Synopsis

Morten Brøns-Pedersen Jesper Reenberg

September 10, 2009

1 Title

"Programmer tools based on static semantic analysis of SML programs, with special emphasis on semi-automatic rewriting of declarations to predefined forms."

2 Motivation

Today functional programming languages are not very widespread outside academic circles. A reason for this is that not many programmer assisting tools exist for functional programming languages compared to certain imperative languages.

We wish to make functional programming, specifically in SML, easier to approach for the beginner as well as more advantageous for the veteran. The means by which we hope to accomplish this task is the development of a framework for implementing programmer assisting tools, and one or more actual tools.

Functional programming languages has a wide variety of advantages that make them specially suited for quick and correct statical analysis. The effect of this is that it will be possible to make more complex tools for functional programming languages than for imperative ones.

3 Elaboration

The project can be divided into three parts.

Front end The front end is responsible for communicating with the programmer. This communication goes both ways. Some tools will work automatically waiting for the programmer to accept corrections and/or suggestions. Some will be initiated by the user.

Back end The back end will perform tasks common to most tools. This includes communicating with the front end in a suitable format, reading and possibly writing project files, reading and parsing source files,

defining a representation of syntax trees, keeping an internal representation of the source code up to date and defining auxiliary functions (e.g. functions for converting syntax trees to source code, converting data to a format suitable for communication with the front end, etc.).

Tools The tools will be initiated by the back end from which they will get a representation of the source code as a syntax tree or the source code itself. Messages to the individual tools from the front end will be delivered via the back end. The tools can send messages back to the front end through the back end. In this project we will focus on a single tool. Se section 4.1 below.

The back end and the tools are split up because most tools will have a lot of tasks in common as described above.

So while the back end and the tools are actually the same piece of software, the distinction is an important one.

The importance of this distinctions implies that we develop a well defined API for communication between the back end and the tools.

4 Primary goals

4.1 A refactoring tool

bleh

5 Secondary goals

6 Limitation

7 Tasks and timeline