Paolo Petta
Jörg P. Müller
Matthias Klusch
Michael Georgeff (Eds.)

LNAI 4687

Multiagent System Technologies

5th German Conference, MATES 2007
Leipzig, Germany, September 2007
Proceedings

Springer

Paolo Petta Jörg P. Müller
Matthias Klusch Michael Georgeff (Eds.)

Multiagent System Technologies

5th German Conference, MATES 2007
Leipzig, Germany, September 24-26, 2007
Proceedings

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Paolo Petta
Medizinische Universität Wien
Freyung 6/II, 1010 Vienna, Austria
E-mail: Paolo.Petta@MeduniWien.ac.at

Jörg P. Müller
Technische Universität Clausthal
Julius-Albert-Str. 4, 38678 Clausthal-Zellerfeld, Germany
E-mail: joerg.mueller@tu-clausthal.de

Matthias Klusch
Deutsches Forschungsinstitut für Künstliche Intelligenz (DFKI)
Stuhlsatzenhausweg 3, 66123 Saarbruecken, Germany
E-mail: klusch@dfki.de

Michael Georgeff
Monash University
Locked Bag 29, Clayton, Victoria 3168, Australia
E-mail: michael.georgeff@med.monash.edu.au

Library of Congress Control Number: 2007934801

CR Subject Classification (1998): I.2.11, I.2, C.2.4, D.1.12, D.1.3, J.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-540-74948-9 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-74948-6 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12123544 06/3180 5 4 3 2 1 0

Lecture Notes in Artificial Intelligence 4687

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Preface

The German conference on Multi-Agent System Technologies (MATES) provides an interdisciplinary forum for researchers, users, and developers to present and discuss the latest advances in research work as well as prototyped or fielded systems of intelligent agents and multi-agent systems. The conference aims to promote theory and applications and covers the whole range of agent- and multi-agent technologies.

For the fifth time, the German special interest group on Distributed Artificial Intelligence in cooperation with the Steering Committee of MATES organized this international conference. Building on the four successful predecessors in 2003, 2004, 2005, and 2006, MATES 2007 took place September 24–26, 2007 under the umbrella of the SABRE (Software, Agents, and Services for Business, Research, and E-Sciences) event organized by the University of Leipzig.

Situated in the lively scene of agent-based computing in Europe that is exemplified by federated events such as Durham Agents'007; SABRE itself —that also included the Central and Eastern European Conference on Multi-Agent Systems (CEEMAS); and the subsequent European Conference on Complex Systems (ECCS 2007) in Dresden, MATES 2007 not only succeeded in attracting 27 submissions, out of which 17 could be accepted for presentation and discussion, but also in holding an edition of the doctoral mentoring programme, an important occasion for both students and established researchers to interact and discuss scientific and managerial aspects of activities. The programme of MATES 2007 was rounded off with invited presentations by the distinguished speakers Michael Georgeff and Rafael Bordini.

Our thanks go to the Programme Committee for their diligent, careful, and constructive work; the local organizers of SABRE for their constant support; and foremost to all authors of submitted papers: the present selection stands to testify the important contribution of the MATES conference series to the rich international agent-oriented systems research landscape.

July 2007

Paolo Petta
Jörg P. Müller
Matthias Klusch
Michael Georgeff

Organization

General Co-chairs

Matthias Klusch	DFKI, Germany
Michael Georgeff	Monash University, Australia

Program Co-chairs

Paolo Petta	Medical University of Vienna and OFAI, Austria
Jörg P. Müller	TU Clausthal, Germany

Doctoral Consortium Chair

Franziska Klügl	Universität Würzburg, Germany

Steering Committee

Hans-Dieter Burkhard	Humboldt-Universität zu Berlin, Germany
Stefan Kirn	Universität Hohenheim, Germany
Matthias Klusch	DFKI, Germany
Jörg P. Müller	TU Clausthal, Germany
Rainer Unland	Universität Duisburg-Essen, Germany
Gerhard Weiss	SCCH Hagenberg, Austria

