JOT
JOURNAL OF
OBJECT TECHNOLOGY

Preface to the JOT Special Issue on ECOOP 2021 Selected Workshop Papers

Davide Ancona[*], **Olivier Flückiger**[†], **Adrian Francalanza**[‡], **Robert Hirschfeld**[§§], **Atsushi Igarashi**[††], **Yu David Liu**[**], **Hidehiko Masuhara**[***], **Chris Seaton**[§], **and Harumi Watanabe**[‡‡]

[*]Università degli Studi di Genova, Italy
[†]Northeastern University, USA
[‡]University of Malta, Malta
[§§]HPI, University of Potsdam, Germany
[††]Kyoto University, Japan
[**]State University of New York at Binghamton, USA
[***]Tokyo Institute of Technology, Japan
[§]Shopify, UK
[‡‡]Tokai University, Japan

ABSTRACT In this preface, the editors present an overview of the topics and scope of the ECOOP workshops on *Context-Oriented Programming (COP)*, on *Implementation, Compilation, Optimization of OO Languages, Programs and Systems (ICOOOLPS)*, and on *Verification and Monitoring at Runtime Execution (VORTEX)*. They further describe the editorial and reviewing process for their editions at ECOOP 2021. The papers selected for publication are presented and briefly described.

KEYWORDS context-oriented programming, runtime verification, runtime monitoring

1. Introduction

The COP, ICOOLPS, and VORTEX workshops were virtually hosted in Aarhus, Denmark, July 12-13, 2021, co-located with ECOOP/ISSTA 2021.

1.1. About COP

The 13th Workshop on Context-Oriented Programming and Advanced Modularity advances the state of the art of modularity support in programming languages, especially through context-oriented programming. The goal of the workshop is to further establish context orientation as a common thread throughout language design, application development, and system support.

Contextual information plays an ever-increasing role in our information-centric world. Current-day software systems adapt continuously to changing execution and usage contexts, even while running. Unfortunately, mainstream programming languages and development environments still do not support this kind of dynamicity very well, leading developers to implement complex designs to anticipate various dimensions of variability.

Context-Oriented Programming directly supports variability at the programming level, depending on a wide range of dynamic attributes. It enables run-time behavior to be dispatched directly on any detected properties of the execution or user context. Since more than a decade, researchers have been working on a variety of notions approaching that idea. Implementations ranging from first prototypes to mature platform extensions used in commercial deployments have illustrated how multi-dimensional dispatch can be supported effectively to achieve expressive run-time variation in behavior.

1.2. About ICOOOLPS

The 16th Workshop on Implementation, Compilation, Optimization of Object-Oriented Languages, Programs and Systems brings together researchers and practitioners working in the field of language implementation and optimization. The goal of the workshop is to discuss emerging problems and research

JOT reference format:
Davide Ancona, Olivier Flückiger, Adrian Francalanza, Robert Hirschfeld, Atsushi Igarashi, Yu David Liu, Hidehiko Masuhara, Chris Seaton, and Harumi Watanabe. *Preface to the JOT Special Issue on:ECOOP 2021 Selected Workshop Papers*. Journal of Object Technology. Vol. 21, No. 2, 2022. Licensed under Attribution 4.0 International (CC BY 4.0)
http://dx.doi.org/10.5381/jot.2022.21.2.e1

directions, as well as new solutions and techniques.

ICOOOLPS prides itself on being an accessible workshop that has still featured influential research in recent years.

1.3. About VORTEX

VORTEX brings together researchers working on all aspects of Runtime Monitoring (RM) with emphasis on integration with formal verification and testing.

RM is concerned with the runtime analysis of software and hardware system executions in order to infer properties relating to system behaviour. Example applications include telemetry, log aggregation, threshold alerting, performance monitoring and adherence to correctness properties (more commonly referred to as Runtime Verification).

RM has gained popularity as a solution to ensure software reliability, bridging the gap between formal verification and testing: on the one hand, the notion of event trace abstracts over system executions, thus favoring system agnosticism to better support reuse and interoperability; on the other hand, monitoring a system offers more opportunities for addressing error recovery, self-adaptation, and issues that go beyond software reliability.

The goal of VORTEX is to bring together researchers contributing on all aspects of RM covering and possibly integrating both theoretical and practical aspects. The workshop focuses on hybrid approaches that take inspiration from, and provide unified solutions with, other techniques for improving the quality of software such as formal methods, program analysis, and testing.

2. Submission and review process

2.1. COP

The program of the COP workshop featured 9 talks, including 3 talks from regular paper submissions, 2 talks from student paper submissions, and 4 invited talks. After the event, we invited authors and presenters to submit the extended and updated drafts of their work to this special issue. The call attracted 3 submissions, out of which 2 were accepted. The special issue submissions went through two rounds of rigorous reviewing, with reviewers drawn from the program committee of the COP workshop.

