A Spanish Verbalization Template
for ORM conceptual models and rules

Mustafa Jarrar, Vrije Universiteit Brussel, Belgium. (Contact Author)
Maria Keet, Free University of Bozen-Bolzano, Italy.
Núria Casellas, Universidad Autónoma de Barcelona, Spain.

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 Spanish verbalization template file. Given an ORM schema (or an ORM-ML file), and given the verbalization template, a Spanish 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 Spanish verbalization of the ORM models and rules.* The verbalization of several other languages (including, but not limited to: German, Italian, Arabic, Russian, Spanish, French, and Lithuanian) 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

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

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., Casellas, N.: A Spanish 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 Spanish 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.

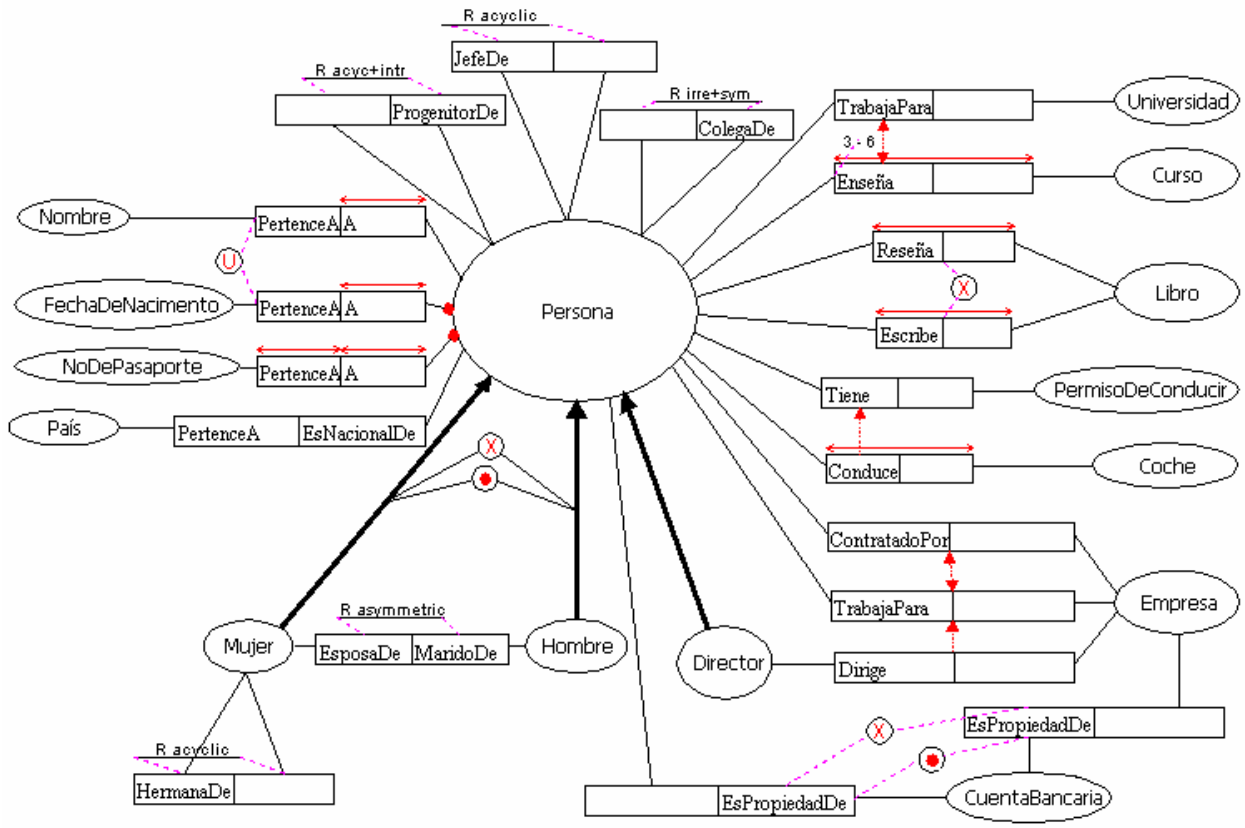

Fig. 1. Example of ORM rules, in Spanish.