Program Committee

Bernhard Bauer	Universität Augsburg, Germany
Federico Bergenti	Università degli Studi di Parma, Italy
Lars Braubach	Universität Hamburg, Germany
Hans-Dieter Burkhard	Humboldt-Universität zu Berlin, Germany
Cristiano Castelfranchi	Università degli Studi di Siena and CNR Roma, Italy
Rosaria Conte	CNR Roma, Italy
Hans Czap	Universität Trier, Germany
Mehdi Dastani	Universiteit Utrecht, The Netherlands
Jörg Denzinger	University of Calgary, Canada
Jürgen Dix	TU Clausthal, Germany
Torsten Eymann	Universität Bayreuth, Germany
Klaus Fischer	DFKI GmbH, Germany

Jean-Pierre Georgé	Université Paul Sabatier, France
Paolo Giorgini	Università degli Studi di Trento, Italy
Rune Gustavsson	Blekinge Institute of Technology, Sweden
Heikki Helin	TeliaSonera, Finland
Wiebe van der Hoek	University of Liverpool, UK
Stefan Kirn	Universität Hohenheim, Germany
Franziska Klügl	Universität Würzburg, Germany
Ryszard Kowalczyk	Swinburne University of Technology, Australia
Daniel Kudenko	University of York, UK
Jürgen Lind	iteratec GmbH, Germany
Gabriela Lindemann	Humboldt-Universität zu Berlin, Germany
Jiming Liu	Baptist University, Hong Kong
Stefano Lodi	Università di Bologna, Italy
Beatriz López	Universitat de Girona, Spain
Volker Nissen	TU Ilmenau, Germany
James Odell	Agentis Software, USA
Andrea Omicini	Università di Bologna, Italy
Sascha Ossowski	Universidad Rey Juan Carlos, Spain
Stefan Poslad	Queen Mary University of London, UK
Von-Wun Soo	National Tsing-hua University, Taiwan
Steffen Staab	Universität Koblenz-Landau, Germany
Ingo Timm	Universität Frankfurt am Main, Germany
Robert Tolksdorf	Freie Universität Berlin, Germany
Adelinde Uhrmacher	Universität Rostock, Germany
Rainer Unland	Universität Duisburg-Essen, Germany
Liászló Zsolt Varga	MTA SZTAKI, Hungary
Danny Weyns	K.U. Leuven, The Netherlands
Cees Witteveen	TU Delft, The Netherlands
Michael Wooldridge	University of Liverpool, UK

Auxiliary Referees

Tina Balke	Alexander Kubias	Fernando Silva Parreiras
Ralf Berger	Ingo Müller	Diemo Urbig
Javesh Boodnah	Michele Piunti	Jian Feng Zhang

Lecture Notes in Artificial Intelligence (LNAI)

Vol. 4733: R. Basili, M.T. Pazienza (Eds.), AI*IA 2007: Artificial Intelligence and Human-Oriented Computing. XVII, 858 pages. 2007.

Vol. 4720: B. Konev, F. Wolter (Eds.), Frontiers of Combining Systems. X, 283 pages. 2007.

Vol. 4702: J.N. Kok, J. Koronacki, R. López de Mántaras, S. Matwin, D. Mladenič, A. Skowron (Eds.), Knowledge Discovery in Databases: PKDD 2007. XXIV, 640 pages. 2007.

Vol. 4701: J.N. Kok, J. Koronacki, R.L.d. Mantaras, S. Matwin, D. Mladenič, A. Skowron (Eds.), Machine Learning: ECML 2007. XXII, 809 pages. 2007.

Vol. 4694: B. Apolloni, R.J. Howlett, L. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part III. XXIX, 1126 pages. 2007.

Vol. 4693: B. Apolloni, R.J. Howlett, L. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part II. XXXII, 1380 pages. 2007.

