# Sharing OWL/SWRL and UML/OCL Rules

Milan Milanović[1], Dragan Gašević[2], Adrian Giurca[3], Gerd Wagner[3], and Vladan Devedžić[1]

[1] FON-School of Business Administration, University of Belgrade, Serbia
milan@milanovic.org, devedzic@etf.bg.ac.yu
[2] School of Interactive Arts and Technology, Simon Fraser University Surrey, Canada
dgasevic@sfu.ca
[3] Institute of Informatics, Brandenburg Technical University at Cottbus, Germany
Giurca@tu-cottbus.de, G.Wagner@tu-cottbus.de

**Abstract.** The paper presents a metamodel-driven model transformation approach to sharing rules between the Semantic Web Rule Language along with the Web Ontology Language (OWL/SWRL) and Object Constraint Language (OCL) along with UML (UML/OCL). The solution is based on the REWERSE Rule Markup Language (R2ML), a MOF-defined general rule language, as a pivotal metamodel and the bi-directional transformations between OWL/SWRL and R2ML and between UML/OCL and R2ML.

## 1. Introduction

In this paper, we further extend the research in approaching the Semantic Web and MDA by proposing a solution to interchanging rules between two technologies. More specifically, we address the problem of mapping between the Object Constraint Language (OCL), a language for defining constrains and rules on UML and MOF models and metamodels, and the Semantic Web Rule Language (SWRL), a language complementing the OWL language with features for defining rules. In fact, our proposal covers the mapping between OCL along with UML (i.e., UML/OCL) and SWRL along with OWL (OWL/SWRL).

In our solution, we use R2ML [1], a MOF-defined general rule language capturing integrity, derivation, production, and reaction rules, which covers almost all of the use cases requirements of the W3C RIF WG [4]. R2ML is a pivotal metamodel for interchanging between OWL/SWRL and UML/OCL. This means that we have to provide a two way mappings for either of two rule languages with R2ML. The main benefit of such an approach is that we can actually map UML/OCL rules into all other rule languages (e.g., Jess, F-Logic, and Prolog) that have mappings defined with R2ML. Since various abstract and concrete syntax are used for representing and sharing all three metamodels (e.g., R2ML XMI, R2ML XML, OWL XML, OCL XMI, UML XMI, OCL text-based syntax), the implementation is done by using Atlas Transformation Language (ATL) and by applying the metamodel-driven model transformation principle.

## 4. Transformations

***Mapping between OWL/SWRL and R2ML.*** In a nutshell, this mapping consists of two transformations. The first one is from OWL/SWRL rules represented in the OWL/SWRL XML format into the models compliant to the RDM (Rule Definition Metamodel) [2]. Second, such RDM-based models are transformed into R2ML models, which are compliant to the R2ML metamodel and this represents the core of the transformation between the OWL/SWRL and R2ML. The rationale for introducing one more metamodel, i.e. RDM, is that it represents an abstract syntax of the SWRL (with OWL) language in the MOF technical space.

**Step 1.** This step consists of injecting OWL/SWRL rules from the XML technical space into the MOF technical space. Such a process is shown in detail for R2ML XML and the R2ML metamodel in.

**Step 2.** In this step, we transform the XML model obtained in Step 1 into the RDM-compliant model. This transformation is done by using the ATL transformation named *XML2RDM.atl*. The output RDM model conforms to the RDM metamodel.

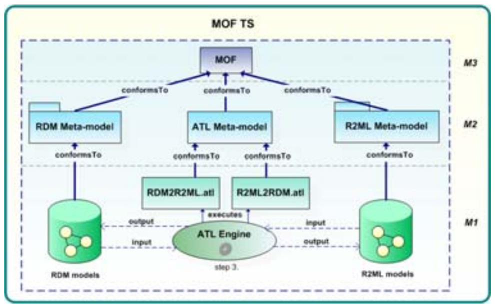

**Fig. 1.** The transformation of the models compliant to the RDM metamodel into the models compliant to the R2ML metamodel

**Step 3.** The last step in this transformation process is the most important transformation where we transforming RDM model to R2ML model (Fig. 1). This means that this step represents the transformation of the OWL/SWRL abstract syntax into the R2ML abstract syntax. In Table 1, we give an excerpt of mappings between the SWRL XML schema, XML metamodel, RDM metamodel and R2ML metamodel.

