



The Mobius Program Verification Environment

Joe Kiniry, University College Dublin

<http://mobius.inria.fr/>



Enabling proof-carrying code for Java on mobile devices

Contract n° IST 015905

This work was funded in part by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2005-015905 MOBIUS Project.

This presentation reflects only the author's views the Community is not liable for any use that may be made of the information contained therein





- leverage existing software foundations as much as is possible and reasonable
 - primarily Eclipse, the JML tool suite, Jack, and ESC/Java2
- integrate tools developed by others
 - mainly other static checkers and rigorous software engineering subsystems
- leverage the ESC/Java2 user-base
 - large set of industrial users, academic researchers/users, and student users that are potential Mobius PVE target users



- Java program code features
 - writing new code
 - type-aware completion
 - compiling
 - debugging
 - refactoring
 - folding code
 - generate Javadoc documentation
 - *analyze code complexity*
 - *analyze coding standard conformance*
 - *detecting common programming errors*



- Java Modeling Language features
 - writing new specifications
 - compiling specifications to runtime tests
 - generate Javadoc documentation
 - *context-aware specification folding*
- Bytecode Modeling Language features
 - *compile JML to BML*
 - *display BML-annotated Java VM bytecode*
 - *edit BML*



- JML-annotated programs features
 - unit test generation
 - specification generation
 - class and *loop invariant generation*
 - translation to guarded commands
 - existing ESC/Java GC and *BoogiePL*
- theorem prover features
 - *use interactive provers in a natural way*
 - *integrate proving and programming in UI*
 - *support several automatic provers*
 - *user- and tool-customization for prover use*



- Java, JML, and *BML* lexer, parser, type checker, and transformation subsystem
 - *generates, visualizes, and manipulates Java VM bytecode, JML annotations, Javadocs, BML-annotated bytecode, and DOT files*
- FreeBoogie subsystem
 - *FreeBoogiePL—FLOSS BoogiePL*
 - *FreeBoogie VC generation*
 - *targets Mobius VC back-end, thus*
 - *will support multiple theorem provers*



- Mobius VC back-end
 - unsorted and *sorted VC representations*
 - *logic-aware syntax generation to several automatic and interactive theorem provers*
 - *e.g., generation of Mobius VCs in Coq, PVS, Simplify, SMT, etc.*
- Mobius ESC VC generator
 - *generation of ESC VCs in several ESC logics*
 - *extended static checking of ESC VCs with rich in-editor feedback*



- Mobius Prover back-end
 - *generic interaction with a variety of automatic and interactive theorem provers*
 - *automatic provers supported*
 - *Simplify, SMT, Fx7, (CVC3, Yices)*
 - *interactive provers supported*
 - *Coq and (PVS)*
- integration of several support tools
 - e.g., CheckStyle, FindBugs, PMD, etc.
 - *the Race Condition Checker (RCC)*



- full support available for:
 - all Java and nearly all JML features
 - editing, compilation, doc generation, etc.
 - code complexity and style checking
 - partial BML support
 - no full compilation of JML to BML as of yet
 - Mobius VC back-end
 - advanced ESC VC generation
 - Mobius Prover back-end
 - interactive proof support for Coq



- several other groups are using PVE subsystems for their own research
 - prover back-end and VC representation
 - Fx7 improvements
 - ESC experimentation (KSU, MIT, others)
- ...and teaching
 - UCD using PVE subsystems for undergraduate instruction in programming and software engineering
 - groups using static checkers for instruction include Univ. of Wash., CMU, MIT, others