Vol. 4692: B. Apolloni, R.J. Howlett, L. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part I. LV, 882 pages. 2007.

Vol. 4687: P. Petta, J.P. Müller, M. Klusch, M. Georgeff (Eds.), Multiagent System Technologies. X, 207 pages. 2007.

Vol. 4682: D.-S. Huang, L. Heutte, M. Loog (Eds.), Advanced Intelligent Computing Theories and Applications. XXVII, 1373 pages. 2007.

Vol. 4676: M. Klusch, K. Hindriks, M.P. Papazoglou, L. Sterling (Eds.), Cooperative Information Agents XI. XI, 361 pages. 2007.

Vol. 4667: J. Hertzberg, M. Beetz, R. Englert (Eds.), KI 2007: Advances in Artificial Intelligence. IX, 516 pages. 2007.

Vol. 4660: S. Džeroski, J. Todorovski (Eds.), Computational Discovery of Scientific Knowledge. X, 327 pages. 2007.

Vol. 4659: V. Mařík, V. Vyatkin, A.W. Colombo (Eds.), Holonic and Multi-Agent Systems for Manufacturing. VIII, 456 pages. 2007.

Vol. 4651: F. Azevedo, P. Barahona, F. Fages, F. Rossi (Eds.), Recent Advances in Constraints. VIII, 185 pages. 2007.

Vol. 4648: F. Almeida e Costa, L.M. Rocha, E. Costa, I. Harvey, A. Coutinho (Eds.), Advances in Artificial Life. XVIII, 1215 pages. 2007.

Vol. 4635: B. Kokinov, D.C. Richardson, T.R. Roth-Berghofer, L. Vieu (Eds.), Modeling and Using Context. XIV, 574 pages. 2007.

Vol. 4632: R. Alhajj, H. Gao, X. Li, J. Li, O.R. Zaïane (Eds.), Advanced Data Mining and Applications. XV, 634 pages. 2007.

Vol. 4629: V. Matoušek, P. Mautner (Eds.), Text, Speech and Dialogue. XVII, 663 pages. 2007.

Vol. 4626: R.O. Weber, M.M. Richter (Eds.), Case-Based Reasoning Research and Development. XIII, 534 pages. 2007.

Vol. 4617: V. Torra, Y. Narukawa, Y. Yoshida (Eds.), Modeling Decisions for Artificial Intelligence. XII, 502 pages. 2007.

Vol. 4612: I. Miguel, W. Ruml (Eds.), Abstraction, Reformulation, and Approximation. XI, 418 pages. 2007.

Vol. 4604: U. Priss, S. Polovina, R. Hill (Eds.), Conceptual Structures: Knowledge Architectures for Smart Applications. XII, 514 pages. 2007.

Vol. 4603: F. Pfenning (Ed.), Automated Deduction – CADE-21. XII, 522 pages. 2007.

Vol. 4597: P. Perner (Ed.), Advances in Data Mining. XI, 353 pages. 2007.

Vol. 4594: R. Bellazzi, A. Abu-Hanna, J. Hunter (Eds.), Artificial Intelligence in Medicine. XVI, 509 pages. 2007.

Vol. 4585: M. Kryszkiewicz, J.F. Peters, H. Rybinski, A. Skowron (Eds.), Rough Sets and Intelligent Systems Paradigms. XIX, 836 pages. 2007.

Vol. 4578: F. Masulli, S. Mitra, G. Pasi (Eds.), Applications of Fuzzy Sets Theory. XVIII, 693 pages. 2007.

Vol. 4573: M. Kauers, M. Kerber, R. Miner, W. Windsteiger (Eds.), Towards Mechanized Mathematical Assistants. XIII, 407 pages. 2007.

Vol. 4571: P. Perner (Ed.), Machine Learning and Data Mining in Pattern Recognition. XIV, 913 pages. 2007.

