Workshop on OCL and Textual Modeling: OCL 2012

Mira Balaban
Ben-Gurion University of the Negev, Israel

Jordi Cabot
INRIA-Ecole des Mines de Nantes, France

Martin Gogolla
University of Bremen, Germany

Claas Wilke
Technische Universität Dresden, Germany

Modeling started out with visual notations such as UML and its precursors. As the modeling paradigm evolves and matures, there is a growing need in a proliferation of well-founded, but also easy-to-use modeling languages, adapted for specific tasks and domains, that might need to work in synergy for achieving complex applications. Past experience suggests that precise modeling often goes hand in hand with textual notations. The lack of precision that characterizes most visual modeling languages does not go along with the emerging Model Driven Engineerin (MDE) approach, which puts models at the center of the software development process. Being based on successive model transformations, MDE requires well defined, formally specified modeling languages, at a level of precision visual notations lack.

The limitations of visual notations encouraged the development of text-based modeling languages that either integrate with or replace graphical notations for modeling. Examples are OCL, textual MOF, Epsilon, and Alloy.

OCL originated as a textual add-on to UML. In recent years, OCL has evolved into an important textual constraint language in connection with languages beyond UML. It has, thus, broadened its scope. Connecting OCL to other modeling languages, as well as providing extensions to the language for coping with new tasks are currently hot topics within the OCL community.

At the MoDELS 2012 conference, the "OCL and Textual Modeling" workshop is a forum where researchers and practitioners interested in building models using OCL or other kinds of textual languages can directly interact, report advances, share results, identify tools for language development, and discuss appropriate standards. The workshop encouraged discussions for achieving synergy from different modeling language concepts and modeling language use. The close interaction enabled researchers and practitioners to identify common interests and options for potential cooperation.

The workshop emerged from a successful series of workshops addressing issues relating to OCL and other textual modeling languages. These workshops were held as part of the UML/MODELS conferences. Previous editions of the workshop have successfully led to new collaborations between researchers in the OCL community.

The key objectives of this workshop series are:

- To provide a forum for the dissemination and analysis of central topics regarding OCL, including OCL evolution, the OCL standard, applications of OCL, and tool support for OCL.
- To provide a forum for dissemination and analysis of textual modeling languages, including textual alternatives for graphical notations, text-based extensions to standard modeling languages, executable modeling notations, and formal semantics of languages. Of special interest are discussions that highlight the correlation between textual modeling languages, clarifying boundaries and integration modes between textual modeling languages, and pragmatic use and formal aspects of the languages. Additional aspects involve the general purpose vs. task focused nature of the languages, and domain specific and visualization aspects.

The "OCL and Textual Modeling" workshop at MoDELS 2012 (`http://st.inf.tu-dresden.de/OCL2012/`) attracted more than 30 participants. Apart from people mainly interested in OCL, there were participants (researchers and practitioners) working on application domains requiring the use of constraint, rule, query, or transformation languages. In particular, participants have contributed to practical applicability of OCL.

Workshop Program

The workshop had four sessions, devoted to *OCL study and tools*, *use of high-order functional languages and modeling techniques*, *OCL uses*, and *OCL use reports.*

OCL Study and Tools.

The three contributions in this session deal with improvements in OCL tools, from three different perspectives:

1. *Tool Supported OCL Refactoring Catalog* by Jan Reimann, Claas Wilke, Birgit Demuth, Michael Muck, and Uwe Aßmann: The paper presents an implemented catalog of OCL refactoring. The catalog is aimed to support the growing complexity of OCL constraints along the evolution of a software model.
2. *An extensible OCL Virtual Machine and Code Generator* by Ed Willink: This paper presents an OCL virtual machine for Eclipse OCL, describes optimizations that give dramatic performance improvements, and discuss extensions of the presented framework to OCL-based languages like QVT.

OCL'12 September 30 2012, Innsbruck, Austria
2012 ACM 978-1-4503-1799-3/12/09 .

3. *Featherweight OCL: A study for the consistent semantics of OCL 2.3 in HOL* by Achim D. Brucker and Burkhart Wolff: This paper discusses a recent formulation of OCL in Higher-Order-Logic (HOL). The formulation reveals several inconsistencies and contradictions in the current version of the OCL standard. These problems challenge OCL implementations.

