

Lecture Notes in Artificial Intelligence 7879

Subseries of Lecture Notes in Computer Science

LNAI Series Editors

Randy Goebel

University of Alberta, Edmonton, Canada

Yuzuru Tanaka

Hokkaido University, Sapporo, Japan

Wolfgang Wahlster

DFKI and Saarland University, Saarbrücken, Germany

LNAI Founding Series Editor

Joerg Siekmann

DFKI and Saarland University, Saarbrücken, Germany

Yves Demazeau Toru Ishida
Juan M. Corchado Javier Bajo (Eds.)

Advances on Practical Applications of Agents and Multi-Agent Systems

11th International Conference, PAAMS 2013
Salamanca, Spain, May 22-24, 2013
Proceedings



Springer

Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada
Jörg Siekmann, University of Saarland, Saarbrücken, Germany
Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

Volume Editors

Yves Demazeau
Centre National de la Recherche Scientifique
Grenoble, France
E-mail: yves.demazeau@imag.fr

Toru Ishida
Kyoto University, Japan
E-mail: ishida@i.kyoto-u.ac.jp

Juan M. Corchado
Universidad de Salamanca, Spain
E-mail: corchado@usal.es

Javier Bajo
Universidad Politécnica de Madrid, Spain
E-mail: javier.bajo@upm.es

ISSN 0302-9743
ISBN 978-3-642-38072-3
DOI 10.1007/978-3-642-38073-0
Springer Heidelberg Dordrecht London New York

e-ISSN 1611-3349
e-ISBN 978-3-642-38073-0

Library of Congress Control Number: Applied for

CR Subject Classification (1998): I.2.11, I.2, I.6, K.3, K.4, J.1, J.2, J.7, C.2, K.4.4, H.3, H.4

LNCS Sublibrary: SL 7 – Artificial Intelligence

© Springer-Verlag Berlin Heidelberg 2013

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Research on agents and multi-agent systems has matured during the last decade and many effective applications of this technology are now deployed. An international forum to present and discuss the latest scientific developments and their effective applications, to assess the impact of the approach, and to facilitate technology transfer has become a necessity and was in fact created a few years ago.

PAAMS, the International Conference on Practical Applications of Agents and Multi-Agent Systems, is the international yearly platform to present, to discuss, and to disseminate the latest developments and the most important outcomes related to real-world applications. It provides a unique opportunity to bring multi-disciplinary experts, academics, and practitioners together to exchange their experience in the development and deployment of agents and multi-agent systems.

This volume presents the papers that were accepted for the 2013 edition of PAAMS. These articles report on the application and validation of agent-based models, methods, and technologies in a number of key application areas, including: agents for real-world Problems; crowds modelling and analysis; decision making and discovery; interaction with artificial agents; mobility, ubiquity, and clouds; (multi-)agent design technology; and simulation and organization. Each paper submitted to PAAMS went through a stringent peer-review by three members of the international committee composed of 93 internationally renowned researchers from 24 countries. From the 70 submissions received, 14 were selected for full presentation at the conference; another nine papers were accepted as short presentations. In addition, a demonstration session featuring innovative and emergent applications of agent and multi-agent systems and technologies in real-world domains was organized. In all, 16 demonstrations were shown, and this volume contains a description of each of them.

We would like to thank all the contributing authors, the members of the Program Committee, the sponsors (IEEE SMC Spain, IBM, AEPIA, AFIA, University of Salamanca and CNRS), and the Organizing Committee for their hard and highly valuable work. Their work has helped to contribute to the success of the PAAMS 2013 event. Thanks for your help - PAAMS 2013 would not exist without your contribution.

Yves Demazeau
Toru Ishida
Juan Manuel Corchado
Javier Bajo

Organization

General Co-chairs

Yves Demazeau	Centre National de la Recherche Scientifique, France
Toru Ishida	University of Kyoto, Japan
Juan M. Corchado	University of Salamanca, Spain
Javier Bajo	Polytechnic University of Madrid, Spain

Advisory Board

Frank Dignum	Utrecht University, The Netherlands
Jörg P. Müller	Technische Universität Clausthal, Germany
Juan Pavón	Universidad Complutense de Madrid, Spain
Michal Pěchouček	Czech Technical University in Prague, Czech Republic

Program Committee

Carole Adam	University of Grenoble, France
Frederic Amblard	University of Toulouse, France
Francesco Amigoni	Politecnico di Milano, Italy
Luis Antunes	University of Lisbon, Portugal
Javier Bajo	Polytechnic University of Madrid, Spain
Jeremy Baxter	QinetiQ
Michael Berger	Docuware AG, Germany
Olivier Boissier	Ecole Nationale Supérieure des Mines de Saint Etienne, France
Vicente Botti	Polytechnic University of Valencia, Spain
Lars Braubach	Universität Hamburg, Germany
France Brazier	TU Delft, The Netherlands
Stefano Bromuri	University of Applied Sciences Western Switzerland
Valerie Camps	University of Toulouse, France
Longbing Cao	University of Technology Sydney, Australia
Javier Carbo	Carlos III University of Madrid, Spain
Lawrence Cavedon	RMIT Melbourne, Australia
Pierre Chevaillier	University of Brest, France
Helder Coelho	University of Lisbon, Portugal
Juan Manuel Corchado	University of Salamanca, Spain
Keith Decker	University of Delaware, USA

