Verbalization of Business Rules
Application to OCL Constraints in the Utility Domain

Rayhana Baghli[1, 2] and Bruno Traverson[1, 2]
[1]*PRiSM, University of Versailles Saint-Quentin-en-Yvelines, Versailles, France*
[2]*ICAME, EDF R&D, Clamart, France*

Keywords: Business Rule, Modelling, Information System, OCL Constraint, Verbalization.

Abstract: Business rules are defined, specified and validated by business experts but they are designed and implemented by technical implementers. Each of them uses languages adapted to their activity and skill. Verbalization of business rules permits to business experts to get a semi-natural expression of rules designed by technical implementers thus facilitating their task of validation. A transformation tool is proposed to automate verbalization and applied to OCL (Object Constraint Language) constraints in the Utility domain.

1 INTRODUCTION

Business rules are described and specified by domain experts. Then, they are designed and implemented by technical implementers. These two distinct modelling phases are known as the specification phase and the design phase. During the specification phase, the business rules are expressed in a language close to the language used by business experts. They are then translated into another language in the design phase. It is equally important to choose a specification language suited to business experts, to choose a design language suitable for technical implementers and to allow the passage of the specification to design and design to specification without loss of information.

Verbalization of business rules translates the rules expressed in a design language into semi-natural expressions. This allows business experts to validate models expressed in a design language without implying any skill on this language.

This principle is widely used in ORM (Object Role Modeling) (Halpin & Morgan, 2010) but less commonly used in object-oriented approaches like UML (Unified Modeling Language) (OMG, 2011) and OCL (Object Constraint Language) (OMG, 2012).

The present paper reports on applying verbalization principle on UML/OCL constraints. This is achieved by the cost of some extensions in UML meta-model and some design rules.

Also, the verbalization tool has been applied on business rules taken from the Utility domain.

Section 2 summarizes state of the art and especially standards in both specification and design languages. Section 3 describes the extensions to UML and the main features of the verbalization tool. Section 4 shows some examples taken from the Utility domain. Section 5 recaps and draws some perspectives.

2 BUSINESS RULE MODELLING

In fact, specification and design may be both used to model business rules but for different purposes. The specification is used to describe constraints in a language close to the business area, while the design is used to implement these constraints in a computer language.

In the specification, the words tend more towards the business vocabulary than to the computer vocabulary. It is necessary to go through this phase. Indeed, the business rules applied to Information Systems are often specified and validated by domain experts not IT, it is essential to express these rules in a language that they understand.

The design is more focused on the way to express business rules in a language for the implementation of IT solutions. This phase is as important as the previous one because the business rules defined by domain experts are to be

Baghli R. and Traverson B..
Verbalization of Business Rules - Application to OCL Constraints in the Utility Domain.
DOI: 10.5220/0004713503480355
In *Proceedings of the 2nd International Conference on Model-Driven Engineering and Software Development* (MODELSWARD-2014), pages 348-355
ISBN: 978-989-758-007-9

implemented and integrated into computer systems.

The two next sections are respectively devoted to specification approaches and to design approaches.

2.1 Specification

During the specification phase, several more or less structured approaches are possible. We identified three main approaches.

The less structured approach is the specification of business rules in natural language. This approach has the advantage of being intuitive, natural and understandable by business experts. However, it is often ambiguous and imprecise. It is not very suitable for expressing complex constraints.

The more structured approach is the specification of business rules in some contractual language. This approach has the merit of being clear and organized but it is formulated in a language not suitable for domain experts.

The third approach is a semi-structured approach. It combines the two previous approaches retaining accessibility of the natural language while framing it in order to reduce ambiguities.

Our choice was the semi-structured approach because, besides the fact that it combines the two former approaches, it is standardized by the OMG (Object Management Group) with the SBVR (Semantics of Business Vocabulary and Business Rules) standard (Chapin & al., 2008).

