A Lithuanian Verbalization Template
for ORM conceptual models and rules

Mustafa Jarrar, Vrije Universiteit Brussel, Belgium. (Contact Author)
Maria Keet, Free University of Bozen-Bolzano, Italy.
Juozas Gordevicius, Free University of Bozen-Bolzano, Italy.

A technical report of the article[1]: *Jarrar, M., Keet, C.M., Dongilli, P.* ***Multilingual verbalization of ORM conceptual models and axiomatized ontologies****. [Submitted].*
URL: http://www.starlab.vub.ac.be/staff/mustafa/orm/verbalization/

Abstract. In the above-mentioned article we describe a novel approach to support *multilingual* verbalization of logical theories, axiomatizations, and other specifications such as business rules. This engineering solution is demonstrated with the Object Role Modeling language, although its underlying principles can be reused with other conceptual models and formal languages, such as Description Logics, to improve its understandability and usability by the domain expert. The engineering solution for multilingual verbalization is characterized by its flexibility, extensibility and maintainability of the verbalization templates, which allow for easy augmentation with other languages than the 11 currently supported.

This report presents the Lithuanian verbalization template file. Given an ORM schema (or an ORM-ML file), and given the verbalization template, a Lithuanian verbalization of the rules and fact types (in the schema) is generated automatically. A comprehensive example of an ORM schema and its corresponding verbalization is generated and given in this report.

1 Introduction

In the above-mentioned article, we present a novel approach to support multilingual verbalization of logical theories, formal axiomatizations, and other specifications such as business rules, ontologies, etc. We demonstrate our approach by providing a flexible and extensible verbalization template for the Object Role Modeling language. This template can be easily customized and translated into other human languages. *This technical report provides the Lithuanian verbalization of the ORM models and rules.* The verbalization of several other languages (including, but not limited to: German, Italian, Arabic, Russian, Dutch, Spanish, and French) can be found at the above-mentioned URL.

The underlying principles of our approach can be reused for other conceptual models and formal languages, such as Description Logics. The objective was to define a template parameterized over a given set of rules, models, or axioms, with as output fixed-syntax pseudo natural language sentences. A simple example is the following: the formal rule

$$\forall x\ (Book(x) \rightarrow \exists y\ (ISBN(y) \wedge Has(x,y)))$$

can be translated into

It is mandatory that each **Book Has** an **ISBN**.

In this way, we enable domain experts themselves to build and/or validate the formal specifications of their domains, without having to know that these sentences are formal axioms; i.e. the underpinning

[1] For Citation use: *Jarrar, M., Keet, C.M., Gordevicius, J.: A Lithuanian Verbalization Template for ORM conceptual models and rules. A technical report of the article: Jarrar, M., Keet, C.M., Dongilli, P. Multilingual verbalization of ORM conceptual models and axiomatized ontologies. [Submitted].*

logics and reasoning services are hidden from the user. Our approach with the provided templates can be reused in modeling business rules, ontologies, knowledge bases, etc. See [H04] for a similar approach to ORM business rules verbalization.

In the following section, we present an ORM example followed by verbalization of all rules in this ORM schema. These verbalizations are generated *automatically*, according to the Lithuanian verbalization template presented in section 3. This approach is fully implemented and supported in the DogmaModeler ontology modeling tool [J05]. It is worth noting that DogmaModeler's automated verbalization has been used by tens of lawyers to build the Customer Complaint Ontology [J05][JVM03].

Remark on Modality: Our verbalization template can be adapted easily according to the application/reasoning scenario, whether it is used as integrity constraints, derivation/inference rules, business rules, etc. For example, the above mandatory constraint can be verbalized in different ways, such as: 1) Each **Book** must **Has** at least one **ISBN**. 2) Each **Book** **Has** some **ISBN** values. 3) If there is a **Book** then it **Has** an **ISBN** value. 4) A **Book** that does not **Has** an **ISBN** is not allowed. 5) If a **Book** does not **Has** an **ISBN** value then....

2 Example of an ORM Schema

We illustrate in one diagram most types of rules supported in ORM. Our article [JKD06] describes technical details on *how* our verbalization approach is implemented, see [H01] to know more about ORM, and [J05] to know about the DogmaModeler tool that we use to build and automatically verbalize ORM models.