Yves Demazeau	Laboratoire d'Informatique de Grenoble, France
Frank Dignum	Utrecht University, The Netherlands
Virginia Dignum	TU Delft, The Netherlands
Alexis Drogoul	Institut de Recherche pour le Développement, Vietnam
Julie Dugdale	University of Grenoble, France
Amal Elfallah	University of Paris 6, France
Maksims Fiosins	Clausthal University of Technology, Germany
Klaus Fischer	DFKI, Germany
Rubén Fuentes	Complutense University of Madrid, Spain
Sylvain Giroux	University of Sherbrooke, Canada
Marie-Pierre Gleizes	University of Toulouse, France
Pierre Glize	University of Toulouse, France
Daniela Godoy	ISISTAN, Argentina
Jorge Gomez-Sanz	Complutense University of Madrid, Spain
Vladimir Gorodetski	University of Saint Petersburg, Russia
Olivier Gutknecht	ACM, USA
Kasper Hallenborg	University of Southern Denmark, Denmark
Koen Hindriks	University of Delft, The Netherlands
Benjamin Hirsch	Technical University of Berlin, Germany
Martin Hofmann	Lockheed Martin, USA
Tom Holvoet	Catholic University of Leuven, Belgium
Shinichi Honiden	National Institute of Informatics Tokyo, Japan
Jomi Fred Hubner	Universidad Federale de Santa Catarina, Florianopolis
Toru Ishida	University of Kyoto, Japan
Takayuki Ito	Massachusetts Institute of Technology, USA
Michal Jakob	Czech Technical University in Prague, Czech Republic
Vicente Julian	Polytechnic University of Valencia, Spain
Achilles Kameas	University of Patras, Greece
Takahiro Kawamura	Toshiba, Japan
Stefan Kirn	Universität Hohenheim, Germany
Franziska Kluegl	University of Örebro, Sweden
Matthias Klusch	DFKI, Germany
Martin Kollingbaum	University of Aberdeen, UK
Ryszard Kowalczyk	Swinburne University of Technology, Australia
Jaroslawn Kozlak	University of Science and Technology in Krakow, Poland
Jiming Liu	Hong Kong Baptist University, China
Beatriz López	Universitat de Girona, Spain
Adolfo López Paredes	University of Valladolid, Spain
Zakaria Maamar	Zayed University, United Arab Emirates
Rene Mandiau	University of Valenciennes, France

Philippe Mathieu	University of Lille, France
Eric Matson	Purdue University, USA
Fabien Michel	University of Reims, France
José M. Molina	Universidad Carlos III de Madrid, Spain
Mirko Morandini	University of Trento, Italy
Jean-Pierre Muller	CIRAD, France
Jörg P. Müller	Clausthal University of Technology, Germany
Peter Novak	Czech Technical University in Prague, Czech Republic
Akhihiko Ohsuga	University of Electro-Communications, Japan
Eugenio Oliveira	University of Porto, Portugal
Andrea Omicini	University of Bologna, Italy
Sascha Ossowski	University of Rey Juan Carlos, Spain
Julian Padget	University of Bath, UK
Van Parunak	New Vectors, USA
Juan Pavon	Complutense University of Madrid, Spain
Michal Pechoucek	Czech Technical University, Czech Republic
Paolo Petta	University of Vienna, Austria
Michael Pirker	Siemens AG, Germany
Alessandro Ricci	University of Bologna, Italy
Juan Antonio Rodriguez Aguilar	AI Research Institute, Spain
Alex Rogers	University of Southampton, UK
Nicolas Sabouret	University of Paris 11, France
Silvia Schiaffino	ISISTAN, Argentina
Toshiharu Sugawara	Waseda University, Japan
Simon Thompson	British Telecom IIS Research Centre, UK
Paolo Torroni	University of Bologna, Italy
Rainer Unland	University of Duisburg, Germany
Domenico Ursino	University of Reggio Calabria, Italy
Jacques Verriet	Embedded Systems Institute, The Netherlands
José Villar	University of Oviedo, Spain
Jiri Vokrinek	Czech Technical University in Prague, Czech Republic
Gerhard Weiss	University of Maastricht, The Netherlands
Danny Weyns	Linnaeus University, Sweden
Niek Wijngaards	Thales, D-CIS lab, The Netherlands
Gaku Yamamoto	IBM, Japan

Organizing Committee

Juan M. Corchado (Chair)	University of Salamanca, Spain
Javier Bajo (Co-chair)	Polytechnic University of Madrid, Spain
Juan F. De Paz	University of Salamanca, Spain
Sara Rodríguez	University of Salamanca, Spain
Dante I. Tapia	University of Salamanca, Spain
Fernando de la Prieta Pintado	University of Salamanca, Spain
Davinia Carolina Zato Domínguez	University of Salamanca, Spain
Gabriel Villarrubia González	University of Salamanca, Spain
Alejandro Sánchez Yuste	University of Salamanca, Spain
Antonio Juan Sánchez Martín	University of Salamanca, Spain
Cristian I. Pinzón	University of Salamanca, Spain
Rosa Cano	University of Salamanca, Spain
Emilio S. Corchado	University of Salamanca, Spain
Eugenio Aguirre	University of Granada, Spain
Manuel P. Rubio	University of Salamanca, Spain
Belén Pérez Lancho	University of Salamanca, Spain
Angélica González Arrieta	University of Salamanca, Spain
Vivian F. López	University of Salamanca, Spain
Ana de Luís	University of Salamanca, Spain
Ana B. Gil	University of Salamanca, Spain
M ^a Dolores Muñoz Vicente	University of Salamanca, Spain
Jesús García Herrero	Carlos III University of Madrid, Spain
Elena García	University of Salamanca, Spain
Roberto González	University of Salamanca, Spain

