OCL and its relationship to meta-modeling

Presented by

Victoria Yang

Overview

- Intro to OCL
- Evolution of UML & OCL
- OCL Metamodel and its relationship to UML Metamodel
- Work Currently done on OCL Meta-modeling
- Future Work

A Brief Intro of OCL

- Why OCL?
  - Describe constraints
  - An unambiguous formal language
  - Suitable for both business use and persons with strong math background
  - A pure specification language
  - Has no side effect

A Brief Intro to OCL (Cont)

- Characteristic of OCL 2.0
  - Both query and constraint language
  - Mathematical foundation, but no mathematical symbols
  - Strongly typed language
  - Declarative Language

Where to use OCL

- As a query language
- To specify invariants on classes and types in class model
- To specify type invariant for stereotypes
- To describe pre and post conditions on operations and methods
- To describe guards
- To specify target (sets) for message and actions
- To specify constraints on operations
- To specify derivation rules for attributes for any expression over a UML model

From 1.1 to 2.0

- Syntax Changes
- New types
- Extra predefined operations
- New options in post conditions
- Other changes

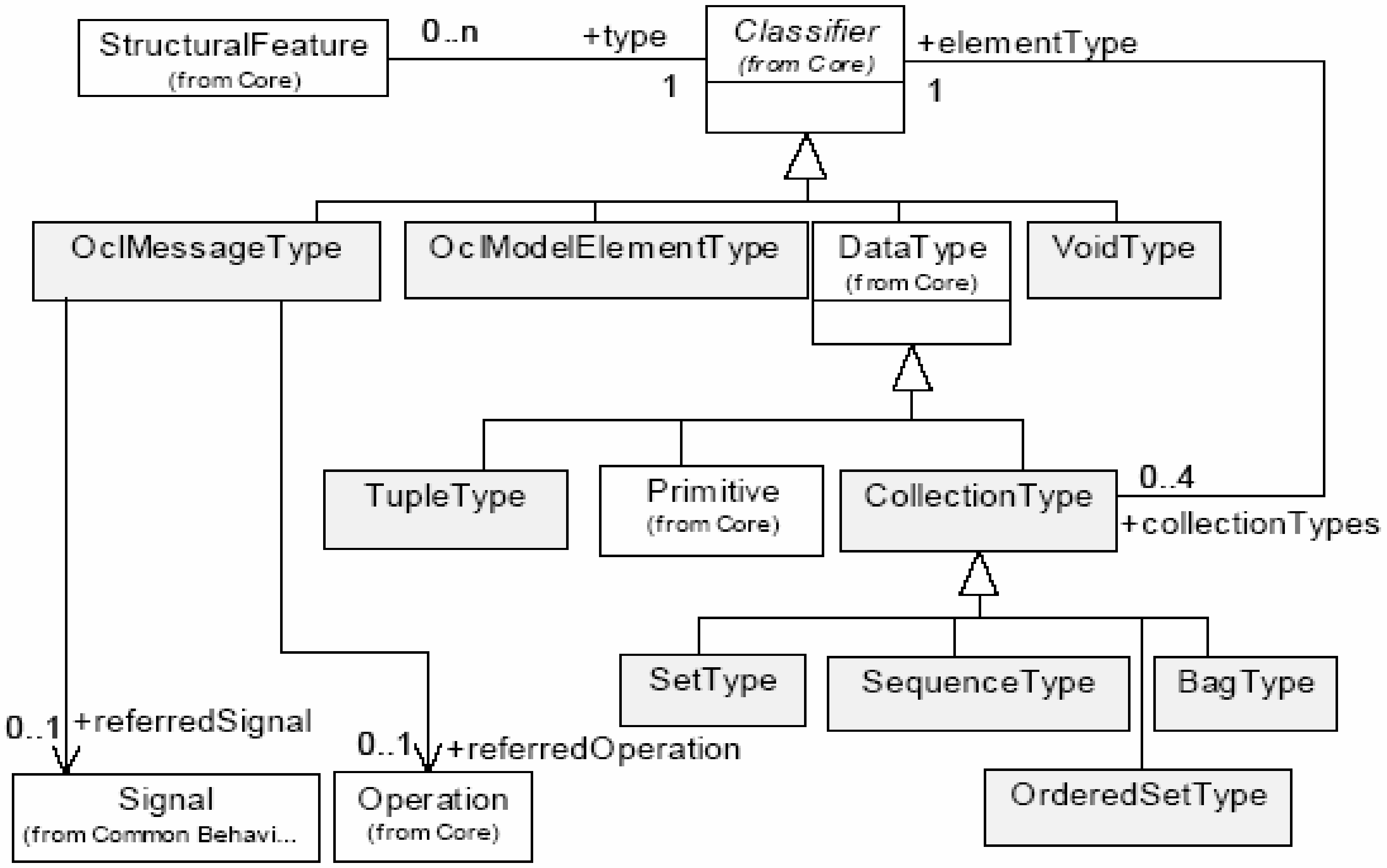
Abstract Syntax VS Concrete Syntax

- *Concrete Syntax*: part of language definition.
- *Abstract Syntax*: presentation used for encoding concrete syntax

