[illegible]

Lilian Burdy Mariela Pavlova

August 25, 2005

Abstract

We propose

proof for its safety w.r.t. to some safety property and the code receiver has just to generate the veri cation conditions and type check the proof against them. The proof is generated automatically by the certifying compiler for properties like well typedness or safe memory access. As the certifying compiler is designed to be

The remainder of the paper is organized as follows: Section 2 reviews scenarios in which the architecture is appropriate to use; Section 3 presents the bytecode speci cation language BCSL and the JML compiler; Section 4 discusses the main features of the weakest precondition calculus; Section 5 discusses the relationship between the veri cation conditions for


theorem prover

contain the BCSL speci cation, resulting from the compilation of the JML speci cation in the Java source le.

The produc64ro

```
public class ListArray {
  Object[] list;
  //@requires list != null;
  //@ensures \result ==  (\exists int i; 0 <= i && i <
```

instruction). this is di erent from JML where loop invariants are written at the beginning of the declaration of the loop statement, while the BCSL speci cation are separated from the bytecode

predicates from rst order logic

Line_Number_Table and Local_Variable_Table attributes. The presence in the Java class le

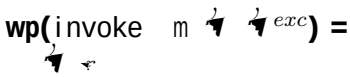
```
nresult = 1
 ( )
9
```

tion like preconditions, normal and exceptional postconditions, class invariants, assertions at particular program point among which loop invariants (if there is no explicite speci cation the default one is taken into account: preconditions, postconditions and invariants are taken to be true, exceptional postcondition is by default false) is taken into account. In the rest of the section, we consider these speci c features - treating instance elds, method invocations and loops.

The Java bytecode languago is stack based, i.e. the instructions take their arguments from the method execution stack and put the result on the stack. In Fig.

4.0.2 Method calls

Method calls are handled by using their speci cation. A method speci - cation is a contract between callers and callees | the precondition of the called method must be established by the caller at the program point where the method is invok



which the JACK source veri cation condition generator will discard.

Hypothesis on bytecode:	Hypothesis on source level:
lv[2]_a_r_20	i_a_

[4] L. Burdy and M. Pavlova. From JML to BCSL. Technical6