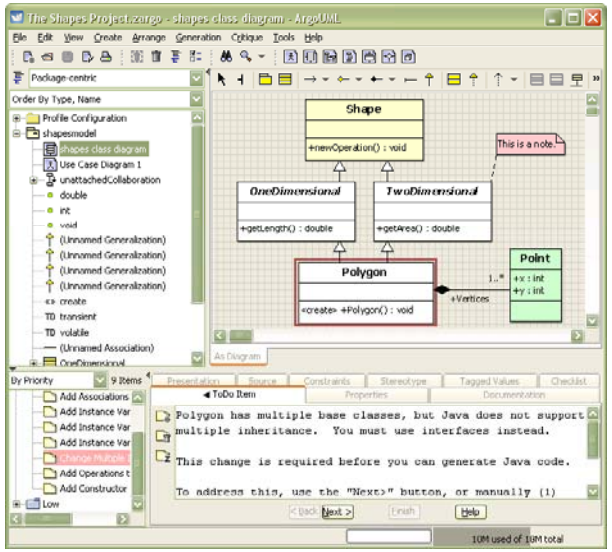
UML/OCL Verification in Practice

Jordi Cabot [1+2], Robert Clarisó [1]
(1) Universitat Oberta de Catalunya
(2) University of Toronto

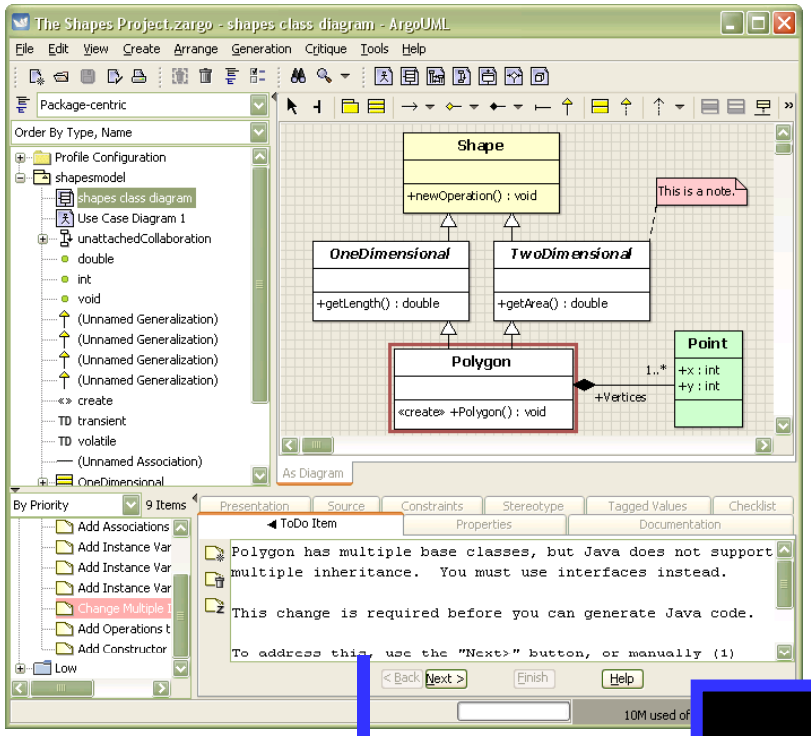
Users experience

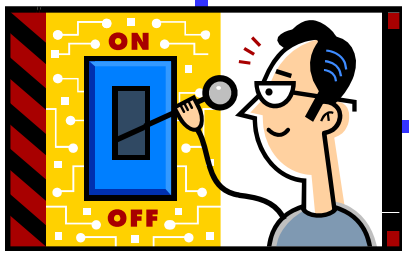


CASE tool

Verification tools

Users wishlist





- Inside CASE tool
- Input = model as is
- No interaction needed
- Meaningful results

```
Class Polygon (creation):
  Cannot instantiate class
  Check multiplicity of Vertices

Class Shape (invariant minArea):
  Invariant is unsatisfiable
```