Use of high-order functional languages and modeling techniques.

The three contributions in this session present three OCL alternatives. The first uses Scala as a host language for defining an internal constraint language; the second uses Lua for defining a library for model query and transformation; the third uses an ontology language for capturing requirements:

1. *On the Use of an Internal DSL for Enriching EMF Models* by Filip Křikava and Philippe Collet: This paper analyzes some scalability problems in OCL, e.g., in enriching large EMF models, claiming that they result from the general purpose nature of OCL. The paper suggests an alternative route of defining an OCL-like language as a DSL that is embedded within a powerful language, thereby taking full advantage of the host language, including state-of-the-art tool support. The paper describes a Scala-based implementation, in which Scala concepts are used for defining such a DSL, and discusses the mapping of OCL into such a language.

2. *Library for Model Querying – IQuery* by Renārs Liepiņš: This paper addresses query and transformation problems in projects where data is stored in different repositories, using different representations. The paper suggests an approach based on a query and transformation library, built in the general purpose language being used in the project (assuming that there is one). The paper shortly describes *IQuery*, a Lua-based library for EMOF-like data stores, and shortly compares it with other similar libraries or languages.

3. *Ontology Driven Design of EMF Metamodels and Well-formedness Constraints* by Benedek Izsó, Zoltán Szatmári, Gábor Bergmann, Akos Horváth, István Ráth and Dániel Varró: This paper proposes a combined use of ontologies and DSM techniques, in which domain requirements that are captured in a textual ontology language are automatically translated into EMF metamodels that are evaluated by EMF tools.

OCL applications.

This session presented two OCL-based applications dealing with model-based testing, and one application for workflow verification:

1. *Modeling and Executing ConcurTaskTrees using a UML- and SOIL-based Metamodel* by Jens Brüning, Martin Kunert and Birger Lantow: This paper describes an implemented approach where workflow models represented in the ComcurTaskTrees language are verified using a meta-model-specification in which soundness properties and operational semantics are captured. The OCL-like imperative language SOIL is further used for testing dynamic control flow properties of models.

2. *Automatic Generation of Test Models and Properties from UML Models with OCL Constraints* by Miguel A. Francisco and Laura M. Castro: This paper presents a method for generating test suites from UML/OCL models. The generated tests are then executed and evaluated by an off-the-shelf testing tool.

3. *Transformation rules from UML4MBT meta-model to SMT meta-model for model animation* by Jérôme Cantenot, Fabrice Ambert and Fabrice Bouquet: This paper describes an application where specifications in the UML4MBT language, a subset of UML/OCL intended for supporting model based testing, are translated into SMT-lib, the input language of the SMT solver which is used for test generation. This approach enables to combine the high level abstraction of a UML/OCL language with the theoretical power of a formal solver.

Reports on OCL in project environments.

The final session of the workshop involved three short reports about using OCL in various applications:

1. *Model-based formal specification of a DSL library for a qualified code generator* by Arnaud Dieumegard, Andres Toom and Marc Pantel: A report on using OCL for specifying graphical modeling languages.

2. *The Secret Life of OCL Constraints* by Oliver Hofrichter, Lars Hamann and Martin Gogolla: This paper reports on pragmatic aspects of using OCL within a large project.

3. *Experiences using OCL for Business Rules on Financial Messaging* by David Garry: This paper reports on learned experience in using OCL within a commercial environment.

The discussions in the workshop have shown that many research questions and practical problems remain to be solved, hopefully in future workshops on this subject. In particular, two key conclusions from this edition were:

- Many people use variants of the standard OCL. These variants usually involve (at the same time) restricting the language so that only a core OCL is used and extending it with new types/functions needed for the domain they are using for. Clearly, there is a need to add a modularization mechanism to OCL that allow people to import (domain-specific) libraries on top a core OCL language. The new version of OCL (to appear in 2013) will advance in this direction.

- Several types of formalisms are used to analyze/reason on OCL expressions (CSPs, SAT, SMT, HOL, ...). The choice of a formalisms seems to be closely related to the available expertise of the research group. A comparison between the trade-offs of the different formalisms when reasoning on OCL expressions could be very useful. Martin Gogolla and Fabian Büttner agreed to take the lead on creating a working group on this, involving representatives of the different OCL-to-X tools available.