2.2. ICOOOLPS

The ICOOOLPS workshop presented ten research talks and one invited talk by Dr Stephen Kell on *The Two Cultures of Language Implementation*. Three presentations have been extended into high-quality peer-reviewed papers which we are proud to present in this journal.

2.3. VORTEX

At the workshop six extended abstracts have been presented together with three invited keynotes. After the event, only two works have been invited to contribute to this special issue with extended versions of the papers submitted to the workshop; in the end, one manuscript was withdrawn, while the other underwent a major revision and were resubmitted for a second round of review. The deadlines for the first and second submission rounds were October 31st and December 17th, 2021. We warmly thank the three reviewers selected from the PC of VORTEX who managed the review process: Giorgio Audrito, Angelo Ferrando and Alceste Scalas.

3. Accepted papers

- *First-Class Concepts: Reified Architectural Knowledge Beyond Dominant Decompositions. Toni Mattis, Tom Beckmann, Patrick Rein, and Robert Hirschfeld.* (presented at COP)
  In this paper, the authors discuss novel ways to express a wide range of *concepts*, including cross-cutting concerns, patterns, and lifecycle artifacts independently of the dominant decomposition imposed by an existing architecture. They propose the representation of concepts as first-class objects inside the programming environment that retain the capability to change as easily as code comments. They explore new tools that allow programmers to view, navigate, and change programs based on conceptual perspectives. In a case study, they demonstrate how such views can be created and how the programming experience changes from programmers' scattered attention across existing modules toward cohesively presented concepts. Their designs are geared toward facilitating multiple *secondary* perspectives on a system to co-exist in symbiosis with the original architecture, hence making it easier to explore, understand, and explain complex contexts and narratives that are hard or impossible to express using primary modularity constructs.
- *Next Generation Context-oriented Programming: Embracing Dynamic Generation of Adaptations.Nicolás Cardozo and Ivana Dusparic.* (presented at COP)
  COP languages are used in numerous adaptive systems areas, enabling dynamic swapping and composition of adaptive behavior at run-time. However, until recently, all approaches relied on the offline pre-definition of adaptive behavior, limiting the adaptations to only those foreseen at design time. AutoCOP recently emerged as an approach to shift adaptation definition to run-time, if and when the need for adaptations to new contexts arises, by utilizing reinforcement learning techniques. In this paper, the authors use AutoCOP as a starting point to discuss the research path to achieve a completely dynamic adaptive system. They discuss the potential benefits of such an automated AI-based approach, present several application domain categories where dynamic adaptation definition would enable adaptivity breakthroughs, and discuss open challenges in developing such a fully automated approach.
- *Threaded Code Generation with a Meta-Tracing JIT Compiler. Yusuke Izawa, Hidehiko Masuhara, Carl Friedrich Bolz-Tereick, and Youyou Cong.* (presented at ICOOOLPS)
  Language implementation frameworks, e.g., RPython and Truffle/Graal, are practical tools for creating efficient virtual machines, including a well-functioning just-in-time (JIT) compiler. It is demanding to support multitier JIT compilation in such a framework for language develop-

ers. This paper presents an idea to generate threaded code by reusing an existing meta-tracing JIT compiler, as well as an interpreter design for it. Their approach does not largely modify RPython itself but constructs an effective interpreter definition to enable threaded code generation in RPython. They expect our system to be extended to support multilevel JIT compilation in the RPython framework. They measured the potential performance of our threaded code generation by simulating its behavior in PyPy. They confirmed that our approach reduced code sizes by 80% and compilation times by 60% compared to PyPy's JIT compiler on average, and ran about 7% faster than the interpreter-only execution.

- *Implementation Strategies for Mutable Value Semantics. Dimitri Racordon, Denys Shabalin, Daniel Zheng, Dave Abrahams, and Brennan Saeta.* (presented at ICOOOLPS) This paper presents implementation strategies for compiling programs with mutable value semantics into efficient native code. We study Swift, a programming language based on that discipline, through the lens of a core language that strips some of Swift's features to focus on the semantics of its value types. The strategies that they introduce leverage the inherent properties of mutable value semantics to unlock aggressive optimizations. Fixed-size values are allocated on the stack, thereby enabling numerous off-the-shelf compiler optimizations, while dynamically sized containers use copy-on-write to mitigate copying costs.
- *Toward a Lingua Franca for Memory Safety. Dimitri Racordon, Aurélien Coet, and Didier Buchs.* (presented at ICOOOLPS) Memory safety checking seeks to protect programs from a wide spectrum of software problems related to memory 2 access and management, such as using unallocated or uninitialized buffers. Despite decades of research, it remains an active 3 and fruitful research topic, as issues of scalability and adoption continue to present open challenges. A popular approach to 4 overcome these obstacles is to rely on type checking. Types are arguably one of the most scalable techniques to reason about 5 a program's structural properties. They also offer a convenient tool to impose restrictions on source code, either to prohibit 6 undesirable behaviors or to facilitate other analyses.
- *RV-TEE-Based Trustworthy Secure Shell Deployment: An Empirical Evaluation. Axel Curmi, Christian Colombo, and Mark Vella.* (presented at VORTEX) Computer security is one of the most interesting application of Runtime Monitoring and Verification. This paper investigates the practicality of the runtime-verification-centric trusted execution environment (RV-TEE) approach to the verification of cryptographic protocol implementation as provided by Paramiko, a popular SSH deployment. The studied solution is based on the LARVA RV tool and the USB-connected hardware security module SEcube; as a result, two security vulnerabilities have been identified. The paper provides an empirical evaluation of the practicality of the approach from the perspective of incurred performance and memory overheads.