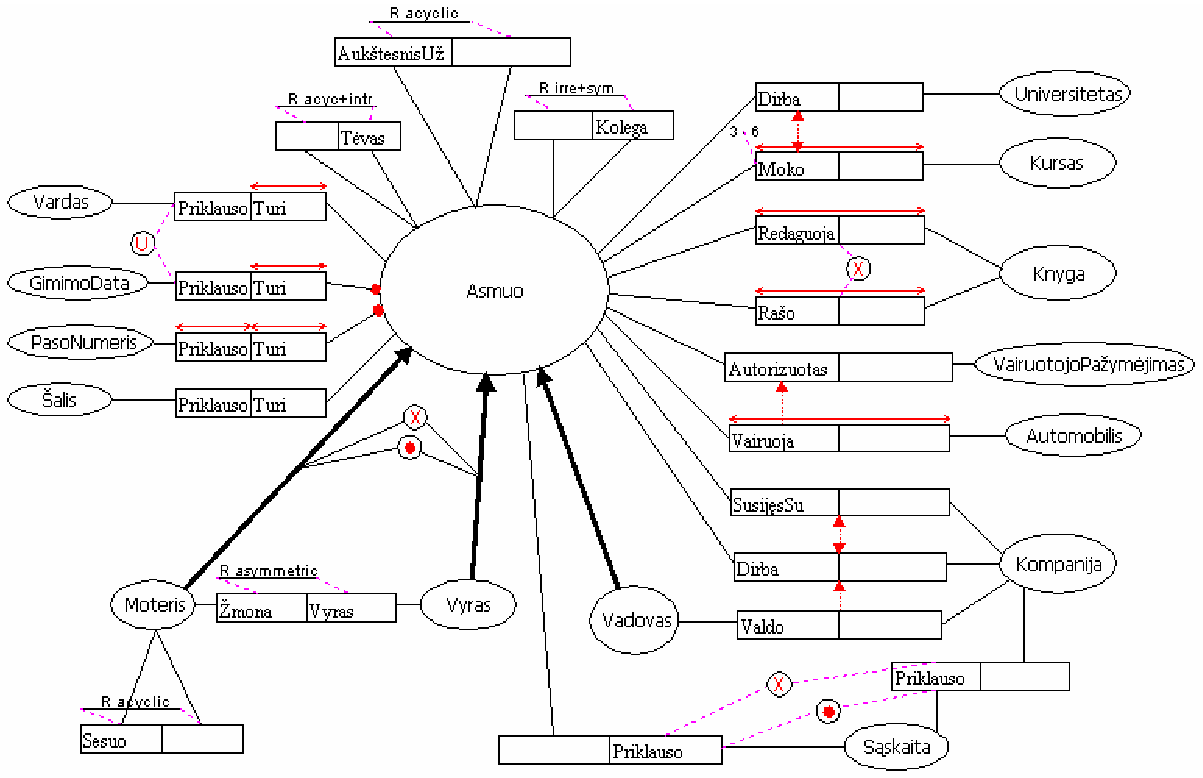

Fig. 1. Example of ORM rules, in Lithuanian.

