5][10][1][7].
An approach combining OWL 1 [19] and SWRL expressions [20] is explored in [4]. Aim of the approach is to obtain a Platform Independent Model (PIM) [11] from the SBVR business vocabulary and rules expressions. That goal is achieved by performing an intermediate SBVR to OWL/SWRL translation process as a way to check consistency and expand the knowledge by means of inference procedures. The work explores the feasibility of the proposed solution and the main problems arising. Although it illustrates some mappings between SVBR and OWL 1 through an example, the transformations are not explicitly formalized and therefore they can not be generalized to other situations. A model transformation chain is presented in [5] as a way to translate SBVR based vocabularies to OWL 1 and R2ML statements [16]. Conducted by the insight that business rules should be guaranteed by all IT applications of an enterprise, the paper presents the needs for a technology to transform the SBVR expressions into a PIM. The work meets such need by presenting a chain of metamodel-based transformations from a conceptual point of view. The transformations are rooted on an intermediate metamodel of SBVR and the QVT language [11]. The proposal use SBVR Vocabulary-to-MOF/XMI Mapping Rule Set to produce a MOF model and XML schema [11], the ODM [12] in order to describe OWL models within the MOF context, and the MOF-based metamodel of R2ML to generate the final version of the rules. The authors highlight the contribution to the model-driven integrity engineering of their approach given that the model transformation chain is completely based on the abstract syntax of the involved languages. Some basics concepts and problems in transforming SBVR expressions to OWL 2 are introduced in [10]. The aim of the work is to allow business users to describe ontologies in a similar way to their everyday business language, which will allow to prove consistency of business vocabulary and rules by means of OWL 2 reasoners. Although these proposals show the usefulness of using SBVR expressions as starting point for ontology development, the first ones do not use the most mature and powerful technological solutions available, while the last one just answer some primary questions.
In the other hand, [1] and [7] propose a set of transformations but following a different approach. These works are rooted in the ORM conceptual modelling language [14], which is at the core of the SBVR proposal. Definition and application of an integrated method that uses ORM to generate an ontology-based query mechanism is presented in [1]. Finally, a set-theoretic semantic for ORM 2 and a formal approach to map a fragment of such language to the $\mathcal{ALCQI}$ fragment of Descriptions Logics [2] is introduced in [7]. Although the work also depicts a tool which implements the translations of a set of ORM 2 constraints into a OWL 2 ontology, the proposal follows a formally grounded approach by stating the logical foundations of the translations between the underlying theories of SBVR and OWL 2.

Instead, the present work depicts a broad and detailed set of transformations that allows the automatable generation of an OWL 2 ontology from the SBVR specifications of a business domain. Transformations are rooted on the structural specification of both standards rather than theoretic considerations of the language, with the aim to provide a set of mappings readily usable for business people or developers concerned with the implementation of a mapping tool.

3 SBVR Overview

SBVR defines the vocabulary and rules for documenting the semantics of business vocabularies, business facts, and business rules; which allows their verbalization in a controlled vocabulary readily understandable by business people.
The fact-oriented approach of SBVR stems from the Business Rules Manifesto[1], stating that *rules builds on facts, and facts build on concepts as expressed by terms*. Therefore, *terms express business concepts, facts make assertions about these concepts, and rules constrain and support these facts*. SBVR supports such approach by providing *noun concepts* and *verb concepts* respectively corresponding to the notions of *terms* and *facts*. Figure 1 shows the structural organization of such components.
A *noun concept* is a concept that is the meaning of a noun or noun phrase, which is specialized by: (1) *object types*, which are noun concepts classifying things on the basis of their common properties; (2) *individual concepts*, which are concept corresponding to only one object thing and (3) *roles*, which are noun concepts corresponding to things based on their playing a part, assuming a function or being used in some situation. Additionally, *fact type roles* are defined as those roles that specifically characterizes its instances by their involvement in an instance of a given fact type.
A *verb concept* - also named *fact type* - is a concept that is the meaning of a verb phrase that involves one or more noun concepts, representing unary, binary or n-ary relations.

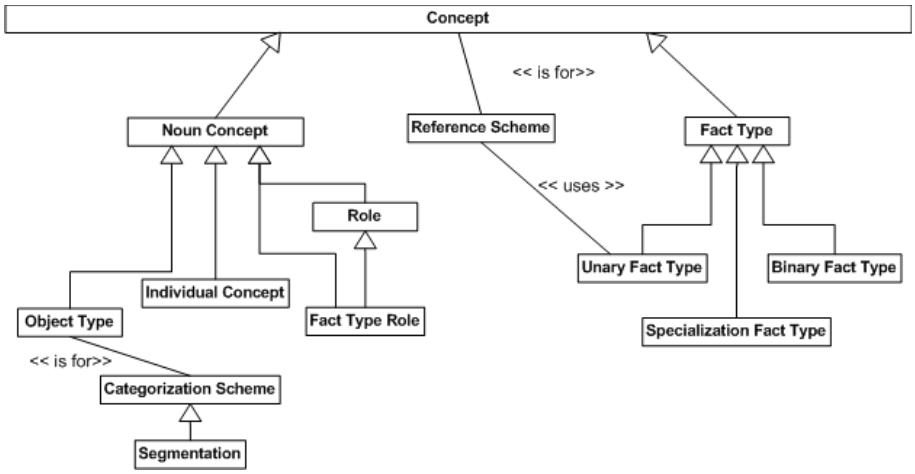

Figure 1: Noun and verb concepts in SBVR

Finally, a SBVR rule is an element of guidance that introduces an obligation or necessity and distinguishing two general types: (1) *structural rules*, which describe the way the business chooses to organize the things it deals with; and (2) *operative rules*, which govern the conduct of business activity by describing business processes. Both types of rules are built imposing restrictions over fact types by using quantifiers, logical operators, etc. Figure 2 shows the structural organization of quantifications and logical operations.
As early stated, SBVR adopts a linguistic approach that allows to define vocabularies and express operative rules. According to this insight, SBVR defines a Controlled Natural Language (CNL) - named SBVR Structured English - and describes the way to mechanically mapping such CNL expressions to SBVR formal concepts.