Vol. 4570: H.G. Okuno, M. Ali (Eds.), New Trends in Applied Artificial Intelligence. XXI, 1194 pages. 2007.

Vol. 4565: D.D. Schmorrow, L.M. Reeves (Eds.), Foundations of Augmented Cognition. XIX, 450 pages. 2007.

Vol. 4562: D. Harris (Ed.), Engineering Psychology and Cognitive Ergonomics. XXIII, 879 pages. 2007.

Vol. 4548: N. Olivetti (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. X, 245 pages. 2007.

Vol. 4539: N.H. Bshouty, C. Gentile (Eds.), Learning Theory. XII, 634 pages. 2007.

Vol. 4529: P. Melin, O. Castillo, L.T. Aguilar, J. Kacprzyk, W. Pedrycz (Eds.), Foundations of Fuzzy Logic and Soft Computing. XIX, 830 pages. 2007.

Vol. 4520: M.V. Butz, O. Sigaud, G. Pezzulo, G. Baldassarre (Eds.), Anticipatory Behavior in Adaptive Learning Systems. X, 379 pages. 2007.

Vol. 4511: C. Conati, K. McCoy, G. Paliouras (Eds.), User Modeling 2007. XVI, 487 pages. 2007.

Vol. 4509: Z. Kobti, D. Wu (Eds.), Advances in Artificial Intelligence. XII, 552 pages. 2007.

Vol. 4496: N.T. Nguyen, A. Grzech, R.J. Howlett, L.C. Jain (Eds.), Agent and Multi-Agent Systems: Technologies and Applications. XXI, 1046 pages. 2007.

Vol. 4483: C. Baral, G. Brewka, J. Schlipf (Eds.), Logic Programming and Nonmonotonic Reasoning. IX, 327 pages. 2007.

Vol. 4482: A. An, J. Stefanowski, S. Ramanna, C.J. Butz, W. Pedrycz, G. Wang (Eds.), Rough Sets, Fuzzy Sets, Data Mining and Granular Computing. XIV, 585 pages. 2007.

Vol. 4481: J. Yao, P. Lingras, W.-Z. Wu, M. Szczuka, N.J. Cercone, D. Ślęzak (Eds.), Rough Sets and Knowledge Technology. XIV, 576 pages. 2007.

Vol. 4476: V. Gorodetsky, C. Zhang, V.A. Skormin, L. Cao (Eds.), Autonomous Intelligent Systems: Multi-Agents and Data Mining. XIII, 323 pages. 2007.

Vol. 4456: Y. Wang, Y.-m. Cheung, H. Liu (Eds.), Computational Intelligence and Security. XXIII, 1118 pages. 2007.

Vol. 4455: S. Muggleton, R. Otero, A. Tamaddoni-Nezhad (Eds.), Inductive Logic Programming. XII, 456 pages. 2007.

Vol. 4452: M. Fasli, O. Shehory (Eds.), Agent-Mediated Electronic Commerce. VIII, 249 pages. 2007.

Vol. 4451: T.S. Huang, A. Nijholt, M. Pantic, A. Pentland (Eds.), Artifical Intelligence for Human Computing. XVI, 359 pages. 2007.

Vol. 4441: C. Müller (Ed.), Speaker Classification. X, 309 pages. 2007.

Vol. 4438: L. Maicher, A. Sigel, L.M. Garshol (Eds.), Leveraging the Semantics of Topic Maps. X, 257 pages. 2007.

Vol. 4434: G. Lakemeyer, E. Sklar, D.G. Sorrenti, T. Takahashi (Eds.), RoboCup 2006: Robot Soccer World Cup X. XIII, 566 pages. 2007.

Vol. 4429: R. Lu, J.H. Siekmann, C. Ullrich (Eds.), Cognitive Systems. X, 161 pages. 2007.

Vol. 4428: S. Edelkamp, A. Lomuscio (Eds.), Model Checking and Artificial Intelligence. IX, 185 pages. 2007.

