Deutsche Verbalisierungsvorlage
für Konzept- und Regelmodelle in ORM

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

Technischer Bericht des Artikels[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/stuff/mustafa/orm/verbalization/

Abstract. Im oben genannten Artikel beschreiben wir einen neuen Lösungsvorschlag zur Unterstützung der mehrsprachigen Verbalisierung von logischen Theorien, Axiomatisierungen und anderen Spezifikationen wie z.B. Geschäftsregeln (Business Rules). Diese Lösung wird anhand der Sprache Object Role Modeling (ORM) illustriert. Die zugrundeliegenden Techniken lassen sich aber auch mit anderen Konzeptmodellen und formalen Sprachen, wie z.B. Description Logics, wiederverwenden, um eine bessere Verständlichkeit und Verwendbarkeit für Fachexperten (Domain Experts) zu erreichen. Die technische Lösung für die mehrsprachige Verbalisierung ist durch ihre Flexibilität, Erweiterbarkeit und Wartbarkeit der Verbalisierungsvorlagen gekennzeichnet, die eine einfache Erweiterung auf andere Sprachen zu den 11 bereits verfügbaren Sprachen ermöglichen.
Dieser Bericht stellt die deutsche Verbalisierungsvorlage vor. Für ein gegebenes ORM-Schema und eine entsprechende Verbalisierungsvorlage wird automatisch eine deutsche Verbalisierung der Regeln und Fakttypen im Schema generiert. Ein umfassendes Beispiel eines ORM-Schemas mit der entsprechenden Verbalisierung wird im Bericht diskutiert.

1 Einführung

Im oben genannten Artikel beschreiben wir einen neuen Lösungsvorschlag, der die mehrsprachige Verbalisierung von logischen Theorien, Axiomatisierungen und anderen Spezifikationen, wie z.B. Geschäftsregeln (Business Rules), Ontologien, usw. unterstützt. Wir zeigen unsere Lösung anhand einer Verbalisierungsvorlage für die Sprache ORM. Diese Vorlage kann auf einfache Art und Weise an verschiedene natürliche Sprachen angepasst und übersetzt werden. Dieser technische Bericht liefert die deutsche Verbalisierung der ORM-Modelle und -Regeln. Die Verbalisierung anderer Sprachen (einschliesslich von Italienisch, Arabisch, Russich, Spanisch, Holländisch, Französisch und Litauisch) findet man unter der oben gezeigten URL.
Die zugrundeliegenden Techniken unserer Lösung können mit anderen Konzeptmodellen und formalen Sprachen, wie z.B. Description Logics, wiederverwendet werden. Die Zielsetzung war also die Definition einer parametrisierten Vorlage auf einer gegebenen Menge von Regeln, Modellen oder Axiomen, wobei die Ausgabe in Form von Sätzen in pseudo-natürlicher Sprache mit fester Syntax dargestellt ist.
Betrachten wir als einfaches Besipiel die folgende formale Regel:

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

Sie könnte folgendermassen in natürliche Sprache übersetzt werden:

Jedes **Buch hat** eine **ISBN**.

[1] Als Literaturstelle: *Jarrar, M., Keet, C.M.: A German 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]*.

Auf diese Weise wird den Fachexperten ermöglicht, den Aufbau und/oder die Gültigkeit einer formalen Spezifikation ihres Fachgebietes zu überprüfen, ohne dass sie wissen müssen, dass diese Sätze formale Axiome darstellen. Das bedeutet, dass die zugrundeliegenden logischen und beweisführenden Dienste dem Benutzer verdeckt bleiben. Unsere Lösung mit den gegebenen Vorlagen kann für die Modellierung von Geschäftsregeln, Ontologien, Wissensbasen, usw. wiederverwendet werden.

Im folgenden Kapitel stellen wir ein ORM-Beispiel mit der Verbalisierung aller Regeln aus dem ORM-Schema vor. Diese Verbalisierungen werden automatisch entsprechend der deutschen Vorlage generiert, die im dritten Kapitel vorgestellt wird. Diese Lösung ist vollständig implementiert und wird in DogmaModeler, einem Werkzeug für ontologische Modellierung [J05], integriert. Es sei bemerkt, dass die automatische Verbalisierung in DogmaModeler von mehreren Dutzenden Juristen beim Aufbau der Customer Compliant Ontology [J05][JVM03] benutzt wurde.

