Master Thesis
Formalization and implementation of translation from Java Bytecode to Guarded Commands

Alex Suzuki

April 14, 2006

Introduction Boogie is a modular program verifier for verifying Spec# programs in the Microsoft .NET framework. The Spec# compiler generates CIL (Common Intermediate Language) bytecode from Spec# source code, a superset of C#, and transforms the resulting CIL to a guarded command language called BoogiePL. Boogie creates verification conditions from BoogiePL, and passes them to an automatic theorem prover, currently Simplify.

While Boogie is part of the Spec# Programming System, there is no reason why it should not be leveraged to verify programs written in other object-oriented languages, namely Java, as long as we supply an appropriate transformation to BoogiePL. Also Spec# currently does not generate BoogiePL code for exception handling, so errors in exception handlers are currently not detectable by Boogie.

The goal of this master's project is to develop a formalization in Coq of both the semantics of the BoogiePL language and a translation from Java Bytecode to guarded commands, including support for exceptions. The *Bicolano* project, which already provides a formalization of Java Bytecode semantics, will be used as a starting point. As soon as the formalization is present, a mapping of the concepts to CIL and subsequent implementation of exception support in Boogie is desirable.

The main parts of this project are:

1. A formalization of a translation from Java Bytecode to guarded commands, including exceptions, in Coq.
2. A formalization of the semantics of the BoogiePL programming language in Coq.

Possible extensions include the implementation of the translation of exception handling in Spec# to BoogiePL for exceptions thrown by methods and the `throw` statement, and optionally support for runtime exceptions.

Supervised by: Prof. Peter Müller and Hermann Lehner