Bidirectional Transformations - Exercises

Harald König
FHDW Hannover / HVL Bergen
harald.koenig@fhdw.de

March 17, 2021

Exercises marked with ($*$) are a bit harder and require a little extra contemplation. Exercises marked with ($**$) are harder and are optional.

Exercise 1: Below, you find a list of transformations. For each case you will have to decide in what direction you can define a get and in which direction you require a put. Maybe it is also sometimes possible to define a get in both directions or the lens may even be symmetric? You should also propose an implementation for get and put.

a) Pairs of numbers (Int,Int) $\leftrightarrow$ their product (Int)

b) Person entities with firstName and lastName $\leftrightarrow$ Persons with name

c) Sets (unordered, unique) $\leftrightarrow$ lists (ordered, non-unique)

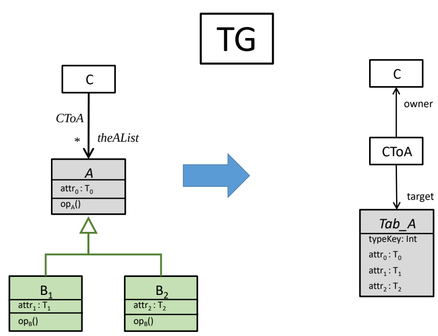

Figure 1: Object Relational Mapping

Exercise 2: We assume class and data models to be structured according to the meta-models from slide 32 and we consider the following object-relational mapping from class models to data models (TG, see Figure 1):

- Each association with target a class is transformed into a (linking) table with two foreign key columns, e.g. table CToA in Figure 1.
- Each class inheritance hierarchy is mapped to exactly one table with an extra column, in which an identifier for the concrete class is stored.
- Associations with target a PrimType (aka attributes) are stored as usual columns.
- Of course operations are not mapped to the data model.

Primary keys are not shown in the figure and foreign keys are depicted as arrows between the tables.

In order to keep both sides in sync after a change of a class model or a data model, would you propose to use a symmetric or an asymmetric lens? How do you implement the $\overrightarrow{R}/\overleftarrow{R}$, the put/get, resp.? If implementations of some restorers are not unique, you must only give one reasonable proposal.

Exercise 3: About the composer example:

a) Justify correctness and hippocraticness for the implementations of $\overrightarrow{R}$ and $\overleftarrow{R}$ on Slide 40.

b) Let (A_1, A_2) be a consistent pair and $A_1 \to A_1'$ be an update. Describe the requirement given by the following equation:

$$\overrightarrow{R}(A_1, \overrightarrow{R}(A_1', A_2)) = A_2$$

and check whether it is satisfied in the composer example.

Exercise 4: A mathematical example: We consider the following asymmetric lens: $\mathbb{S}$ contains pairs (X, Y) of arbitrary sets X and Y. $\mathbb{V}$ just contains all sets. Furthermore let $\mathsf{get}(X, Y) := X \cap Y$ and

$$\mathsf{put}((X, Y), Z') := (Z' \cup (X - Y), Z' \cup Y)$$

a) Why is information asymmetrically distributed?

b) Is the lens hippocratic and correct?

c) $(*)$ Is it history ignorant?

Exercise 5: The special setting $\mathbb{S} \cong \mathbb{V} \times C$ for some set C with

$$\mathsf{get}(v, c) = v \text{ and } \mathsf{put}((v, c), v') = (v', c) \tag{1}$$

for an asymmetric lens was dubbed "the constant complement setting". We pointed out that such a lens is *history ignorant* (the put-put-law).

a) Is a constant complement decomposition of $\mathfrak{M}_1$ into $\mathfrak{M}_2$ and C possible for the composer example on slide 52? (If yes, by the above statement, there would be a very simple implementation of put!)

b) Let in the constant complement setting $\mathbb{V}$ be the collection of all class models, elements of $\mathbb{S}$ the resp. generated code (e.g. by EMF), which has regions (elements in C) that can manually be enhanced and are protected during re-generation. Describe the practical action of get, put in this case, especially why manual enhancements are really protected. What means history ignorance in this case? Why is the put still problematic w.r.t. syntax errors?

c) (∗∗) History ignorance is a nice feature, because one needs to synchronize only once in the end after several updates, saving a lot of performance. The question is, for how many situations this feature holds. By the above statement it holds at least for the lenses with constant complement property.

   Show that – unfortunately – no other asymmetric lens L as the ones, for which the constant complement property holds, are history ignorant, i.e. the converse of the above statement holds:

$$L \text{ is history ignorant } \Rightarrow L \text{ must have the form (1)}$$

   Hint: For $v \in \mathbb{V}$ define $\mathbb{S}_v = \{s \in \mathbb{S} \mid get(s) = v\}$ and show that for any $v_1, v_2 \in \mathbb{V}$ the function $\mathsf{put}(s, v_2) : \mathbb{S}_{v_1} \to \mathbb{S}_{v_2}$ is bijective by the put-put-law. From this, you can determine C: Look at the modulo arithmetic example and find the bijection $\mathbb{S} \cong \mathbb{V} \times C$.