Table of Contents

Regular Papers

Preventing Elderly from Falls: The Agent Perspective in EPRs	1
<i>Sebastian Ahrndt, Johannes Fähndrich, and Sahin Albayrak</i>	
Dynamic Organisation of the Household Activities for Energy Consumption Simulation	13
<i>Edouard Amouroux and François Sempé</i>	
MAKKSIm: MAS-Based Crowd Simulations for Designer’s Decision Support	25
<i>Luca Crociani, Lorenza Manenti, and Giuseppe Vizzari</i>	
A Multiagent System for Resource Distribution into a Cloud Computing Environment	37
<i>Fernando De la Prieta, Sara Rodríguez, Javier Bajo, and Juan Manuel Corchado</i>	
Empirical Specification of Dialogue Games for an Interactive Agent	49
<i>Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Jean-Philippe Kotowicz, Alexandre Pauchet, and Jean-Pierre Pécuchet</i>	
MAS-BPM: Multi-Agent System Bomber Problem Model	61
<i>Zina Elguedria, Boutheina Jlifi, and Khaled Ghédira</i>	
Agent Perception Modeling for Movement in Crowds	73
<i>Katayoun Farrahi, Kashif Zia, Alexei Sharpanskykh, Alois Ferscha, and Lev Muchnik</i>	
SERENA: A Multi-site Pervasive Agent Environment That Supports Serendipitous Discovery in Research	85
<i>Jamie Forth, Thanasis Giannimaras, Geraint A. Wiggins, Robert Stewart, Diana Bental, Ruth Aylett, Deborah Maxwell, Hadi Mehrpouya, Jamie Shek, and Mel Woods</i>	
Towards a Multi-avatar Macroeconomic System	97
<i>Gianfranco Giulioni, Edgardo Bucciarelli, Marcello Silvestri, and Paola D’Orazio</i>	
Dynamic Filtering of Useless Data in an Adaptive Multi-Agent System: Evaluation in the Ambient Domain	110
<i>Valérian Guivarch, Valérie Camps, André Péninou, and Simon Stuker</i>	

ARMAN: Agent-based Reputation for Mobile Ad hoc Networks	122
<i>Guy Guemkam, Djamel Khadraoui, Benjamin Gâteau, and Zahia Guessoum</i>	
Decentralized Intelligent Real World Embedded Systems: A Tool to Tune Design and Deployment	133
<i>Jean-Paul Jamont, Michel Occello, and Eduardo Mendes</i>	
Multi-agent Models for Transportation Problems with Different Strategies of Environment Information Propagation	145
<i>Jarosław Koźlak, Sebastian Pisarski, and Małgorzata Żabińska</i>	
How to Build the Best Macroscopic Description of Your Multi-Agent System?	157
<i>Robin Lamarche-Perrin, Yves Demazeau, and Jean-Marc Vincent</i>	
Multi-layered Satisficing Decision Making in Oil and Gas Production Platforms	170
<i>Lars Lindegaard Mikkelsen, Yves Demazeau, and Bo Nørregaard Jørgensen</i>	
A Security Response Approach Based on the Deployment of Mobile Agents	182
<i>Roberto Magán-Carrión, José Camacho-Páez, and Pedro García-Teodoro</i>	
Modeling Dependence Networks for Agent Based Simulation of Online and Offline Communities	192
<i>Francesca Marzo, Stefano Za, and Paolo Spagnoletti</i>	
Improving Classifier Agents with Order Book Information	204
<i>Philippe Mathieu and Matthis Gaciarz</i>	
From Real Purchase to Realistic Populations of Simulated Customers	216
<i>Philippe Mathieu and Sébastien Picault</i>	
REAGENT: Reverse Engineering of Multi-Agent Systems	228
<i>Ricardo Pérez-Castillo, Juan Pavón, Jorge J. Gómez-Sanz, and Mario Piattini</i>	
An Agent-Based Analyses of F-formations	239
<i>Kavin Preethi Narasimhan and Graham White</i>	
X-CAMPUS: A Proactive Agent for Ubiquitous Services	251
<i>Hajer Sassi and José Rowillard</i>	

Demo Papers

Agents Vote against Falls: The Agent Perspective in EPRs	263
<i>Sebastian Ahrndt, Johannes Fährndrich, and Sahin Albayrak</i>	
Pedestrians and Crowd Simulations with MAKKSIM - A Demonstration	267
<i>Luca Crociani, Lorenza Manenti, and Giuseppe Vizzari</i>	
GAMA: A Spatially Explicit, Multi-level, Agent-Based Modeling and Simulation Platform	271
<i>Alexis Drogoul, Edouard Amouroux, Philippe Caillou, Benoit Gaudou, Arnaud Grignard, Nicolas Marilleau, Patrick Taillandier, Maroussia Vavasseur, Duc-An Vo, and Jean-Daniel Zucker</i>	
Demonstrating SERENA: Chance Encounters in the Space of Ideas	275
<i>Jamie Forth, Athanasios Giannimaras, Geraint A. Wiggins, Robert Stewart, Diana Bental, Ruth Aylett, Deborah Maxwell, Hadi Mehrpouya, Jamie Shek, and Mel Woods</i>	
Automatic Deployment of a Consensus Networks MAS	279
<i>Yolanda Gómez, Alberto Palomares, Carlos Carrascosa, and Miguel Rebollo</i>	
Using MASH in the Context of the Design of Embedded Multiagent System	283
<i>Jean-Paul Jamont and Michel Occello</i>	
A Brownian Agent Model for Analyzing Changes in Product Space Structure in China	287
<i>Bin Jiang, Chao Yang, Shuming Peng, Renfa Li, and Takao Terano</i>	
ArgCBR-CallCentre: A Call Centre Based on CBR Argumentative Agents	292
<i>Jaume Jordán, Stella Heras, Soledad Valero, and Vicente Julián</i>	
Analysis of International Relations through Spatial and Temporal Aggregation	296
<i>Robin Lamarche-Perrin, Yves Demazeau, and Jean-Marc Vincent</i>	
Demonstrating Multi-layered MAS in Control of Offshore Oil and Gas Production	300
<i>Lars Lindegaard Mikkelsen, Jørgen Rentler Næumann, Yves Demazeau, and Bo Nørregaard Jørgensen</i>	
IMOSHION: A Simulation Framework Using Virtual Intelligent Agents for Workplace Evacuation in Case of Emergency Situation	304
<i>Stéphane Maruejols and Caroline Chopinaud</i>	