4 OWL 2 Overview

The OWL 2 Web Ontology Language (OWL 2) is the latest version of an ontology language proposed by the World Wide Web Consortium (W3C) for the development of the Semantic Web [22], but it has gradually evolved as a de-facto standard for a broad spectrum of applications.

[1] http://www.businessrulesgroup.org/brmanifesto.htm

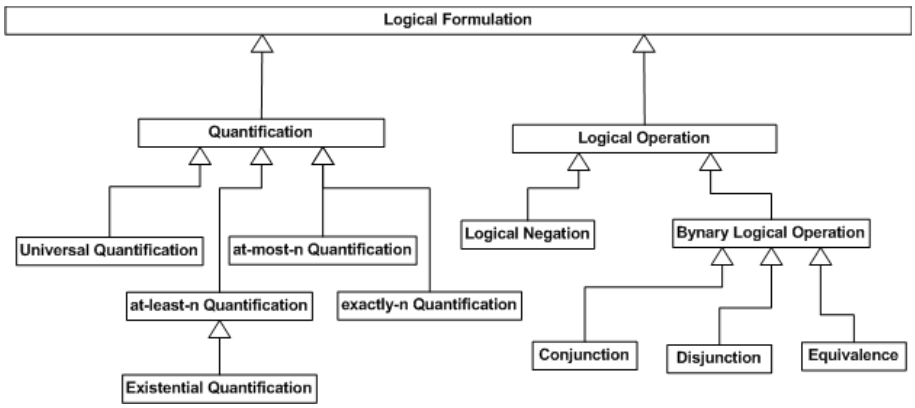

Figure 2: Quantifications and logical operations in SBVR

OWL 2 ontologies provide classes, properties, individuals, and data values, and are stored as Semantic Web documents. An OWL 2 ontology is a formal description of a domain of interest rooted in three syntactic categories that are interpreted under a standardized semantics, which allows useful inferences to be drawn. Figure 3, Figure 4 and Figure 5 depict the structural specification of such categories:

- *Entities* such as classes, properties, and individuals. They are the basic elements of an ontology and are identified by Internationalized Resource Identifiers (IRIs) [8]. For example, a class *a:Person* can be used to represent the set of all people, the object property *a:parentOf* can be used to represent the parent-child relationship and the individual *a:Peter* can be used to represent a particular person called "Peter".

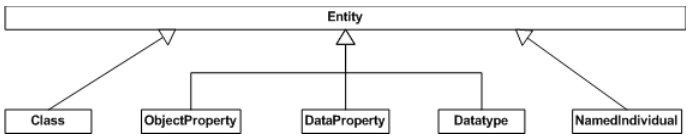

Figure 3: Entities in OWL 2 ontologies

- *Expressions*, representing complex notions in the domain being described. For example, a class expression describes a set of individuals in terms of the restrictions on the individuals characteristics.

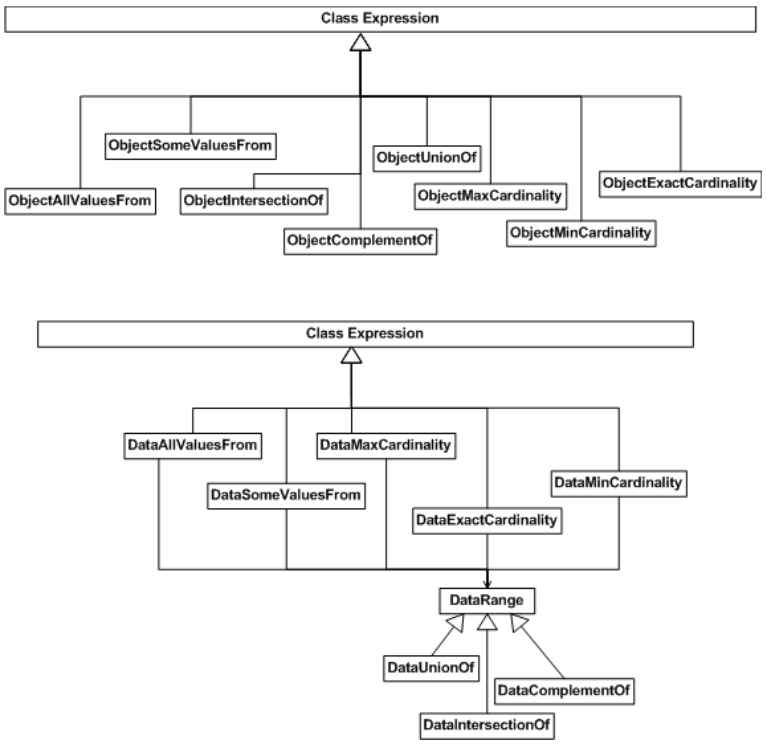

Figure 4: Expressions in OWL 2 ontologies

- *Axioms*, which are statements asserted to be true in the domain being described. For example, a subclass axiom state that the class *a:Student* is a subclass of the class *a:Person*.

OWL 2 ontology language defines several concrete syntaxes that can be used to serialize and exchange ontologies. Among them, the functional style syntax is defined in the OWL 2 structural specification [22] with the aim to state the semantics of OWL 2 constructors and allow a compact writing of ontologies. Following the structural specification insight, the rest of this paper uses such syntax to state OWL 2 ontology expressions.

