Preface to the special issue on Coordination Models and Languages (COORDINATION 2010)

This issue of Science of Computer Programming is devoted to papers selected from the 12th International Conference on Coordination Models and Languages (COORDINATION 2010), which took place as a part of the federated conferences on Distributed Computing Techniques (DisCoTec’10), held in Amsterdam, The Netherlands, 7–9 June 2010. COORDINATION focuses on software technologies for collaboration and coordination in concurrent, distributed, and socio-technical systems. A distinctive feature of COORDINATION is the emphasis on high level abstractions capturing interaction patterns at all levels of a system and extending into the realm of the end-user and social domain. It is an exciting time for COORDINATION, with the emergence of new and important technologies such as wireless sensor networks, cloud computing, and multicore processors, along with the increasing prevalence of software in society via the multitude of mobile networked devices.

The best papers of the conference were selected by the programming committee after the conference. Extended versions of these papers were subjected to additional rounds of anonymous peer reviews. The revised versions are included in this special issue, covering topics including session types, service orchestration in nomadic networks, flexible and efficient service provisioning in wireless sensor networks, type system support for object-oriented languages, and behavioural contracts.

Multi-party session types provide a means for rigorously specifying interaction protocols for multiple distributed participants, thereby allowing the static verification of compliance of the corresponding distributed programs. In Efficient sessions, K.C. Sivaramakrishnan, Mohammad Qudeisat, Lukasz Ziarek, Karthik Nagaraj, and Patrick Eugster observe that explicit control flow information manifested by multi-party session types opens intriguing avenues for performance improvements. They present a Java-based session type-guided performance enhancement framework for distributed object interaction. An experimental evaluation in the context of Emulab and Amazon’s Elastic Compute Cloud (EC2) illustrates the benefits of their approach.

Nomadic or mobile ad hoc networks make service orchestration challenging due to the high volatility of connections – temporary network failures – between services residing on mobile devices. As network failures are considered the rule rather than the exception in this setting, failure handling must be a core element of applicable programming languages and coordination models. NOW: Orchestrating services in a nomadic network using a dedicated workflow language by Eline Philips, Ragnhild Van Der Straeten, and Viviane Jonckers applies this idea to a workflow language built on top of an ambient-oriented programming language, thereby adding this extra layer of abstraction to assist programmers in developing applications for nomadic networks. The proposed workflow language, NOW, has support for high level workflow abstractions for control flow, rich network and service failure detection, failure handling through compensating actions, and dynamic data flow between the services in the environment.

Wireless sensor networks (WSNs) are difficult to program and use due to energy constraints and high connectivity dynamics. Software applications must also be coordinated in terms of resource utilization and adaptation to a dynamic environment. In Adaptive service provisioning for enhanced energy efficiency and flexibility in wireless sensor networks, Chien-Liang Fok, Gruia-Catalin Roman, and Chenyang Lu present Adaptive Servilla, a middleware that provides adaptive service provisioning capabilities for coordinating the resources used by WSN applications. The paper demonstrates how adaptive service provisioning enables WSN applications to be more energy efficient while being better able to adapt to the changing availability of network resources. This is achieved through novel service binding strategies that automatically adapt application behaviour when opportunities for energy savings surface, and switch providers in response to changes in the network topology. Adaptive Servilla was implemented on TinyOS and evaluated using the TelosB and Imote2 WSN platforms. Adaptive Servilla is evaluated empirically and its applicability is demonstrated with two e-health applications.

Two developments in object-oriented programming over recent years are traits, which increase the flexibility of composing classes by offering fine-grained units of behaviour, and ownership types, which impose hierarchical structure on the run-time object heap. Combining traits with boxes and ownership types in a Java-like setting, by Lorenzo Bettini, Ferruccio
Damiani, Kathrin Geilmann, and Jan Schaefer, explore how these notions can be combined in the context of the box model. The combination offers static guarantees of object encapsulation and fine-grained reuse among classes. More importantly, it solves a problem specific to the box model’s ownership type system, namely that box classes could not inherit from standard classes and vice versa, which means that sharing code between these two kinds of classes, which was previously not possible, is now possible.

The paper *Behavioural contracts with request–response operations*, by Lucia Acciai, Michele Boreale, and Gianluigi Zavattaro, presents two models of behavioural contracts in the context of service-oriented computing. Behavioural contracts are abstract descriptions of the message-passing behaviour of services and can be used to check properties of service compositions such as service compliance. The notions of contract considered in this paper include bidirectional request–response operations in the presence of an unbounded number of clients and servers. In this setting, client–server compliance is shown to be decidable in a model based on the abstract service interface language WSCL and undecidable in a model based on abstract WS-BPEL, showing an expressiveness gap between the two models.

The proceedings of COORDINATION 2010, where the original version of these papers appear, were published as volume 6116 in Springer's Lecture Notes in Computer Science series. We gratefully acknowledge the anonymous reviewers as well as the program committee members of COORDINATION 2010 for their help in selecting the papers for this special issue. Finally we would like to thank the referees for this special issue for the high quality of their reviews.

Dave Clarke*

*Katholieke Universiteit Leuven, Belgium
E-mail address: Dave.Clarke@cs.kuleuven.be.

Gul Agha**

*University of Illinois at Urbana-Champaign, United States
E-mail address: agha@cs.uiuc.edu.

Available online 9 May 2012

* Corresponding editor.
** Corresponding editor.