A Security Response Approach Based on the Deployment of Mobile Agents: A Practical Vision	308
<i>Roberto Magán-Carrión, José Camacho-Páez, and Pedro García-Teodoro</i>	
The Galaxian Project: A 3D Interaction-Based Animation Engine	312
<i>Philippe Mathieu and Sébastien Picault</i>	
Demonstration of the Multi-Agent Simulator of Competitive Electricity Markets	316
<i>Tiago Pinto, Isabel Praça, Gabriel Santos, and Zita Vale</i>	
Multi-Agent Systems Platform for Mobile Robots Collision Avoidance	320
<i>Angel Soriano, Enrique J. Bernabeu, Angel Valera, and Marina Vallés</i>	
Parallel and Distributed Simulation of Large-Scale Cognitive Agents	324
<i>Kashif Zia, Katayoun Farrahi, Alexei Sharpanskykh, Alois Ferscha, and Lev Muchnik</i>	
Author Index	329

REAGENT: Reverse Engineering of Multi-Agent Systems

Ricardo Pérez-Castillo¹, Juan Pavón², Jorge J. Gómez-Sanz², and Mario Piattini¹

¹ Instituto de Tecnologías y Sistemas de Información (ITSI), University of Castilla-La Mancha
Paseo de la Universidad, 4 13071, Ciudad Real, Spain

{ricardo.pdelcastillo,mario.piattini}@uclm.es

² Department of Software Engineering and Artificial Intelligence,
Universidad Complutense de Madrid

Ciudad Universitaria s/n, 28040, Madrid, Spain

{jpavon, jjgomez}@fdi.ucm.es

Abstract. Agent-based technology is being used in an increasing variety of applications and domains. Despite the substantial research effort on methodologies for analyzing, designing and implementing multi-agent systems (MAS), maintenance and evolution of MAS software is nowadays challenging. This paper presents REAGENT, a reverse engineering technique for retrieving MAS design models from the source code of Jade based MAS implementations. REAGENT support tools have been experimentally evaluated by means of a case study with 19 benchmark programs from textbook. REAGENT has proved to be able to collect accurate and complete agent models in a linear time regarding the size of agent models, which facilitates its applicability to large, complex, industrial MAS.

Keywords: Multi-Agent System, Reverse Engineering, Maintenance, REAGENT.

1 Introduction

Multi-Agent Systems (MAS) are being used in an increasing variety of applications, ranging from comparatively small systems for personal assistance to open, complex, mission-critical systems for industrial applications such as process control, system diagnostics, manufacturing, transportation logistics and network management.

Similarly to traditional information systems, MAS have to be maintained in order to solve or prevent faults, and evolve adding new functionalities or meeting new requirements. During software maintenance, sometimes the only reliable information is embedded in the source code since the documentation is missing or outdated. Reverse engineering of source code aims at creating high-level representations for the existing software system to support its comprehension and evolution [5]. Program comprehension is time-consuming and entails the largest portion of the maintenance effort, especially because it is often performed manually with simple general-purpose tools, such as editors and regular expression matchers. There are also some reverse engineering tools for traditional information systems, which can facilitate the task, for example to

extract facts from source code/binaries, execution traces, or historical data, to query the extracted facts, and to build high-level views of the software system (e.g., UML diagrams) [4].

While reverse engineering of information systems has been widely addressed in literature and is a mature area, there are no similar achievements on research about reverse engineering of MAS software [10]. One of the reason of this fact is the nature of MAS (e.g., communication and interaction protocols, belief sharing and propagation, behavior adaptation, etc.), which entails particular challenges making it difficult to address reverse engineering of MAS [2].

Despite the mentioned challenges, there are some approaches for reversing MAS, such as *Moreno et al.* [10], which uses natural language processing, *Bosse et al.* [3], which records agent activities by considering execution traces of MAS, or *Sauvage* [15], which employs design patterns to collect and express agent concepts. Unfortunately, most current reverse engineering techniques are *ad hoc* solutions that only focus on a particular agent framework. This entails a lack of standardization, and therefore, a lack of automation. This problem makes it difficult to reuse these techniques with large and complex MAS.

The main contribution of this paper is REAGENT, a reverse engineering technique for obtaining design models by analyzing the source code of MAS. REAGENT takes JADE-based systems as input and provides abstract design models according to INGENIAS [12], an agent-oriented development methodology. Although REAGENT currently works only with JADE and INGENIAS as the operational/execution environment, ideas from this approach can be adapted to other agent platforms and methodologies. JADE has been chosen because there is a vast number of MAS that are implemented with this platform, as it follows the FIPA standard. INGENIAS has been chosen for MAS modeling because it provides a set of tools that are based on meta-modeling techniques, which facilitate the editing and transformation of models. The conformance with these well-known platforms together with the proposal is supported by a tool that ensures the reuse of this technique to large, complex and industrial MAS. This paper additionally provides experimental results about the applicability of REAGENT to a set of example JADE programs [2] that have been used as benchmarks in this context.

The remaining of the paper is organized as follows: Section 2 summarizes related work. Section 3 presents the reverse engineering approach for recovering MAS models that are specified with the INGENIAS modeling language. Section 4 shows the multi-case study with MAS benchmarks. Finally, Section 5 discusses conclusions and future work.

2 Related Work

There is some precursory work in the literature about reverse engineering of MAS. For example, *Moreno et al.* [10] have applied natural language processing techniques throughout the entire lifecycle of MAS in order to obtain a design model from existing MAS. In the same line, *Hirst* [8] has proposed analysis of pre-existing Soar agents by automatically reverse engineering them. This has been shown useful for

maintaining the agents themselves, and also suggests the possibility of knowledge reuse across agent architectures. The limitation of this approach is that it proposes an *ad hoc* technique specially developed for Soar architectures.

While previous approaches mainly focus on static analysis, i.e., a syntactic analysis of source code line by line, other approaches are based on the dynamic analysis. *Bosse et al.* [3] uses the dynamic analysis of MAS by recording agent activities as execution traces to verify that the traces satisfy some specified properties. This approach ensures that the user's comprehension of the system behavior is accurate with respect to the execution traces detecting anomalous behavior. *Adra et al.* [1] have proposed a dynamic inference mechanism for the refinement of agent-based model specifications that helps to establish a confidence level about the implemented model and reveals discrepancies between observed and normal or expected behavior.

