

Generic Language Technology (2008-2009)

Code Generation

Exercise 7 (deadline: December 19th 2008)

Introduction

In this exercise you will create a grammar which enables creating simple class models. You will also create a transformation that generates Java code from such a class model.

Setup

1. Read the entire exercise!
2. Create a new sub-directory where you wish to store your exercise files, e.g. `exercise7`.
3. Change to the just created sub-directory.
4. Start the ASF+SDF Meta-Environment with the command `asfsdf-meta`.

A Grammar for Creating Simple Class Models

In Figure 1 a metamodel for a simple class model is depicted. Classes in this class model have an identifier (its name), zero or more attributes, and zero or more operations. Every attribute has a modifier to indicate its visibility, a type, and an identifier. Every operation also has a modifier, a type, and an identifier. In addition, an operator can have zero or more parameters, which have a type and a name. The modifiers that can be used are `private`, `protected` and `public`. The types and identifiers are the same as available in the Java language.

The simple class model allows inheritance. A class can inherit methods and attributes from other classes using inheritance. This is expressed by the “is parent of” relation.

The goal of this part of the exercise is to create an SDF grammar that allows specification of class models adhering to the metamodel of Figure 1.

- Create a new module named `ClassModel.sdf`.
- Create a grammar that allows specification of class models adhering to the metamodel of Figure 1.
- Hint: choose the syntax for your types and names carefully.
- Hint: think of a clever way to represent inheritance.

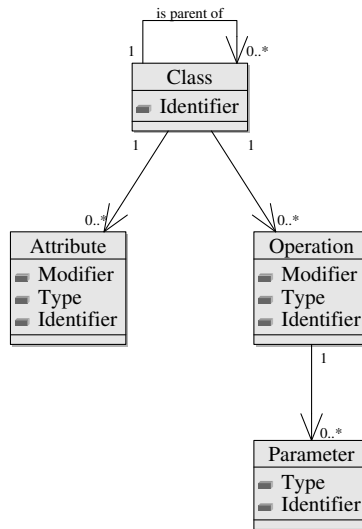


Figure 1: Metamodel of a simple class model

Skeleton Code Generation

The goal of this part of the exercise is to transform class models specified in the grammar you just created into Java classes. Use the Java syntax from the standard SDF grammar library as target for your transformation.

- Create a new module named `ClassModel2Java.sdf`.
- Import the Java syntax from the standard SDF grammar library.
- Create a transformation that transforms class models specified in the grammar you created for the first part of the exercise into Java classes.

Submission

Submit the following, well-documented files via PEACH.

- `ClassModel.sdf`,
- `ClassModel2Java.sdf`,
- `ClassModel2Java.asf`, and
- a test term.