5 SBVR to OWL 2 Mappings

Mappings presented in this section allow the automatable generation of an OWL 2 ontology from the SBVR specifications of a business domain. Transformations are rooted on the structural specification of both standards and are depicted in subsections below by grouping and sequencing them according to the inherent logical order of the subject matter itself. In addition to their theoretical expression, the mappings are illustrated by building an ontology that reflects the business knowledge exposed by a case study[2].

[2]Ontology generated from the example can be found in http://code.google.com/p/eurent-mapping-case-study/

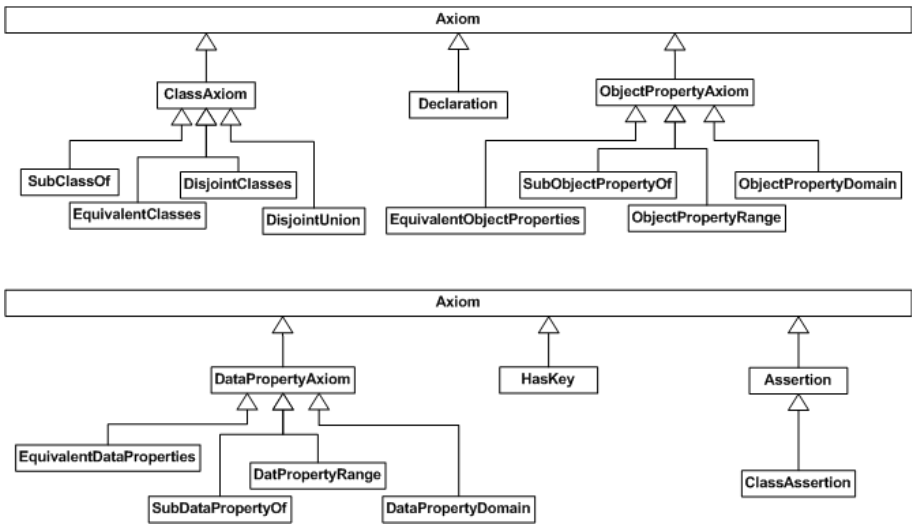

Figure 5: Axioms in OWL 2 ontologies

The case study has been drawn from the Annex E of the SBVR specification [3], depicting the business service of a fictitious car rental company with branches in several countries. This case study has been introduced by the OMG as a common real world model of business enterprise that researchers and system developers can use to illustrate the capabilities of their proposals. Although just a fragment of the case study is presented, it result complete enough in order to depict the usefulness of the proposed mappings. Figure 6 exposes the movements of cars among branches used to illustrate the mappings. EU-Rent rents cars to its customers. Different models of car are offered, organized into groups. All cars in a group are charged at the same rates. A rental booking specifies the car group required, the start and end dates/times of the rental and the EU-Rent branch from which the rental is to start. Optionally, the reservation may specify a one way rental - in which the car is returned to a branch different from the pick up branch - and may request a specific car model within the required group. Furthermore, the rentals are classified based on whether it crosses local area or international boundaries.

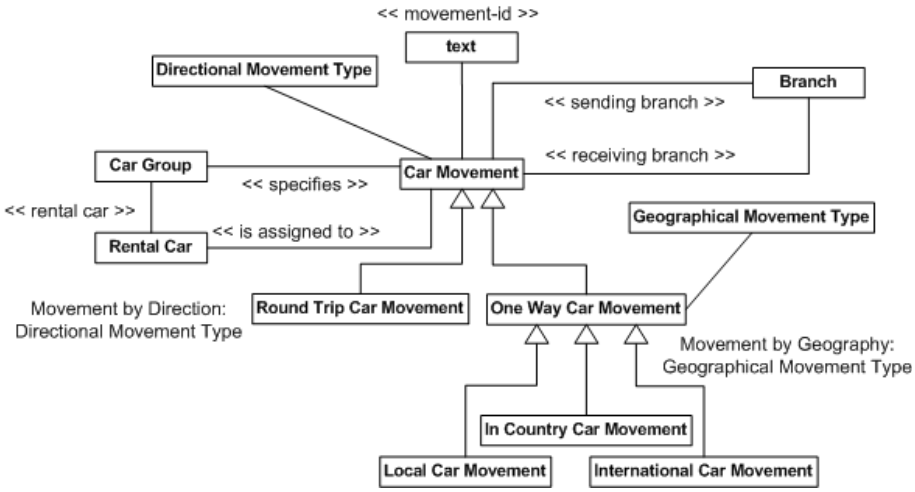

Figure 6: EU-Rent car movement among branches

5.1 Core Mappings

As early stated, *object types*, *individual concepts*, *fact types* and *fact type roles* constitute the core of SBVR metamodel. Therefore, their mappings to OWL 2 expressions are required for more complex translations:

1. Each *object type ot* is mapped to *Declaration(Class(a:ot))*

[3] Annex E (Pp. 267) of the SBVR specification. Accessible in: http://www.omg.org/spec/SBVR/1.0/

2. Each *individual concept ic* of an object type *ot* is mapped to:
   Declaration(NamedIndividual(a:ic)
   ClassAssertion(a:ot a:ic)

3. Each *unary fact type uft* is mapped to:
   Declaration(DataProperty(a:uft)
   DataPropertyDomain(a:uft a:ClassOne)
   DataPropertyRange(a:uft a:DataRangeOne)

4. Each *binary fact type bft* is mapped to:
   Declaration(ObjectProperty(a:bft)
   ObjectPropertyDomain(a:bft a:ClassOne)
   ObjectPropertyRange(a:bft a:ClassTwo)

5. Each *fact type role ftr* is mapped by using the *SubObjectPropertyOf* and *ObjectPropertyChain* OWL 2 axioms, as proposed in [10].

