Bidirectional model transformations with Echo

Nuno Macedo Tiago Guimarães Alcino Cunha

HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY
INESCTEC

Universidade do Minho

FATBIT Workshop 2013
October 3, Braga, Portugal

Introduction

- In MDE engineering models are the primary development artifact;
- Models must conform to their meta-models...
- ...and also coexist in a consistent manner;
- OMG has proposed standards for the specification of models (MOF) and constraints over them (OCL);
- The *QVT* standard has been proposed to specify model transformations and consistency.

Query/View/Transformation

- The QVT standard proposes three different languages;
- We focus on *QVT Relations* (QVT-R);
- Declarative language where the specification denotes the consistency relation between models;
- Two running modes should be derived:
  - *checkonly* mode (checks consistency);
  - *enforce* mode (updates are propagated in one direction in order to restore consistency).

Bidirectional Transformations

- Since updates can be propagated to either model, QVT-R transformations entail *bidirectional transformations* (BX);
- These need to be inferred from a single specification;
- Models may contain different information and are *not bijective*;
- The exact edit-sequence of the update is unknown;
- Information from the *original* target model must be retrieved.

QVT-R Checking semantics

- *Checking*: for every candidate in the source there must exist a candidate in the target that matches it;
- The standard is omissive about what should happen in circular recursion;
- We chose *not* to allow circular recursion;
- However, we can resort to the *transitive closure* (which has recently been added to the OCL standard);
- We were able to rewrite the classic recursive QVT-R examples to use the transitive closure.

QVT-R Enforcement semantics

- The standard is ambiguous and incomplete;
- Strong syntactic restrictions are required to guarantee determinism:
  - Writing BX with the expected behavior becomes difficult (not even the example from the standard is bidirectional!);
  - Deterministic but unpredictable;
- Disregards the OCL constraints of the meta-model;
- Instead, we follow the clear and predictable *principle of least change*.

Formalization

- For every transformation T between M and N we have:
  - a relation $\mathbf{T} \subseteq M \times N$ that checks the consistency;
  - transformations $\overrightarrow{\mathbf{T}} : M \times N \to N$ and $\overleftarrow{\mathbf{T}} : M \times N \to M$ that propagate updates;
- For every meta-model M, $\Delta_M : M \times M \to \mathbb{N}$ calculates the distance between two models.

Formalization

- Correctness:

$$\forall m \in M, n \in N : m\mathbf{T}(\overrightarrow{\mathbf{T}}(m,n))$$
$$\forall m \in M, n \in N : (\overleftarrow{\mathbf{T}}(m,n))\mathbf{T}n$$

- Hippocraticness (check-before-enforce):

$$\forall m \in M, n \in N : m\mathbf{T}n \Rightarrow m = \overrightarrow{\mathbf{T}}(m,n) \wedge n = \overleftarrow{\mathbf{T}}(m,n)$$

Formalization

- Correctness:

$$\forall m \in M, n \in N : m\mathbf{T}(\overrightarrow{\mathbf{T}}(m, n))$$
$$\forall m \in M, n \in N : (\overleftarrow{\mathbf{T}}(m, n))\mathbf{T}n$$

- Hippocraticness (check-before-enforce):

$$\forall m \in M, n \in N : m\mathbf{T}n \Rightarrow m = \overrightarrow{\mathbf{T}}(m, n) \wedge n = \overleftarrow{\mathbf{T}}(m, n)$$

- Principle of least change ($\Rightarrow$ hippocraticness for $\Delta = 0$):

$$\forall m \in M, n, n' \in N : m\mathbf{T}n' \Rightarrow \Delta_N(\overrightarrow{\mathbf{T}}(m, n), n) \leq \Delta_N(n', n)$$
$$\forall m, m' \in M, n \in N : m'\mathbf{T}n \Rightarrow \Delta_M(\overleftarrow{\mathbf{T}}(m, n), m) \leq \Delta_M(m', m)$$

Model distance

- Graph edit distance:
  - counts insertions and deletions of nodes and edges;
  - meta-model independent metric;
  - automatically inferred;
- Operation-based distance:
  - sequence of operations required to obtain the model;
  - user specified metric (operations as OCL predicates);
  - allows finer control over updates;
- Both suitable for different contexts.

