Modeling Semantics of Business Rules

Paolo Ceravolo  Cristiano Fugazza  Marcello Leida

Università degli Studi di Milano,
Dipartimento di Tecnologie dell’Informazione
via Bramante, 65
26013 Crema (CR), Italy
Email: \{ceravolo,fugazza,leida\}@dit.unimi.it

Abstract—Organizations are showing growing interest in paradigms where business models and services compatibility is adaptively tested, e.g., by applying automatic systems to check business rules consistency. In this paper, we build on the original proposal by OMG of using first-order logics for representing business vocabularies and propose an approach based on Description Logics (DL) as formal logic support for business rules. By translating SBVR business vocabularies and rules into OWL DL ontologies, standard inference procedures of DL can be applied to check the business model consistency in the open-world, which is the default interpretation of SBVR models. Moreover, SBVR facts that cannot be expressed with OWL DL are translated into SWRL rules so that they can then be integrated with the starting ontology and evaluated, albeit restricted to the closed-world made of known facts. We exemplify this process by translating a fragment of the EU-Rent example, drawn from the SBVR specification, into a OWL+SWRL knowledge base.

I. INTRODUCTION

Knowledge base representation and reasoning systems are central tasks in automatic enterprise modeling. International organizations like OMG [17] are specifying new architectures aimed at supporting the migration of enterprises to Virtual E-Enterprise (VEE) systems. The primary goals of this architecture are portability, interoperability and reusability through architectural separation of concerns. One interesting aspect of this systems will allow business experts to model the desired business domain in a natural language style supported by a sound and firm vocabulary that supply the necessary semantics to describe the system independently of the platform that will support it. The OMG Specification on Semantics of Business Vocabulary & Business Rules (SBVR) [5], inserted in the wider Model Driven Architecture (MDA) [14] project, is based on this assumption. The Knowledge Management Group of the Department of Information Technologies [12] is developing a tentative solution to implement a SVBR-based system using Description Logics (DL) [1] to support reasoning, where possible, and First Order Logic (FOL) in case the expressivity of DL lacks in to model the semantics of rules. This work is a position paper mainly aimed at exploring the feasibility of the proposed solution and the main problems arising. Specifically, the paper illustrates some mapping between SVBR and OWL DL and shows that the only use of DL is not enough to model the complete semantics of SBVR vocabularies.

The paper is organized as follows: Sec. II describes the proposal for standardization on the Semantics of Vocabularies & Business Rules made by OMG: it gives a general overview of the MDA, dwelling upon the the Business Semantics of Business Rules (BSBR) module, described in [5] and in particular its grounding in First Order Logic (FOL). Sec. III presents our approach based on DL as underpinning logic formalism for the SBVR, and the problems that arise with this assumption. Sec. IV shows the steps involved in the translation process from business rules expressed in controlled English to a Platform Independent Model (PIM). Sec. V describes an example of a practical application of the solution proposed in Sec. IV. This main section shows a partial mapping of the EU-Rent example with the OWL and SWRL languages, highlights the open problems we have to deal with, and propose a tentative solution. Finally, Sec. VI focuses on the future steps of our work.

II. BUSINESS SEMANTICS OF BUSINESS RULES

The global information infrastructure has made it possible for different enterprises to integrate their business activities as a VEE. A key success factor for VEEs is achieving seamless integration of legacy applications, as well as heterogeneous network resources. Companies that have attempted “best of breed” business process integration have found it to be expensive, time consuming and fraught with difficulties. For this reason, organizations wishing to start a virtual business are showing growing interest in automatically testing their business models and services compatibility. As a response to this demand, OMG published a set of documents, grouped under the Business Modeling & Integration Domain Task Force (BEIDTF) project [4]. The SBVR model has been presented as result of the Request for Proposal on Business Semantics of Business Rules (BSBR) made by OMG, which is a part of the Business Model Layer in the MDA, as shown in Fig. 1. The purpose of SBVR is to describe formally and without ambiguities the semantics of a business model, allowing for generating a Platform Independent Model (PIM) of the corresponding business system. The beneficiaries of SBVR include business analysts and modellers, as well as Business Vocabulary and Rules administrators and software tool
developers. These communities are sub-communities of the business enterprise; they share a common base of knowledge made by vocabularies, facts and rules called, in the SBVR, Body of Shared Meanings (BSM). The BSM can be seen as a structured vocabulary of terms and rules usable by the various communities (i.e., UML models for tool builders, Vocabularies and Rules for the business modeler, etc.). However, OMG has chosen to ground BSM representation in logics, in order to enable vocabularies’ validation and further development via a reasoning process. As shown in Fig. 1, SBVR has a sound theoretical foundation in formal logics, which is the backbone of the Business Representation production process (representation and reasoning). The formal logic block of SBVR is based on First Order Predicate Logic (FOL), with some extensions to higher-order and Modal Logic. SBVR does not specify a FOL to use.