The core of the ontology reflecting the business knowledge exposed by the EU-Rent example is built by applying the translations before depicted. As first step, the mappings of the main object types of the case study are shown below.

Table 1: SBVR specification of "car movement" concept

car movement	
Concept Type:	*object type*
Definition:	Planned movement of a rental car of a specified car group from a sending branch to a receiving branch.
Description:	A car movement meets the business requirement that a car of a given group has to be moved between branches. A specific car will be assigned to it at some time, not necessarily when the requirement is first identified.

Table 2: SBVR specification of "branch" concept

branch	
Concept Type:	*object type*
Definition:	Rental organization unit that has rental responsibility.

Table 3: Mapping of main object types

Declaration(Class(eurent:Car_Movement))
Declaration(Class(eurent:Branch))

Characteristics of object types are expressed. In this case, just the *Car_Movement* class have such kind of attribute:

Table 4: SBVR specification of "has movement ID" fact type

car movement has movement-id	
Concept Type:	unary fact type
Necessity:	Each car movement has exactly one movement-id

Table 5: Mapping of unary fact type

Declaration(DataProperty(eurent:has_Movement-ID))
DataPropertyDomain(eurent:has_Movement-ID eurent:Car_Movement)
DataPropertyRange(eurent:has_Movement-ID xsd:string)

Binary fact types are expressed as properties of classes as shown following. A binary fact type translation is presented below, and the rest of them share the same structure.

Table 6: SBVR specification of "specifies car group" fact type

car movement specifies car group	
Concept Type:	binary fact type
Synonymous Form:	Car group is specified in car movement.
Necessity:	Each car movement specifies exactly one car group.

Table 7: Mapping of binary fact type

Declaration(ObjectProperty(eurent:specifies_Car_Group))
ObjectPropertyDomain(eurent:specifies_Car_Group eurent:Car_Movement)
ObjectPropertyRange(eurent:specifies_Car_Group eurent:Car_Group)

In the case study, *Branch* object type adopts two roles by assuming the function of sending or receiving branch of a car movement. Expressions below shows the mapping of the *sending* role - the other one is made in the same way -.

Table 8: SBVR specification of "sending branch" role

sending branch	
Concept Type:	role
Definition:	Branch that is the origin of a car movement.

Table 9: Mapping of role

Declaration(Class(eurent:Sending_Branch))
SubObjectPropertyOf(ObjectPropertyChain(eurent:has_Sending_Branch eurent:is_role_of) eurent:has_Origin_Branch)
SubObjectPropertyOf(eurent:has_Sending_Branch eurent:has_Origin_Branch)

Where *is_role_of* is an OWL 2 object property whose domain and range are the *Sending Branch* and the *Branch* classes, respectively.
Finally, although this portion of the case study does not consider the modelling of individuals, to the aim of illustrate the translations it is possible to suppose the existence of a *Branch* named *LAX_Airport_Agency*.

Table 10: SBVR specification of "LAX Airport Agency" individual concept

LAX_Airport_Agency	
Concept Type:	individual concept
General Concept:	branch

Table 11: Mapping of individual concept

Declaration(NamedIndividual(eurent:LAX_Airport_Agency))
ClassAssertion(eurent:Branch eurent:LAX_Airport_Agency)

5.2 Quantifications Mappings

Quantifications are defined by SBVR as logical formulations introducing a variable and having either the meaning: all referents of the variable satisfy a scope formulation; or a bounded number of referents of the variable exist and satisfy a scope formulation, if there is one. According to such definition and the previous translations, the mappings of the SBVR quantifications are presented depending on the arity of the fact type they ranges over:

1. *Universal quantification*

   If the logical formulation scopes over a unary fact type, the expression is mapped to *DataAllValuesFrom(a:DataPropertyOne a:DataRangeOne)*

   If the logical formulation scopes over a binary fact type, the expression is mapped to *ObjectAllValuesFrom(a:ObjectPropertyOne a:ClassOne)*

2. *Existential quantification*

   If the logical formulation scopes over a unary fact type, the expression is mapped to *DataSomeValuesFrom(a:DataPropertyOne a:DataRangeOne)*

   If the logical formulation scopes over a binary fact type, the expression is mapped to *ObjectSomeValuesFrom(a:ObjectPropertyOne a:ClassOne)*

3. *at-most-n quantification*, where "n" is a non-negative integer

   If the logical formulation scopes over a unary fact type, the expression is mapped to *DataMaxCardinality(n a:DataPropertyOne a:DataRangeOne)*

   If the logical formulation scopes over a binary fact type, the expression is mapped to *ObjectMaxCardinality(n a:ObjectPropertyOne a:ClassOne)*

4. *at-least-n quantification*, where "n" is a non-negative integer

   If the logical formulation scopes over a unary fact type, the expression is mapped to *DataMinCardinality(n a:DataPropertyOne a:DataRangeOne)*

   If the logical formulation scopes over a binary fact type, the expression is mapped to *ObjectMinCardinality(n a:ObjectPropertyOne a:ClassOne)*

5. *exactly-n Quantification*, where "n" is a non-negative integer

   If the logical formulation scopes over a unary fact type, the expression is mapped to *DataExactCardinality(n a:DataPropertyOne a:DataRangeOne)*

   If the logical formulation scopes over a binary fact type, the expression is mapped to *ObjectExactCardinality(n a:ObjectPropertyOne a:ClassOne)*

Remaining SBVR quantifications - *at-most-one*, *exactly-one*, and *numeric-range* quantifications - are easily translatable in terms of the above presented mappings.
Three exactly-n quantification mapping (where n = 1) is shown following.