Other works focus on the migration of traditional information systems to MAS. In this line, *Chen et al.* [6] developed an evolutionary approach to reengineer legacy systems into agent-based Web services.

Studies about reverse engineering of MAS are often *ad hoc* techniques without empirical validation. As a result, the above mentioned methods are difficult to be reused with different MAS, and, therefore, their applicability to the industry cannot be ensured. To mitigate this threat, *Regli et al.* [13] made an effort to provide a reference model for agent-based systems. The purpose of a reference model is to provide a common conceptual basis for comparing systems and driving the development of software architectures and other standards. As part of this study, the team applied software reverse engineering techniques to perform static and dynamic analysis of operational agent-based systems, which enabled identification of key common concepts across different agent frameworks. Similarly, *Sauvage* [15] proposed a set of design patterns to collect and express agent concepts, which can be adapted to various MAS developing challenges. The agent patterns cover all the development stages, from analysis to implementation, including reengineering through *antipatterns*.

Despite recent software engineering paradigms like Model-Driven Development (MDD) could help with the formalization and automation of reverse engineering techniques, it is rarely incorporated to current approaches. An exception is the approach proposed by *Warwas et al.* [16] which presents a model-driven reverse engineering approach for lifting the underlying design of implemented MAS to a platform independent level. For this purpose the proposal provides conceptual mappings from the platform to a platform independent modeling language. The extracted structures can be re-used as blue print for solving similar problems on similar execution platforms. MDD has been used as well for reverse engineering associated to modernization of software systems in the MOMOCS project [7], but without considering MAS.

In summary, current approaches address testing and agent verification or focus on the analysis and measurement of MAS. So far, there are no mechanisms to abstract code-level MAS to high-level design models.

3 Reverse Engineering of JADE-based Systems

REAGENT supports reverse engineering of JADE-based MAS into a high-level design that is represented using the INGENIAS agent-oriented modeling

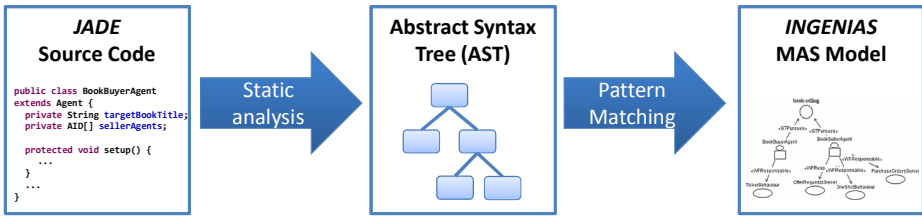


Fig. 1. The REAGENT overview: stages and artifacts involved

language. This is done in two steps, which are illustrated in Fig. 1: a static analysis of source code followed by a pattern matching recognition, which will provide a MAS model.

3.1 Static Analysis

The first stage of REAGENT employs a static analysis of JADE-based MAS implementation. Static analysis consists of the syntactic analysis of source code line by line. Static analysis can be easily carried out by means of parsers that recognize certain syntax depicted by metamodels of grammars. In comparison to dynamic analysis, which considers runtime information, static analysis is not able to take into account actual values for program variables. Nevertheless, the static analysis is more exhaustive than the dynamic analysis, since static analysis considers all parts of source code while the dynamic analysis takes into account only those parts that were reached in a particular execution. Additionally, dynamic analysis is more time-consuming than static one due to dynamic analysis firstly records traces during execution and secondly examines traces to extract relevant information.

The static analysis in this stage takes as input JADE-based source code, which is, in Java source code. REAGENT builds an abstract syntax tree (AST) using the analysis of JADE source code. AST is an intermediate representation of source code in a tree structure which can be used to easily recognize some particular structures in the pattern matching stage (see Fig. 1). The static analysis was implemented as an extension of the *Java 1.5* parser developed with JavaCC [11], a parser generator tool.


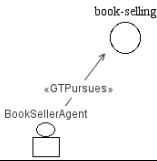

3.2 Pattern Matching

Having obtained an AST from source code, REAGENT applies a pattern matching technique for generating agent models according to the INGENIAS methodology. Pattern matching checks a perceived sequence or tree structure of tokens from the AST for the presence of the constituents of a certain pattern. Pattern matching includes the definition of the respective output structures which will be built in the outgoing agent model.

REAGENT provides three main patterns: agents, goals and tasks. Table 1 shows an example of the result of pattern matching applied to the *BookSellerAgent* class, which corresponds to one of the benchmark programs used in the case study (cf. Section 4). These patterns correspond with core concepts of the INGENIAS methodology [12].

INGENIAS provides a notation for modeling MAS and a well-defined collection of activities to guide the development of an MAS in the tasks of analysis, design, verification, and code generation, supported by an integrated set of tools. An MAS in INGENIAS is specified from different viewpoints: organization, agent, goals and tasks, interaction, and environment. REAGENT focuses on the agent model, which depicts the agents of an MAS, their goals and the tasks for which they are responsible.

Table 1. A running example for the pattern reconginition [2]

	JADE Code	INGENIAS Agent Model
Agents	<pre>public class BookSellerAgent extends Agent { ... }</pre>	
Goals	<pre>public class BookSellerAgent extends Agent { ... DFAgentDescription dfd = new DFAgentDescription(); dfd.setName(getAID()); ServiceDescription sd = new ServiceDescription(); sd.setType("book-selling"); }</pre>	
Tasks	<pre>public class BookSellerAgent extends Agent { public void updateCatalogue(final String title, final int price) { addBehaviour(new OneShotBehaviour() { public void action() { ... } }); } }</pre>	

Agents attempts to identify Java classes that extend the *jade.core.Agent* class. The list of the *extends* clauses of a class in an AST is provided by the path *CompilationUnit > PackageDeclaration > TypeDeclaration > ClassOrInterfaceDeclaration > ExtendsList*. This pattern builds an Agent element in the agent model for each recognized agent class (see Table 1).