Anmerkung zur Vorgehensweise: Unsere Verbalisierungsvorlage kann an das Anwendungs-/Beweisführungsszenario auf einfache Art und Weise angepasst werden unabhängig davon, ob es sich um eine Integritätsbedingung, Ableitungs-/Inferenzregel, Geschäftsregel, usw. handelt. Zum Beispiel, die oben gezeigte **Zwangsbedingung** kann verschiedenartig verbalisiert werden, wie: 1) Jedes **Buch** muss mindestens eine **ISBN** haben. 2) Jedes **Buch hat** einige **ISBN** Nummern. 3) Falls es ein **Buch** gibt, dann **hat** es eine **ISBN** Nummer. 4) Ein **Buch,** das keine **ISBN** hat, ist nicht erlaubt. 5) Falls ein **Buch** keine **ISBN** hat, dann ...

2 Beispiel von ORM-Schema

In folgenden Diagramm zeigen wir die meisten Typen von Regeln, die von ORM vorgesehen sind. Unser Artikel [JKD06] beschreibt die technischen Details darüber, wie unsere Lösung für die Verbalisierung implementiert ist. Weitere Informationen über ORM findet man in [H01]. Das Werkzeug DogmaModeler, mit dem wir ORM-Modelle aufbauen und automatisch verbalisieren können, ist in [J05] beschrieben.

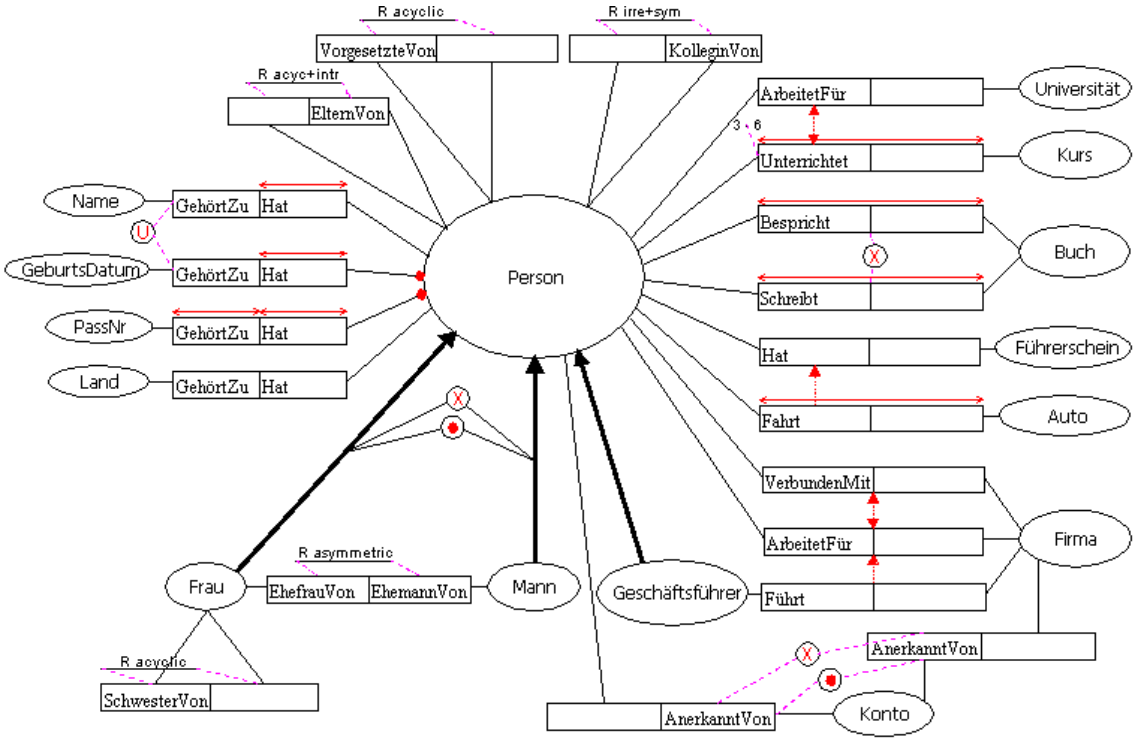

Fig. 1. Beispiel von ORM Regeln auf Deutsch.

