BML: Specification and Verification at the Bytecode Level

Aleksy Schubert
Institute of Informatics
Warsaw University
ul. Banacha 2
02–097 Warsaw
Poland

October 23, 2008

Outline

MOBIUS

BML

BML related tools

Work in progress

MOBIUS – Mobility, Ubiquity and Security

- European Integrated Project in global computing

MOBIUS – Mobility, Ubiquity and Security

- European Integrated Project in global computing
- Security guarantees using *proof-carrying code*

MOBIUS – Mobility, Ubiquity and Security

- European Integrated Project in global computing
- Security guarantees using *proof-carrying code*
- Focus on Java MIDP platform

MOBIUS – Mobility, Ubiquity and Security

- European Integrated Project in global computing
- Security guarantees using *proof-carrying code*
- Focus on Java MIDP platform
- Techniques: static analysis, types, program verification

BML – Bytecode Modeling Language

- Bytecode specification language

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)
- Main features:

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)
- Main features:
  - similar to JML (Java Modeling Language)

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)
- Main features:
  - similar to JML (Java Modeling Language)
  - based on design-by-contract principles

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)
- Main features:
  - similar to JML (Java Modeling Language)
  - based on design-by-contract principles
  - covers (JML0):

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)
- Main features:
  - similar to JML (Java Modeling Language)
  - based on design-by-contract principles
  - covers (JML0):
    - invariants (static & instance), history constraints, simple form of represents clauses

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)
- Main features:
  - similar to JML (Java Modeling Language)
  - based on design-by-contract principles
  - covers (JML0):
    - invariants (static & instance), history constraints, simple form of represents clauses
    - pre- and post- conditions (with exceptions), modifies clauses

BML – Bytecode Modeling Language

- Bytecode specification language
- Proposed by: Lilian Burdy, Marieke Huisman, and Mariela Pavlova (FASE'07)
- Main features:
  - similar to JML (Java Modeling Language)
  - based on design-by-contract principles
  - covers (JML0):
    - invariants (static & instance), history constraints, simple form of represents clauses
    - pre- and post- conditions (with exceptions), modifies clauses
    - asserts, assumes, loop invariants, decreases clauses, loop modifies clauses

BML – Bytecode Modeling Language

- Additional features:
  - access to local variables and operand stack
  - representation in class format

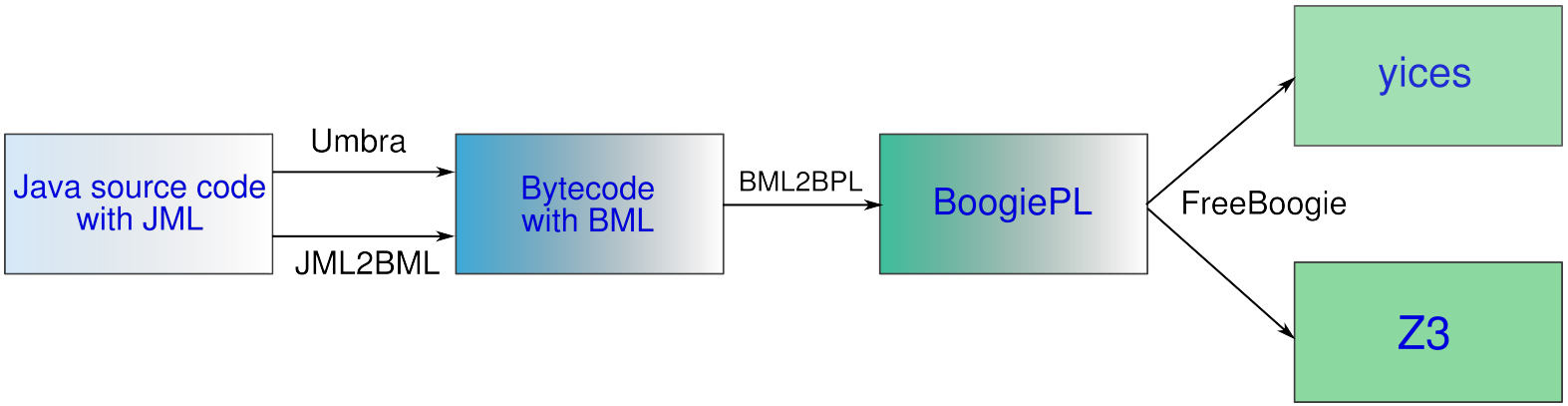
BML Reference Manual

- people involved: Jacek Chrząszcz, Marieke Huisman, Aleksy Schubert,
  and Joe Kiniry, Erik Poll, Mariela Pavlova
- covers:
  - definition of the textual format
  - definition of the bytecode format
  - definition of a translation from JML to BML
- work in progress (80% ready)
- web page: `http://bml.mimuw.edu.pl/`
  also available from `http://www.jmlspecs.org`

