Available at
www.ElsevierMathematics.com
POWERED BY SCIENCE DIRECT®
The Journal of Logic and Algebraic Programming 58 (2004) 1–2

THE JOURNAL OF LOGIC AND ALGEBRAIC PROGRAMMING

www.elsevier.com/locate/jlap

Guest editors' introduction: Special issue on Formal Methods for Smart Cards

Marieke Huisman[a], Thomas Jensen[b]

[a] *INRIA Sophia-Antipolis 2004, route des Lucioles BP 93, 06902 Sophia-Antipolis, France*
[b] *IRISA/CNRS, Campus de Beaulieu, 35042 Rennes, France*

Over the last decade, the programming of smart cards has evolved significantly due to the advent of high-level programming languages tailored to this application domain. In particular, Java Card—a dialect of Java—has been designed with the limited resources of smart cards in mind, while retaining the standard Java architecture with a virtual machine.

Smart cards find some of their most important applications in sectors such as finance and health-care, where safety and security are important. Consequently, there is a high demand for methods and tools that can assist in the development and validation of smart card software. This has triggered a number of research projects on formal techniques (automated deduction, program analysis, model checking, etc.) for establishing whether a given smart card application satisfies certain desired properties.

The European projects VERIFICARD and SECSAFE are concerned with formal methods for analysing and proving correctness of programs, with smart card software as privileged application domain. These projects have held several VERISAFE workshops together. The current special issue results from the workshop held in Nice in September 2002. The articles in this special issue describe various results of the two projects.

The article by Siveroni presents an operational semantics of the Carmel language, a synthesis of the Java Card byte code language, developed in the SECSAFE project. This semantics provides a formal definition of card-specific language features such as transient data and the applet firewall.

Klein and Strecker have focused on the byte code verification that takes place before an applet is executed by the Java virtual machine. This verification is a fundamental part of the security architecture of Java Card. They model it using the theorem prover ISABELLE and in addition they show how to issue a type certificate that allows to verify the type correctness of the compiled code.

Two articles deal with tools and techniques for reasoning about source-code level Java Card programs, annotated using the JML specification language. Jacobs shows how to reason using weakest preconditions within the LOOP tool, relying on the theorem prover PVS. Marché et al. have developed the KRAKATOA tool for reasoning about JML-annotated applets, which works by translating code into an intermediate representation in the Coq proof assistant.

E-mail addresses: marieke.huisman@inria.fr (M. Huisman), jensen@irisa.fr (T. Jensen).

1567-8326/$ - see front matter
doi:10.1016/j.jlap.2003.07.002

The article by Jacobs et al. details a case study in applet verification using the LOOP tool. They study a Java Card electronic purse applet that uses a challenge-response mechanism. The verification relies on an encoding of a finite-state automaton in JML to represent particular aspects of the applet's behaviour.

Special thanks are due to the reviewers for their magnificent efforts. You have been most helpful for putting together this special issue!