A Relational Approach to Bidirectional Transformation

Nuno Macedo

October 29 2014, Braga, Portugal
PhD Thesis Defense
MAPi Doctoral Programme

Universidade do Minho

universidade
de aveiro

U.PORTO

Model-driven Engineering

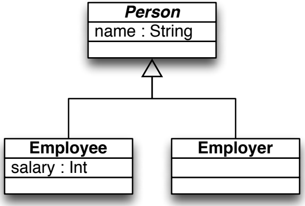
- Model-driven engineering (MDE) focuses on *models* as the primary development artifacts;
- Dynamic environment where coexisting models evolve simultaneously;
- Must be kept consistent with *meta-models* and with *each other*.

Bidirectional Transformation

- Updates on a model must be propagated to others in order to restore consistency;
- Maintaining individual transformations is troublesome and error-prone;
- *Bidirectional transformation* (BX):
  - Single specification entails both transformations.

BX: Object-relational Mapping

- Update propagation:

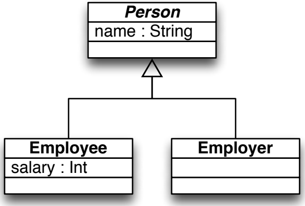


```
CREATE TABLE Employee (
  _Id     int ,
  name    varchar (255),
  salary  int ,
  PRIMARY KEY (_Id) );
CREATE TABLE Employer (
  _Id     int ,
  name    varchar (255),

  PRIMARY KEY (_Id) );
```

BX: Object-relational Mapping

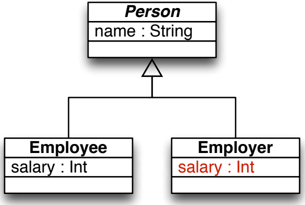
- Update propagation:



```
CREATE TABLE Employee (
   _Id      int ,
   name     varchar (255),
   salary   int ,
   PRIMARY KEY ( _Id) );
CREATE TABLE Employer (
   _Id      int ,
   name     varchar (255),
   salary   int ,
   PRIMARY KEY ( _Id) );
```

BX: Object-relational Mapping

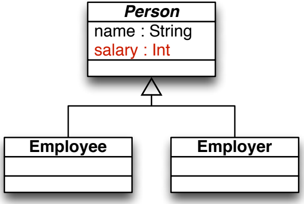
- Update propagation:



```
CREATE TABLE Employee (
  _Id     int ,
  name    varchar (255),
  salary  int ,
  PRIMARY KEY (_Id) );
CREATE TABLE Employer (
  _Id     int ,
  name    varchar (255),
  salary  int ,
  PRIMARY KEY (_Id) );
```

BX: Object-relational Mapping

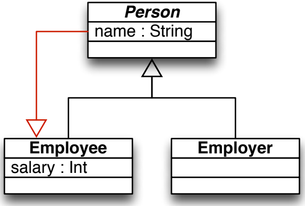
- Update propagation:



```
CREATE TABLE Employee (
   _Id     int ,
   name    varchar (255),
   salary  int ,
  PRIMARY KEY (_Id) );
CREATE TABLE Employer (
   _Id     int ,
   name    varchar (255),
   salary  int ,
  PRIMARY KEY (_Id) );
```

Constrained Datatypes

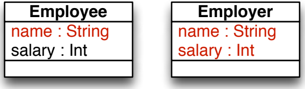
- To be useful, the BX must consider meta-model constraints:



```
CREATE TABLE Employee (
  _Id    int ,
  name   varchar (255),
  salary int ,
  PRIMARY KEY (_Id) );
CREATE TABLE Employer (
  _Id    int ,
  name   varchar (255),
  salary int ,
  PRIMARY KEY (_Id) );
```

Least-change Updates

- To be predictable, the BX should be least-change:



Employee
name : String
salary : Int

Employer
name : String
salary : Int

```
CREATE TABLE Employee (
   _Id     int ,
   name    varchar (255),
   salary  int ,
   PRIMARY KEY (_Id) );
CREATE TABLE Employer (
   _Id     int ,
   name    varchar (255),
   salary  int ,
   PRIMARY KEY (_Id) );
```

Goal

- Two popular BX frameworks:
  - **Lenses**: from a transformation, derive a *putback*;
  - **Constraint maintainers**: from a consistency relation, derive *both* transformations;
- Address the problem of **least-change** BXs over **constrained datatypes** in the context of *lens* and *constraint maintainer* frameworks.