4. Acknowledgments

The organizers of COP would like to thank Mehdi Bagherzadeh, Nicolás Cardozo, Kenji Hisazumi, Natsuko Noda, and Patrick Rein for their reviews of the special issue submissions. We would like to thank Alfonso Pierantonio and Jan Vitek for their editorial guidance and help.

The Program Committee of ICOOOLPS would like to thank Carl Friedrich Bolz-Tereick, Clément Béra, Lukas Diekmann, Tim Felgentreff, Magnus Madsen, Andrew Nisbet, Olivier Zendra, for their reviews, Jan Viket for his guidance, and Stephen Kell for his invited talk.

We would like to thank all authors, the invited speakers Michael Fisher, Cesar Sanchez and Julien Signoles and the PC members who contributed to the success of VORTEX 2021 and of this special issue. We are grateful to Anders Möller (ECOOP 2021 General Chair), Manu Sridharan (ECOOP 2021 PC Chair), Marcel Böhme and Maria Christakis (ECOOP 2021 WS Co-Chairs) for the successful organization of ECOOP and co-located events; finally, a special thank goes to Jan Vitek, whose contribution has been essential for making this issue possible, and to Alfonso Pierantonio for his precious support in the preparation of the camera-ready versions of the papers.

About the authors

Davide Ancona is associate professor at DIBRIS, Università degli Studi di Genova (Italy). His research interests include Runtime Monitoring and Verification, IoT programming and formal methods. You can contact him at davide.ancona@unige.it or visit https://person.dibris.unige.it/ancona-davide/.

Olivier Flückiger is a PhD student at Northeastern, working on verification of just-in-time compilers and optimizing optimization-hostile languages such as R. You can contact him at o@o1o.ch or visit https://www.o1o.ch/.

Adrian Francalanza is associate professor at the Computer Science Department of the University of Malta (Malta). His research interests include programming language semantics, software verification and concurrency You can contact him at adrian.francalanza@um.edu.mt or visit https://www.um.edu.mt/profile/adrianfrancalanza.

Robert Hirschfeld is a professor at the Hasso Plattner Institute at the University of Potsdam. His research interests include dynamic programming languages, development tools, and runtime environments to make live, exploratory programming more approachable. You can contact him at robert.hirschfeld@hpi.uni-potsdam.de or visit https://www.hpi.uni-potsdam.de/swa/.

Atsushi Igarashi is professor at Kyoto University, Japan. His research interests include program verification, programming language design, and type systems. You can contact him at igarashi@kuis.kyoto-u.ac.jp or visit https://www.fos.kuis.kyoto-u.ac.jp/~igarashi.

Yu David Liu is professor at State University of New York (SUNY) at Binghamton. His research interests include programming languages, software engineering, and computer systems. You can contact him at davidl@binghamton.edu or visit http://www.cs.binghamton.edu/~davidl.

Hidehiko Masuhara is professor at Tokyo Institute of Technology, Japan. His research interests are programming languages, especially on aspect- and context-oriented programming, partial evaluation, computational reflection, meta-level architectures, parallel/concurrent computing, and programming environments. You can contact him at masuhara@is.titech.ac.jp or visit https://prg.is.titech.ac.jp/people/masuhara/.

Chris Seaton is a Researcher (Senior Staff Engineer) at Shopify, where he works on the Ruby programming Language, and a Visitor at the University of Manchester. He was formerly a Research Manager at the Oracle Labs Virtual Machine Research Group, where he led the TruffleRuby implementation of Ruby, and worked on other language and virtual machine projects. Before this he completed a PhD at Manchester where he researched programming languages and irregular parallelism, and an MEng at the University of Bristol on languages with mutable syntax and semantics. You can contact him at chris@chrisseaton.com or visit https://chrisseaton.com/.

Harumi Watanabe is professor at Tokai University in Japan. Her research interests include programming languages, embedded software engineering, and Digital Transformation. You can contact her at harumi-w@tsc.u-tokai.ac.jp or visit https://researchmap.jp/HarumiWatanabe?lang=en