Lars Braubach Jean-Pierre Briot
John Thangarajah (Eds.)

Programming Multi-Agent Systems

7th International Workshop, ProMAS 2009
Budapest, Hungary, May 10-15, 2009
Revised Selected Papers

Volume Editors

Lars Braubach
University of Hamburg
Hamburg, Germany
E-mail: braubach@informatik.uni-hamburg.de

Jean-Pierre Briot
LIP6, Paris 6 - CNRS
Paris, France
E-mail: jean-pierre.briot@lip6.fr

John Thangarajah
RMIT University
Melbourne, Australia
E-mail: johthan@cs.rmit.edu.au

Library of Congress Control Number: 2010932226

CR Subject Classification (1998): I.2, D.2, C.2.4, I.2.11, I.6, D.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-642-14842-5 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-14842-2 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Foreword

The earliest work on agents may be traced at least to the first conceptualization of the actor model by Carl Hewitt. In a paper in an AI conference in the early 1970s, Hewitt described actors as entities with knowledge and goals. Research on actors continued to focus on AI with the development of the Sprites model in which a monotonically growing knowledge base could be accessed by actors (inspired by what Hewitt called "the Scientific Computing Metaphor"). In the late 1970s and well into 1980s, controversy raged in AI between those arguing for declarative languages and those arguing for procedural ones. Actor researchers stood on the side of a procedural view of knowledge, arguing for an open systems perspective rather than the closed world hypothesis necessary for a logical, declarative view. In the open systems view, agents had arms length relationships and could not be expected to store consistent facts, nor could the information in a system be considered complete (the "negation as failure" model).

Subsequent work on actors, including my own, focused on using actors for general purpose concurrent and distributed programming. In the late 1980s, a number of actor languages and frameworks were built. These included Act++ (in C++) by Dennis Kafura and Actalk (in Smalltalk) by Jean-Pierre Briot. In recent times, the use of the Actor model, in various guises, has proliferated as new parallel and distributed computing platforms and applications have become common: clusters, Web services, P2P networks, client programming on multicore processors, and cloud computing. These applications have motivated the use of actors as it is proving to be more natural to program large-scale actor systems than write programs with large numbers of threads, locks, and shared variables. Among the many actor languages and frameworks now being used or developed are Erlang, E, Scala, Salsa, Kilim, ActorFoundry, and Axum, besides many other frameworks in Python, Java, and Ruby.

Work on agents from its earliest stages has concentrated on distributed problem solving. Unlike the turn actor research had taken by focusing on the development of actors as a programming model, the agents research community worked on ways in which agents could be used to solve problems more abstractly. By embedding more abstract declarative notions of knowledge within agents, it can be said that while at their core agents remain procedural, agents are also overcoming the schism between the declarative and the procedural camps.

It can be safely asserted that research in agents as "intelligent" actors gathered momentum in the 1990s. However, the bulk of this research focused on building models and algorithms. The community relied on theoretical analyses and simulations as their primary tools. The need for building intelligent agent languages and development tools continued to grow. It is true that the formal programming model for agents is actors, in that agents are autonomous, operate asynchronously, are distributed, may create new agents and acquire knowledge

about other agents, all properties of the actor programming model. However, supporting multi-agent programming requires more than this: agents have a complex semantics which needs to be supported by providing ways of easily specifying concepts such as mental models, beliefs, goals, intentions, and plans. Although certain frameworks for agents, such as BDI, have also become popular, programming support for these frameworks remains a work in progress. Part of the reason for this is the need for research which will enable us to understand how to build debugging, profiling, and monitoring tools for agents.

The ProMAS community is addressing this critical need. The current volume is a good representative of the state of the art; it documents both the progress that has been made and the challenges that lie ahead. For example, research on the notion of commitments shows how a more semantic model of communication can be interpreted to specify behaviors of agents instead of the traditional rigid method of specifying interactions by building specific protocols. Research in reasoning and verification of agents has also focused on the semantic content of agent behavior, applying techniques such as constraint solving, hybrid automata, situation calculus, and other formal methods.