Relational Logic

- First-order logic with relational operations and extended with transitive closure;
- Suitable to handle MDE problems;
- *Partial* and *multi-valued* transformations are natural concepts;
- Is the unifying formalism behind the thesis.

Invariant-constrained Lenses

- Problem solved when invariants are exact matches:
  - What if they are not?
- Solutions has two steps:
  - Calculate the restricted transformation domains;
  - Derive constraint-aware putbacks;
- Round-tripping laws defined modulo invariants.

N. Macedo, H. Pacheco and A. Cunha.
Relations as executable specifications: taming partiality and non-determinism using invariants.
In *RAMiCS 2012*. Springer, 2012.

Invariant-constrained Lenses

- Proposed framework is generic but impractical;
- Instantiated by defining controlled invariant languages and associated operations;
- E.g., *spreadsheet formulas*;
- Invariants built over data validation features of spreadsheets.

N. Macedo, H. Pacheco, A. Cunha and N. R. Sousa.
Bidirectional spreadsheet formulas.
In *VL/HCC 2014*. IEEE, 2014.

Least-change Lenses

- In general, least-change updates are not preserved by *composition*:
  - When are they?
- Dual formalizations:
  - return at most a minimal update;
  - return at least all minimal updates;
- Set of criteria under which composition is least-change.

N. Macedo, H. Pacheco, A. Cunha and J. N. Oliveira.
Composing least-change lenses.
In *BX 2013*. EASST, 2013.

Constraint Maintainers as Model Finding

- Deriving update procedures from consistency relations is complex
  - Even more so if they are to be constraint-aware and least-change;
- In a sense, it amounts to finding models that conform to certain constraints:
  - Can constraint maintainers be deployed over relational model finders?
- A problem consists of a ***constraint*** (the consistency relation) and ***bounds*** (which model is modified).

QVT-R

- QVT is OMG's standard for model transformation;
- QVT-R has bidirectional concerns, but problematic semantics:
  - Would our approach be feasible?
- Our embedding provides a correct and least-change bidirectional semantics.

N. Macedo and A. Cunha.
Implementing QVT-R bidirectional model transformations using Alloy.
In *FASE 2013*. Springer, 2013.

ATL

- One of the most popular model transformation languages;
- Inherently *unidirectional*:
  - How can constraint maintainers be derived?
- Bidirectional embedding allows maintaining consistency after batch transformation.

N. Macedo and A. Cunha.
Least-change bidirectional model transformation with QVT-R and ATL.
Software and System Modeling. Springer, 2014.

Beyond Bidirectional Transformation

- Formalization as model finding is easily extended:
  - Can it be generalized to other application scenarios?
- E.g., *multi-directional transformation*:
  - Bounds define which set of models are updated.

N. Macedo, A. Cunha and H. Pacheco.
Towards a framework for multi-directional model transformations.
In *BX 2014*. CEUR-WS, 2014.

Echo

- The constraint maintainer formalization was deployed as the **Echo** tool;
- Seamless integration:
  - *Eclipse* plug-in
  - Standard MDE file formats (Ecore, XML, OCL, QVT-R, ATL);
- Built over the *Alloy* model finder.

📄 N. Macedo, T. Guimarães and A. Cunha.
Model repair and transformation with Echo.
In *ASE 2013*. IEEE, 2013.

Summary

- Lens framework:
  - Invariant-constrained lenses (+ spreadsheet instantiation);
  - Least-change lenses;
- Constraint maintainer framework:
  - Invariant-constrained least-change constraint maintainers;
  - Language embeddings;
  - Deployment: `http://haslab.io.github/echo`.

Future Work

- Flexible metrics;

  N. Macedo, A. Cunha and T. Guimarães.
  Exploring scenario exploration.
  Submitted. 2014

- Finder performance;

  A. Cunha, N. Macedo and T. Guimarães.
  Target-oriented relational model finding.
  In *FASE 2014*. Springer, 2013.

- Transformation validation.