Tools and formalisms

- BML – specification language (FASE'07)
- JACK – Java Card verification environment (FMCO'06)
- Umbra – specification editor (Bytecode'08)
- BMLLib – library to parse and store BML specifications (Bytecode'08)
- JML2BML – compiler of JML to BML (CEE-SET'08)
- BML to BoogiePL – translator (Bytecode'07)
- FreeBoogie – translator to FOL provers

Tools and formalisms



JACK

Java Card verification environment (FMCO'06)

- preliminary work on BML
- people involved: Gilles Barthe, Lilian Burdy, Julien Charles, Benjamin Grégoire, Marieke Huisman, Jean-Louis Lanet, Mariela Pavlova, and Antoine Requet
- features:
  - storing BML in class files
  - editing BML specifications
  - generation of proof obligations
- web page:

  `http://www-sop.inria.fr/everest/soft/Jack/jack.html`

Umbra

Bytecode and BML specification language editor (Bytecode'08)

- current main platform of BML-related tools
- people involved: Jacek Chrząszcz, Tomasz Batkiewicz, Wojciech Wąs, Aleksy Schubert,
- features:
  - one can transform an existing Java source code file (with JML) to bytecode (with BML),
  - one can view an existing class file,
  - one can add, delete, and edit bytecode mnemonics,
  - one can add, delete, and edit BML specifications,
  - one can validate a class using BoogiePL backend.
- web page: `http://www.mimuw.edu.pl/~alx/umbra/`

BMLLib

Library to parse and store BML specifications (Bytecode'08)

- the core representation of BML
- people involved: Jacek Chrząszcz, Tomasz Batkiewicz, and Aleksy Schubert
- features:
  - one can parse textual BML specifications
  - one can print out textual BML specifications
  - one can read BML specifications from class files
  - one can write BML specifications to class files
  - one can manipulate BML specifications programmatically
- based on BCEL bytecode library
- web page: `http://www.mimuw.edu.pl/~alx/umbra/`

JML2BML

Compiler of JML specifications into BML ones (CEE-SET'08)

- standalone compiler of JML specifications to BML specifications
- people involved: Jędrzej Fulara, Krzysztof Jakubczyk, Aleksy Schubert
- integrated with Umbra
- it takes Java source code with JML annotations + compiled class file
  and returns class file with BML attributes
- web page: http://www.mimuw.edu.pl/ alx/jml2bml/

BML to BoogiePL

A translator of BML to BoogiePL (MSc thesis in ETH Zürich)

BoogiePL

A formal intermediate language for automatic proving of program properties

- the formal basis of Spec#
- people involved: Robert DeLine, Rustan Leino
- typed procedural language
- few instructions: assume, assert, havoc, goto, assignment

BML to BoogiePL

A translator of BML to BoogiePL (Bytecode'2007)

- a tool which transforms BML annotated bytecode to BoogiePL
- people involved: Hermann Lehner, Peter Müller, Ovidio Mallo
- integrated with Umbra
- features:
  - reading class files with BML specifications
  - writing text files with BoogiePL result
  - based on ASM bytecode library

FreeBoogie

A translator of BoogiePL to format of FOL provers

- a tool which transforms BoogiePL to
- people involved: Radu Grigore, Joseph Kiniry
- partly integrated with Umbra
- features:
  - parsing BoogiePL files
  - typechecking of BoogiePL code
  - generation of FOL formulae (work in progress)

Related tools

- CCT — toolset to embed certificates into class files (Tadeusz Sznuk)

Tool presentation

Work in progress

- translation from BML to Coq
- translation of non-interference type system to BML
- case study

Thank you!