The constraints/rules in the above ORM example are verbalized automatically (according to the verbalization template presented in section 2):

```
-[Mandatory]  Kiekvienas Asmuo privalo Turi bent vieną PasoNumeris.
-[Mandatory]  Kiekvienas Asmuo privalo Turi bent vieną GimimoData.
-[Mandatory]  Kiekvienas Sąskaita turėtų būti Asmuo arba Priklauso Kompanija  Priklauso.
-[Uniqueness]  Kiekvienas Asmuo privalo Turi daugiausia vieną GimimoData.
-[Uniqueness]  Kiekvienas Asmuo privalo Turi daugiausia vieną Vardas.
-[Uniqueness]  Kiekvienas Asmuo privalo Turi daugiausia vieną PasoNumeris.
-[Uniqueness]  Kiekvienas PasoNumeris privalo Priklauso daugiausia vieną Asmuo.
-[Uniqueness]  įmanoma, kad  Asmuo Moko daugiau nei vieną Kursas  ir atvirkščiai.
-[Uniqueness]  įmanoma, kad  Asmuo Redaguoja daugiau nei vieną Knyga  ir atvirkščiai.
-[Uniqueness]  įmanoma, kad  Asmuo Rašo daugiau nei vieną Knyga  ir atvirkščiai.
-[Uniqueness]  įmanoma, kad  Asmuo Vairuoja daugiau nei vieną Automobilis  ir atvirkščiai.
-[Uniqueness]  { GimimoData ir Vardas } turi atitikti daugiausia vieną Asmuo.
-[Exclusive]  Kiekvienas Asmuo turėtų būti arba Vyras , arba Moteris.
-[Totality]  Kiekvienas Asmuo privalo būti mažiausiai Vyras , arba Moteris.
-[Subset]  Jei Asmuo  Vairuoja Automobilis , tai Asmuo  Autorizuotas VairuotojoPažymėjimas.
-[Subset]  Jei Vadovas  Valdo Kompanija , tai šis Asmuo  Dirba tam/tai Kompanija.
-[Equality]  Asmuo  Dirba  Universitetas tada ir tik tada, kai Asmuo  Moko  Kursas.
-[Equality]   Asmuo  SusijęsSu Kompanija tada ir tik tada, kai šis Asmuo  Dirba tam/tai Kompanija.
-[Exclusion]  Nei vienas Sąskaita ne- Priklauso  Asmuo , kai  Priklauso  Kompanija.
-[Exclusion]  Niekada Asmuo , kuris  Redaguoja Knyga , tos Knyga ne-  Rašo.
-[Value] Galimos Šalis reikšmės yra: { Belgija , Prancūzija, Vokietija }.
-[Irreflexive] Joks Asmuo nėra Kolega pats sau.
-[Symmetric] Jei Asmuo X Kolega Asmuo Y, tai ir atvirkščiai.
-[Acyclic]   Asmuo negali būti tiesiogiai (arba netiesiogiai)  AukštesnisUž pats sau .
-[Acyclic]   Moteris negali būti tiesiogiai (arba netiesiogiai)  Sesuo pats sau .
-[Asymmetric]  Jei Moteris X  Žmona  Moteris Y, tai ne atvirkščiai .
-[Intransitive] Jei Asmuo X Tėvas Asmuo Y, ir Y Tėvas Z, tai negali bti, kad X Tėvas Z.
-[Frequency] Jei Asmuo Moko Kursas, tai šis Asmuo Moko mažiausiai 3 ir daugiausiai 6 Kursas.
```