Vol. 4426: Z.-H. Zhou, H. Li, Q. Yang (Eds.), Advances in Knowledge Discovery and Data Mining. XXV, 1161 pages. 2007.

Vol. 4411: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 249 pages. 2007.

Vol. 4410: A. Branco (Ed.), Anaphora: Analysis, Algorithms and Applications. X, 191 pages. 2007.

Vol. 4399: T. Kovacs, X. Llorà, K. Takadama, P.L. Lanzi, W. Stolzmann, S.W. Wilson (Eds.), Learning Classifier Systems. XII, 345 pages. 2007.

Vol. 4390: S.O. Kuznetsov, S. Schmidt (Eds.), Formal Concept Analysis. X, 329 pages. 2007.

Vol. 4389: D. Weyns, H. Van Dyke Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems III. X, 273 pages. 2007.

Vol. 4386: P. Noriega, J. Vázquez-Salceda, G. Boella, O. Boissier, V. Dignum, N. Fornara, E. Matson (Eds.), Coordination, Organizations, Institutions, and Norms in Agent Systems II. XI, 373 pages. 2007.

Vol. 4384: T. Washio, K. Satoh, H. Takeda, A. Inokuchi (Eds.), New Frontiers in Artificial Intelligence. IX, 401 pages. 2007.

Vol. 4371: K. Inoue, K. Satoh, F. Toni (Eds.), Computational Logic in Multi-Agent Systems. X, 315 pages. 2007.

Vol. 4369: M. Umeda, A. Wolf, O. Bartenstein, U. Geske, D. Seipel, O. Takata (Eds.), Declarative Programming for Knowledge Management. X, 229 pages. 2006.

Vol. 4343: C. Müller (Ed.), Speaker Classification I. X, 355 pages. 2007.

Vol. 4342: H. de Swart, E. Orłowska, G. Schmidt, M. Roubens (Eds.), Theory and Applications of Relational Structures as Knowledge Instruments II. X, 373 pages. 2006.

Vol. 4335: S.A. Brueckner, S. Hassas, M. Jelasity, D. Yamins (Eds.), Engineering Self-Organising Systems. XII, 212 pages. 2007.

Vol. 4334: B. Beckert, R. Hähnle, P.H. Schmitt (Eds.), Verification of Object-Oriented Software. XXIX, 658 pages. 2007.

Vol. 4333: U. Reimer, D. Karagiannis (Eds.), Practical Aspects of Knowledge Management. XII, 338 pages. 2006.

Vol. 4327: M. Baldoni, U. Endriss (Eds.), Declarative Agent Languages and Technologies IV. VIII, 257 pages. 2006.

Vol. 4314: C. Freksa, M. Kohlhase, K. Schill (Eds.), KI 2006: Advances in Artificial Intelligence. XII, 458 pages. 2007.

Vol. 4304: A. Sattar, B.-h. Kang (Eds.), AI 2006: Advances in Artificial Intelligence. XXVII, 1303 pages. 2006.

Vol. 4303: A. Hoffmann, B.-h. Kang, D. Richards, S. Tsumoto (Eds.), Advances in Knowledge Acquisition and Management. XI, 259 pages. 2006.

Vol. 4293: A. Gelbukh, C.A. Reyes-Garcia (Eds.), MICAI 2006: Advances in Artificial Intelligence. XXVIII, 1232 pages. 2006.

Vol. 4289: M. Ackermann, B. Berendt, M. Grobelnik, A. Hotho, D. Mladenič, G. Semeraro, M. Spiliopoulou, G. Stumme, V. Svátek, M. van Someren (Eds.), Semantics, Web and Mining. X, 197 pages. 2006.

Vol. 4285: Y. Matsumoto, R.W. Sproat, K.-F. Wong, M. Zhang (Eds.), Computer Processing of Oriental Languages. XVII, 544 pages. 2006.

