

Eiffel library to generate Java bytecodes

DIPLOMA PROJECT DESCRIPTION

Project period: 2003-05-26 Monday – 2003-09-25 Thursday

Student name: Daniel Gisel

Email address: daniel@gisel.ch

Supervising Assistant: Karine Arnout

Supervising Professor: Bertrand Meyer

Overview

One goal of the Java language environment is making “write once, run anywhere” possible. To achieve this, the Java compiler translates Java programs into an intermediate language called Java bytecodes – the platform-independent codes executed by the Java Virtual Machine (JVM) [6]. This means that as long as a computer has a JVM, a program in Java bytecodes can run on it, independent of the operating system or the hardware of the computer.

Even if the JVM and the Java bytecodes are designed and optimized for the Java programming language, the JVM does not assume that the instructions it executes were generated from Java code. It is also possible to compile other programming languages to Java bytecodes.

The Java bytecodes of one class is always stored in class files (name.class). This file contains all information about the class, like the access rights, references to the parent class and all the interfaces the class implements, tables with all the fields and the methods (including their code) of the class and the constant pool. The constant pool is a table of structures representing various string constants, class and interface names, field names and other constants that are referred to within the class file structure and its substructures.

Scope of the work

The main work of my diploma project is to design and implement an Eiffel library which is able to generate Java class files that can be executed by a Java Virtual Machine. To achieve this goal, I will do the following steps.

First I have to create an object oriented data structure that maps the structure of a Java class file. This data structure will contain classes that represent a Java class, a method, a field, the constant pool, and so on. These classes will be fully contracted to catch as many errors as possible in the code generation to avoid invalid class files.

Based on this data structure I will build a layer that allows an easy Java class file creation for the client. In this layer there will be generators for classes, methods and fields.

To test this library and also to give examples how to use the framework, I will create a simple language with a corresponding compiler that uses the library as backend. The language will be similar to the Java language, but much simpler.

Intended results

Library:

The created library will be fully contracted and produce correct Java class files. It will be built up of two layers. The first layer is very close to the structure of the class files and allows direct access to the elements. The second layer implements generators for classes and methods and allows an easy creation of class files.

Compiler:

I will design a simple language similar to Java that is compiled to Java bytecodes. The backend of the compiler will be the library created in the first part of my project. The goal of this compiler is to test the library and to show how to use it.

Documentation:

The documentation will contain three documents. The first one is the report of the whole project. The other two documents are a user and a developer manual for the library.

REFERENCES

- [1] Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman: *Compilers: Principles, Techniques and Tools*, Addison-Wesley, 1986.
- [2] Eric Bezault: *Gobo Eiffel Project*, Online at <http://www.gobosoft.com>, consulted in April 2003.
- [3] *Byte Code Engineering Library (BCEL)*, Online at: <http://jakarta.apache.org/bcel/>, consulted in May 2003.
- [4] Chair of Software Engineering: *Semester-/Diplomarbeiten*, Online at: <http://se.inf.ethz.ch/projects/index.html>, consulted in March 2003.
- [5] James Gosling, Bill Joy, Guy Steele, Gilad Bracha: *The Java Language Specification, 2nd edition*, Addison-Wesley, 2000, <http://java.sun.com/docs/books/jls/>.
- [6] Tim Lindholm, Frank Yellin: *The Java Virtual Machine Specification, 2nd edition*, Addison-Wesley, 1999, <http://java.sun.com/docs/books/vmspec/>.
- [7] Bertrand Meyer: *Object-Oriented Software Construction, 2nd edition*, Prentice Hall, 1997.
- [8] Seven Muchnick, *Advanced Compiler Design and Implementation*, Morgan Kaufmann Publishers, 1997.
- [9] Java 2 Platform, available from <http://java.sun.com>.