The challenge

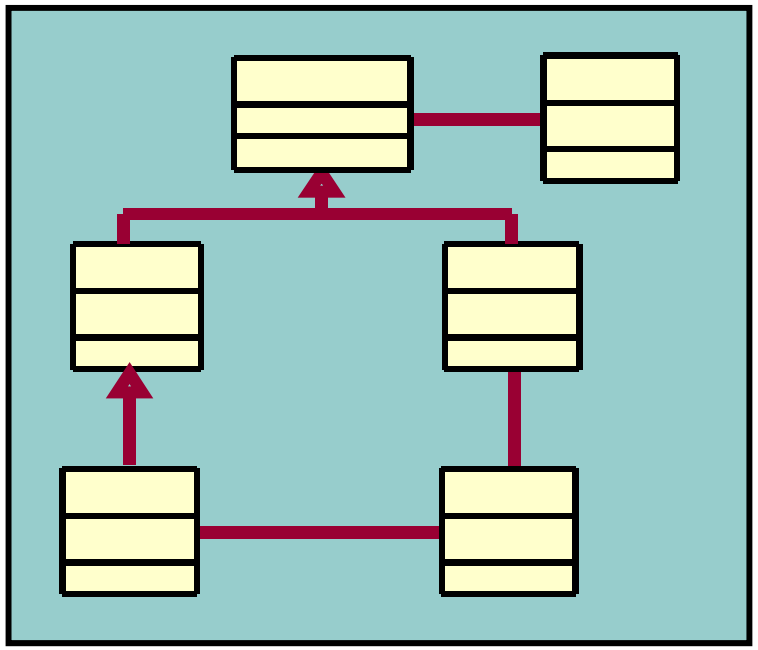
Verification without…

- Theoretical background
- User interaction

Verification with…

- Efficiency
- Meaningful results

UML/OCL class diagrams



Graphical constraints:

Multiplicities of associations

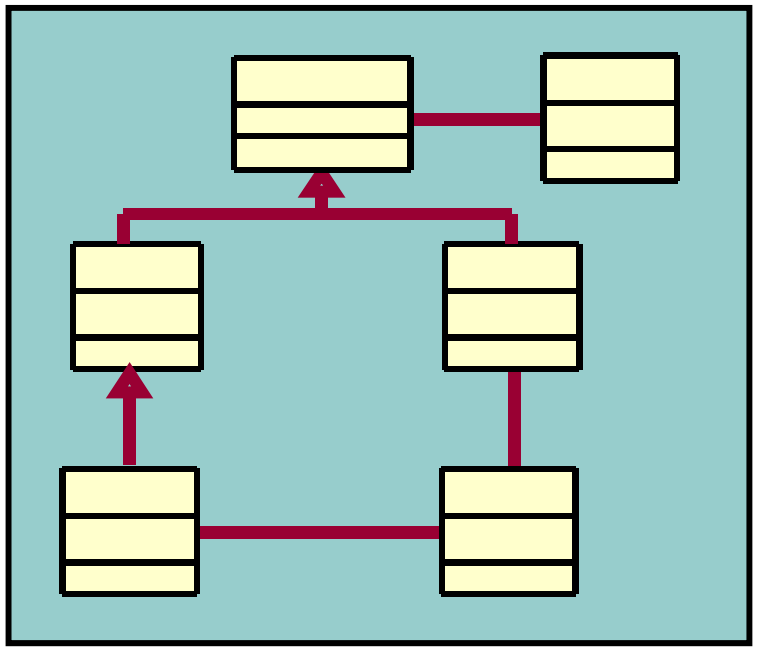
Inheritance hierarchies

context Person **inv**:
self.age ≥ 0

Textual constraints:

Invariants, pre/post

Model satisfiability



context Person **inv**:
self.age ≥ 0

“Create an instance which satisfies all the contraints”

State-of-the-art

Validation tools

- Visual animators — USE, MOVA, …
- Constraint evaluators — Dresden OCL, OCLE, …

Verification tools

- Model checkers — Alloy, SPIN, …
- Theorem provers — HOL-OCL, RACER, KeY, …
- SAT/LP/CP solvers — Chaff, CPLEX, UMLtoCSP, …

Even more in demos & tutorials at MODELS

Researcher POV

Efficiency	Scalability
Expressiveness	Modeling notation vs formalism
Integration	Many case tools Model interchange formats
Usability	Diagnostic information?

User POV

Efficiency	Select most appropiate method Abstract & partition input model
Expressiveness	Translate to formal notation Adapt unsupported constructs
Integration	Import/export model
Usability	Understand output and fixes

Where to go

- Community benchmarks
  - Objective measure of progress
  - Incentive for improvement
  - Identify best approach for each problem
  - Interoperability improvements!

- Examples: SAT
  Circuit layout

Research directions

Efficiency	Choice of most appropiate method Model abstraction and partition Incremental verification
Expressiveness	Model normalization
Usability	Feedback improvements

Long term target

Verification in MDE is taken for granted

e.g. a type checker in a compiler

a syntax checker in a text editor