Vol. 4274: Q. Huo, B. Ma, E.-S. Chng, H. Li (Eds.), Chinese Spoken Language Processing. XXIV, 805 pages. 2006.

¥452.00元

Table of Contents

Multi-agent Planning and Simulation

Trust and Reputation

Multi-Agent System: A Guiding Metaphor for the Organization of Software Development Projects

Lawrence Cabac

University of Hamburg, Dept. of Informatics, Vogt-Kölln-Str. 30, D-22527 Hamburg
http://www.informatik.uni-hamburg.de/TGI

Abstract. In this work we propose the introduction of multi-agent concepts for the organization of software development projects of (especially multi-agent) application design and implementation. This is expressed by the guiding metaphor (German: Leitbild) of a *multi-agent system of developers*.

Team orientation and concurrent development are two aspects that are crucial in every large development project. Consequently, the organizational structure of the programming team has to take account for both. If the developed application is distributed, concurrent and team-oriented – e.g. a multi-agent application – one approach is to aim for a comparable (homomorphic) structure of a developed system and development team. We achieve this by reintroducing the multi-agent system metaphor into the organizational structure of the development team.

Agent attributes such as autonomy, communication, cooperation, self-organization and the capacity for teamwork are transferred by the guiding metaphor back to team members. Concurrency and distribution of resources and processes is naturally supported by the guiding metaphor.

This guiding metaphor can be applied to any project organization. However, it is best suited for the organization of multi-agent application development, due to the similarity in structure.

Keywords: agents, guiding metaphors, *multi-agent system of developers*, Leitbild, metaphor, project management, software development approach, team organization.

1 Introduction

Multi-agent systems are applications based on encapsulated, autonomous software entities that can flexibly achieve their objectives by interacting with one another in terms of high-level interaction protocols and languages. Agents balance their reactive behavior in response to influences from the environment with their proactive behavior towards the achievement of design objectives.

The agent metaphor is highly abstract and it is necessary to develop software engineering techniques and methodologies that particularly fit the agent-oriented paradigm. Traditional software development techniques such as for example object-oriented analysis and design are inadequate to capture the flexibility and autonomy of an agent's problem-solving capabilities, the richness

P. Petta et al. (Eds.): MATES 2007, LNAI 4687, pp. 1–12, 2007.

of agent interactions and the (social) organizational structure of a multi-agent system as a whole. Many agent-oriented software development methodologies have been brought forward over the last years, many of them already in mature state.

Agent-oriented development methodologies such as GAIA [1,2], MASE [3] or PROMETHEUS [4] are well-established. Similarities can be found in methods and abstractions such as use cases, system structure (organization) diagrams, role models, interaction diagrams and interaction protocols. However, it is not a trivial task to decide on a suitable implementation platform as pointed out by Sudeikat et al. [5].

Similar claims hold for the management of development processes, the organization and guidance of a team as well as for project management. As well as for methodologies and techniques of software development, there also exists a necessity to develop approaches for the management of projects that particularly fit the agent-oriented paradigm. As already proposed by Petrie et al. [6] the organization of projects can be oriented towards the agent concept. The proposal here is to increase even more the symmetry between the project management and the software being build.

We present a guiding metaphor that is capable to dynamically adapt to the needs of the team and development processes. Criteria for a powerful and acceptable metaphor is its simplicity, flexibility and the range of the commonly known concepts. It should take account of the main concepts and design objectives of the developed system; e.g. for a multi-agent application these are concepts such as distribution, concurrency and dynamical structures.

Section 2 introduces the term *guiding metaphor* and explains the guiding metaphor *multi-agent system of developers*[1] for the development of multi-agent-based projects in detail. Section 3 describes the utilization and our experiences with this guiding metaphor.