SBVR is intended to serve as a basis for a declarative description of a complex entity such as an Enterprise Information System. It thus aims to express complex rules based on business semantic. It is optimized for business experts and designed to be used for business regardless of any implementation in a computer system. SBVR allows the production of vocabulary and business rules, based on the constraints expressed by domain experts. Thus, it represents the first specification of the OMG that includes the official use of natural language modelling.

2.1.1 Vocabulary Concepts

Figure 1, taken from the standard, shows the concepts used for the vocabulary part.

The concepts are classified into noun concepts and verb concepts.

The noun concepts are organized into individual concepts, object types, concept types, roles and fact type roles.

Object types, also called general concepts, correspond to two or more objects that form a group

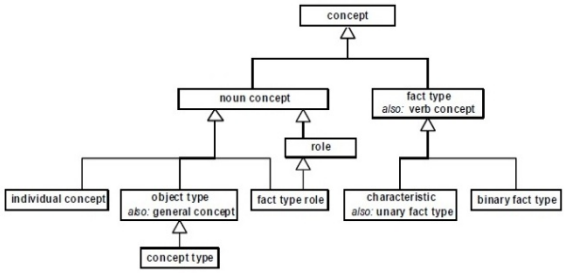

Figure 1: Concepts for business vocabulary.

because of common properties. "Tower" and "Car" are object type concepts.

Individual concepts represent only single objects. For instance, "Eiffel Tower" is an individual concept.

Concept types are object types that specialize conceptual concepts – "role" is a concept type.

Roles correspond to objects that play roles in a function or are used in a given situation. The fact type roles match the roles of objects in the fact types. For example, "driver" and "passenger" are fact type roles done in the following fact type: "In France, the driver sits on the left of the passenger."

The verb concepts are organized into unary fact types and binary fact types.

Unary fact types, also known as characteristics, are the fact types with exactly one role. "A car driver is at least 18 years old" is actually a unary fact type.

In contrast, the binary fact types have exactly two roles. "In France, the driver sits on the left of the passenger" is actually a binary fact type.

2.1.2 Rule Concepts

Figure 2, a subset of a figure taken from the standard, shows the concepts used for the rule part.

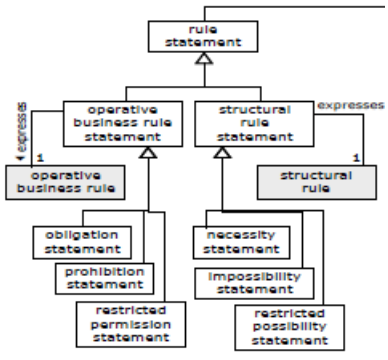

Figure 2: Concepts for business rules.

Rules are logical propositions based on fact types.

Operative rules can be expressed using deontic

logic. Obligation formulates that the proposition is true in all acceptable worlds. Permission formulates that the proposition is true in some acceptable world.

Structural rules can be expressed using alethic logic. Necessity formulates that the proposition is true in all possible worlds. Possibility formulates that the proposition is true in some possible world.

2.2 Design

Three main design approaches have been distinguished: object-oriented approaches, fact-oriented approaches and ontologies.

The object-oriented approach is the most traditional approach and the most widely used of the three. The most often used object-oriented modelling language is UML complemented by OCL for modelling constraints.

The fact-oriented approach considers the real world as objects playing roles in events. The ORM language illustrates this approach.

The ontology approach is illustrated by the ontology languages from the W3C consortium: RDF (Resource Description Framework), RDFS (RDF Schema) and OWL (Web Ontology Language).

2.2.1 Object-Oriented Approach

Nowadays, it is undoubtedly the most widely used design approach and is illustrated by the UML standard.

UML allows structural and behavioral design of Information Systems. UML is extended by OCL that is a declarative language for expressing constraints. This is a deliberately simple access language and has an elementary grammar based on predicate logic and set theory. It can be interpreted by tools.