III. DESCRIPTION LOGICS AS FORMAL LOGIC SUPPORT TO SBVR

SBVR is intended to model and capture the semantics of business facts and business rules that are expressed either explicitly or implicitly. Business domain experts are in charge to model business behavior using SBVR vocabularies and it can be a hard task for complex systems. SBVR has a strong formal logic underpinning with the purpose to make automated reasoning on rules that may contain inconsistencies and lead to contradictory situations. Modeling semantics in a firm and sound logic formalism also help to discover implicit knowledge that is hidden in the rules. This additional knowledge is discovered by reasoning on the initial knowledge base provided by rules, helping the business domain experts to discover inconsistent situations and tool developers to find out mechanisms implicitly described by rules. In this paper we propose, as formal logic support for the SBVR production process, an approach based on DL. Our choice of DL as a knowledge representation formalism for business rules is due to the fact that DL is based on a subset of FOL, providing i) a declarative formalism for representation and expression of knowledge and ii) sound, tractable reasoning methods [1]. However, as shown in Sec. V, DL is not expressive enough to model all the possible semantics of business vocabularies, which will be modeled using Horn Rules expressed in the SWRL formalism [24]. SWRL is an extension to OWL made by the combination of RuleML and OWL, grounded in a FOL, and with more expressive power than DL. This extension of our initial DL-based solution with SWRL rules, even a DL-safe subset of them [15], brings to undecidability when reasoning according to the Open World Assumption (OWA), which is the basic world assumption in SBVR. A solution is to introduce an epistemic operator in the DL, but this operation leads to a Closed World Assumption (CWA) reasoning which is not included in the functionalities of firm reasoners. Some prototypal applications to handle OWL+SWRL are being developed, like Hoolet [11]. Hoolet is an implementation of an OWL DL reasoner that uses a first order theorem prover. The ontology is translated to collection of axioms (in an obvious way based on the OWL semantics) and this collection of axioms is then given to a first order prover for consistency checking. The reasoner integrated in Hoolet is Vampire [26], an automatic theorem prover for first-order classical logic developed in the University of Manchester. It uses a number of standard redundancy criteria and simplification techniques for pruning the search space and simplify the reasoning process. A feasible approach is to switch from OWA to CWA and adopt Prolog-like languages to infer from knowledge represented as rules but, as shown in [15], this process cannot be executed easily and sequentially without losing derivable information.

A DL knowledge base is traditionally divided into two main parts: the terminology or schema, i.e. a vocabulary of the application domain called the TBox, and assertions, which are named individuals expressed in terms of the vocabulary, called the ABox. TBox and ABox elements are expressed in a description language and represent two separate meta-levels in the application domain. In other words, elements in the TBox are the definitions of elements instantiated in the ABox. Using DL as the underlying logics for business rules, we can adopt a well-known semantic Web-style description language, OWL [21], as our standard metadata language for business rules. The OWL syntax is defined as an XML serialization and encoded into RDF Graphs composed of triples. The OWL specification provides three sublanguages, providing different expressive power. The sublanguage we are using, OWL DL, was designed to take advantage of DL decision procedures and reasoning systems. Inference in the underlying $SHOIN(D)$ [10] DL is complete (given a OWL DL system, all its logic entailments are guaranteed to be computed) and decidable (all computations will finish in finite time). Thanks to the
decidability of $\text{SHOIN}(D)$, we can take advantage of a suite of inference services. In particular, we have:

- **Consistency checking**, which ensures that an ontology does not contain any contradictory fact. In DL jargon, this is the operation of checking the consistency of an ABox with respect to a TBox.
- **Concept satisfiability**, which checks if it is possible for a class to have any instances. If a class is unsatisfiable, then defining an instance of that class will cause the whole class hierarchy to be inconsistent.
- **Classification**, which computes the subclass relations between every named class to create the complete class hierarchy. The class hierarchy can be used to answer queries such as getting all or only the direct subclasses of a class.
- **Realization**, which finds the most specific classes that an individual belongs to or, in other words, computes the direct types for each of the individuals. Realization can only be performed after classification since direct types are defined with respect to a class hierarchy. Using the classification hierarchy, it is also possible to get all types for a given individual.