Goals searches for *DFAgentDescription* instances which are added to the yellow pages with a concrete *ServiceDescription*. The service description label is used to provide the name of the goal, which is pursued by the respective agent (see Table 1).

Tasks search for different behaviors of agent classes to figure out the tasks of each agent element in the INGENIAS agent model. This pattern searches for calls to the *addBehaviour* method of the *jade.core.Agent* class, which adds a subclass of the *jade.core.behaviours.Behaviour* class. A task is built for each recognized behavior.

4 Empirical Validation

This section presents a case study by using the example JADE programs provided in [2], which are considered as benchmark programs in this context. The study applies the proposed technique to these benchmarks to obtain design MAS models. The case study was conducted according to the formal protocol proposed by *Runeson et al.* [14] for designing, conducting and reporting case studies in the software engineering field.

The following subsections show the adaptation to MAS of the stages of this protocol: design, execution procedure, data collection, and analysis and interpretation.

4.1 Design

The *object of study* is the reverse engineering technique proposed in Section 3, while the *purpose of this study* is the assessment of specific properties of the proposed technique such as their effectiveness and efficiency. Taking into account the object and purpose of the study, two main research questions are defined. Q1 is related to the effectiveness evaluation, whilst Q2 is associated with the efficiency assessment.

Q1. Is the technique able to retrieve accurate and complete agent design model from existing MAS?

Q2. Is the technique scalable to larger and more complex MAS?

Concerning the formal design, the study is considered a multi-case study since it focuses on various agent programs. The study also follows a *holistic* design since it is applied to each case as a whole, and does not consider several analysis subunits. As a result, MAS programs are considered as the independent variable.

In order to quantitatively answer the research questions, various measures are considered as dependent variables. On the one hand, to evaluate effectiveness according to Q1, recall and precision are used. Recall is a measure of completeness whereas precision can be seen as a measure of exactness or fidelity. Recall (1) represents the number of relevant elements retrieved as a function of the total of relevant elements (retrieved and not retrieved) depicting the MAS through an agent model. Precision (2) represents the number of relevant elements retrieved within the set of retrieved elements in an agent model. Together with recall and precision, F-measure (3) is considered for aggregating precision and recall values into a sole value by means of a harmonic mean. F-measure is necessary, since there is an inverse relationship between precision and recall. On the other hand, the study measures the time spent on the execution of each transformation so that Q2 can be answered.

Apart from these measures, other measures to characterize the input of MAS programs are used, like the number of Java files, lines of source code (LOC) and cyclomatic complexity (4). The cyclomatic complexity is the number of linearly independent paths through the source code, and is related to the intricacy of source code.

$$P = \frac{|{\text{relevant recovered tasks}}|}{|{\text{recovered tasks}}|} \quad (1)$$

$$R = \frac{|{\text{relevant recovered tasks}}|}{|{\text{relevant tasks}}|} \quad (2)$$

$$F = \frac{2 \cdot P \cdot R}{P + R} \quad (3)$$

$$CC = |{\text{call graph edges}}| - |{\text{call graph nodes}}| + 2 \cdot |{\text{connected components}}| \quad (4)$$

4.2 Execution Procedure

The case study is executed in a finite set of steps, which are partially supported by the REAGENT tool, a plug-in to the INGENIAS Development Kit (IDK) to retrieve MAS models from JADE code.

1. Source code of 19 MAS provided as examples in textbook [2] are considered. These examples are: *base64*, *behaviours*, *bookTrading*, *content*, *hello*, *inprocess*, *ja-deJessProtege*, *jess*, *messaging*, *mobile*, *O2AInterface*, *ontology*, *party*, *pingAgent*, *protocols*, *service*, *thanksAgent*, *topic* and *yellowPages*. Benchmarks are manually analyzed to obtain a reference agent model to be used as the *gold standard*.
2. REAGENT v1.0 is applied to each benchmark obtaining an agent model that is graphically visualized with the IDK tool. REAGENT was executed in a computer with a 2.66 GHz dual processor and 4.0 GB RAM.
3. The first sketches of agent models are then analyzed and compared with the *gold standard* to obtain base metrics for computing precision and recall (e.g., the number of retrieved non-relevant elements or non-retrieved relevant elements).
4. Having computed all the mentioned measures, collected data are analyzed and interpreted to draw conclusions in order to answer the research questions.

4.3 Data Collection

Table 2 summarizes all the measured values collected during the execution of study for each system. Table 2 shows for each (i) mentioned benchmark, (ii) the number of Java files, (iii) the number of lines of source code, (iv) the cyclomatic complexity average, and (v) the execution time in milliseconds. For each respective agent model, Table 2 provides the number of different INGENIAS elements such as (vi) agents,

