A Formal Methods Environment for OCL: HOL-OCL 2.0

Achim D. Brucker, Frédéric Tuong, and Burkhart Wolff

16th International Workshop in OCL and Textual Modeling
October 2, 2016, Saint-Malo, France

HOL-OCL 2.0: Our Goal

Our goal:

- A *certified* formal tool for UML/OCL:
  HOL-OCL 2.0 is guaranteed (by construction) to be
  - consistent and
  - compliant to a formal semantics of UML/OCL
- A tool that allows to use UML/OCL in formally *certified* development processes
  HOL-OCL 2.0 provides
  - Interactive theorem proving in terms of UML/OCL constructs
  - Generation of specification and proof documents
  - Code generation
  - . . .

HOL-OCL 2.0: Implementation

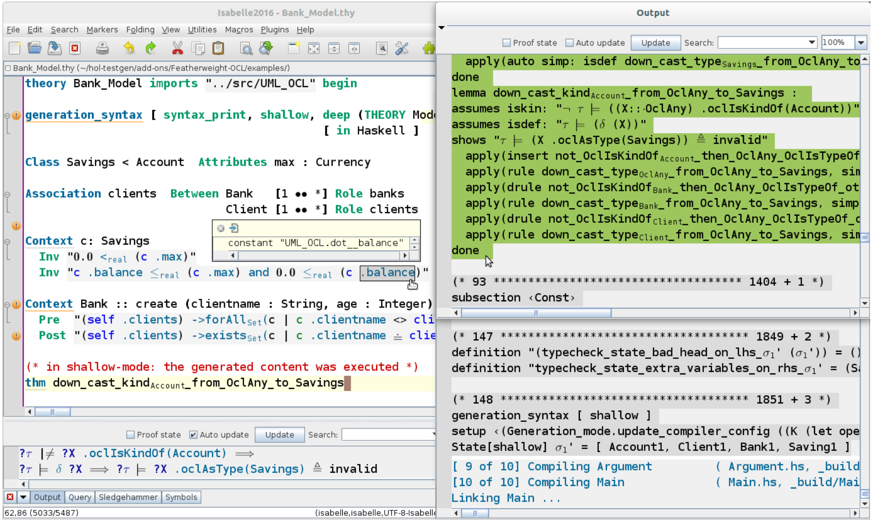
Implementation:

- Based on Isabelle 2016
- Based on a reflexive implementation approach (formal meta-modelling approach)

Relation to HOL-OCL 1.x:

- Both share the same goals and vision
- HOL-OCL 2.0 is a complete re-implementation:
  - using a modern Isabelle (modern IDE, more powerful proof methods, etc.)
  - using a formal meta-modelling approach (instead of traditional datatype packages)
  - supporting OCL with `invalid` and `null`

Tool Demo



Isabelle2016 - Bank_Model.thy
File Edit Search Markers Folding View Utilities Macros Plugins Help
Bank_Model.thy (~/hol-testgen/add-ons/Featherweight-OCL/examples/)

```
theory Bank_Model imports "../src/UML_OCL" begin

generation_syntax [ syntax_print, shallow, deep (THEORY Mode
                                             [ in Haskell ]

Class Savings < Account  Attributes max : Currency

Association clients  Between Bank   [1 •• *] Role banks
                             Client [1 •• *] Role clients

Context c: Savings
  Inv "0.0 <real (c .max)"
  Inv "c .balance ≤real (c .max) and 0.0 ≤real (c .balance)"

Context Bank :: create (clientname : String, age : Integer)
  Pre  "(self .clients) ->forAllSet(c | c .clientname <> cli
  Post "(self .clients) ->existsSet(c | c .clientname ≐ clie

(* in shallow-mode: the generated content was executed *)
thm down_cast_kindAccount_from_OclAny_to_Savings
```

```
constant "UML_OCL.dot__balance"
```

```
?τ |≠ ?X .oclIsKindOf(Account) ⟹
?τ ⊨ δ ?X ⟹ ?τ ⊨ ?X .oclAsType(Savings) ≜ invalid
```

