VMIL 2010

4th Workshop on Virtual Machines and Intermediate Languages

Hridesh Rajan[λ], Michael Haupt[φ], Christoph Bockisch[β], Robert Dyer[λ]

[λ]Iowa State University, [φ]Hasso Plattner Institute, University of Potsdam, and [β]Universiteit Twente

[λ]{hridesh,rdyer}@iastate.edu, [φ]michael.haupt@hpi.uni-potsdam.de, and [β]c.m.bockisch@cs.utwente.nl

Abstract

The VMIL workshop is a forum for research in virtual machines (VMs) 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 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 elegantly (and reusably) expressed at the intermediate language level (e.g., in bytecode), how their implementations can be optimized, and how VM architectures might be shaped to facilitate such implementation efforts.

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

General Terms Design, Languages, Performance

Keywords Virtual machine, intermediate language

1. Motivations and Themes

An increasing number of high-level programming language implementations is realized using standard virtual machines (VMs). 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.

Standard VM vendors have started to adopt extensions supporting this trend. For instance, the Oracle standard JVM will include the *INVOKEDYNAMIC* instruction, which will facilitate 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 VM implementation. Dedicated support for language constructs enables sophisticated optimization by direct access to the running system. This approach has been adopted by several projects aiming at providing support for aspect-oriented programming or dynamic dispatch in general-purpose VMs (Steamloom, Nu, ALIA4J).

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) expressed at the intermediate language level (e.g., in bytecode), how their implementations can be optimized, and how VM 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 VMs 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 VM components required to manage them.

2. Goals and Expected Results

We intend to solicit both technical and position papers. Our expectation is to receive contributions that, on the one hand,

SPLASH'10, October 17–21, 2010, Reno/Tahoe, Nevada, USA.
ACM 978-1-4503-0240-1/10/10.

point out mechanisms and concepts worth to be supported at the level of the execution environment; and, on the other, provide more detailed descriptions of implementation approaches for such mechanisms and concepts. These papers should motivate new researchers to include the workshop topics into their research. To accomplish this, we will make all papers available on the workshop web page; and we intend to publish the workshop proceedings—consisting of selected high-quality papers and extended abstracts of the remaining accepted papers—in the ACM digital library. The proceedings of the first three VMIL workshops have already been published in the ACM digital library. We also intend to open the workshop to researchers without accepted papers.

It is our intention to receive submissions from researchers new to the field as well as experienced researchers and practitioners. For the former, we want to offer a platform for discussing their ideas and receiving feedback on them. This will be supported by question and answer sessions as well as by a session of group discussions.

3. Organizers and Program Committee

Hridesh Rajan is an Assistant 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 and a 2010 Early Achievement in Research Award from Iowa State University. He has published in several conferences such as ECOOP, ESOP, ICSE, ESEC/FSE, ASE, AOSD, and IEEE Software. He has served on the program committees of Onward, AOSD, OOPSLA, NWeSP, ACP4IS, FOAL, and as a referee for top journals in his area such as IEEE Transactions on Software Engineering, ACM Transactions on Software Engineering and Methodology, and IEEE Software. He has also served as an external referee for several international conferences namely IEEE Infocom, ESEC/FSE, ICSE, OOPSLA, Formal Methods (FM), COMPSAC, etc. and program committee of the FOAL, VMIL, and ACP4IS workshops. He was also co-organizer and co-chair of the 2007-09 editions of this workshop. He is a member of IEEE, IEEE Computer Society, ACM, SIGSOFT, and SIGPLAN.

Michael Haupt is a post-doctoral researcher in the Software Architecture Group at Hasso-Plattner-Institut in Potsdam. His research interests are in improving the modularity of systems software 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 Universit"at Darmstadt, where he has worked on the Steamloom VM 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 two 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. Christoph is one of two project supervisors and lead programmers of the ALIA4J project, which comprises the STEAMLOOM^{ALIA} VM, the successor of Steamloom. He authored and co-authored several papers about compilation techniques and VM support for aspect-oriented programming languages, published amongst others by the OOPSLA, AOSD, and VEE conferences. To provide VM support, Christoph researches extensions to high-performing Java VMs 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. Christoph was co-organizer and co-chair of all prior editions of the VMIL workshop and he was program co-chair of the AOSD-Europe Summer School 2009. He was reviewer for the TAOSD and TSE journal, member of the program committee of the ODAL workshop 2006, and supporting reviewer for the conferences Compiler Construction, AOSD, and ECOOP and for the TOSEM journal. Christoph was and continues to be involved in teaching courses to graduate students about aspect-oriented software development, concepts of programming languages and VM.

Robert Dyer is a fourth year Ph. D. student with a research focus on the design of intermediate language models and VM support for advanced modularization techniques. He is currently the lead developer on the Nu project, which includes an intermediate language model and VM support for aspect-oriented programming languages. He has authored and co-authored papers about advanced intermediate languages and dedicated VM caching mechanisms in the ACM TOSEM journal and AOSD conference. He was co-organizer of the previous two editions of the VMIL workshop. Robert served as a supporting reviewer for the OOPSLA and AOSD conferences and FOAL workshop. He is a student member of ACM, SIGSOFT, and SIGPLAN.

Program Committee

Walter Binder (University of Lugano, Switzerland)
Steve Blackburn (Australian National University, Australia)
Erik Ernst (University of Aarhus, Denmark)
Naveen Kumar (Intel Corporation, USA)
Doug Simon (Oracle, Switzerland)
Roel Wuyts (IMEC, Belgium)