2.2.2 Fact-Oriented Approach

The second approach considered is the approach based on the facts and illustrated by the ORM solution. ORM is a logical modelling method based on the facts. It is so called because it sees the world in terms of objects that play roles. This approach has been proposed to counter the lack of extensibility of the object-oriented modelling approach.

There are therefore no more notions of classes and attributes. This approach relies on a graphical notation that is complemented with textual formulations. The graphical notation for modelling data allows a visual representation variety of constraints.

ORM has been specially designed to provide models that can be validated, which are semantically stable, expressive and orthogonal. To do this, ORM is based on two principles: the principle of validation and the principle of semantic stability.

The principle of validation is to ensure that the models provided should facilitate validation by domain experts. This is achieved via the principles of population and verbalization. The principle of population specifies that all structures must be easily filled with concrete examples. The principle of verbalization states that all aspects of the model can be verbalized in a language that is understandable by business experts.

The principle of semantic stability requires that the representation of a fact in the model should not be affected unless the meaning of the fact has been changed. This requires avoiding structures based on attributes and offers extensibility.

2.2.3 Ontology-based Approach

The third approach considered is that based on ontologies. In computer science, an ontology is a structured set of terms and concepts representing the meaning of an information field, either by metadata namespace or by the elements of a knowledge domain.

The ontology is itself a model representative of a set of data concepts in a domain and the relationships between these concepts. It is used to reason about the objects in the field. The primary purpose of an ontology is to represent a body of knowledge in a given field. Ontologies are closely related to the Semantic Web.

- RDF (Resource Description Framework) is a data model for the description of objects (resources) and relations between them. It provides for the concept of semantic data model. Data models can be represented in XML syntax.
- RDFS (RDF Schema) provides a vocabulary for describing properties and classes of RDF resources, and semantics for generalization of properties and classes.
- OWL (Web Ontology Language) adds more capabilities to describe properties and classes: relations between classes (e.g. disjunction), cardinality (e.g. "only one"), equality, properties and characteristics (e.g. symmetry).

2.2.4 Discussion

Table 1 synthesizes and compares the three approaches to designing business rules together.

Thus, the three approaches can express the SBVR vocabulary and business rules. But only the

Table 1: Comparison of design approaches.

Criteria	UML/OCL	ORM	RDF(S)/OWL
Expression of business vocabulary	X	X	X
Expression of business rules	X	X	X
Expression of modal logics		X	
Maturity	X		
Extensibility		X	X
Validation	X	X	
Verbalization		X	

fact-oriented approach handles alethic and deontic rules.

Models from the fact-oriented and ontology-based approaches are extensible but the object-oriented approach is the most mature of the three approaches.

Also, the object-oriented and the fact-oriented approaches operate in a closed world in contrast to the ontology-based approach that operates in an open world.

Finally, verbalization of business rules is only handled in fact-oriented approach.

3 APPLICATION TO UML/OCL

To verbalize a model and business rules written in UML/OCL, it is necessary to extend the UML metamodel in order to support different types of modal rules. The verbalization module of business rules has then been developed taking into account the extension of the UML metamodel.

3.1 Meta-Model

OCL constraints are defined for the elements of a UML model. However, UML does not handle alethic and deontic constraints. So, there is a loss of information when SBVR rules are translated to UML/OCL language.

Two solutions have emerged. The first solution was to create a DSL (Domain Specific Language) for modal rules and the second solution was to extend UML using the profile extension mechanism to integrate the concepts of modal logic.

The first solution leads to describe a complete meta-model. In contrast, the second approach extends and specializes the concepts previously defined by UML.

We opted for the second solution because it allows reuse of the UML concepts (classes, associations, attributes, operations ...).

The only need we have is to specialize the concept of constraint (constraint) to specify the type of modal logic. Figure 3 shows the stereotypes that we have defined.

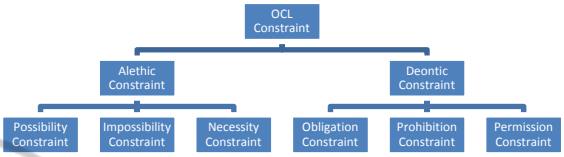

Figure 3: Extending the UML metamodel.