Output Query Sledgehammer Symbols
62,86 (5033/5487)
(isabelle,isabelle,UTF-8-Isabelle
Output
□ Proof state □ Auto update Update Search: 100%

```
  apply(auto simp: isdef down_cast_typeSavings_from_OclAny_to
done
lemma down_cast_kindAccount_from_OclAny_to_Savings :
assumes iskin: "¬ τ ⊨ ((X::·OclAny) .oclIsKindOf(Account))"
assumes isdef: "τ ⊨ (δ (X))"
shows "τ ⊨ (X .oclAsType(Savings)) ≜ invalid"
  apply(insert not_OclIsKindOfAccount_then_OclAny_OclIsTypeOf
  apply(rule down_cast_typeOclAny_from_OclAny_to_Savings, sim
  apply(drule not_OclIsKindOfBank_then_OclAny_OclIsTypeOf_ot
  apply(rule down_cast_typeBank_from_OclAny_to_Savings, simp
  apply(drule not_OclIsKindOfClient_then_OclAny_OclIsTypeOf_c
  apply(rule down_cast_typeClient_from_OclAny_to_Savings, sim
done

(* 93 ************************************ 1404 + 1 *)
subsection ‹Const›
```

```
(* 147 ************************************ 1849 + 2 *)
definition "(typecheck_state_bad_head_on_lhs_σ1' (σ1')) = ()
definition "typecheck_state_extra_variables_on_rhs_σ1' = (Sa

(* 148 ************************************ 1851 + 3 *)
generation_syntax [ shallow ]
setup ‹(Generation_mode.update_compiler_config ((K (let ope
State[shallow] σ1' = [ Account1, Client1, Bank1, Saving1 ]
[ 9 of 10] Compiling Argument        ( Argument.hs, _build
[10 of 10] Compiling Main            ( Main.hs, _build/Mai
Linking Main ...
```

Thank you for your attention!

Any questions or remarks?

Related Publications

Achim D. Brucker, Frédéric Tuong, and Burkhart Wolff.
Featherweight ocl: A proposal for a machine-checked formal semantics for ocl 2.5.
Archive of Formal Proofs, January 2014.
ISSN 2150-914x.
http://www.brucker.ch/bibliography/abstract/brucker.ea-featherweight-2014.
http://www.isa-afp.org/entries/Featherweight_OCL.shtml, Formal proof development.

Achim D. Brucker, Frédéric Tuong, and Burkhart Wolff.
Featherweight ocl: A proposal for a machine-checked formal semantics for ocl 2.5.
Technical Report 1582, lri, Univ Paris Sud, cnrs, Centrale Supélec, Université Paris-Saclay, France, September 2015.
http://www.brucker.ch/bibliography/abstract/brucker.ea-formal-semantics-ocl-2.5-2015.

Delphine Longuet, Frédéric Tuong, and Burkhart Wolff.
Towards a tool for featherweight ocl: A case study on semantic reflection.
In Achim D. Brucker, Carolina Dania, Geri Georg, and Martin Gogolla, editors, *Proceedings of the* models *2014* ocl *Workshop (*ocl *2014)*, volume 1285 of ceur *Workshop Proceedings*, pages 43–52. ceur-ws.org, 2014.
http://www.brucker.ch/bibliography/abstract/longuet.ea.ea-ocl-reflection-2014.

Frédéric Tuong.
Constructing Semantically Sound Object-Logics for UML/OCL Based Domain-Specific Languages.
Ph.D. thesis, University of Paris-Saclay, France, 2016.
https://tel.archives-ouvertes.fr/tel-01318156.

Frédéric Tuong and Burkhart Wolff.
A meta-model for the isabelle api.
Archive of Formal Proofs, 2015.
ISSN 2150-914x.
http://www.brucker.ch/bibliography/abstract/tuong.ea-meta-model-2015.