Lecture Notes in Artificial Intelligence 2977

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Giovanna Di Marzo Serugendo
Anthony Karageorgos Omer F. Rana
Franco Zambonelli (Eds.)

Engineering Self-Organising Systems

Nature-Inspired Approaches to Software Engineering

Springer

Volume Editors

Giovanna Di Marzo Serugendo
University of Geneva
Centre Universitaire d'Informatique, 1211 Geneva 4, Switzerland
E-mail: Giovanna.Dimarzo@cui.unige.ch

Anthony Karageorgos
University of Thessaly
Department of Computer and Communication Engineering
37 Glavani, 28th October Str., Deligiorgi Building
4th floor, room D3/4, 382 21 Volos, Greece
E-mail: karageorgos@computer.org

Omer F. Rana
Cardiff University
School of Computer Science
5 The Parade, P.O.Box 916, Cardiff CF24 3XF, UK
E-mail: o.f.rana@cs.cardiff.ac.uk

Franco Zambonelli
Università di Modena e Reggio Emilia
Dipartimento di Scienze e Metodi dell'Ingegneria
42100 Reggio Emilia, Italy
E-mail: franco.zambonelli@unimore.it

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): I.2.11, I.2, C.2.4, F.1.1, D.2

ISSN 0302-9743
ISBN 3-540-21201-9 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by DA-TeX Gerd Blumenstein
Printed on acid-free paper SPIN: 10989791 06/3142 5 4 3 2 1 0

Preface

The spread of the Internet, mobile communications and the evolution of traditional market models, such as the proliferation of e-commerce, has led to much existing IT infrastructure operating as a global dynamic system. It is therefore important that software that operates over this infrastructure adapt accordingly, to provide highly customised services to a huge user population. New maintenance requirements have to be met, for example software that cannot be stopped must still evolve. Furthermore, software must cope with changes in interconnectivity requirements, such as network topology workload fluctuations, mobility of devices, node failures, and interoperation of unreliable, possibly erroneous distributed code.

As information handling systems get more complex, it becomes more difficult to manage them using traditional approaches based on centralised and predefined control mechanisms. Over recent years there has been a significant increase in interest in taking inspiration from biology, the physical world, chemistry, or social systems to more effectively manage such systems – generally based on the concept of Self-organisation. This gave rise to Self-organising Applications (SOA) – generally applications that undertake problem solving (in a particular application domain) based on distributed interactions between uncoupled system components. Typical examples of such distributed interactions in SOA include systems that reproduce behaviours of social insects or multicellular organisms. Another example is multiagent systems that aim to reproduce behaviours found in human societies – thereby providing a useful paradigm for managing large, distributed information handling systems. In multiagent systems self-organising behaviour results from the autonomous behaviour of system components and the locality of their interactions. The key characteristic of all these applications is their ability to achieve complex collective tasks with relatively simple individual behaviours.

There is currently a consensus that modern applications can gain in robustness, management, and simplicity if they are developed according to the principles of self-organisation (which one finds in nature). The central question related to the engineering of SOA is how to specify the local behaviours of individual components to achieve the desired (self-organising) global system behaviour. This involves, in particular, defining a global goal, designing local behaviours and interactions that will result in the desired global behaviour. Subsequently, it is necessary to monitor system execution to verify the emergence of the correct system behaviour, and the achievement of the global goal. This is a difficult exercise because the global system behaviour is not directly predictable as the "sum" of the local behaviours. It is also important to note that self-organisation is not useful in all instances – as it may lead to "parasitic" and unwanted behaviours. The need to monitor the system as it evolves from some initial state is therefore an important requirement in many applications that make use of self-organising principles.

Traditional software engineering methodologies also make it possible to define a global behaviour only as a closed function of the behaviour of the various parts of the system. They involve techniques based on assumptions, such as interfaces fixed at design time and statically established ontologies, which therefore makes them insufficient for engineering self-organisation. Furthermore, traditional practices in multiagent systems have introduced basic techniques for autonomously interacting or retrieving information, such as agent coordination, service description, and ontology-based interaction. However, these techniques rely on preprogrammed interaction patterns, preventing large-scale adaptation to unexpected environmental changes.

Current engineering practices that directly address self-organisation primarily involve designing distributed algorithms according to natural life metaphors, such as the social insect metaphor and the immune system. More recently, appropriate interaction mechanisms and middleware technologies specifically targeting the development of self-organising applications have been introduced. However, many issues such as verification and systematic methodologies covering the whole engineering life cycle still remain open.