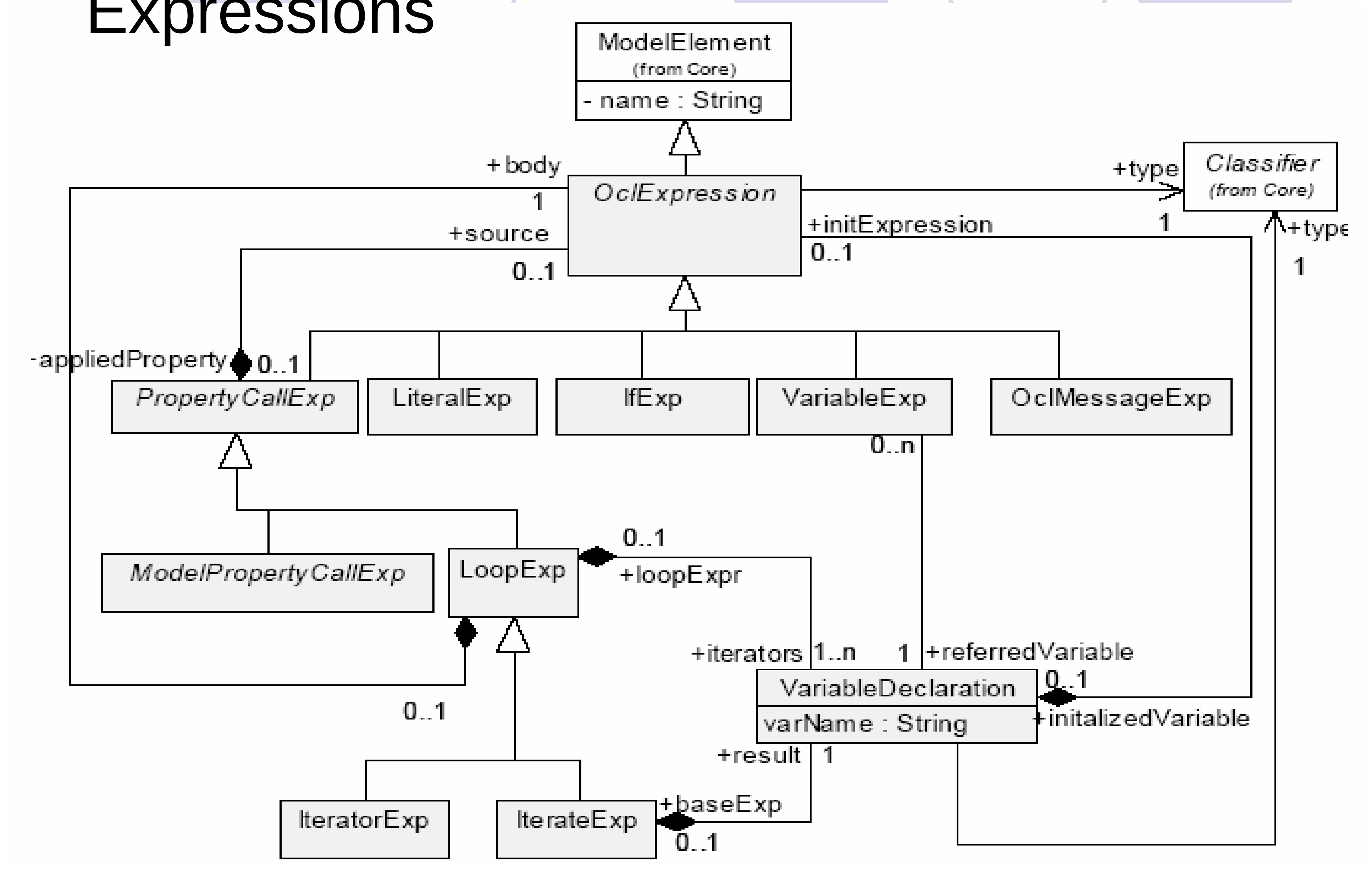
Abstract & Concrete Syntax in OCL

- Abstract Syntax
  - The *Type* package
  - The *Expression* package

Abstract Syntax Metamodel for OCL Types



Abstract Syntax Metamodel for Expressions



Abstract & Concrete Syntax in OCL (Cont)

- Concrete Syntax
  - Synthesized attributes, *ast*
  - Inherited attributes, *env*
  - Multiple production rules
  - Multiple occurrences of production names
  - Disambiguating rules