These algorithms are implemented in a number of software tools collectively known as reasoners. With respect to generic FOL, for which few sound and mature reasoners are available, well-engineered reasoners and other tools do exist for DL, supporting advanced features like standard query interfaces and clean separation between facts (ABox) and fact types (TBox). Available reasoners include RacerPro [8], FaCT [9] and Pellet [25]. Other tools, like Mandarin and Prova, are being tested for business applications [19]. Also, the DL Implementation Group has developed an interface [2] which is now an emerging standard for accessing DL reasoners via an HTTP-based interface.

IV. FROM DL-SUPPORTED BUSINESS RULES TO A PLATFORM INDEPENDENT MODEL

Our approach is aimed at translating DL-based business rules into a PIM (Platform Independent Model, see Fig. 1). For each business rule (BR), our procedure involves the following phases:

- **Identify vocabulary symbols**: BR is tagged according to the Controlled Language used to express business rules. Here, we rely on SBVR Structured English which is a set of terms, verbs and keywords with their mapping to a logical formulation.
- **Parse according to language rules**: tagged BR is parsed to identify names, terms, concepts, verbs, keywords and relations between them.
- **Restate as facts of logical formulation**: parsed BR is restated/expanded using DL inference procedures.\(^1\)
- **Represent facts of logical formulation as objects**: the DL-based formulation of BR is expressed in OWL DL.

- **Write objects as XML**: the entire SBVR knowledge representation (including current BR) is serialized as a XML file that will be used to generate the PMI MOF model (e.g., the UML representation) of the system.

Of course, mapping SBVR into a MOF model is not straightforward; rather, it will require two additional steps. First, SBVR Vocabularies will be mapped (using mapping rules described in [5]) to a MOF model representing facts that can be meant by any atomic formulation expressible via the business vocabulary.\(^2\) Then, the full SBVR will have to be captured in terms of the MOF model created by the first mapping. This second step includes the definitions of concepts, terms, business rules and other facts of the SBVR Metamodel in terms of SBVR Vocabularies.

V. THE EU-RENT EXAMPLE

In this section, we provide the translation between a small fragment of the EU-Rent example\(^3\), presented in Annex E of [5] and OWL DL, enhanced with SWRL constructs. In this phase, we are primarily interested in identifying which aspects of business vocabularies and rules can be effectively modelled with OWL DL (and can therefore take advantage of DL reasoners) and which, on the other hand, require closed-world theorem provers to enforce more general first-order logic constraints stemming from SBVR facts. The UML diagram in Fig. 3 displays some of the generalization and association relationships related to term movement, considered by our example. At a first glance, describing vocabulary structures with UML models and relying on XMI techniques to derive a machine-processable description of the business domain may look appealing. On the other hand, some constraints can be missing as a consequence of such a translation: as an example, terms round-trip car movement and one-way car movement in Fig. 3 are disjoint concepts, because the latter is indirectly defined as the complement of the former, but this cannot be rendered in the diagram without introducing OCL constructs [16] or inter-relationship constraints [7]. We will therefore consider UML diagrams as an intuitive means for sketching the business model and, conversely, rely on the facts expressed in natural language to ground the actual translation process.

A. Context Modeling through DL Definitions

Business vocabularies are, w.r.t. more general business rules, the best candidates to adopt a semantics-aware translation into OWL DL ontologies, inasmuch the vocabulary portrays static information and requirements that maps fairly well to DL concept and role definitions. As an example, a first typology of facts in the EU-Rent example describes the distinguishing features that univocally describe a car movement, as an example, movement-id, sending branch, and receiving branch. Note that, the first role is not of particular interest since it can be

\(^1\)A bootstrap process, aimed at providing the reasoner with an initial knowledge base, may be necessary before this phase.

\(^2\)This mapping does not capture the full semantics of SBVR.