As shown in figure 3, we extended the concept of UML constraint to six kinds of constraints, one for each type of modal rule: Possibility, Impossibility, Necessity, Obligation, Prohibition and Permission. Thus, each modal rule expressed in SBVR finds correspondence in UML extended meta-model.

More precisely, the UML profile definition is described in figure 4.

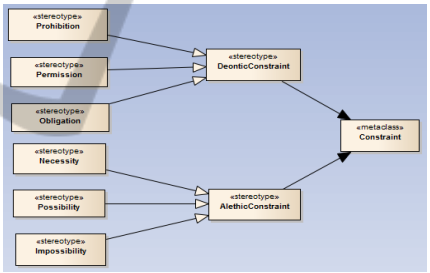

Figure 4: COVE UML Profile Definition.

In fact, there are two other types of modal rules in SBVR: Restrictive permission and restrictive possibility.

Permissions and restrictive permissions are represented by a single stereotype because restrictive permission is actually a rule including an implication together with a permission. OCL is a formal language that implements the “implies” construction and, in order to avoid redundancy, we have chosen to represent permission and restrictive permission by a single stereotype.

Possibility and restrictive possibility are represented by the same stereotype for the same reason.

3.2 Verbalization Module

To implement the verbalization module, we first had to choose a tool supporting UML and OCL. Our

choice fell on " Modelio " tool (Modelio) because it is an Open Source tool dedicated to UML which allows the definition of profiles. Moreover, it can easily integrate additional modules as plug-ins.

The verbalization module was appointed COVE, which is the concatenation of the first two letters of the two words "Constraint Verbalizator".

During the implementation of the COVE module, we first described the UML profile for the extension of the meta-model to take into consideration modal rules. This profile, noted CoveProfile, was then integrated to the UML metamodel. Thus, when creating a constraint, the user can choose from among the six stereotypes defined by the profile. Figure 5 is a screenshot showing the different choices when creating constraints.

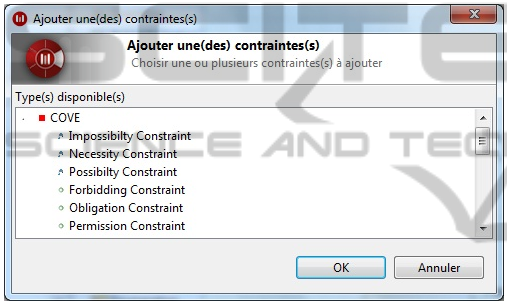

Figure 5: COVE constraints.

Alethic and deontic constraints are distinguished by associating them to different icons, a green gear for deontic constraints and a blue file for alethic constraints.

After the CoveProfile profile has been defined, we have implemented the code for verbalizing constraints. Thus, each type of constraint is verbalized differently following a predefined template. The COVE Module retrieves the text of the OCL constraint and stereotype. The text is then parsed and keywords recovered.

Concerning the verbalization of stereotypes, for each stereotype, a modal message is defined. For instance, the modal message that corresponds to the "Possibility Constraint" stereotype is "It is possible".

The modal message is then concatenated to other parts of the sentence to build the verbalization form of the business rule.

The parser begins with looking for the type of constraint. The three types of constraints supported by the COVE are the three main types of OCL constraints, i.e. invariants, pre-conditions and post-conditions.

Once the modal message and the type of constraint detected, the analyzer proceed in scanning the text of the constraint. During the analysis stage, the verbalizer detects and verbalizes three logical connectors: "and", "or" and "implies". This analysis also helps in separating logical expressions containing only simple variables and/or comparator expressions.

Simple expressions are then analyzed to be verbalized, the six types of comparison are taken into account (=, !=, <, >, <=, >=). The COVE module verbalizes boolean variables differently than other types of variables.