Die Zwangsbedingungen/Regeln im obigen ORM-Beispiel werden automatisch wie folgt verbalisiert (unter Verwendung der Verbalisierungsvorlage in Kapitel 3):

-[Mandatory] Jeder/s Person Hat mindestens 1 PassNr.
-[Mandatory] Jeder/s Person Hat mindestens 1 GeburtsDatum.
-[Mandatory] Jeder/s Konto entweder AnerkanntVon ein Person oder AnerkanntVon ein Firma.
-[Uniqueness] Jeder/s Person Hat höchstens 1 GeburtsDatum.
-[Uniqueness] Jeder/s Person Hat höchstens 1 Name.
-[Uniqueness] Jeder/s Person Hat höchstens 1 PassNr.
-[Uniqueness] Jeder/s PassNr GehörtZu höchstens 1 Person.
-[Uniqueness] Es ist möglich das ein Person mehr als 1 Kurs Unterrichtet , und umgekehrt .
-[Uniqueness] Es ist möglich das ein Person mehr als 1 Buch Bespricht , und umgekehrt .
-[Uniqueness] Es ist möglich das ein Person mehr als 1 Buch Schreibt , und umgekehrt .
-[Uniqueness] Es ist möglich das ein Person mehr als 1 Auto Fahrt , und umgekehrt .
-[Uniqueness] Jeder Kombination von { GeburtsDatum und Name } ist verbunden mit nur 1 Person.
-[Exclusive] Jeder/s Person kann entweder ein Mann oder ein Frau sein.
-[Totality] Jeder/s Person ist mindestens ein Frau oder ein Mann.
-[Subset] Wenn ein Person Fahrt ein Auto , dann soll diese Person auch Hat ein Führerschein.
-[Subset] Wenn ein Geschäftsführer ein Firma Führt , dann soll diese Person auch ArbeitetFür der/den/dem Firma.
-[Equality] Ein Person ArbeitetFür ein Universität nur wenn diese Person ein Kurs Unterrichtet.
-[Equality] Ein Person VerbundenMit Firma nur wenn diese Person ArbeitetFür der/dem/den Firma.
-[Exclusion] Kein Konto ist AnerkanntVon ein Person und auch AnerkanntVon ein Firma.
-[Exclusion] Kein Person Bespricht ein Buch und auch Schreibt das/der/dieselbe Buch.
-[Value] Die mögliche Instanzen von Land sind : {Belgien, Frankreich, Deutschland}.
-[Irreflexive] Kein einzige Person KolleginVon sich selbst.
-[Symmetric] Als Person X KolleginVon Person Y, dann auch umgekehrt.
-[Acyclic] Person kann nicht direkt (oder indirekt via eine Verkettung) VorgesetzteVon sich selbst sein
-[Acyclic] Frau kann nicht direkt (oder indirekt via eine Verkettung) SchwesterVon sich selbst sein .
-[Asymmetric] Als Frau X EhefrauVon Frau Y, dann ist umgekehrt unmöglich .
-[Intransitive] Als Person X ElternVon Person Y, und Y ElternVon Z, dann ist es nicht möglich das X ElternVon Z.
-[Frequency] Wenn ein Person Unterrichtet ein Kurs, dann diese Person Unterrichtet mindestens 3 und höchstens 6 Kurs.