Table 12: SBVR specification of "car movement has receiving branch" fact type

car movement has receiving branch	
Concept Type:	binary fact type
Necessity:	Each car movement has exactly one receiving branch.

Table 13: SBVR specification of “car movement has sending branch” fact type

car movement has sending branch	
Concept Type:	binary fact type
Necessity:	Each car movement has exactly one sending branch.

Table 14: Mapping of exact quantification

ObjectExactCardinality(1 eurent:has_Receiving_Branch eurent:Receiving_Branch)

Table 15: Mapping of exact quantification

ObjectExactCardinality(1 eurent:has_Sending_Branch eurent:Sending_Branch)

5.3 Logical Operations Mappings

SBVR defines logical operations as those formulations of meaning based on only the truth or falseness of the meanings of its logical operands. In correspondence with the previous translations, the mappings of the SBVR logical operations are presented depending on the types of the involved logical operands:

1. *Logical Negation*

   If the logical operand is an object type, then the expression is mapped to *ObjectComplementOf(a:operand)*

   If the logical operand is a literal, then the expression is mapped to *DataComplementOf(a:operand)*

2. *Conjunction*

   If both logical operands are object types, then the expression is mapped to *ObjectIntersectionOf(a:operand1 a:operand2)*

   If both logical operands are literals, then the expression is mapped to *DataIntersectionOf(a:operand1 a:operand2)*

3. *Disjunction*

   If both logical operands are object types, then the expression is mapped to *ObjectUnionOf(a:operand1 a:operand2)*

   If both logical operands are literals, then the expression is mapped to *DataUnionOf(a:operand1 a:operand2)*

4. *Equivalence*

   If both logical operands are object types, then the expression is mapped to *EquivalentClasses(a:operand1 a:operand2)*

   If both logical operands are individual concepts, then the expression is mapped to *SameIndividual(a:operand1 a:operand2)*

   If both logical operands are unary fact types, then the expression is mapped to *EquivalentDataProperties(a:logicaloperand1 a:logicaloperand2)*

   If both logical operands are binary fact types, then the expression is mapped to *EquivalentObjectProperties(a:operand1 a:operand2)*

Remaining SBVR logical operations - *exclusive disjunction*, *nand-formulation*, *nor formulation*, and *whether-or-not formulation* - are translatable by the logical combination of the above presented mappings.
As an example, the mapping below depicts a disjunction expression stating the geographical criteria of rentals classification are based on whether it crosses local area or international boundaries.

Table 16: Mapping of disjunction

Declaration(Class(eurent:Geographical_Movement_Type))
Declaration(Class(eurent:InCountry_Car_Movement))
Declaration(Class(eurent:International_Car_Movement))
Declaration(Class(eurent:Local_Car_Movement))
SubClassOf(eurent:Geographical_Movement_Type ObjectUnionOf(eurent:InCountry_Car_Movement eurent:International_Car_Movement eurent:Local_Car_Movement))

5.4 Identifiers, Specializations and Classification Mappings

1. *Reference Scheme*, is the SBVR way of identifying instances of a given concept. However, SBVR considers only characteristics to be used as reference schemes, so the expression is mapped to *HasKey(a:ClassExpression a:DataPropertyExpressionOne)*
   In the case study, the *has_Movement-ID* characteristic is used as the identifier of the individuals belonging to the *Car_Movement* object type. The corresponding translation is shown below.
   HasKey(eurent:Car_Movement () (eurent:has_MovementID))

2. *Specialization*, is a fact type representing relationships between a more general and a more specific concept.

   If both concepts are object types, then the expression is mapped to *SubClassOf(a:concept1 a:concept2)*

   If both concepts are unary fact types, then the expression is mapped to *SubDataPropertyOf(a:concept1 a:concept2)*

   If both concepts are binary fact types, then the expression is mapped to *SubObjectPropertyOf(a:concept1 a:concept2)*

   If concept1 is an object type and concept2 is an individual concept, then the expression is mapped to *ClassAssertion(a:concept1 a:concept2)*
   An application of the specialization mapping over the case study has been shown in the disjunction expression stated in the previous subsection.

3. *Categorization* and *Segmentation*
   A possible situation is the need of categorize the individuals belonging to certain entity according to a set of different criteria. Such situation is presented in the case study, where the car movements are classified based on whether the car is returned to a branch different from the pick up branch - directional categorization criteria - and on whether the car crosses local area or international boundaries - geographical categorization criteria -. Moreover, the categories belonging to each criteria are disjoint between them. In the SBVR context, such modelling are called categorization and segmentation, respectively. While OWL 2 *ObjectUnionOf* is the way to map a SBVR categorization, OWL 2 *DisjointUnion* are used to translate SBVR segmentations.
   Multiple categorization criteria has been already depicted in the case study through the rentals classifications based on directional or geographical aspects. The translations of such business issue is presented following.