Also, in OCL, it is possible to navigate associations to put constraints on attributes and/or operations of related classes. In this case, the verbalizer will navigate the association and verbalize the constraint using the roles of the association and the classes involved in the constraint.

The verbalizer differentiates attributes and operations. Verbalization form is a function of the nature of the elements to compare. Also, the conjugation of verbs in the indicative and subjunctive is supported by the module.

3.3 Design Rules

To ensure consistency of verbalized constraints, four main recommendations are made when developing the UML model.

- All role names on associations shall be defined as the verbalizer uses them in the verbalization.
- All role names must be expressed as verbs in the third person singular.
- All attribute names must be nouns or noun phrases (e.g. age, electrical diagnosis ...)
- All operation names must be expressed in an infinitive verbal form (e.g. renovate, ...).

4 APPLICATION TO UTILITY DOMAIN

Verbalization solution we have proposed and implemented was applied to two real case studies in the fields of electricity.

The first case study includes rules defined by the Promotelec association in order to ensure the safety of electrical installations in private homes.

The second case study includes rules for the installation of pipes in nuclear power plants.

We present in this section some rules taken from these two case studies. Each business rule is first described in natural language as it is present on the original documents. Then, this rule is transformed

into an SBVR specification and designed in UML/OCL. Finally, automatic verbalization of the rule is presented.

4.1 Promotelec Rules

In this section, we present a business rule from the rules set defined by the Promotelec association for the safety of electrical installation. It concerns the renovation of the electrical installation. The business rule is of deontic kind and is an obligation.

> Business Rule 1 (original text): The renovation of an electrical installation must be done by the electrician who performed the installation or by an electrician who has a "Qualifelec" qualification.

In analyzing the business rule expressed in natural language, we can extract its noun concepts (electrical installation, electrician, qualification), verb concepts (renovate, perform the installation) and modal logic kind (obligation). Thus, this rule can be reformulated in the semi-natural language SBVR by a business analyst.

> Business Rule 1 (SBVR form): It is mandatory to renovate the electrical installation by the electrician who performed the installation or by an electrician who has a "Qualifelec" qualification.

We can model this business rule in UML/OCL. Figure 6 shows the classes necessary to model the business rule. Thus, the concepts of names "electrical installation" and "electrician" are translated into classes. The concepts of "qualification" and "perform installation" are translated into class attributes. The verb concept "renovate" is translated in an operation.

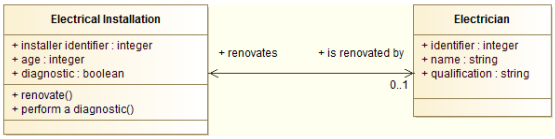

Figure 6: UML model for Business Rule 1.

The type of modal rule is specified during the creation of OCL constraints by a designer. In this example, it is an obligation. The type and the text of the OCL constraint is the following. The constraint is a pre-condition checked before the execution of a renovate operation.

> Business Rule 1 (OCL text):
> Context : Electrical Installation::renovate()
> pre : self.installer identifier = is renovated by.identifier or self.is renovated by.qualification = 'Qualifelec'

Once the UML model and the OCL constraint have been modeled by the designer, the verbalization module produces the verbalized form of the business rule. Figure 7 shows the result of the verbalization.

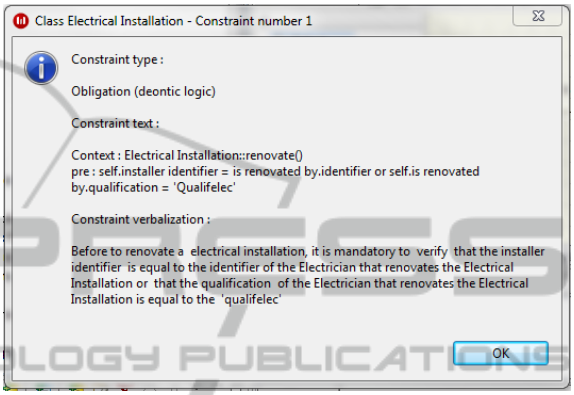

Figure 7: Verbalization of Business Rule 1.