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

-[Mandatory] Cada Persona debe A al menos un(a) NoDePasaporte.
-[Mandatory] Cada Persona debe A al menos un(a) FechaDeNacimento.
-[Mandatory] Cada CuentaBancaria debería ser EsPropiedadDe Empresa o EsPropiedadDe Persona.
-[Uniqueness] Cada Persona debe A como máximo un(a) FechaDeNacimento.
-[Uniqueness] Cada Persona debe A como máximo un(a) Nombre.
-[Uniqueness] Cada Persona debe A como máximo un(a) NoDePasaporte.
-[Uniqueness] Cada NoDePasaporte debe PertenceA como máximo un(a) Persona.
-[Uniqueness] Es posible que Persona Enseña más de un(a) Curso , y viceversa.
-[Uniqueness] Es posible que Persona Reseña más de un(a) Libro , y viceversa.
-[Uniqueness] Es posible que Persona Escribe más de un(a) Libro , y viceversa.
-[Uniqueness] Es posible que Persona Conduce más de un(a) Coche , y viceversa.
-[Uniqueness] la combinación de { FechaDeNacimento y Nombre } debe referirse sólo a un(a) Persona.
-[Exclusive] Cada Persona debería ser o bien Mujer o Hombre.
-[Totality] Cada Persona debe ser, al menos, Mujer o Hombre.
-[Subset] Si un(a) Persona Conduce un(a) Coche , entonces eso(a) Persona Tiene un(a) PermisoDeConducir.
-[Subset] Si un(a) Director Dirige un(a) Empresa , entonces este(a) Persona TrabajaPara eso/a Empresa.
-[Equality] Persona TrabajaPara un(a) Universidad si y sólo si este(a) Persona Enseña Curso.
-[Equality] Persona ContratadoPor Empresa si y sólo si este(a) Persona TrabajaPara ese Empresa.
-[Exclusion] Ningún(a) CuentaBancaria EsPropiedadDeEmpresa y también EsPropiedadDe Persona.
-[Exclusion] Ningún(a) Persona Escribe Libro y también Reseña este(a) Libro.
- [Value] Las posibles instancias de País son:{ Bélgica, Francia, Alemania}
-[Irreflexive] Ningún(a) Persona ColegaDe él(ella) mismo(a).
- [Symmetric] Si Persona X ColegaDe Persona Y , debe suceder también a la inversa.
-[Acyclic] Persona no puede ser directamente (o indirectamente a través de una cadena) JefeDe él(ella) mismo(a) .
-[Acyclic] Mujer no puede ser directamente (o indirectamente a través de una cadena) HermanaDe él(ella) mismo(a) .
-[Asymmetric] Si Persona X EsposaDe Persona Y, no puede suceder a la inversa .
-[Intransitive] Si Persona X ProgenitorDe Persona Y, y Y ProgenitorDe Z, entonces no puede suceder que X ProgenitorDe Z.
-[Frequency] Si Persona Enseña Curso, entonces este/a Persona Enseña al menos 3 y como máximo 6 Curso(s).

3 The Spanish Verbalization Template

The template is presented in an XML syntax, it is being implemented in the DogmaModeler tool to support the Spanish 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
/verbalization/'>

<ORMNLMeta>
 <Meta name="DC.Title" content="Spanish verbalization template (Ver0.1)"/>
 <Meta name="DC.Version" content="0.1"/>
 <Meta name="DC.Creator" content="Mustafa Jarrar"/>
```

```
 <Meta name="DC.Contributor" content="Maria Keet"/>
 <Meta name="DC.Language" content="Spanish"/>
 </ORMNLMeta>

<ORMNLBody>

<FactType xsi:type="FactType" >
<Text></Text>
<Object index="0" />
<Role index="0" />
<Object index="1" />
<Text> / </Text>
<Object index="1" />
<Role index="1" />
<Object index="0" />
</FactType>