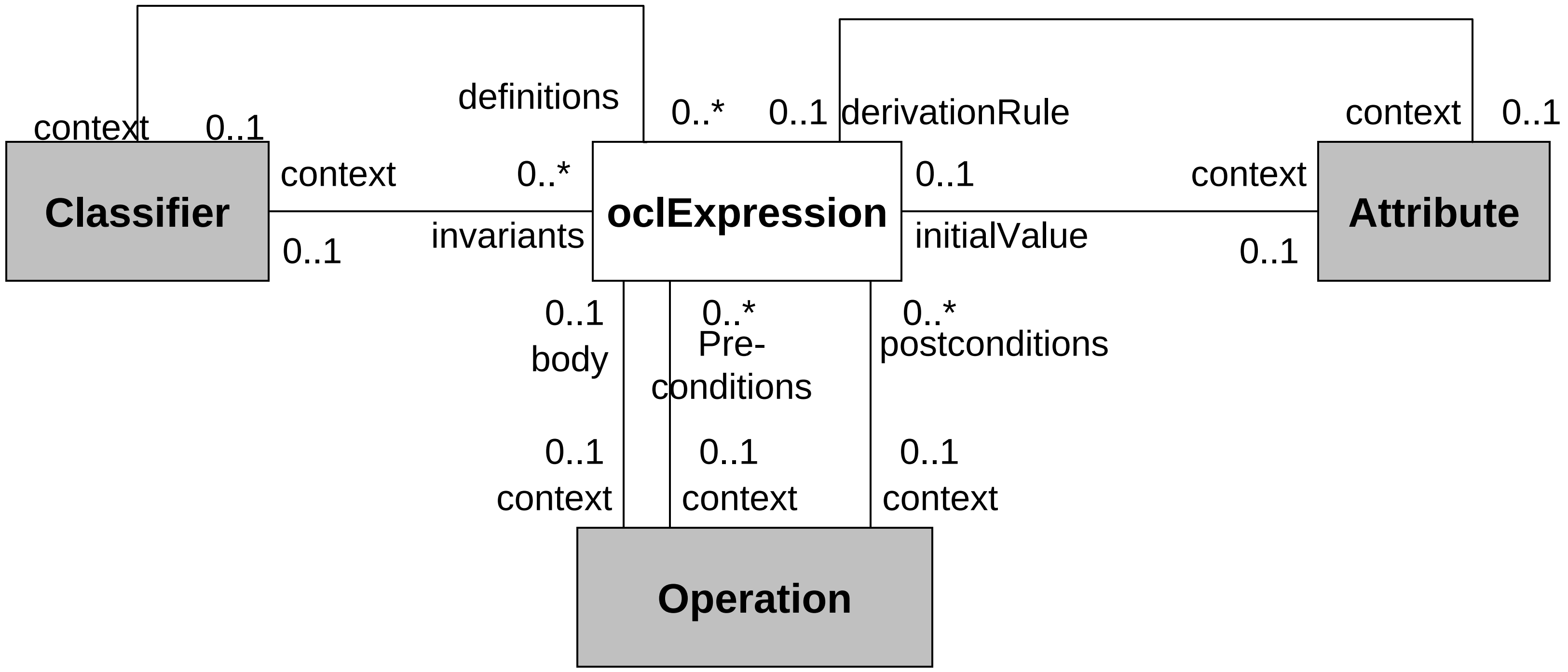
OCL Abstract & Concrete Syntax Mapping

- Concrete to Abstract Syntax Mapping
  - Adding a synthesized attribute
- Abstract Syntax to Concrete Syntax Mapping
  - Applying production rules from left to right

OCL Metamodel & UML Metamodel

- The UML metamodel
  - modelElement & Classifier
- The OCL metamodel
- The relationship between the UML and OCL metamodels