In the final analysis, research on multi-agent systems should not be viewed as research in a specialized area; rather it as an ambitious enterprise whose goal is to change the future of computing. Multi-agent programming is a way of more abstractly expressing general purpose real-world programs which are concurrent and distributed, open to interaction, and extensible. By enabling more declarative specifications in the small and resilience through cooperation in the large, agent programming provides the robustness currently missing from software. To reach that vision, multi-agent research must focus not only on making actors more intelligent, it must focus on providing models and tools which enable large organizations or ensembles of agents to be programmed effectively.

November 2009

Gul Agha
University of Illinois at Urbana-Champaign

Preface

These are the proceedings of the International Workshop on Programming Multi-Agent Systems (ProMAS 2009), the seventh of a series of workshops that has the main objective of giving an overview of current research for programming multi-agent systems and providing an interactive discussion forum for agent researchers.

The ProMAS workshop series aims at promoting and contributing to the establishment of multi-agent systems as a mainstream approach to the development of industrial-strength software. More specifically, the workshop facilitates the discussion and exchange of ideas concerning the concepts, techniques, and tools that are important for establishing multi-agent programming platforms that are useful in practice and have a theoretically sound basis.

In its previous editions, ProMAS constituted an invaluable occasion bringing together leading researchers from both academia and industry to discuss issues on the design of programming languages and tools for multi-agent systems. We were very pleased to be able to again present a range of high-quality papers at ProMAS 2009. After six successful editions of the ProMAS workshop series, which took place at AAMAS 2003 (Melbourne, Australia), AAMAS 2004 (New York, USA), AAMAS 2005 (Utrecht, The Netherlands), AAMAS 2006 (Hakodate, Japan), AAMAS 2007 (Honolulu, Hawai'i), and AAMAS 2008 (Estoril, Portugal), the seventh edition took place during May 11–12 in Budapest, Hungary, in conjunction with AAMAS 2009, the main international conference on autonomous agents and multi-agent systems. ProMAS 2009 received 34 submissions. Each of these papers was carefully reviewed by three members of the Program Committee. As a result, nine contributions were accepted as full presentations and seven as short ones. Due to the high number of quality contributions received this year, it was decided to extend ProMAS 2009 to a two-day workshop.

At the workshop, in addition to the presentation of regular papers, Munindar Singh (North Carolina State University) gave an invited talk about *commitment communication*. The main idea is that communication relations are often specified in a too constrained way when traditional description techniques like AUML sequence diagrams are used. In this case, too much attention is paid to the exact definition of how an interaction takes place, i.e., what the possible message sequences are. In contrast to this rigid way, a commitment-based specification is proposed. Commitments are a mechanism for specifying the different responsibilities of the participating communication parties. The communication then follows the commitments made by these parties, which means the focus of the interaction shifts from flow orientation to reason orientation. In this way, the concrete message ordering loses importance and alternatives may be exploited. We are also happy that Munindar Singh accepted the invitation for an invited paper on this topic and provided it for this ProMAS proceedings volume.

Following the workshop, we set up a new submission, evaluation, and revision process for publishing these proceedings. The authors of the papers accepted at the workshop were invited to submit revised papers. Each paper was reviewed by a member of the Program Committee and by the editors. Authors were then requested to further revise their submissions and the resulting papers are what forms this volume.

This volume also includes a foreword by Gul Agha (University of Illinois at Urbana-Champaign). In his foreword, Gul Agha traces back work on multi-agent programming to the early proposal about actors by Carl Hewitt in the early 1970s and discusses the relations between multi-agent programming and actor programming, pointed out as a possible programming model foundation. He also discusses the needs for better understanding and for building debugging, profiling and monitoring tools for large-scale multi-agent programs.

The workshop addressed a broad range of mostly practical topics. This year the topics included practical examples of applying agent technology in interesting application domains such as computer games and boat monitoring. Another focus was on typical programming aspects such as debugging and profiling, which are common in standard programming languages but very new to agent languages. Finally, more formal aspects were also covered e.g., those that address artifact environments and verification with hybrid automata.