Table 2. Data collected during the case study execution

MAS ID	# Files	LOC	C. Complexity	Exec. Time (ms)	# Agents	# Tasks	# Goals	# WFResponsible	# GTPursues	# Rt.	# Rt. Rv.	# Rt. NRv.	# NRt. Rv.	Recall	Precision	F-Measure
1	3	292	3.0	2321	2	0	2	0	1	5	4	1	1	0.80	0.80	0.80
2	4	370	1.5	2858	4	12	0	12	0	28	20	8	0	1.00	0.71	0.83
3	3	456	3.3	2779	2	4	1	4	2	13	13	0	0	1.00	1.00	1.00
4	41	3222	1.3	3027	4	10	0	10	0	24	15	9	10	0.60	0.63	0.61
5	1	44	1.0	2149	1	0	0	0	0	1	1	0	2	0.33	1.00	0.50
6	2	346	2.2	2435	1	1	0	1	0	3	3	0	0	1.00	1.00	1.00
7	7	154	1.7	2080	1	0	0	0	0	1	1	0	3	0.25	1.00	0.40
8	4	677	3.8	2320	1	1	0	1	0	3	3	0	0	1.00	1.00	1.00
9	4	245	2.4	2577	4	3	0	3	0	10	10	0	0	1.00	1.00	1.00
10	6	938	1.8	2455	1	3	0	3	0	7	7	0	0	1.00	1.00	1.00
11	4	185	1.4	2308	1	0	0	0	0	1	1	0	0	1.00	1.00	1.00
12	15	1390	1.7	2764	4	6	0	6	0	16	12	4	2	0.86	0.75	0.80
13	3	994	2.3	2273	2	0	0	0	0	2	2	0	8	0.20	1.00	0.33
14	1	106	2.7	2214	0	1	1	0	0	2	2	0	3	0.40	1.00	0.57
15	5	572	2.9	2671	5	5	0	5	0	15	15	0	0	1.00	1.00	1.00
16	1	62	1.7	1895	0	0	0	0	0	0	0	0	0	Nan	Nan	Nan
17	1	218	9.0	2368	1	1	0	1	0	3	3	0	0	1.00	1.00	1.00
18	2	123	2.5	2278	2	0	0	0	0	2	2	0	4	0.33	1.00	0.50
19	4	347	4.3	2521	3	1	1	1	1	7	7	0	2	0.78	1.00	0.88
Total	111	10741	-	46293	39	48	5	47	4	143	121	22	35	-	-	-
Mean	6	565	2.7	2436	2	3	0	2	0	8	6	1	2	0.75	0.94	0.79
S. Des.	9	736	1.8	287	2	4	1	4	1	8	6	3	3	0.31	0.12	0.24

(vii) tasks, and (viii) goals, as well as relationships such as (ix) *WFResponsible* between agents and tasks, and (x) *GTPursues* between agents and goals. After that, the evaluation of base measures to compute precision and recall are also provided such as the number of (xi) total retrieved elements, (xii) retrieved relevant, (xiii) retrieved non-relevant, and (xiv) non-retrieved relevant elements. Finally, Table 2 provides (xv) precision, (xvi) recall, and (xvii) F-measure.

4.4 Analysis and Interpretation

After the execution of the study, collected data is analyzed to answer research questions. Regarding the effectiveness question (Q1) the analysis of precision and recall is shown in Fig. 2. The recall mean is 0.75 while the precision mean is 0.94. Firstly, a higher precision shows that almost all the retrieved elements in agent models correspond to expected elements. Secondly, a lower recall indicates that there are various actual elements that were not retrieved. However, recall distribution has a high standard deviation, which means that the mentioned problem happens only in some benchmarks (see cases 5, 7, 13, 14 and 18 in Table 2). Anyway, the F-measure is 0.79 on average. This F-measure value is above most reference values of similar works from the literature [9, 17], which define 0.6 as a threshold of applicability.

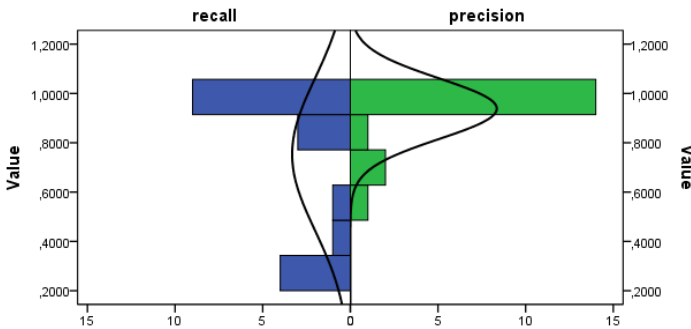


Fig. 2. Comparison of the recall and precision distribution

Despite precision and recall proved to be appropriate, the effect of complexity to these values is also analyzed by means of the anova test. This test analyzes the variance of various sub-samples with respect to a factor. Thus, the null hypothesis is $H_0: \mu_1 = \mu_2 = \mu_n$, while the alternative hypothesis means that there is a significant difference between the means of sub-samples, i.e., $H_1: \mu_1 \neq \mu_2 \neq \mu_n$. In this study the factor of the anova test is complexity, which was differentiated between low, medium and high regarding percentiles 1/3 and 2/3. Table 3 provides the results of the three anova tests for recall, precision and F-measure. The null hypotheses cannot be rejected in all the cases.

Despite the anova test result is not significant, the effect size of each distribution can be analyzed. Recall improves for high complexity (with a negative sign) and is almost the same for medium complexity. Similarly, precision improves for both

medium and high complexity. In general, F-measure follows the same trend. These results imply that complex source code provides better results since there are less missing agent elements and relationships than in toy agent programs with a low cyclo-matic complexity.

In addition to Q1, the efficiency of this approach is analyzed regarding Q2. The scalability of REAGENT is tested by a linear regression model (see Fig. 3), which considers the execution time for each benchmark as a dependent variable, and the size of the agent model as the independent variable. Fig. 3 shows the regression line ($y = 31.963 \cdot x - 2195.9$), which presents a positive linear relationship with $R^2=0.8244$. The correlation coefficient R^2 (between -1 and 1) is the degree to which the real values of the dependent variable are close to the predicted values. The R^2 value obtained is high, and very close to 1, thus the proposed linear regression model is suitable for explaining the data obtained in this study, i.e., there is no quadratic or exponential relationship between the clustering time and the size. The increase in time for larger MAS will consequently be linear, and this time may be assumable. The conclusion is that Q2 can be answered positively.