3 Die deutsche Verbalisierungsvorlage

Die Vorlage ist in XML-Syntax dargestellt und in DogmaModeler integriert, um die deutsche Verbalisierung von ORM-Modellen zu unterstützen. Weitere Einzelheiten dieser Lösung und über DogmaModeler finden sich in [JKD06] und [J05].

```
<?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="German verbalization template (Ver0.2)"/>
<Meta name="DC.Version" content="0.3"/>
<Meta name="DC.Creator" content="Mustafa Jarrar"/>
<Meta name="DC.Contributor" content="C. Maria Keet"/>
<Meta name="DC.Language" content="German"/>
</ORMNLMeta>
```

```
<ORMNLBody>

<FactType xsi:type="FactType" >
<Text>Ein(e)</Text>
<Object index="0" />
<Role index="0" />
<Text>/</Text>
<Role index="1" />
<Text> ein(e)</Text>
<Object index="1" />
</FactType>

<Constraint xsi:type="Mandatory" >
<Text> -[Mandatory] Jeder/s</Text>
<Object index="0" />
<Role index="0" />
<Text> mindestens 1</Text>
<Object index="1" />
</Constraint>

<Constraint xsi:type="Backward Mandatory" >
<Text> -[Mandatory] Für jeder/s</Text>
<Object index="0" />
<Text>gibt es mindestens 1</Text>
<Object index="1" />
<Text> das</Text>
<Role index="1" />
<Text> diesen/m</Text>
<Object index="0" />
</Constraint>

<Constraint xsi:type="Disjunctive Mandatory" >
<Text> -[Mandatory] Jeder/s</Text>
<Object index="0" />
<Text>entweder</Text>
<Role index="0" />
<Text>ein</Text>
<Object index="1" />
<Loop index="1" >
<Text>oder </Text>
<Role index="n" />
<Text>ein</Text>
<Object index="1" />
</Loop>
</Constraint>

<Constraint xsi:type="Uniqueness" >
<Text> -[Uniqueness] Jeder/s </Text>
<Object index="0" />
<Role index="0" />
<Text> höchstens 1 </Text>
<Object index="1" />
</Constraint>

<Constraint xsi:type="Backward Uniqueness" >
<Text> -[Uniqueness] Für jeder/s </Text>
<Object index="0" />
<Text> gibt es höchstens ein </Text>
<Object index="1" />
<Text> das/wer </Text>
<Role index="1" />
<Text> diese/r/m </Text>
```

```
<Object index="0" />
</Constraint>

<Constraint xsi:type="External Uniqueness" >
<Text> -[Uniqueness] Jeder Kombination von { </Text>
<Object index="1" />
<Loop index="1">
<Text>und</Text>
<Object index="n" />
</Loop>
<Text>} ist verbunden mit nur 1</Text>
<Object index="0" />
</Constraint>

<Constraint xsi:type="Many Uniqueness" >
<Text> -[Uniqueness] Es ist möglich das ein </Text>
<Object index="0" />
<Text>mehr als 1</Text>
<Object index="1" />
<Role index="0"></Role>
<Text>, und umgekehrt </Text>
</Constraint>
<Constraint xsi:type="Subtype" >
<Text> -[Subtype] Jeder/s</Text>
<Object index="child" />
<Text>ist auch ein </Text>
<Object index="parent" />
</Constraint>

<Constraint xsi:type="Value">
 <Text> -[Value]  Die mögliche Instanzen von </Text>
 <Object index="0"/>
 <Text> sind :{</Text>
 <Value index="0"/>
 <Loop index="1">
  <Text>,</Text>
  <Value index="n"/>
 </Loop>
 <Text> }</Text>
 </Constraint>

<Constraint xsi:type="Exclusive" >
<Text> -[Exclusive] Jeder/s</Text>
<Object index="0"/>
<Text> kann entweder ein </Text>
<Object index="1"/>
<Loop index="1">
<Text>oder ein</Text>
<Object index="n" />
</Loop>
<Text>sein</Text>
</Constraint>

<Constraint xsi:type="Total" >
<Text> -[Total] Jeder/s</Text>
<Object index="0" />
<Text>ist mindestens ein</Text>
<Object index="1" />
<Loop index="1" >
<Text>oder ein</Text>
<Object index="n" />
</Loop>
</Constraint>
```

```
<Constraint xsi:type="Subset" >
<Text> -[Subset] Wenn ein</Text>
<Object index="0" />
<Role index="child" />
<Text>ein</Text>
<Object index="child" />
<Text>, dann soll diese </Text>