4.2 Pipe Installation Rules

In this section, we present a business rule extracted from the case study on the installation of pipes in nuclear power plants. This is a prohibition rule (deontic logic). It expresses the prohibition to install brackets on sections that can be periodically removed.

> Business Rule 2 (original text): It is prohibited to install brackets on sections that can be removed periodically.

In analyzing the business rule expressed in natural language, we can extract noun concepts (bracket, section), verb concepts (install) and modal rule (prohibition rule). Thus, the business rule can be reformulated in SBVR by the business analyst.

> Business Rule 2 (SBVR): It is forbidden to install a bracket on sections that are removed periodically.

We model this business rule in UML/OCL. Figure 8 shows the classes necessary to represent the business rule. Thus, the noun concepts "bracket" and "section" are represented by classes. The noun concept "remove periodically" is represented by a boolean attribute "periodic disassembly". The

relationship between the bracket and the section is expressed by an association "support".

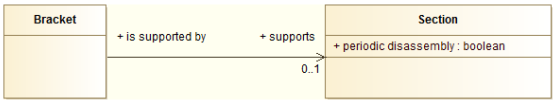

Figure 8: UML model for Business Rule 2.

The OCL constraint is an invariant type constraint.

```
Business Rule 2 (OCL text):
Context Bracket
inv : self.supports.periodic disassembly = true
```

Once the UML model and the OCL constraint have been described by the designer, the Verbalization module produces the verbalized form of the business rule. Figure 9 shows the result of the verbalization.

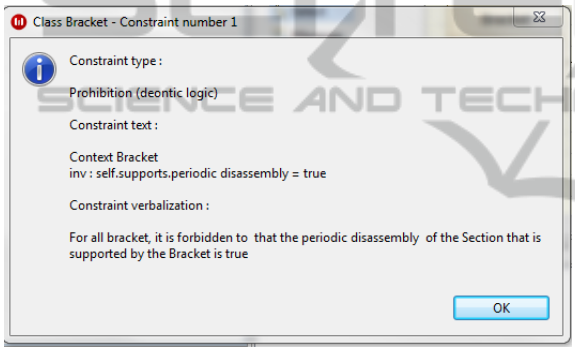

Figure 9: Verbalization of Business Rule 2.

5 CONCLUSIONS

The work carried out at the state of the art has helped us to realize that no modeling approach is better than the other approaches. Each approach is suited to certain needs and contexts. In our environment, the modeling approach that best fit our needs is the object-oriented approach.

However, in the study of other modeling approaches, we found that the fact-oriented approach provides the ability to verbalize the business rules. Indeed, the verbalization in ORM is a real advantage for validation.

Also, the fact-oriented approach and the specification language SBVR have a common benefit that is the support of alethic and deontic rules.

Thus, our proposed solution permits to support deontic and alethic rules and performs verbalization of business rules expressed in UML/OCL. The solution was developed as a COVE module and then integrated into the Modelio environment.

Finally, both the development of the module and its applications on case studies allowed us to discover the benefits and the challenges offered by the solution.

Thus, the verbalization module actually facilitates the validation of business rules by business experts. Also, the verbalized business rules are often more accurate than rules expressed in natural language.

However, the result of verbalization can sometimes be more verbose than the constraint expressed in natural language, especially when it contains multiple logical connectors. Also, like other automated translation systems, verbalizing business rules can sometimes contain language errors such as wrong conjugate verbs or past participles. However, verbalization, even with some natural language errors, is a significant help to business rules validation by business experts.

Looking ahead to our research, several other lines of research can be investigated to enrich the object-oriented modelling approach. Indeed, the different approaches to modeling business rules have great methodological diversity. The only types of OCL rules considered by the verbalizer are invariants, pre-conditions and post-conditions. Thus, we wish to extend the module to verbalize other forms of OCL rules, such as rules implementing sets, collections and OCL functions for these sets.