\(^3\)The EU-Rent example is a case-study introduced by the OGM group as a common real world model of business enterprise, which researchers and system developers can use to illustrate the capabilities of their products.
straightforwardly rendered as a datatype property. However, all these fact types have the following form:

\[
\text{car movement has sending branch}
\]

\[
\text{each car movement has exactly one sending branch}
\]

Firstly, the fact type introduce concept definitions CarMovement and Branch, which are addressed by both associations sending branch and receiving branch in Fig. 3. We adopt CamelCase notation to turn vocabulary terms into valid tokens for naming concepts, we also distinguish concept names from role names by capitalizing the first term. Secondly, rather than introducing general purpose object properties modelling verbs in the EU-Rent vocabulary, such as \textit{has} in (1), we coin specific properties \textit{hasSendingBranch} and \textit{hasReceivingBranch} \footnote{This approach agrees with UML models, where associations are always specific to the intervening entities (i.e., are not shared between distinct classes) and univocally determine domain and range properties.}. The \texttt{each} keyword in the original fact type defines a necessary condition that ought to be shared by all instances of class CarMovement. Properties are also made functional to express the \texttt{exactly one} constraint. The assertions derivable from (1) are the following:

\[
\text{CarMovement} \subseteq \exists \text{hasSendingBranch}.\text{Branch} \\
\text{T} \subseteq \leq 1 \text{hasSendingBranch}
\]

Although these axioms do not actually constrain the domain and range of property \textit{hasSendingBranch} (primarily to keep the example simple), they are sufficient to require CarMovements to having an associated sending Branch and also disallow any individual to having more than one. Another set of facts specializes term \textit{car movement} with terms expressing directional properties. Note that, the definition of \textit{one-way car movement} indirectly relies on the definition of \textit{round-trip car movement} by means of characteristic \textit{being round-trip}:

\[
\text{round-trip car movement:}
\]

\[
\text{car movement that is roundTrip}
\]

\[
\text{one-way car movement:}
\]

\[
\text{car movement that is not roundTrip}
\]

Moreover, \textit{one-way car movement} is further specialized according to the geographic distance between sending branch and receiving branch:

\[
\text{local car movement:}
\]

\[
\text{one-way car movement that is local}
\]

\[
\text{international car movement:}
\]

\[
\text{one-way car movement that is international}
\]

\[
\text{in-country car movement:}
\]

\[
\text{one-way car movement that is not local and is not international}
\]

Here again, the notion of \textit{in-country car movement} relies on the characteristics associated with the other terms (i.e., \textit{being local} and \textit{being international}) for its definition. As for \textit{being round-trip}, we refrain from defining these characteristics because, as we will see in the next section, they cannot be rendered with OWL DL constructs and require the adjacent SWRL to be expressed. However, the facts categorizing the specific types of \textit{car movement} can be expressed with the following inclusion axioms:

\[
\text{OneWayCarMovement} \subseteq \text{CarMovement} \\
\text{RoundTripCarMovement} \subseteq \text{CarMovement} \\
\text{LocalCarMovement} \subseteq \text{OneWayCarMovement} \\
\text{InternationalCarMovement} \subseteq \text{OneWayCarMovement} \\
\text{InCountryCarMovement} \subseteq \text{InternationalCarMovement}
\]
Note that, the indirect complementarity between concepts OneWayCarMovement and RoundTripCarMovement is rendered by defining the latter as the complement of the former. Therefore, in the OWL representation, characteristic being round-trip does not contribute to the definition of subsumption relationships between these concept. Anyway, the characteristic will determine which CarMovements will be asserted as RoundTripCarMovements. To model characteristics, the notion of branch, local area, operating company, and operating country has to be defined. The facts pertaining our example, with some simplification to make it more easily intelligible, are the following:

- Each branch is included in exactly one local area.
- Each local area is included in one operating company.
- Each operating company operates in exactly one operating country.
- Each branch has one country.

