Software Component Technology Group

Semester Project

Slicing Spec# programs

Sebastian Grössl
Supervisors: Prof. Peter Müller, Joseph N. Ruskiewicz

26. September 2007

Introduction

Spec# is a new programming system for specification and verification of object-oriented software. The Spec# language is a superset of the programming language C# extending C# by non-null types, method contracts, object invariants and an ownership type system. The behavior of a Spec# program is checked at runtime and statically verified by Boogie, the Spec# static program verifier. This works by transforming Spec# code into an intermediate language called BoogiePL, on which program analysis and verification can be performed.

There exists an implementation [1] of a program slicer for BoogiePL which helps isolate code that led to a verification error. But because this slicer only operates on BoogiePL programs, it cannot directly be used to output Spec# slices.

Goal of this semester project

The goal is to be able to slice Spec# programs extracting code that led to a verification error. Since Spec# programmers shouldn't be required to know anything about BoogiePL it wouldn't make sense to present them the sliced BoogiePL code. Instead, code in their original Spec# program should be shown.

We will use the Spec# compiler that translates the program into BoogiePL, on which the statical verifier Boogie can then be run. If verification errors are found, the code will be sliced, so that only relevant portions of the BoogiePL code remain. In a final step, this slice will be transformed back into the original Spec# code. The main labor of this thesis therefore is to make the transformation of Spec# to BoogiePL reversible.

As an extension, this functionality should be integrated in the Spec# Visual Studio plug-in.

References

[1] Karin Freiermuth. Using program slicing to improve error reporting in Boogie. Master's thesis, Swiss Federal Institute of Technology Zurich (ETHZ), Department of Computer Science, 2007.