Table 17: Mapping of disjunction

Declaration(Class(eurent:Directional_Movement_Type))
SubClassOf(eurent:Directional_Movement_Type ObjectUnionOf(eurent:Round_Trip_Car_Movement eurent:OneWay_Car_Movement))
DisjointClasses(eurent:Round_Trip_Car_Movement eurent:OneWay_Car_Movement)
DisjointClasses(eurent:OneWay_Car_Movement eurent:Round_Trip_Car_Movement)
Declaration(Class(eurent:Geographical_Movement_Type))
SubClassOf(eurent:Geographical_Movement_Type ObjectUnionOf(eurent:InCountry_Car_Movement eurent:International_Car_Movement eurent:Local_Car_Movement))
DisjointClasses(eurent:InCountry_Car_Movement eurent:International_Car_Movement)
DisjointClasses(eurent:International_Car_Movement eurent:InCountry_Car_Movement)
DisjointClasses(eurent:InCountry_Car_Movement eurent:Local_Car_Movement)
DisjointClasses(eurent:Local_Car_Movement eurent:InCountry_Car_Movement)
DisjointClasses(eurent:International_Car_Movement eurent:Local_Car_Movement)
DisjointClasses(eurent:Local_Car_Movement eurent:International_Car_Movement)

6 A Real Case Validation Example

The real case validation example depicted in this section was performed with the aim of approach the feasibility of the proposed mappings. Such feasibility study is based on the quality assessment of the developed ontology. The example was performed by a three-member group conformed by engineering students, in the context of the last year of the course program of Information System Engineering and entitled "Ontology-based Informations Systems Development", at the Argentinian Technological University in the province of Santa Fe. By taking part in the experiment, participants earned educational credits.
Development of the example involved the ontological specification of the policies governing the student fellowship program of the university. Such policies are stated in a natural language written document [4]. Table 18 depicts a excerpt of the fellowship program policies. Table 19, Table 21, Table 23, Table 25, and Table 27 shows the SBVR specification of the main domain concepts. Table 20, Table 22, Table 24, Table 26, and Table 28 presents the OWL mappings of the SBVR policies expressions. The students made use of text editors to model the business domain according to the followed approach. Ontology implementation was performed by means of Protégé, a free and open source ontology editor [5].

[4] Institutional document depicting such policies can be found in http://csu.rec.utn.edu.ar/docs/php/salida.php3?tipo=ORD&numero=1180&
[5] Support, downloads and documentation can be found in http://protege.stanford.edu/

Table 18: An excerpt of the policies governing the student fellowship program

The profile of the students involve basic information as detailed following: full name, birthday, genre, postal address, email and telephone number.
The students have an identification number assigned by the administration of the university.
The students should follow at least one engineering student program.
The students who have approved two test will be considered regular students.
Students who have started an engineering student program in the current school year will be considered starter students.
Starter students will be considered regular students.
The candidates to a scholarship should follow an engineering study program.
The candidates should be regular students.
The candidates should register to a scholarship program in the corresponding registration period.
The university offers two kinds of scholarships programs: the research and service and the economic assistance programs.
The scholarships programs are defined in the context of an unique school year.
The scholarships programs define a fixed registration period.

Table 19: SBVR specification of "student" concept

student	
Concept Type:	*object type*
Necessity:	Each student has exactly one full name. Each student has exactly one identification number. Each student has exactly one birthday. Each student has exactly one genre. Each student is enrolled on at least one engineering student program.

Table 20: OWL specification of "student" concept

Declaration(Class(policies:Student))
SubClassOf(policies:Student DataExactCardinality(1 policies:has_full_name xsd:string))
SubClassOf(policies:Student DataExactCardinality(1 policies:has_Identification_Number xsd:unsignedLong)))
SubClassOf(policies:Student DataExactCardinality(1 policies:has_birthday xsd:dateTime))
SubClassOf(policies:Student DataExactCardinality(1 policies:has_genre xsd:string))
SubClassOf(policies:Student ObjectMinCardinality(1 policies:is_enrolled_on policies:Engineering_Student_Program))

Table 21: SBVR specification of "regular student" concept

regular student	
Concept Type:	*object type*
General Concept:	*student*
Definition:	Each regular student is an student that has at least two approved test or is a starter student.

Table 22: OWL specification of "regular student" concept

Declaration(Class(policies:Regular_Student))
SubClassOf(policies:Regular_Student ObjectUnionOf(ObjectMinCardinality(2 policies:has_approved policies:Test) policies:Starter_Student))

Table 23: SBVR specification of "studentship" concept

studentship	
Concept Type:	*object type*
Definition:	Each studentship is a research and service studentship or an economic assistance studentship but not both.
Necessity:	Each studentship belongs to exactly one school year. Each studentship has exactly one registration period.

Table 24: OWL specification of "studentship" concept

Declaration(Class(policies:Studentship))
DisjointUnion(policies:Studentship policies:Research_and_Service_Studentship policies:Economic_Assistance_Studentship)
SubClassOf(policies:Studentship ObjectExactCardinality(1 policies:belongs_to policies:School_Year))
SubClassOf(policies:Studentship ObjectExactCardinality(1 policies:has_registration_period policies:Registration_Period))

Table 25: SBVR specification of "research and service studentship" concept

research and service studentship	
Concept Type:	*object type*
General Concept:	*studentship*

Table 26: OWL specification of "research and service studentship" concept

Declaration(Class(policies:Research_and_Service_Studentship))
SubClassOf(policies:Research_and_Service_Studentship policies:Studentship)

Table 27: SBVR specification of "economic assistance studentship" concept

economic assistance studentship	
Concept Type:	*object type*
General Concept:	*studentship*

Table 28: OWL specification of "economic assistance studentship" concept

Declaration(Class(policies:Economic_Assistance_Studentship))
SubClassOf(policies:Economic_Assistance_Studentship policies:Studentship)