3 The Lithuanian Verbalization Template

The template is presented in an XML syntax, it is being implemented in the DogmaModeler tool to support the Lithuanian verbalization of ORM models. More details about this approach can be found [JKD06], and refer to [J05] about DogmaModeler.

```
<?xml version='1.0' encoding='UTF-8'?>
<ORMSchema xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
xsi:noNamespaceSchemaLocation='http://www.starlab.vub.ac.be/staff/mustafa/orm/verbaliz
ation/'>

<ORMNLMeta>
 <Meta name="DC.Title" content="Lithuanian verbalization template (Ver0.1)"/>
 <Meta name="DC.Version" content="0.1"/>
 <Meta name="DC.Creator" content="Mustafa Jarrar"/>
 <Meta name="DC.Contributor" content="Juozas Gordevicius"/>
 <Meta name="DC.Language" content="Lithuanian"/>
 </ORMNLMeta>

<ORMNLBody>

<FactType xsi:type="FactType" >
<Text></Text>
```

```
<Object index="0" />
<Role index="0" />
<Text>/</Text>
<Role index="1" />
<Text></Text>
<Object index="1" />
</FactType>

<Constraint xsi:type="Mandatory">
 <Text> -[Mandatory]  Kiekvienas</Text>
 <Object index="0"/>
 <Text>privalo</Text>
 <Role index="0"/>
 <Text>bent vieną</Text>
 <Object index="1"/>
</Constraint>

<Constraint xsi:type="Backward Mandatory">
 <Text> -[Mandatory]  Kiekvienam</Text>
 <Object index="0"/>
 <Text>yra bent vienas</Text>
 <Object index="1"/>
 <Text>, kuris</Text>
 <Role index="1"/>
 <Text>tam</Text>
 <Object index="0"/>
</Constraint>

<Constraint xsi:type="Disjunctive Mandatory">
 <Text> -[Mandatory]  Kiekvienas</Text>
 <Object index="0"/>
 <Text>turėtų būti</Text>
 <Object index="1"/>
 <Loop index="1" >
  <Text>arba</Text>
  <Role index="n"/>
  <Object index="n"/>
 </Loop>
 <Role index="0"/>
</Constraint>

<Constraint xsi:type="Uniqueness">
 <Text> -[Uniqueness]  Kiekvienas</Text>
 <Object index="0"/>
 <Text>privalo</Text>
 <Role index="0"/>
 <Text>daugiausia vieną</Text>
 <Object index="1"/>
</Constraint>

<Constraint xsi:type="Backward Uniqueness">
 <Text> -[Uniqueness]  Kiekvienas</Text>
 <Object index="0"/>
 <Text>turi turėti daugiausia vieną</Text>
 <Object index="1"/>
 <Text>kuris</Text>
 <Role index="1"/>
 <Text>tam</Text>
 <Object index="0"/>
</Constraint>

<Constraint xsi:type="Many Uniqueness">
 <Text> -[Uniqueness]  įmanoma, kad </Text>
 <Object index="0"/>
<Role index="0"></Role>
 <Text>daugiau nei vieną</Text>
 <Object index="1"/>
 <Text> ir atvirkščiai</Text>
</Constraint>

<Constraint xsi:type="External Uniqueness">
 <Text> -[Uniqueness]  {</Text>
 <Object index="1"/>
 <Loop index="1">
  <Text>ir</Text>
  <Object index="n"/>
 </Loop>
 <Text>} turi atitikti daugiausia vieną</Text>
 <Object index="0"/>
</Constraint>

<Constraint xsi:type="Subtype">
 <Text> -[Subtype]  Kiekvienas</Text>
 <Object index="child"/>
 <Text>taip pat yra</Text>
 <Object index="parent"/>
</Constraint>

<Constraint xsi:type="Value">
 <Text> -[Value]  Galimos </Text>
 <Object index="0"/>
 <Text> reikšmės yra: {</Text>
 <Value index="0"/>
 <Loop index="1">
  <Text>,</Text>
  <Value index="n"/>
 </Loop>
 <Text> }</Text>
 </Constraint>

<Constraint xsi:type="Exclusive">
<Text> -[Exclusive]  Kiekvienas</Text>
<Object index="0"/>
<Text>turėtų būti arba</Text>
<Object index="1"/>
<Loop index="1">
  <Text>, arba</Text>
  <Object index="n"/>
</Loop>
</Constraint>

<Constraint xsi:type="Total">
 <Text> -[Total]  Kiekvienas</Text>
 <Object index="0"/>
 <Text>privalo būti mažiausiai</Text>
 <Object index="1"/>
 <Loop index="1">
  <Text>, arba</Text>
  <Object index="n"/>
 </Loop>
</Constraint>

<Constraint xsi:type="Partition">
 <Text> -[Partition]  Kiekvienas</Text>
 <Object index="0"/>
 <Text>yra bent vienas iš</Text>
 <Object index="1"/>
 <Loop index="1">
  <Text>, arba</Text>
  <Object index="n"/>
 </Loop>
 <Text>, bet ne visi kartu</Text>
</Constraint>

<Constraint xsi:type="Subset">
 <Text> -[Subset]  Jei</Text>
 <Object index="0"/>
 <Role index="child"/>
 <Object index="child"/>
 <Text>, tai</Text>
 <Object index="0"/>
 <Role index="parent"/>
 <Object index="parent"/>
</Constraint>

<Constraint xsi:type="Subset FactType">
 <Text> -[Subset]  Jei</Text>
 <Object index="0"/>
 <Role index="child"/>
 <Object index="child"/>
 <Text>, tai šis</Text>
 <Object index="1"  />
 <Role index="parent"/>
 <Text>tam/tai</Text>
 <Object index="parent"/>
</Constraint>

<Constraint xsi:type="Equality">
 <Text> -[Equality]  </Text>
 <Object index="0"/>
 <Role index="first"/>
 <Text> </Text>
 <Object index="first"/>
 <Text>tada ir tik tada, kai </Text>
 <Object index="0"/>
 <Role index="second"/>
<Text> </Text>
 <Object index="second"/>
</Constraint>

<Constraint xsi:type="Equality FactType">
 <Text> -[Equality]  </Text>
 <Object index="0"/>
 <Role index="First"/>
 <Object index="First"/>
 <Text>tada ir tik tada, kai</Text>
 <Text>šis</Text>
 <Object index="1"/>
 <Role index="Second"/>
 <Text>tam/tai</Text>
 <Object index="Second"/>
</Constraint>


<Constraint xsi:type="Exclusion">
 <Text> -[Exclusion]  Nei vienas </Text>
 <Object index="0"/>
 <Text>ne-</Text>
 <Role index="first"/>
 <Text> </Text>
 <Object index="first"/>
 <Text>, kai</Text>
 <Role index="second"/>
 <Text> </Text>
 <Object index="second"/>
</Constraint>

<Constraint xsi:type="Exclusion FactType">
 <Text> -[Exclusion]  Niekada</Text>
 <Object index="0"/>
 <Text>, kuris</text>
 <Role index="first"/>
 <Object index="first"/>
 <Text>, tos</Text>
 <Object index="second"/>
 <Text>ne-</Text>
 <Role index="second"/>
</Constraint>

<Constraint xsi:type="Frequency">
 <Text> -[Frequency] Jei </Text>
 <Object index="0"/>
 <Role index="0"/>
 <Object index="1"/>
 <Text>, tai šis </Text>
 <Object index="0"/>
 <Role index="0"/>
 <Text>mažiausiai </Text>
 <Minimum/>
 <Object index="1"/>
 <Text> ir daugiausiai </Text>
 <Maximum/>
 <Object index="1"/>
</Constraint>

<Constraint xsi:type="Irreflexive">
 <Text> -[Irreflexive]  Joks</Text>
 <Object index="0"/>
 <Text>nėra</Text>
 <Role index="0"/>
 <Text> pats sau</Text>
</Constraint>

<Constraint xsi:type="Symmetric">
 <Text> -[Symmetric]  Jei</Text>
 <Object index="0"/>
 <Text>X</Text>
 <Role index="0"/>
 <Object index="0"/>
 <Text>Y</Text>
 <Text>, tai ir atvirkščiai</Text>
</Constraint>


<Constraint xsi:type="Asymmetric">
 <Text> -[Asymmetric]  Jei</Text>
 <Object index="0"/>
 <Text> X</Text>
 <Role index="0"/>
 <Text> </Text>
 <Object index="0"/>
<Text> Y, tai ne atvirkščiai</Text>
```

```
</Constraint>

<Constraint xsi:type="Acyclic">
 <Text> -[Acyclic]  </Text>
 <Object index="0"/>
 <Text> negali būti tiesiogiai (arba netiesiogiai)</Text>
 <Role index="0"/>
 <Text> pats sau</Text>
</Constraint>

<Constraint xsi:type="Transitve">
 <Text> -[Intransitve]  Jei</Text>
 <Object index="0"/>
 <Text>X</Text>
 <Role index="0"/>
 <Object index="0"/>
 <Text>Y, ir Y</Text>
 <Role index="0"/>
 <Text> Z, tai negali bti, kad X</Text>
 <Role index="0"/>
 <Text>Z</Text>
</Constraint>

</ORMNLBody>
</ORMSchema>
```

