Generating Efficient Code from Logic*

Natarajan Shankar

Computer Science Laboratory
SRI International
Menlo Park CA 94025 USA
{shankar, owre}@csl.sri.com
URL: http://www.csl.sri.com/~shankar/
Phone: +1 (650) 859-5272

Abstract. We explore the seemingly far-fetched thesis that program development and maintenance are best carried out within a logic rather than a programming language. In such a framework, programs are specified, verified, analyzed, refined, transformed, and modified as definitions within a logic. This logic must be rich enough to capture common computational idioms and software engineering principles, while containing useful executable sublanguages. Code in conventional programming languages can be generated from the executable fragments of the logic. The approach of developing programs within a logic offers several advantages over writing coding directly within a programming language:

1. Logics are usually semantically simpler than conventional programming languages, and can therefore support more effective forms of static analysis.
2. The correspondence between a program and its proof can be maintained even as the program changes.
3. Executable specifications can be used as runtime-checkable assertions.
4. Semantics-based optimizations can be carried out within the logic, and machine-specific optimizations can be left to the compilers for the target programming language.

We illustrate the thesis with the PVS logic and its accompanying proof checker and code generator. We list some of the open problems in achieving ultra-efficient execution from logic.

* This work was funded by NSF Grant CCR-0082560, DARPA/AFRL Contract F33615-00-C-3043, and NASA Contract NAS1-00079.