  - OCL expression reference model elements
  - UML elements adorned with infromation from OCL expressions

OCL Metamodel & UML Metamodel (Cont)



OCL context in terms of the motamodels

From UML metamodel

Implementing OCL

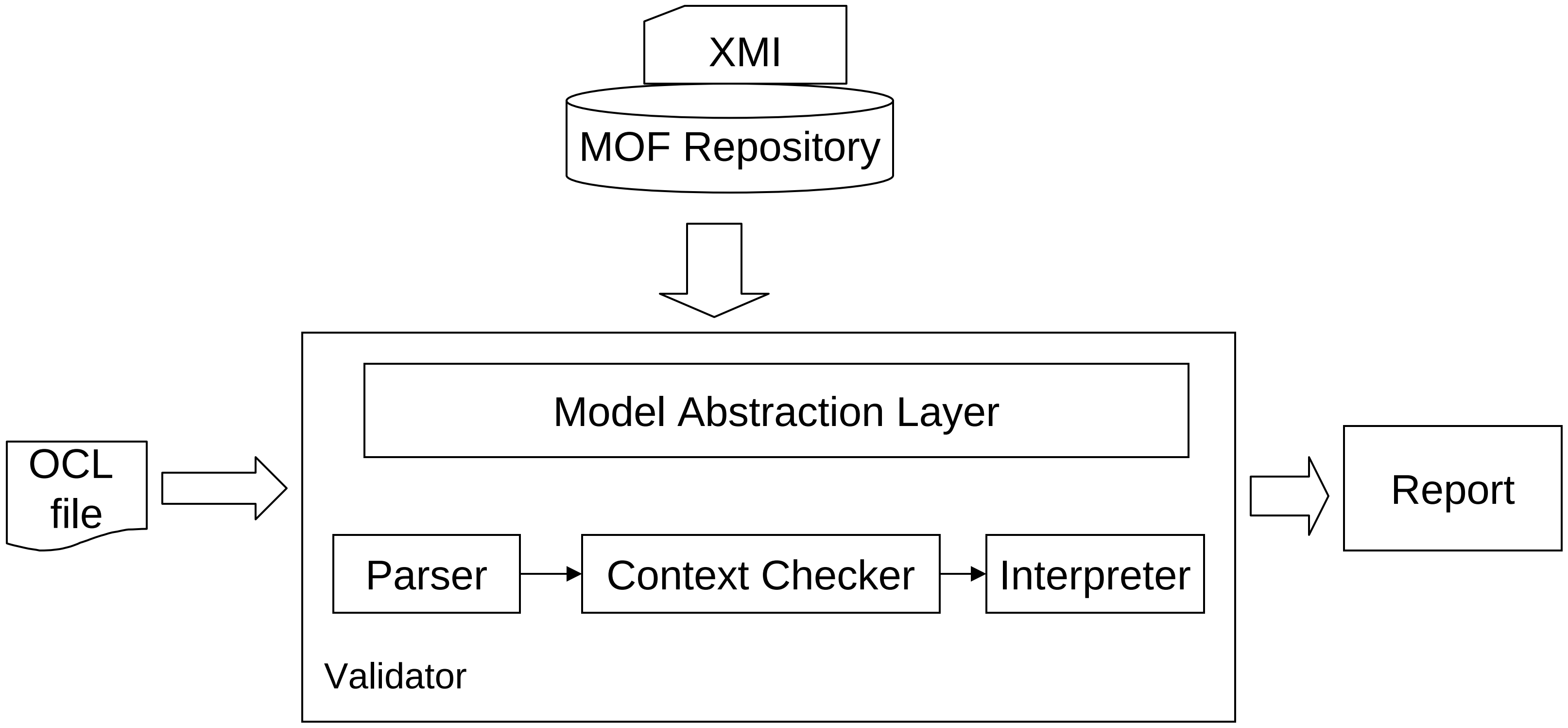
- UML model elements
- OCL standard library
- OCL expressions
- Merge code fragments
- Check invariants, pre and post conditions and perform action when check fails