Acknowledgments

We are in debt to Andriy Lisovoy who helped in the implementation of DogmaModeler, and to Hai Nguyen Hoang who helped in the first implementation of the verbalization component during his Master thesis. This work is partially supported by the EU Knowledge Web NoE project (IST-2004-507482).

References

[JKD06] Jarrar, M., Keet, C.M., Dongilli, P. Multilingual verbalization of ORM conceptual models and axiomatized ontologies. [Submitted].

[J05] Jarrar, M.: Towards Methodological Principles for Ontology Engineering. PhD thesis, Vrije Universiteit Brussel, 2005.

[JVM03] Jarrar, M., Verlinden, R., Meersman, R.: Ontology-based Customer Complaint Management. In: Jarrar M., Salaun A., (eds.): Proceedings of the workshop on regulatory ontologies and the modeling of complaint regulations, Catania, Sicily, Italy. Springer Verlag LNCS. Vol. 2889. November (2003) pp. 594–606

[H01] Halpin, T.: Information Modeling and Relational Databases. 3rd ed. Morgan-Kaufmann. (2001)

[H04] Halpin, T.: Business Rule Verbalization. In Doroshenko, A., Halpin, T., Liddle, S., Mayr H. (eds): Information Systems Technology and its Applications, 3rd International Conference (ISTA'2004), LNI 48 GI ISBN 3-88579-377-6, (2004) pp:39-52.