An additional step is to transform rules from R2ML into the R2ML XML concrete syntax, which we have also implemented by using the ATL language.

**Table 1.** An excerpt of mappings between the OWL/SWRL XML schema, XML metamodel, RDM metamodel, and the R2ML metamodel

| OWL/SWRL | XML metamodel | RDM metamodel | R2ML metamodel |
|---|---|---|---|
| individualPropertyAtom | Element name = 'swrlx:individualPropertyAtom' | Atom | UniversallyQuantified Formula |
| OneOf | Element name = 'owlx:OneOf' | EnumeratedClass | Disjunction |
| var | Element name = 'ruleml:var' | IndividualVariable | ObjectVariable |
| sameIndividualAtom | Element name = 'swrlx:sameIndividualAtom' | Atom | EqualityAtom |
| maxcardinality | Element name = 'owlx:maxcardinality' | MaxCardinality Restriction | AtMostQuantifiedFormula |

***Mapping between UML/OCL and R2ML.*** Since the R2ML and OCL metamodels are both located in the MOF technical space and there is an metamodel for OCL defined in the OCL specification, the transformation by ATL is straightforward in terms of technological requirements, i.e. we do not have to introduce an additional metamodel like we have done with RDM.

**Step 1.** We transform an R2ML model into an OCL model by using an ATL transformation. The output OCL model conforms to the OCL metamodel. In Table 2, we give an excerpt of mappings between the R2ML metamodel and OCL metamodel on which this ATL transformation is based.

**Table 2.** An excerpt of mappings between R2ML metamodel, OCL metamodel, and OCL code

| R2ML metamodel | OCL metamodel | OCL code |
|---|---|---|
| Conjuction | OperationCallExp<br>referredOperation (name = 'and') | Operand and Operand |
| Implication | OperationCallExp<br>referredOperation (name = 'implies') | Expression implies Expression |
| ObjectVariable | Variable | Variable name |
| RoleFunctionTerm | PropertyCallExp<br>referredProperty (name = 'property')<br>source Variable | Variable.property |
| AtMostQuantifiedFormula | OperationCallExp<br>referredOperation (name = '<=')<br>argument maxvalue | Expression <= maxvalue |

**Step 2.** Because the OCL concrete syntax is located in the EBNF technical space, we need to get an instance of the OCL metamodel (abstract syntax) into EBNF technical space. Since the concrete syntax of OCL has been implemented in TCS according to the OCL syntax, we can use to perform this transformation from R2ML to OCL, and final text-based OCL. In the opposite direction, from OCL to R2ML, the solution is to use a TCS for creating model from code. When the OCL model is generated form the OCL code, and our case an OCL model

## 6. Conclusions

The presented research is a next step towards the further reconciliation of MDA and Semantic Web languages, and hence continues the work established by the OMG's ODM specification that only addressed mappings between OWL and UML [3], while we extended it on the accompanying rule languages, i.e., SWRL and OCL. This paper is accompanied by transformations that are available at http://oxygen.informatik.tu-cottbus.de/rewerse-i1/?q=node/15. We also plan to extend our rule transformation framework in order to support other OMG's specifications covering rules, i.e., the ones for business and production rules.

## References

1. Wagner, G., Giurca, A., Lukichev, S. (2005). "R2ML: A General Approach for Marking-up Rules," *In Proc of Dagstuhl Sem. 05371 on Princ. and Practices of Sem. Web Reasoning*.
2. Brockmans, S., Haase, P. (2006). "A Metamodel and UML Profile for Rule-extended OWL DL Ontologies - A Complete Reference," Universität Karlsruhe (TH) - Technical Report.
3. OMG ODM (2006). "Ontology Definition Metamodel," 6th Revised Submission.
4. Rule Interchange Format: Use cases and requirements, W3C Working Draft, http://www.w3.org/TR/rif-ucr/