Work Done on OCL to Meta-modeling

- Fadi Chanbarek’s development of an OCL-parser
  - Based on OCL 1.4
  - Defines on the M3 level (meta-metamodel level)

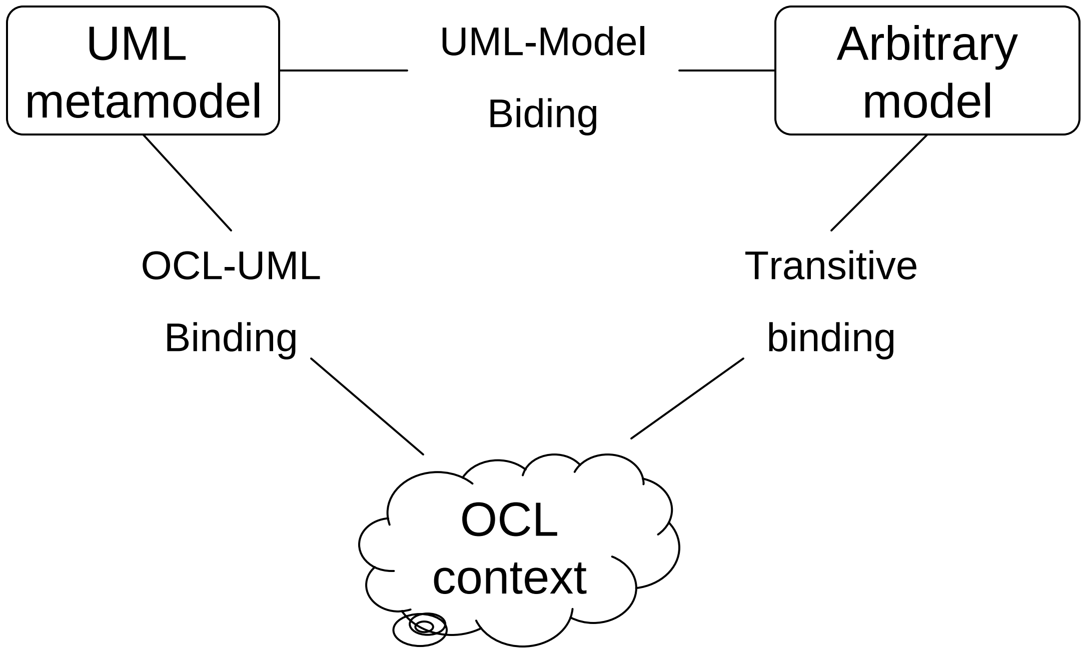
Fadi Chanbarek’s OCL Parser

- Architecture



Fadi Chanbarek’s OCL Parser (Cont)

- Interface



Conclusion & Future Work

- Conclusion

- Future Work
  - Implementation of OCL-parser based on UML 2.0 OCL
  - Integration of OCL-parser to multi-paradigm modeling tools

References

- Jos Warmer, Anneke Kleppe: [*The Object Constraint Language, Getting Your Models Ready for MDA, 2nd Edition*],Addison Wesley, 2003

- Fadi Chabarek: [*Development of an OCL-Parser for UML-Extensions*], Technical University of Berlin, 2004

- Object Management Group, Inc.: [*UML 2.0 OCL Specification*], 2004