6.1 Ontology Quality Assessment

The ontology quality evaluation task was performed by means of OQuaRE [6], a framework conceived for that purpose and based on the SQuaRE standard for software quality evaluation [9].
OQuaRE considers ontologies as artifacts obtained by means of a building process and evaluates them independently of any particular development process. It provides an automatable approach which enables the objective assessment of ontology quality and makes quality evaluation reproducible.
OQuaRe defines a quality model and quality metrics for ontology evaluation. Quality model is divided into a series of dimensions - or *characteristics* - organized into subdimensions - or *subcharacteristics* - which are evaluated by applying a set of automatable metrics. OQuaRE defines the criteria to transform the quantitative scores of each metric into a 1-5 range and establishes that *1 means not acceptable, 3 is minimally acceptable and 5 exceeds the requirements.* After such transformation, score for each subcharacteristic is the

mean of its associated metrics while the score of each characteristic is the mean of its subcharacteristics. The set of characteristics scores is the quality assessment result, enabling the identification of strengths and flaws of an ontology. Characteristics evaluated in the example are defined as follows:

- *Structural* dimension involves formal and semantic properties that are important when evaluating ontologies since it accounts for quality factors such as consistency, formalisation, redundancy or tangledness.
- *Functional adequacy* dimension refers to the appropriateness of the ontology for its intended purpose, according to the categories identified by [18].
- *Maintainability* dimension is related to the capability of the ontologies to be modified for changes in the environment, in requirements or in functional specifications.
- *Compatibility* dimension refers to the ability of two or more ontologies to exchange information and/or to perform their required functions while sharing the same hardware or software environments. The compatibility dimension can be evaluated over a single ontology - although intuitively it involves properties about more than one ontology - given that it is quantitatively assessed by means of a set of metrics applied to each ontology separately.
- *Transferability* dimension is the degree to which the ontology can be transferred from one environment (e.g., operating system) to another.
- *Operability* dimension refers to the effort needed to use the ontology and, in the individual assessment of such use, by a stated or implied set of users.
- *Reliability* dimension is the capability of the ontology to maintain its level of performance under stated conditions for a given period of time.

Three metrics were left out of consideration at the assessment of the aforementioned quality dimensions. *Annotation richness* - i.e., mean number of annotations per class - and *class richness* - i.e., mean number of instances per class - metrics were not assessed since the annotation and instantiation of ontologies were not a part of the task of the experiment. *Attribute richness* metric - i.e., mean number of attributes per class - was not assessed because "attributes" are not a part of the structural specification of the implementation language of the assessed ontologies [22].
OQuaRE also defines *performance efficiency* and *quality in use* dimensions. *Performance efficiency* exposes the relationship between the level of performance of the ontology and the amount of used resources, under stated conditions, taking into account elements such as time of response or memory consumption. *Quality in use* refers to the degree to which the ontology used by specific users meets their needs to achieve specific goals. However, such dimensions were left out of consideration because there was a lack of metrics for their subcharacteristics [6]. Figure 7 shows the quality scores for the ontology developed in the context of the validation example.

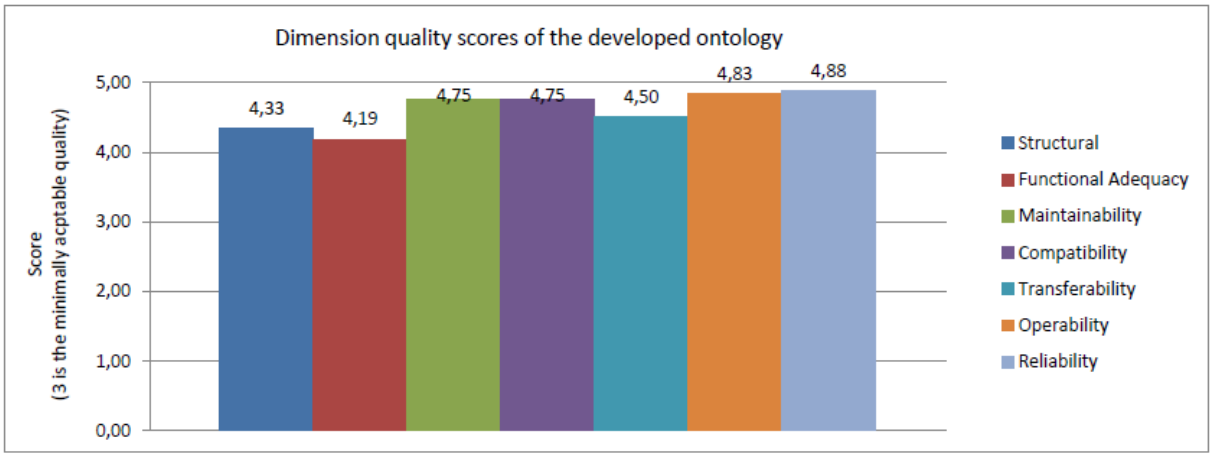

Figure 7: Dimensions quality scores of the developed ontology

[6] A full description of the applied quality model and the obtained results can be found in https://docs.google.com/file/d/0B4QCQLMIO05Yal81VUxlRVhDXzg/edit?usp=sharing

A quickly recognizable outcome is the level of quality shown by the ontology: according to the meaning assigned for OQuaRE to the values of the 1-5 ranking system, it largely outperform the minimally acceptable quality in all considered dimensions. Moreover, the global quality score - which is equal to 4.60 and it is calculated as the mean of all the scores - is very close to the maximal quality value. These results provide a first insight about the feasibility of the proposed mappings, by presenting a real case example where the quality of the developed ontology overcome the acceptable level - as stated by an assessment framework rooted in a widely recognized standard for software quality evaluation.