<Constraint xsi:type="Mandatory">
 <Text> -[Mandatory] Cada</Text>
 <Object index="0"/>
 <Text>debe</Text>
 <Role index="0"/>
 <Text>al menos un(a)</Text>
 <Object index="1"/>
</Constraint>

<Constraint xsi:type="Backward Mandatory">
 <Text> -[M]  Por cada</Text>
 <Object index="0"/>
 <Text>hay al menos un(a)</Text>
 <Object index="1"/>
 <Text>que</Text>
 <Role index="1"/>
 <Text>este/a</Text>
 <Object index="0"/>
</Constraint>

<Constraint xsi:type="Disjunctive Mandatory">
 <Text> -[Mandatory]  Cada</Text>
 <Object index="0"/>
 <Text>debería ser</Text>
 <Role index="0"/>
 <Object index="1"/>
 <Loop index="1" >
  <Text>o</Text>
  <Role index="n"/>
  <Object index="n"/>
 </Loop>
</Constraint>

<Constraint xsi:type="Uniqueness">
 <Text> -[Uniqueness] Cada</Text>
 <Object index="0"/>
 <Text>debe</Text>
 <Role index="0"/>
 <Text>como máximo un(a)</Text>
 <Object index="1"/>
```

```
</Constraint>

<Constraint xsi:type="Backward Uniqueness">
 <Text> -[Uniqueness]  Por cada</Text>
 <Object index="0"/>
 <Text>debe haber sólo un</Text>
 <Object index="1"/>
 <Text>eso/a</Text>
 <Role index="1"/>
 <Text>esto</Text>
 <Object index="0"/>
</Constraint>

<Constraint xsi:type="Many Uniqueness">
 <Text> -[Uniqueness]  Es posible que </Text>
 <Object index="0"/>
<Role index="0"></Role>
 <Text>más de un(a)</Text>
 <Object index="1"/>
 <Text>, y viceversa</Text>
</Constraint>

<Constraint xsi:type="External Uniqueness">
 <Text> -[Uniqueness]  la combinación de {</Text>
 <Object index="1"/>
 <Loop index="1">
  <Text>y</Text>
  <Object index="n"/>
 </Loop>
 <Text>} debe referirse sólo a un(a)</Text>
 <Object index="0"/>
</Constraint>

<Constraint xsi:type="Subtype">
 <Text> -[Subtype]  Cada instancia</Text>
 <Object index="child"/>
 <Text> es también una instancia de </Text>
 <Object index="parent"/>
</Constraint>

<Constraint xsi:type="Value">
 <Text> -[Value]  Las posibles instancias de </Text>
 <Object index="0"/>
 <Text> son :{</Text>
 <Value index="0"/>
 <Loop index="1">
  <Text>,</Text>
  <Value index="n"/>
 </Loop>
 <Text> }</Text>
 </Constraint>

<Constraint xsi:type="Exclusive">
<Text> -[Exclusive]  Cada</Text>
<Object index="0"/>
<Text>debería ser o bien</Text>
<Object index="1"/>
```

```
<Loop index="1">
  <Text>o</Text>
  <Object index="n"/>
</Loop>
</Constraint>

<Constraint xsi:type="Total">
 <Text> -[Totality]  Cada</Text>
 <Object index="0"/>
 <Text>debe ser, al menos, </Text>
 <Object index="1"/>
 <Loop index="1">
  <Text>o</Text>
  <Object index="n"/>
 </Loop>
</Constraint>

<Constraint xsi:type="Partition">
 <Text> -[Partition]  Cada</Text>
 <Object index="0"/>
 <Text>es como mínimo uno de</Text>
 <Object index="1"/>
 <Loop index="1">
  <Text>o</Text>
  <Object index="n"/>
 </Loop>
 <Text>pero no todos</Text>
</Constraint>

