Science of Computer Programming 96 (2014) 175–176

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Preface

Selected and extended papers of the Brazilian Symposium on Programming Languages 2012

The *Brazilian Symposium on Programming Languages* (SBLP) is an annual conference promoted by the Brazilian Computer Society (SBC), which has become an important forum for researchers and practitioners in programming language design and implementation from around the world to interact and present their results and experience. This issue of *Science of Computer Programming* includes extended versions of selected papers from the XVI SBLP, held in Natal, Brazil on September 23–28, 2012.

SBLP 2012 received 27 submissions from Brazil, Czech Republic, France, Netherlands, Portugal, USA and Uruguay. The Program Committee accepted 10 papers (37% acceptance rate), which have been published as volume 7554 of Springer's *Lecture Notes in Computer Science* series. After the conference, extended versions of seven papers have been submitted for publication in a special issue of the Science of Computer Programming journal. After a rigorous reviewing process, in which each paper was reviewed by at least three specialist reviewers, none of them member of the SBLP Program Committee, five submissions were accepted for publication.

The first three papers contribute to formalisms aimed at the specification of programming languages. In particular, the first two propose innovative extensions to Parsing Expression Grammars (PEGs), an expressive formalism for specification of deterministic context-free languages. They are unambiguous by construction and may be viewed as both the specification of a language and the specification of a linear-time top-down parser with an unlimited lookahead. Both papers address different limitations in the expressiveness of this formalism.

Paper *Left Recursion in Parsing Expression Grammars*, by Sérgio Medeiros, Fabio Mascarenhas, and Roberto Ierusalimschy, discusses support for both direct and indirect left-recursive rules in PEGs. For that, it proposes a novel operational semantics, as a conservative extension to the semantics of PEGs. It is the first extension that is not based on packrat parsing as a parsing approach, being actually implementation agnostic. Other contributions include a simple semantic extension to describe expression grammars with multiple levels of operator precedence and associativity, and a semantics for describing left-recursive PEGs as programs in a low-level parsing machine, as an extension to the semantics of a parsing machine for non-left-recursive PEGs developed by Sérgio Medeiros and Roberto Ierusalimschy in an earlier work.

In turn, paper *The Formalization and Implementation of Adaptable Parsing Expression Grammars*, by Leonardo V.S. Reis, Roberto S. Bigonha, Vladimir O. Di Iorio, Luis Eduardo S. Amorim, addresses the specification of extensible programming languages, proposing a new adaptable model based on PEGs for definition of the syntax of extensible programming languages. The paper introduces a complete formalization of this model as well as an implementation on top of which a specification of a real extensible language is discussed. Performance tests are also discussed in the context of this case study.

A third paper on the specification of programming languages, entitled *Attribute Grammar Macros*, by Marcos Viera and Doaitse Swierstra, resorts to a semantic level approach to provide support for extensible programming languages. The problem is stated as how to extend the specification of a programming language with either new abstractions or syntactic sugar, without modifying the base specification. A compiler is regarded as a composition of independently compiled components in such a way that new components may be added at any time to provide incremental extensions to the base language specification. Moreover, the authors propose attribute grammar macros for defining the semantics of the added syntactic constructions in terms of the existing semantics specification, differently from other approaches which only support translation of newly introduced syntactic sugar through its translation to the base syntax. Each semantic extension is encapsulated in a pre-compiled component making possible to check the consistency of the composite system with the Haskell type checker.

In their paper *A Quantitative and Qualitative Assessment of Aspectual Feature Modules for Evolving Software Product Lines*, Felipe Nunes Gaia, Gabriel Coutinho Sousa Ferreira, Eduardo Figueiredo, and Marcelo de Almeida Maia compare four mechanisms for implementing variability along the evolution process of SPLs, with respect to change propagation and modularity aspects: conditional compilation, feature-oriented programming, aspect-oriented programming, and aspectual feature modules. They

http://dx.doi.org/10.1016/j.scico.2014.08.001

resort to two comprehensive case studies with 6 and 5 change scenarios, respectively, to conclude that aspectual feature modules offer increased stability and cope better with crosscutting concerns. The paper also recommends avoiding conditional compilation when feature modularity is a major concern in evolving a SPL.

Spreadsheets may be viewed as special-purpose programming environment targeted at a large set of end-users without programming skills. From the experience of programmers with general-purpose programming environments, a number of recent research initiatives have been proposing techniques to help end-users to avoid redundant work and errors when building large spreadsheets, as well as to increase their productivity and data analysis capability. In the paper *Model-based Programming Environments for Spreadsheets*, Jácome Cunha, Jorge Mendes, João Saraiva and Joost Visser introduce a set of techniques for deriving advanced spreadsheet programming environments from database spreadsheets. In particular, they present a technique for calculating a normalized relational model from an existing spreadsheet, from which a new refactored spreadsheet can be obtained, without redundant data and with editing assistance. Their proposal is validated through a set of spreadsheets from the EUSES Spreadsheet Corpus, a shared resource comprising more than 4500 worksheets gathered from different sources. Finally, the paper reports on an empirical study with 38 participants from medicine, economics, nursing and biology for analyzing the end-user spreadsheet productivity using the proposed approach.

We would like to express our gratitude to all the people who have made this special issue possible. First and foremost, our most sincere appreciation is to the authors for submitting their papers and incorporating all the corrections and improvements as advised by a thorough reviewing process. We are indebted to the reviewers for kindly contributing their time and effort to ensure the highest quality of each paper. Last but not least, we would like to thank Jan A. Bergstra, Bas van Vlijmen, and the editorial staff at Elsevier for agreeing to publish this special issue in *Science of Computer Programming*, and their assistance in bringing it to publication.

Francisco Heron de Carvalho Junior
Universidade Federal do Ceará, Brazil

Luís Soares Barbosa
INESC TEC & Universidade do Minho, Portugal

8 May 2014
Available online 12 August 2014