<Object index="1" />
<Text>auch</Text>
<Role index="parent" />
<Text>ein</Text>
<Object index="parent" />
</Constraint>

<Constraint xsi:type="Subset FactType" >
<Text> -[Subset] Wenn ein </Text>
<Object index="0" />
<Text>ein</Text>
<Object index="child" />
<Role index="child" />
<Text>, dann soll diese </Text>
<Object index="1" />
<Text>auch</Text>
<Role index="parent" />
<Text> der/den/dem </Text>
<Object index="parent" />
</Constraint>

<Constraint xsi:type="Equality" >
<Text> -[Equality] Ein </Text>
<Object index="0" />
<Role index="first" />
<Text>ein </Text>
<Object index="first" />
<Text>nur wenn </Text>
<Text>diese </Text>
<Object index="0" />
<Text>ein </Text>
<Object index="second" />
<Role index="second" />
</Constraint>

<Constraint xsi:type="Equality FactType" >
<Text> -[Equality] Ein</Text>
<Object index="0" />
<Role index="First" />
<Object index="First" />
<Text>nur wenn</Text>
<Text>diese </Text>
<Object index="1" />
<Role index="Second" />
<Text>der/dem/den</Text>
<Object index="Second" />
</Constraint>

<Constraint xsi:type="Exclusion" >
<Text> -[Exclusion] Kein </Text>
<Object index="0" />
<Text> ist </Text>
<Role index="first" />
<Text> ein </Text>
<Object index="first" />
```

```
<Text>und auch</Text>
<Role index="second" />
<Text>ein </Text>
<Object index="second" />
</Constraint>

<Constraint xsi:type="Exclusion FactType" >
<Text> -[Exclusion] Kein </Text>
<Object index="0" />
<Role index="first" />
<Text> ein </Text>
<Object index="first" />
<Text>und auch</Text>
<Role index="second" />
<Text>das/der/dieselbe </Text>
<Object index="second" />
</Constraint>

<Constraint xsi:type="Frequency">
<Text> -[Frequency] Wenn ein </Text>
<Object index="0"/>
<Role index="0"/>
<Object index="1"/>
<Role index="0"/>
<Text>, dann diese </Text>
<Object index="0"/>
<Role index="0"/>
<Text>mindestens </Text>
<Minimum/>
<Text> und höchstens </Text>
<Maximum/>
<Role index="0"/>
<Text>(s)</Text>
</Constraint>

<Constraint xsi:type="Irreflexive">
<Text> -[Irreflexive] Kein einzige</Text>
<Object index="0"/>
<Role index="0"/>
<Text> sich selbst</Text>
</Constraint>

<Constraint xsi:type="Symmetric" >
<Text>-[Symmetric] Als</Text>
<Object index="0"/>
<Text> X</Text>
<Role index="0"/>
<Object index="0"/>
<Text> Y</Text>
<Text> , dann auch umgekehrt</Text>
</Constraint>

<Constraint xsi:type="Asymmetric">
<Text> -[Asymmetric] Als</Text>
<Object index="0"/>
<Text> X</Text>
<Role index="0"/>
<Text> </Text>
<Object index="0"/>
<Text> Y, dann ist umgekehrt unmöglich</Text>
</Constraint>

<Constraint xsi:type="Acyclic">
```

```
<Text> -[Acyclic]</Text>
<Object index="0"/>
<Text> kann nicht direkt (oder indirekt via eine Verkettung)</Text>
<Role index="0"/>
<Text> sich selbst sein</Text>
</Constraint>

<Constraint xsi:type="Transitve">
<Text> -[Intransitve] Als</Text>
<Object index="0"/>
<Text>X</Text>
<Role index="0"/>
<Object index="0"/>
<Text>Y, und Y</Text>
<Role index="0"/>
<Text> Z, dann ist es nicht möglich das X</Text>
<Role index="0"/>
<Text>Z</Text>
</Constraint>

</ORMNLBody>
</ORMSchema>
```

Danksagung

Wir sind Herrn Andriy Lisovoy für seinen Beitrag bei der Implementierung von DogmaModeler zu Dank verpflichtet. Weiters geht unser Dank auch an Herrn Hai Nguyen Hoang für seine erste Implementierung vom Verbalisierungsmodul im Rahmen seiner Diplomarbeit.
Die gesamte Arbeit ist teilweise vom europäischen Projekt Knowledge Web NoE (IST-2004-507482) unterstützt.

Bibliographie

[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.