TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Software- und Multimediatechnik, Lehrstuhl für Softwaretechnologie

Winter term 2015/2016 - Model-Driven Software Development in Technical Spaces

Exercise 3: Model Transformations

The purpose of this exercise is to understand how to realize a model to text transformation. For this, two approaches shall be realized. The first approach represents template-based code generation. The second is using a model-to-model transformation and EMFText.

Task 1: Template-based Code Generation

- Use the integrated DSL for class diagrams and state charts from the last exercise.
- Download and understand the tool Acceleo (http://www.eclipse.org/acceleo/)
- Write Acceleo templates, which generate valid Java code for programs written using the integrated DSL. Hint: use the state pattern to translate state charts into code.

Task 2: Model-to-Model Transformations and JaMoPP

- Again, use the integrated DSL from the last exercise.
- Download and understand the tool ATL (http://www.eclipse.org/atl/).
- Download and understand the EMFText Language JaMoPP (http://www.jamopp.org/).
- Write an ATL transformation from your integrated DSL to JaMoPP. Hint: again, the state pattern can be useful.

Example

Listing 1: Example DSL Instance for Class Model+Statechart

```
class Door {
  boolean isOpen;

  void open();
  void close();
  void lock();
  void unlock();

  statechart {
     state open;
     state closed;
     state locked;
     transition open (close() [isOpen] / isOpen=false) closed
     transition closed (open() [!isOpen] / isOpen=true) open
     transition closed (lock() [] / ) locked
     transition locked (unlock() [] / ) closed
  }
}
```

Listing 2: Exemplary Java Code Generated from Listing 1

```
abstract class Door {
  private boolean isOpen;

  public abstract Door open();
  public abstract Door close();
  public abstract Door lock();
  public abstract Door unlock();

  public Door(boolean open) { isOpen=open; }

  public void setOpen(boolean open) { isOpen=open; }
  public boolean isOpen() { return isOpen; }
}

class OpenDoor extends Door {
  //...
}
//...
```