We thank the authors whose contributions made this book possible. Also, we thank the members of the Program Committee for their dedication on successive rounds of reviewing papers.

As for previous editions, we hope that the work described in these proceedings will contribute to the overall goal of stimulating the uptake of agent programming languages and the adoption of agent-based tools for real-world applications.

January 2010

Lars Braubach
Jean-Pierre Briot
John Thangarajah

Organization

The ProMAS 2009 workshop was held May 11–12, 2009, in Budapest, Hungary. The workshop was part of the AAMAS 2009 Workshop Program.

Program Chairs

Lars Braubach	University of Hamburg, Germany
Jean-Pierre Briot	LIP6, University Paris 6 - CNRS, France
John Thangarajah	RMIT University, Australia

Steering Committee

Rafael Heitor Bordini	Federal University of Rio Grande do Sul, Brazil
Mehdi Dastani	Utrecht University, The Netherlands
Jürgen Dix	Clausthal University of Technology, Germany
Amal El Fallah Seghrouchni	LIP6, University Paris 6 - CNRS, France

Program Committee

Matteo Baldoni	University of Turin, Italy
Guido Boella	University of Turin, Italy
Juan Botía Blaya	Universidad de Murcia, Spain
Keith Clark	Imperial College, UK
Rem Collier	University College Dublin, Ireland
Louise Dennis	University of Liverpool, UK
Ian Dickinson	HP Labs, Bristol, UK
Berndt Farwer	Durham University, UK
Michael Fisher	University of Liverpool, UK
Jorge Gómez-Sanz	Universidad Complutense de Madrid, Spain
Vladimir Gorodetsky	Russian Academy of Sciences, Russian Federation
Dominic Greenwood	Whitestein Technologies, Switzerland
James Harland	RMIT University, Australia
Koen Hindriks	Delft University of Technology, The Netherlands
Benjamin Hirsch	TU-Berlin, Germany
Jomi Fred Hübner	ENS Mines Saint-Etienne, France
João Leite	Universidade Nova de Lisboa, Portugal
Viviana Mascardi	University of Genova, Italy
John-Jules Meyer	Utrecht University, The Netherlands
David Morley	SRI International, USA
Jörg Müller	Clausthal University of Technology, Germany
Peter Novák	Clausthal University of Technology, Germany

Andrea Omicini	University of Bologna, Italy
Frédéric Peschanski	LIP6, University Paris 6 - CNRS, France
Michele Piunti	ISTC - CNR and DEIS Università di Bologna, Italy
Agostino Poggi	University of Parma, Italy
Alexander Pokahr	University of Hamburg, Germany
Alessandro Ricci	DEIS, Università di Bologna, Italy
Ralph Rönnquist	Intendico Pty Ltd, Australia
Sebastian Sardina	RMIT University, Australia
Ichiro Satoh	National Institute of Informatics, Japan
Munindar P. Singh	NCSU, USA
Tran Cao Son	New Mexico State University, USA
Kostas Stathis	Royal Holloway, UK
Paolo Torroni	University of Bologna, Italy
Gerhard Weiß	SCCH GmbH, Austria
Wayne Wobcke	University of New South Wales, Australia
Neil Yorke-Smith	SRI International, USA
Yingqian Zhang	Delft University of Technology, The Netherlands
Olivier Boissier	ENS Mines Saint-Etienne, France
Birna van Riemsdijk	Delft University of Technology, The Netherlands
Leon van der Torre	University of Luxembourg, ILIAS, Luxembourg

Auxiliary Reviewers

Alferes, José	Hepple, Anthony
Bromuri, Stefano	Kaiser, Silvan
Chopra, Amit	Remondino, Marco
Gabaldon, Alfredo	Torres, Viviane
Ghizzioli, Roberto	

Table of Contents

IV Analysis and Debugging

V Agent Architectures

VI Applications