Table 3. ANOVA Test results for effectiveness measures

Measure	Complexity	Quadratic Mean	F-value	Effect Size	p-value
Recall	Low	0.244	2.331	0.472	0.129
	Medium			0.084	
	High			-0.848	
Precision	Low	0.120	2.2267	0.519	0.136
	Medium			-0.373	
	High			-0.431	
F-Measure	Low	0.226	3.262	0.591	0.065
	Medium			0.035	
	High			-0.931	

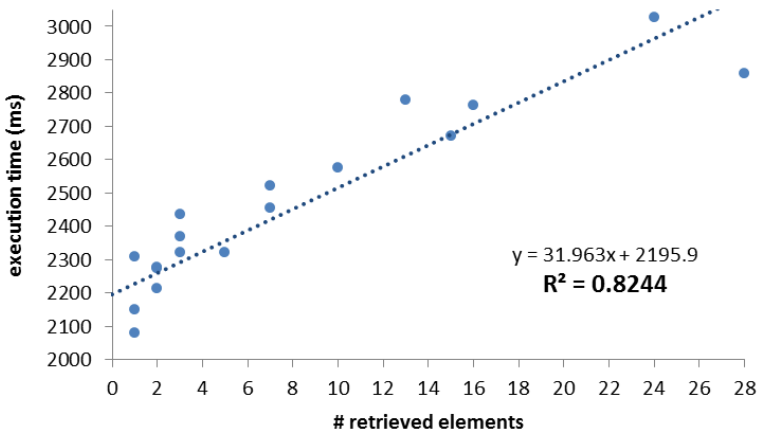


Fig. 3. Linear regression model for the agent model size against execution time

5 Conclusions and Future Work

This paper has presented REAGENT, a reverse engineering technique for retrieving agent design models from legacy Jade source code. The design models are represented according to the INGENIAS methodology. The abstraction of source code toward design models is made following INGENIAS for several reasons: (i) it has been validated in real applications; (ii) it provides useful tools supporting the analysis, design and code generation of based-agent software, (iii) it uses a model-driven approach that facilitates the independence from the implementation platform, and finally (iv) it is not oriented towards a particular agent platform.

A case study with 19 multi-agent systems has been conducted to empirically demonstrate the applicability of REAGENT. The results of the study indicate that REAGENT is able to retrieve accurate and complete agent design models from existing source code. However, particular legacy knowledge is still missing due to the characteristic semantic loss common to all the reverse engineering technique. Despite this fact, REAGENT is able to provide a first sketch of design models, which is obtained in a less error-prone and less time-consuming way than manual modeling from scratch.

The future work will focus on the improvement of REAGENT by incorporating further patterns for considering additional INGENIAS models concerning agent interactions, organization and environment, among others. Additionally, other implementation target platforms will be considered. Finally, a case study involving a large industrial MAS is also expected to be conducted in order to obtain strength results.

Acknowledgements. This work was supported by the FPU Spanish Program and the R&D projects MAGO/PEGASO ([TIN2009-13718-C02-01]; GEODAS-BC (TIN2012-37493-C03-01); and Social Ambient Assisting Living - Methods (TIN2011-28335-C02-01).

References

1. Adra, S.F., Kiran, M., McMinn, P., Walkinshaw, N.: A multiobjective optimisation approach for the dynamic inference and refinement of agent-based model specifications. In: IEEE Congress on Evolutionary Computation (CEC) (2011)
2. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology), 300 p. Wiley (2007)
3. Bosse, T., Lam, D.N., Barber, K.S.: Tools for analyzing intelligent agent systems. *Web Intelli. and Agent Sys.* 6(4), 355–371 (2008)
4. Canfora, G., Di Penta, M.: New Frontiers of Reverse Engineering. In: Future of Software Engineering (FOSE 2007). IEEE Computer Society (2007)
5. Canfora, G., Di Penta, M., Cerulo, L.: Achievements and challenges in software reverse engineering. *Commun. ACM* 54(4), 142–151 (2011)
6. Chen, F., Yang, H., Guo, H., Xu, B.: Agentification for web services (2004)
7. Fuentes-Fernández, R., Pavón, J., Garijo, F.: A model-driven process for the modernization of component-based systems. *Science of Computer Programming* 77(3), 247–269 (2012)

8. Hirst, A.J.: Reverse engineering of Soar agents, pp. 72–73 (2000)
9. Lucrédio, D., de M. Fortes, R.P., Whittle, J.: MOOGLE: A Model Search Engine. In: Czarnecki, K., Ober, I., Bruel, J.-M., Uhl, A., Völter, M. (eds.) MoDELS 2008. LNCS, vol. 5301, pp. 296–310. Springer, Heidelberg (2008)
10. Moreno, J., López, L.: Using Techniques Based on Natural Language in the Development Process of Multiagent Systems. In: Corchado, J.M., Rodríguez, S., Llinas, J., Molina, J.M. (eds.) DCAI 2008, vol. 50, pp. 269–273. Springer, Heidelberg (2009)
11. Open Source Initiative, Java 1.5 grammar for JavaCC (2009),
<https://javacc.dev.java.net/files/documents/17/3131/Java1.5.zip>
12. Pavon, J., Gomez-Sanz, J.J., Fuentes, R.: The INGENIAS Methodology and Tools. In *Agent-Oriented Methodologies*, pp. 236–276. IGI Global (2005)
13. Regli, W.C., Mayk, I., Dugan, C.J., Kopena, J.B., Lass, R.N., Modi, P.J., Mongan, W.M., Salvage, J.K., Sultanik, E.A.: Development and specification of a reference model for agent-based systems. *Trans. Sys. Man Cyber. Part C* 39(5), 572–596 (2009)
14. Runeson, P., Höst, M.: Guidelines for Conducting and Reporting Case Study Research in Software Engineering. *Empirical Softw. Eng.* 14(2), 131–164 (2009)
15. Sauvage, S.: Design patterns for multiagent systems design. In: Monroy, R., Arroyo-Figueroa, G., Sucar, L.E., Sossa, H. (eds.) MICAI 2004. LNCS (LNAI), vol. 2972, pp. 352–361. Springer, Heidelberg (2004)
16. Warwas, S., Klusch, M.: Making Multiagent System Designs Reusable: A Model-Driven Approach. In: *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, (2), pp. 101–108. IEEE C. S. (2011)
17. Ye, Y., Fischer, G.: Supporting reuse by delivering task-relevant and personalized information. In: *24th Int. Conference on Software Engineering*, pp. 513–523. ACM (2002)