Following the study of different approaches to specification and design business rules, several issues were found in each of the modelling phases.

- Choice of specification language: In this phase, we wonder if the SBVR language is sufficient to express any kind of business rules. Indeed, SBVR does not specify the temporal aspect in the business rules. Also, SBVR does not make the link with business processes.

- Choice of design language: In the study of different approaches to the design of business rules, the ability to validate the model seemed to be of great importance. Thus, the question arises whether it is possible and interesting to close the supposed open world in ontologies. Also, the UML modelling is not extensible.

- Transition from specification to design: The result of the specification phase model is intended to be modeled during the design phase, it is important that the design languages support the concepts described by the specification. This is not always the case, especially for UML/OCL and RDF(S)/OWL, which do not support deontic and alethic constraints.

ORM languages and RDF (S)/OWL also do not support the time constraints while UML/OCL offers limited support.

- Transition from design to specification: Once the design is done, it is interesting to compare the result with the specification described by business experts. We raised that the ORM language provides the ability to verbalize the business vocabulary and rules, which significantly facilitates the validation of technical models by business experts. An interesting prolongation of this work may be to extend the principle of verbalization to other design approaches like ontologies.

REFERENCES

Chapin, D., Baisley, D. E., & Hall, H., 2008. *Semantics of Business Vocabulary and Business Rules (SBVR).* Object Management Group.

Chapin, D., Baisley, D. E., & Hall, H., 2005. *Semantics of Business Vocabulary & Business Rules (SBVR).* In Rule Languages for Interoperability.

Chapin, D. 2008. SBVR: *What is now possible and why?*, Business Rules Journal.

Baisley, D. E., Hall, J., & Chapin, D., 2005. *Semantic Formulations in SBVR.* In Rule Languages for Interoperability.

Halpin, T. , 2001, January. Augmenting UML with Fact-orientation. In System Sciences, 2001. *Proceedings of the 34th Annual Hawaii International Conference on (pp. 10-pp).* IEEE.

Halpin, T., 2005. ORM 2 Graphical Notation, *Technical Report ORM2-02*, Neumont University, 2005.

Halpin, T., & Curland, M. 2006. ORM 2 Constraint Verbalization Part.

Halpin, T., & Morgan, T., 2010. *Information modeling and relational databases*. Morgan Kaufmann.

OMG, 2011. *Unified Modeling Language specification. Object Management Group*, 1034.

OMG, 2012. OCL Specification. OMG Document.

Gruber, T. R. 1993. *A translation approach to portable ontology specifications. Knowledge acquisition*, 5(2), 199-220.

Modelio. http://www.modelio.org.

McGuinness, D. L., & Van Harmelen, F., 2004. *OWL web ontology language overview.* W3C recommendation, 10(2004-03), 10.

Klyne, G., Carroll, J. J., & McBride, B., 2004. *Resource description framework (RDF): Concepts and abstract syntax.* W3C recommendation, 10.

Brickley, D., & Guha, R. V. 2000. *Resource Description Framework (RDF) Schema Specification 1.0:* W3C Candidate Recommendation 27 March 2000.

Halpin, T., & Bloesch, A. 1999. Data modeling in UML and ORM: a comparison. *Journal of Database Management (JDM)*, 10(4), 4-13.

Halpin, T., & al 2001. Augmenting UML with Fact Orientation. *In Proceedings of the 34th Annual Hawaii International Conference on System Sciences* (HICSS-34)-Volume (Vol. 3, p. 3053).

Motik, B., Horrocks, I., & Sattler, U. 2009. *Bridging the gap between OWL and relational databases*. Web Semantics: Science, Services and Agents on the World Wide Web, 7(2), 74-89.

Halpin, T., & Curland, M., 2006. Automated verbalization for ORM 2. In *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops (pp. 1181-1190)*. Springer Berlin Heidelberg.