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
theory Bank_Model imports "../src/UML-OCL" begin

generate_syntax [ syntax_print, shallow, deep (THEORY Model) in Haskell ]

Class Savings << Account Attributes max : Currency
Association clients Between Bank [1 .. *] Role banks
Association clients Client [1 .. *] Role clients

Context c : Savings
Inv "0 ∈rat (c.max)"
Inv "c.balance ≤real (c.max) and 0 ≤real (c.balance)"

Context Bank :: create (classname : String, age : Integer)
Pre "(self.clients) ->forall c | c.classname = classname"
Post "(self.clients) ->exists c | c.classname = classname"

/* in shallow-mode: the generated content was executed */

thm down_cast_kindAccount_from_OclAny_to_Savings

apply(auto simp : isdef down_cast_type_Savings_from_OclAny_to_done
lemma down_cast_kindAccount_from_OclAny_to_Savings :
assumes iskin : ""(X::OclAny).oclIsKindOf(Account)"
assumes isdef : ""(\delta(X)"
shows ""(X::OclType(Savings)) ∈ invalid"
apply(insert not_OclIsKindOfAccount_then_OclAny_OclIsTypeOf apply(rule down_cast_type_OclAny_from_OclAny_to_Savings, simp apply(drule not_OclIsKindOfClient_then_OclAny_OclIsTypeOf apply(rule down_cast_type_Bank_from_OclAny_to_Savings, simp apply(drule not_OclIsKindOfClient_then_OclAny_OclIsTypeOf apply(rule down_cast_type_client_from_OclAny_to_Savings, simp done

(* 93 ****************************************************** 1484 + 1 *)

subsection <Const>

(* 147 ****************************************************** 1849 + 2 *)
definition "(typecheck_state_bad_head_on_lhs_\sigma') (\sigma')" = ()
definition "(typecheck_state_extra_variables_on_rhs_\sigma')" = ()

(* 148 ****************************************************** 1851 + 3 *)
generate_syntax [ shallow ]
setup (Generation_mode.update_compiler_config (KV (let open State[shallow] \sigma') = [ Account, Client, Bank1, Savings1 ]

[ 9 of 10 ] Compiling Argument ( Argument.hs, _build
[ 10 of 10 ] Compiling Main ( Main.hs, _build/Main

Linking Main ...

Achim D. Brucker, Frédéric Tuong, and Burkhart Wolff

A Formal Methods Environment for OCL: HOL-OCL 2.0

October 2, 2016
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.
ISSN 2150-914x.

Achim D. Brucker, Frédéric Tuong, and Burkhart Wolff.
Featherweight ocl: A proposal for a machine-checked formal semantics for ocl 2.5.

Delphine Longuet, Frédéric Tuong, and Burkhart Wolff.
Towards a tool for featherweight ocl: A case study on semantic reflection.

Frédéric Tuong.
Constructing Semantically Sound Object-Logics for UML/OCL Based Domain-Specific Languages.

Frédéric Tuong and Burkhart Wolff.
A meta-model for the isabelle api.
ISSN 2150-914x.