This book intends to serve as a reference and starting point for establishing the field of *Engineering Self-Organising Applications*. It comprises papers presented at the Engineering Self-Organising Applications (ESOA 2003) workshop (held at the Autonomous Agents and Multi-Agents Systems conference (AAMAS 2003) in Melbourne in July 2003), and selected invited papers from leading contributors in the self-organisation field.

The book presents the current state-of-the-art perspective on the field. Part I contains several papers related to self-organising applications: Di Marzo Serugendo et al. review natural self-organising systems and their interaction mechanisms and discuss several examples of artificial self-organising applications; Brueckner et al. present SWARMing principles applied to mobile ad hoc networks; Thompson then discusses the use of self-organising systems for agent-based implementations.

In Part II, translations from a diverse collection of natural-life metaphors are presented. Naghpal presents self-organising principles from multicellular organisms and their application to multiagent systems. Tateson et al. apply self-organising behaviour of cells in a developing fruit fly to wireless communication networks. Guo et al. present a genetic algorithm for designing self-assembling objects. This is followed by two papers inspired from ant-based behaviour properties such as indirect communication of interacting entities via the environment (stigmetry): Handl et al. apply ant mechanisms in data set clustering, and Hadeli et al. in manufacturing control. Foukia et al. combine ant-based behaviour and the immune system metaphor to build an intrusion detection and response system in a computer network. Ulieru et al. design logistics infrastructures based on the holonic structure of the Universe. Subsequently, Babanov et al. take their inspiration from evolutionary computation to build an additional layer on top of heterogeneous strategies in electronic markets.

Part III describes artificial self-organisation mechanisms specifically introduced for computer systems applications. Capera et al. propose an adaptive multiagent system theory for providing self-organising behaviour in mechanical design. Hales et al. propose the notion of tags, a marking attached to agents and observable by other agents, as an interaction mechanism for realising self-organisation. Yolum et al. then establish the notion of trust as a self-organising mechanism in referral networks.

In Part IV, self-organisation specific to coordination models is discussed by Menezes et al. in the case of the Linda-based systems. Along this line, Mamei et al. describe the TOTA coordination environment propagating tuples according to a propagation rule.

Part V concludes the book with methods and tools for engineering emergent behaviour. Airiau et al. discuss the COllective INtelligence framework for accessing shared resources among several users. Jelasity et al. describe simple components that can be combined to implement complex functions. Finally, Bernon et al. present the Adelfe methodology that guides a programmer during the design of SOA.

December, 2003

Giovanna Di Marzo Serugendo,
Anthony Karageorgos,
Omer F. Rana,
Franco Zambonelli

Programme Committee

Karl Aberer	EPFL, Switzerland
Yaneer Bar-Yam	New England Complex Systems Institute, USA
Rene te Boekhorst	University of Hertfordshire, UK
Sven Brueckner	Altarum, USA
Marco Dorigo	IRIDIA, Université Libre de Bruxelles, Belgium
Noria Foukia	University of Geneva, Switzerland
Nigel Gilbert	University of Surrey, UK
Maria Gini	University of Minnesota, USA
Salima Hassas	University of Lyon, France
Jeff Kephart	IBM Thomas J. Watson Research Center, USA
Mark Klein	MIT Sloan School of Management, USA
Ghita Kouadri Mostefaoui	University of Fribourg, Switzerland
Soraya Kouadri Mostefaoui	University of Fribourg, Switzerland
Marco Mamei	Università di Modena e Reggio Emilia, Italy
Philippe Massonet	CETIC, Belgium
Evangelos Milios	Dalhousie University, Canada
Radhika Nagpal	MIT Artificial Intelligence Lab, MIT, USA
N.C. Narendra	Hewlett-Packard, India
Andrea Omicini	University of Bologna, Italy
H. Van Dyke Parunak	Altarum, USA
Daniel Polani	University of Hertfordshire, UK
David Servat	Laboratoire d'Informatique Appliquée, LIP6, Paris, France
Simon Thompson	BT Exact Technologies, UK
Mihaela Ulieru	University of Calgary, Canada
Chris Van Aart	University of Amsterdam, Netherlands
Tom Wagner	Honeywell Laboratories, USA
David Wolpert	NASA Ames Research Center, USA

We are grateful to the Programme Committee of the ESOA 2003 workshop for returning their reviews on time, and for offering useful suggestions on improving the workshop. All papers submitted to the workshop were reviewed by a minimum of two members of the programme committee.

Table of Contents

Part I: Applications

Part II: Natural Metaphors

Multi-cells and Genetic Algorithms

Stigmergy

Atoms and Evolution

Part III: Artificial Interaction Mechanisms

Part IV: Middleware

Part V: Methods and Tools