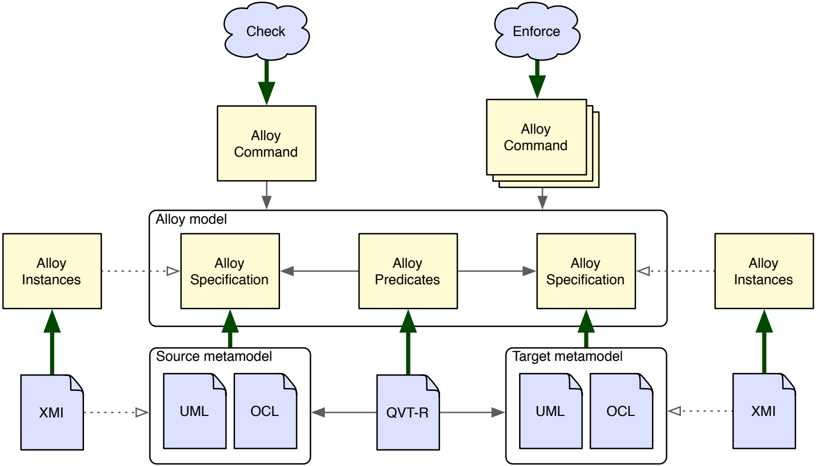
Echo

- *Echo* is a tool for model repair and transformation built over the *Alloy* SAT solver;
- Deployed as an Eclipse plugin over EMF;
- Support for:
  - Model visualization;
  - Model generation;
  - Consistency check;
  - Model repair;
  - *Inter-model consistency check*;
  - *Inter-model consistency repair*;
  - Inter-model generation.

Alloy

- *Alloy* is a lightweight model-checking tool based on relational calculus;
- Allows automatic bounded verification of properties and generation of instances via SAT solving;
- We have already developed a tool for the transformation of MOF+OCL meta-models to Alloy;
- We propose the translation of QVT-R to Alloy on top of that;
- Least-change attained by asking for increasingly distant solutions.

QVT-R to Alloy Translation

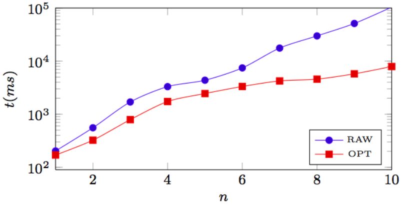


Demo

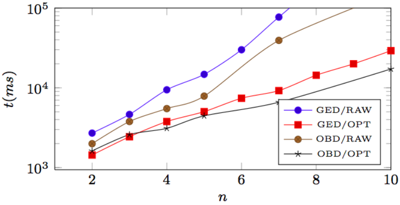
Demo

Evaluation: performance

- Solver-based approaches' main caveat is performance;
- Through Alloy simplifications, we were able to greatly improve performance:
    - Rewriting system aiming at removing quantifications from formulas;



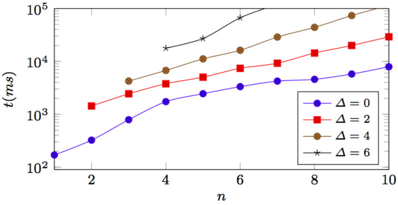
(a) Checkonly.



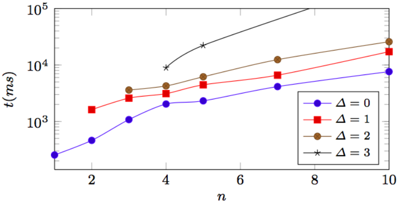
(b) Enforcement ($d = 1$) for GED and OBD.

Evaluation: performance

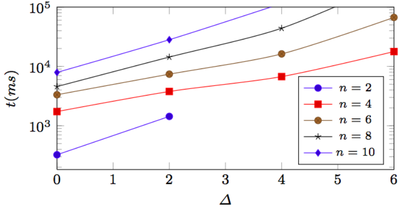
- Is our tool effective in practice?
- How low is good enough? What are the typical model sizes?



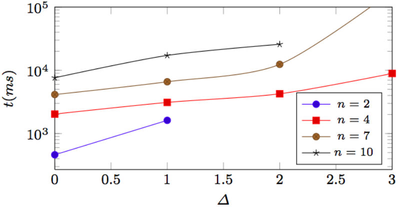
(a) GED.



(b) OBD.



(a) GED.



(b) OBD.

Evaluation: examples

- Tool support for inter-model consistency repair is scarce;
- Need for a repository of standard MDE examples;
- Compare existing techniques:
  - Echo;
  - Lenses (Focal);
  - TGGs;
  - GRoundTram;
  - BX as model repair.
  - ...

Evaluation: language

- Is QVT-R a suitable language?
- Recursion is unpredictable:
  - UML2RDBMS, HSM2NHSM...
- The forall-there-exists test in all directions is too inflexible:
  - UML class diagrams to UML sequence diagrams...
- Structured specifications is desirable:
  - Compare with specifying first-order logic constraints;
- Maybe a compromise could be attained:
  - Mark the domain patterns with multiplicities?

Echo

Available at

http://haslab.github.io/echo