7 Conclusions and Future Work

An exhaustive and automatable approach for SBVR to OWL 2 transformations has been presented and evaluated through a case study.
However, OWL 2 is not expressive enough to model all the possible semantics of business vocabularies. As an example, it can be took the rule of the case study imposing that the sending and receiving branch of a round-trip car movement must be the same. Such restriction can not be expressed by OWL 2 statements. According to that, future theoretic works involve three main issues.
The first one is focused on the validation and formalization of the mappings. Both goals will be achieved by generating an ontological metamodel of SBVR specification by following an similar approach to that adopted in the building of the Ontology Definition Metamodel (ODM) [12]. The second one is related to the definition of mappings from SBVR to Horn Rules expressed in the SWRL language, with the aim to filling the gap between SBVR and OWL 2 expressive power. Such new set of mappings will take advantage of the mappings formalization before mentioned. The third one is about the analysis of the benefits of integrating the mappings approach to EDON [15], an evolutionary method for building ontologies intended to be used as a structural conceptual model of an information system. In a early stage, translations can be useful in a method that make use of ontologies to encapsulate business rules as a mean to raise the flexibility, extensibility and ease of maintenance of enterprise software systems.
Several study groups have been organized with the aim to obtain early feedback about the application of the mappings. Such evaluation effort have resulted in an exploratory experiment comparing the performance and attitudes of some groups of students building an ontology based on SBVR business rules expressions with another set of groups building an ontology based on traditional glossaries describing domain entities. Although a more structured and detailed analysis of the results will be presented in the future, the practitioners mainly highlighted the focus on the declarative specification of business rules that allows to obtain an implemented ontology from business knowledge in a smooth way.
Finally, practical objectives of the research will be directed towards the implementation of a prototype intended to provide automatable translations from SBVR business domain specifications to OWL 2 ontologies.

References

[1] Alberts, R. and Franconi, E.: An Integrated Method Using Conceptual Modelling to Generate an Ontology-based Query Mechanism. In Proc. of OWLED 2012. Greece. (2012)

[2] Baader, F. and Calvanese, D. and McGuiness, D.L. and Nardi, D. and Patel-Schneider, P.F., eds.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press. New York, USA. (2003)

[3] Calero, C. and Ruiz, F. and Piattini, M., eds.: Ontologies for Software Engineering and Software Technology. Springer. (2006)

[4] Ceravolo, P. and Fugazza, C. and M. Leida: Modeling Semantics of Business Rules. In Proc. of EcoSystems, DEST 07. (2007)

[5] Demuth, B. and Liebau H-B.: An Approach for Bridging the Gap Between Business Rules and the Semantic Web. In Proc. of the RuleML 07, Berlin. (2007)

[6] Duque-Ramos, A. and Lopez, U. and Fernandez-Breis, J. T. and Stevens, R. and Aussenac-Gilles, N.: OQuaRE: a SQuaRE-based approach for evaluating the quality of ontologies. Journal of Research and Practice in Information Technology, 43, 159-173.

[7] Franconi, E. and Mosca, A.: The formalisation of ORM2 and its encoding in OWL2. KRDB Research Centre Technical Report KRDB12-2, Faculty of Computer Science, Free University of Bozen-Bolzano. Italy. (2012)

[8] Internet Engineering Task Force (IETF): RFC 3987: Internationalized Resource Identifiers (IRIs). Accessible in: http://www.ietf.org/rfc/rfc3987.txt (2005)

[9] International Organization for Standardization (ISO) ISO/IEC 25000 2005, Software Engineering - Software Product Quality Requirements and Evaluation (SQuaRE)

[10] Karpovic, J. and Nemuraite, L.: Transforming SBVR Business semantics into Web Ontology Language OWL 2: Main Concepts. In Proc. 17th International Conference on Information and Software Technologies IT 2011. (2011)

[11] Object Management Group (OMG): Model Driven Architecture (MDA) Accessible in http://www.omg.org/mda/ (2000)

[12] Object Management Group (OMG): Ontology Definition Metamodel (ODM). Version 1.0. Accessible in: http://www.omg.org/spec/ODM/1.0 (2009)

[13] Object Management Group (OMG): Semantics of Business Vocabulary and Business Rules (SBVR). Version 1.0: Formal Specification. Accessible in: http://www.omg.org/spec/SBVR/1.0/ (2008)

[14] Object Role Modeling. The official site for conceptual data modeling. Accessible in: http://orm.net/

[15] Reynares, E. and Caliusco M. A. and Galli, M. R.: EDON: A Method for Building an Ontology as Software Artefact. In Proc. 41st Argentine Conference on Informatics - 13th Argentine Symposium on Software Engineering (JAIIO - ASSE 2012) La Plata, Buenos Aires, Argentina. (2012)

[16] REWERSE Reasoning on the web: R2ML The REWERSE I1 Rule Markup Language. Accessible in: http://oxygen.informatik.tu-cottbus.de/rewerse-i1/?q=node/6 (2006)

[17] Ruotsal, T.: Methods and Applications for Ontology-Based Recommender Systems (PhD. Thesis). Aalto University School of Science and Technology. Finland. (2010)

[18] Stevens, R. and Wroe, C. and Gobel, C. and Lord, P.: Application of ontologies in bioinformatics. In Handbook of Ontologies in Informations Systems. Staab, S. and Studer, R. (eds). pp 635-658. Springer. (2008)

[19] World Wide Web Consortium (W3C): OWL Web Ontology Language. Guide. Accessible in: http://www.w3.org/TR/owl-guide/ (2004)

[20] World Wide Web Consortium (W3C): SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Accessible in: http://www.w3.org/Submission/SWRL/(2004)

[21] World Wide Web Consortium (W3C): OWL 2 Web Ontology Language. Document Overview. Accessible in: http://www.w3.org/TR/owl2-overview/ (2009)

[22] World Wide Web Consortium (W3C): OWL 2 Web Ontology Language. Structural Specification and Functional-Style Syntax. Accessible in: http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/ (2009)

[23] Zacharias, V.: Rules as simple way to model knowledge: Closing the gap between promise and reality. In Proc. 10th Int. Conf. on Enterprise Information Systems (2008).