Therefore, we need to model part-of relationships between entities in the business domain and these cannot be properly modeled with OWL built-in constructs. Instead, we introduce class CompositeConcept to group class definitions whose individuals are composite entities and define custom properties to take into account functional and non-functional inclusions. Specifically, in the example we model them former with property componentOf and the inverse counterpart hasComponent. The functional relationships between branch, local area, operating company, and operating country can then be portrayed with the following axioms:

- \( \text{Branch} \sqsubseteq \exists \text{componentOf}\text{LocalArea} \)
- \( \text{LocalArea} \sqsubseteq \exists \text{componentOf}\text{OperatingCompany} \)
- \( \text{OperatingCompany} \sqsubseteq \exists \text{operatesIn}\text{OperatingCountry} \)
- \( T \sqsubseteq \leq 1 \text{componentOf} \)
- \( T \sqsubseteq \leq 1 \text{operatesIn} \)

The axioms introduced so far can be loaded onto a reasoner to check the vocabulary consistency, to classify concept definitions, and to populate the ontology with individuals representing the current state of the business domain according to the constraints defined. Fig. 4 shows the hierarchy of concept definitions in the Protégé interface [23]. Standard inference procedures of DL reasoners can then be accessed.

B. Defining SWRL Rules

Unfortunately, the decidable fragment of first-order logic that can be expressed with OWL DL is too limited to encode the possible semantics of business vocabularies. As an example, assigning a country to a branch, i.e. following the graph chain linking a specific instance of concept Branch to the corresponding LocalArea, OperatingCompany, and OperatingCountry, is not possible with OWL DL concept definitions. Moreover, also the simplest characteristic referred to by car movement, the one defining the property of being round-trip, cannot be expressed with OWL DL: defining CarMovements whose fillers of properties hasReceivingBranch and hasSendingBranch are the same individual would involve dynamically defined enumerated classes (with the property filler as the only member) and this is not even possible with OWL Full.

- Car movement being round-trip
- Car movement having sending branch that is the receiving branch of the car movement
- Car movement being local
- Car movement having receiving branch that is included in the local area of the sending branch of the car movement
- Car movement being international
- Car movement having country of sending branch that is not the country of the receiving branch of the car movement

Instead, Horn Rules expressed in the SWRL formalism can compensate for this limitation. Unfortunately, the enhanced expressiveness can easily lead to undecidability unless proper countermeasures are taken [15]: as an example, reasoning in the open-world is no longer advisable. By adding SWRL rules to the OWL DL ontology, the definition of property hasCountry, as well as of the characteristics described above, becomes straightforward. Fig. 5 is showing the FOL definitions expressing the SBVR structures that cannot be rendered with OWL. The enforcement issues introduced by the adoption of generic rules in the modelling of the business domain need to take into account the interchange between the open-world approach provided by DL reasoners and the closed-world assumption of Prolog-like execution models. The work [15] is showing an example of how this integration cannot be accomplished by merely interleaving open- and closed-world evaluations of the business model. The works [6], [13] tackle the problem of integrating both reasoning paradigms. Particularly, the former relies on the distinction between strong and weak negation to indicate translation mechanisms between the open- and closed-world reasoning approaches. Anyway, much work has to be done for the definition of sound and complete decision procedures that can be applied to hybrid knowledge bases of this kind.
VI. CONCLUSIONS AND OUTLOOK

Our future research will focus on DL-based knowledge representation for rule-based languages describing business models and services. Specifically, we intend to investigate DL (as opposed to related formalisms like first-order logic and deontic logics) proposed for representing contracts and obligations [22]) as a way of deriving formal declarative specifications for checking consistency of business rules. We believe our approach can lead to automated consistency checks as well as automated comparison of different rule sets. In the fullness of time, automatic consistency check may lead to dynamically supporting and validating changes to business descriptions. We plan to develop this subject in a future paper.

In the translation example, we have been interpreting the EU-Rent example within the open-world assumption characterizing the novel feature of business domains that could take advantage of DL reasoning services. Consequently, recourse to more general, closed-world inference procedures has been seen as an approximation of the desired behavior. Nevertheless, the semantics of SBVR allows to freely combine local closure in domains that are considered open by default with local openness in closed domains. SBVR also defines semi-closure with respect to fact types, i.e. the behavior implied by functional properties (such as operateIn) and mandatory multiplicities. Moreover, when modeling business domains, SBVR does not encompass the actual enforcement of rules, e.g. the interchange of alethic and deontic interpretations of facts that may coexist in the business domain. Translating vocabularies and rules into logic structures is a first step toward the enforcement of these different categories of constraints.

REFERENCES


[23] Semantic Web Rule Language (SWRL) - http://www.w3.org/SWRL/SWRL/


