6th Workshop on Virtual Machines and Intermediate Languages (VMIL'12)

Hridesh Rajan
Iowa State University, USA
hridesh@iastate.edu

Michael Haupt
Oracle Labs, Potsdam, Germany
michael.haupt@oracle.com

Christoph Bockisch
Universiteit Twente, The Netherlands
c.m.bockisch@cs.utwente.nl

Steve Blackburn
Australian National University, Australia
Steve.Blackburn@anu.edu.au

Abstract

The VMIL workshop is a forum for research in virtual machines and intermediate languages. It is dedicated to identifying programming mechanisms and constructs that are currently realized as code transformations or implemented in libraries but should rather be supported at VM level. Candidates for such mechanisms and constructs include modularity mechanisms (aspects, context-dependent layers), concurrency (threads and locking, actors, software transactional memory), transactions, etc. Topics of interest include the investigation of which such mechanisms are worthwhile candidates for integration with the run-time environment, how said mechanisms can be expressed at the intermediate language level, how their implementations can be optimized, and how virtual machine architectures might be shaped to facilitate such implementation efforts.

Categories and Subject Descriptors D.3.4 [*Programming Languages*]: Processors—runtime environments

Keywords virtual machines, intermediate languages

1. Motivations and Themes

An increasing number of high-level programming language implementations is realized using standard virtual machines. Recent examples of this trend include the Clojure (Lisp) and Potato (Squeak Smalltalk) projects, which are implemented on top of the Java Virtual Machine (JVM); and also F# (ML) and IronPython, which target the .NET CLR. Making diverse languages–possibly even adopting different paradigms–available on a robust and efficient common platform leverages language interoperability. Also new languages with many kinds of mechanisms and concepts researched in various communities are mostly reflected in high-level language and library design. AspectJ, Scala and JPred are examples of languages originally compiled for the JVM platform.

Vendors of standard virtual machine implementations have started to adopt extensions supporting this trend from the run-time environment side. For instance, the recent release of the Oracle standard JVM includes the *invokedynamic* instruction, which facilitates a simpler implementation of dynamic programming languages on the JVM.

The observation that many language constructs are supported in library code, or through code transformations leading to over-generalized results, has led to efforts to make the core mechanisms of certain programming paradigms available at the level of the virtual machine implementation. Thus, dedicated support for language constructs enables sophisticated optimization by direct access to the running system. The workshop's main goal is the discussion of compilation techniques, intermediate languages and execution environments that more naturally support such constructs even within compiled programs. This support will, e.g., facilitate new dynamic optimization, incremental compilation, and improve debugging. It can be expected that language constructs benefit from a more efficient execution and better integration into the development process: increased efficiency and improved integration will raise the acceptance of the concepts which in turn activates further research at the conceptual level.

The main themes of this workshop are to investigate which programming language mechanisms are worthwhile candidates for integration with the run-time environment, how said mechanisms can be declaratively (and re-usably)

SPLASH'12, October 19–26, 2012, Tucson, Arizona, USA.
ACM 978-1-4503-1563-0/12/10.

expressed at the intermediate language level (e.g., in bytecode), how their implementations can be optimized, and how virtual machine architectures might be shaped to facilitate such implementation efforts. Possible candidates for investigation include modularity mechanisms (aspects, context-dependent layers), concurrency (threads and locking, actors, software transactional memory), transactions, paradigm-specific abstractions, and combinations of paradigms.

The areas of interest include, but are not limited to, compilation-based and interpreter-based virtual machines as well as intermediate-language designs with better support for investigated language mechanisms, compilation techniques from high-level languages to enhanced intermediate languages as well as native machine code, optimization strategies for reduction of run-time overhead due to either compilation or interpretation, advanced caching and memory management schemes in support of the mechanisms, and additional virtual machine components for managing them.

2. Workshop Format

The planned workshop agenda interleaves presentations of accepted papers, invited talks from experts in the research area, and discussions to stimulate participants. The accepted workshop papers should act as motivation for new researchers to include the topics of this workshop into their research. To accomplish this, authors of accepted papers will be required to prepare a presentation, and all prospective workshop participants will be asked to read all accepted papers. For this purpose, all papers will be made available on the workshop web page.[1]

3. About the Organizers

Hridesh Rajan is an Associate Professor of Computer Science at the Iowa State University. He received his Ph.D. from the University of Virginia in 2005. He is the recipient of a 2009 US National Science Foundation CAREER award, a 2010 ISU LAS Early Achievement in Research award, and a 2012 Big-12 Fellowship. He was also co-organizer of the 2007-2011 edition of this workshop. His research on programming language and verification support for modular program design has been funded by the US National Science Foundation. He is a member of IEEE, IEEE Computer Society, ACM, SIGSOFT, and SIGPLAN.

Michael Haupt is a researcher and software developer in the Virtual Machine Research Group at Oracle Labs. His research interests are in improving the modularity of complex software system architectures as well as in implementing programming languages, in which latter area his main focus is on faithfully regarding programming paradigms' core mechanisms as primary subjects of language implementation effort. Michael holds a doctoral degree from Technische Universität Darmstadt, where he has worked on the Steamloom virtual machine to provide run-time support for AOP languages. He has published papers on this and other AOSD-related subjects in the L'Objet and IEEE Software journals as well as in the AOSD, VEE, OOPSLA, and ECOOP conference series. Michael has served as PC member for ECOOP 2008 and 2010, as reviewer for TAOSD, and has been supporting reviewer for the AOSD, ECOOP, ICSE, FSE, MODELS, and VEE conference series. He has co-organized the Dynamic Aspects Workshop series in conjunction with the AOSD conferences, and the previous three editions of the VMIL. Michael is a member of the ACM.

Christoph Bockisch is an assistant professor on Software Composition with a research focus on the design and implementation of programming languages with advanced dispatch mechanisms. He received his doctoral degree from the Technische Universität Darmstadt in 2008. To provide virtual machine support, Christoph researches extensions to high-performing Java virtual machines based on just-in time compilation. He furthermore researches meta-models for the definition of arbitrary dispatch mechanisms to act as a first-class representation. He is co-founder and co-organizer of the workshop series on Virtual Machines and Intermediate Languages (VMIL) and Free Composition (FREECO). He is Student-Events Co-Chair and PC member of the AOSD'13 conference.

Stephen M. Blackburn is a Professor at the Australian National University (ANU). He was Program Committee Chair for VMIL'11. He has served on numerous SIGPLAN program committees, including ECOOP'12 and ISMM'12. He is an organizer of the EVALUATE workshop series. His research interests include programming language implementation, architecture, and performance analysis. He received his PhD from the ANU. He is a Distinguished Scientist of the ACM.

4. Program Committee

- David Grove (chair), IBM Research
- Daniel Frampton, Microsoft
- Kawachiya Kiyikuni, IBM Research
- Chandra Krintz, UCSB
- Prasad Kulkarni, University of Kansas
- Christian Probst, Technical University of Denmark
- Ian Rogers, Google
- Jeremy Singer, University of Glasgow
- Witawas Srisa-an, University of Nebraska
- Christian Wimmer, Oracle Labs

Acknowledgments

Hridesh Rajan was supported in part by the US National Science Foundations under grants CCF-11-17937, CCF-10-17334, and CCF-08-46059.

[1] See `http://design.cs.iastate.edu/vmil/2012/`.