<Constraint xsi:type="Subset">
 <Text> -[Subset]  Si un(a)</Text>
 <Object index="0"/>
 <Role index="child"/>
 <Text>un(a)</Text>
 <Object index="child"/>
 <Text>, entonces eso(a) </Text>
 <Object index="0"/>
 <Role index="parent"/>
 <Text>un(a)</Text>
 <Object index="parent"/>
</Constraint>

<Constraint xsi:type="Subset FactType">
 <Text> -[Subset]  Si un(a)</Text>
 <Object index="0"/>
 <Role index="child"/>
 <Text>un(a)</Text>
 <Object index="child"/>
 <Text>, entonces este(a) </Text>
 <Object index="1"  />
 <Role index="parent"/>
 <Text>eso/a</Text>
 <Object index="parent"/>
</Constraint>

<Constraint xsi:type="Equality">
 <Text> -[Equality] </Text>
```

```
 <Object index="0"/>
 <Role index="first"/>
 <Text> un(a) </Text>
 <Object index="first"/>
 <Text>si y sólo si</Text>
 <Text>este(a) </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>si y sólo si</Text>
 <Text>este(a)</Text>
 <Object index="1"/>
 <Role index="Second"/>
 <Text>ese</Text>
 <Object index="Second"/>
</Constraint>

<Constraint xsi:type="Exclusion">
 <Text> -[Exclusion]  Ningún(a)</Text>
 <Text> </Text>
 <Object index="0"/>
 <Text> </Text>
 <Role index="first"/>
 <Object index="first"/>
 <Text>y también</Text>
 <Role index="second"/>
 <Text> </Text>
 <Object index="second"/>
</Constraint>

<Constraint xsi:type="Exclusion FactType">
 <Text> -[Exclusion]  Ningún(a)</Text>
 <Object index="0"/>
 <Role index="first"/>
 <Text> </Text>
 <Object index="first"/>
 <Text>y también</Text>
 <Role index="second"/>
 <Text>este(a)</Text>
 <Object index="second"/>
</Constraint>

<Constraint xsi:type="Frequency">
 <Text> -[Frequency] Si </Text>
 <Object index="0"/>
 <Role index="0"/>
 <Object index="1"/>
 <Text>, entonces este(a) </Text>
 <Object index="0"/>
```

```
 <Role index="0"/>
 <Text>al menos </Text>
 <Minimum/>
 <Text> y como máximo </Text>
 <Maximum/>
 <Role index="0"/>
 <Text>(os)</Text>
</Constraint>

<Constraint xsi:type="Irreflexive">
 <Text> -[Irreflexive]  Ningún(a)</Text>
 <Object index="0"/>
 <Role index="0"/>
 <Text> él(ella) mismo(a)</Text>
</Constraint>

<Constraint xsi:type="Symmetric">
 <Text> -[Symmetric]  Si</Text>
 <Object index="0"/>
 <Text>X</Text>
 <Role index="0"/>
 <Object index="0"/>
 <Text>Y</Text>
 <Text>, debe suceder también a la inversa</Text>
</Constraint>

<Constraint xsi:type="Asymmetric">
 <Text> -[Asymmetric]  Si</Text>
 <Object index="0"/>
 <Text>X</Text>
 <Role index="0"/>
 <Text> </Text>
 <Object index="0"/>
<Text> Y, no puede suceder a la inversa</Text>
</Constraint>

<Constraint xsi:type="Acyclic">
 <Text> -[Acyclic] </Text>
 <Object index="0"/>
 <Text> no puede ser directamente (o indirectamente a través de una
cadena)</Text>
 <Role index="0"/>
 <Text> él(ella) mismo(a)</Text>
</Constraint>

<Constraint xsi:type="Transitve">
 <Text> -[Intransitve]  Si</Text>
 <Object index="0"/>
 <Text>X</Text>
 <Role index="0"/>
 <Object index="0"/>
 <Text>Y, y Y</Text>
 <Role index="0"/>
 <Text> Z, entonces no puede suceder que 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.