2 Leitbild: MAS

Before we start with our approach we will elaborate on the notion of the *guiding metaphor*. Then we will describe the guiding metaphor of a *multi-agent system of developers* in regard to three aspects. First, we describe the guiding metaphor in more detail in its role as a Leitbild [7] regarding orientation, notions, strategies and terminology in the environment of multi-agent application development. Second, we go into detail of the guiding metaphor's manifestation in the organizational structure of a (multi-agent application) development project especially in regard to concurrent and distributed development. Third, we focus on communication, coordination, project organization and team management.

[1] We include all participants of a development process, such as programmers, users, supporting staff, etc. We could thus also call the metaphor *multi-agent system of participants* but in the context of system development we regard all participants as developers of the system.

2.1 Guiding Metaphor

A guiding metaphor (German: Leitbild [7]) is a strong and well-established concept that can guide the participants of a development team in a general sense. While the term originated in business management, it is also well established in software engineering. A guiding metaphor should have four functions. It should offer orientation and have a strong integrative force. Decision processes should be supported by the guiding metaphor and it should also be a means of coordination. Züllighoven et al. define a guiding metaphor as follows.

> A guiding metaphor in software development defines a frame of orientation for involved groups in development processes as well as during utilization. It supports the design, utilization and the evaluation of software and is based on values and goals. A guiding metaphor can be used constructively or analytically [7, from German, p. 73].

An important feature is that the guiding metaphor is so general and common that every potentially involved person has at least a good idea of the organizational concepts, structures, notions and rules. A good guiding metaphor comes with a whole set of other metaphors that do not have to be named explicitly.[2] In the context of developing software we can distinguish three different forms of guiding metaphor. It can be used to characterize the software systems, the development process and also the team organization (respectively project management).[3] Examples of guiding metaphors are the *tools & material* approach [7] or the *expert work place* [8] for software systems. Guiding metaphors for team organizations are *the factory*, *the office*, *the workshop* or the *(free) jazz band* [9].

One interesting approach as to how to define a new guiding metaphor for team organization has been done by Mack [10]. He proposes the guiding metaphor of an *expedition* for the development process and derives some aspects that are useful in everyday (work) life of a software developer. Here we will not go into detail of this guiding metaphor but we would like to elaborate on the notions that are instantly linked to this example to show the potentials of a guiding metaphor.

For a (development) expedition one will need a team (developers, supporting users and other staff) and resources (computers, software, rooms, paper, etc.). There should be a good notion on how much everyone can carry (individual capabilities of team members) on the way. The organizers need to work out a plan in advance that is detailed enough to take as many aspects as possible into account and flexible enough to allow the team members to react to sudden changes and dangers. In an expedition it seems clear that all members have to support each other and that conflicts that are left unsolved can lead to difficulties that can endanger the expedition (software project). A good communication

[2] In this way the guiding metaphor can be compared to an *extended metaphor* or even a *parable* as used in literature.

[3] In this work we focus on the function of the guiding metaphor for the team's organizational structures / project management.

between members of the team is essential in all stages of the expedition. We know that an expedition is a socially challenging project that can be adventurous as well as hard work. In addition, the outcome of an expedition is open in the beginning.

The example shows that a strong guiding metaphor offers many notions (common in the team) and a multitude of metaphors. These help team members to find orientation in the project and by this the guiding metaphor succeeds in guiding a team.

In the following sections we describe a guiding metaphor that is well applicable to the development of multi-agent application and is also well known in the multi-agent community. It is the multi-agent system.

2.2 Multi-Agent System of Developers

Our approach of organizing projects for multi-agent application development is described by the guiding metaphor of *multi-agent system of developers*. Developer teams, their members and their actions are characterized by the attributes usually related to agents [11], multi-agent systems [12] and cooperative workflows [13].[4] In the team members are acting in a self-organized, autonomous, independent and cooperative way. They all have individual goals that culminate in a common vision of the system that is to be developed.

Like agents in a multi-agent system, developers are situated in an environment, in which they communicate with other developers and other participants of the development process. Moreover, the environment offers services or restricts the possibilities of actions for the developers.

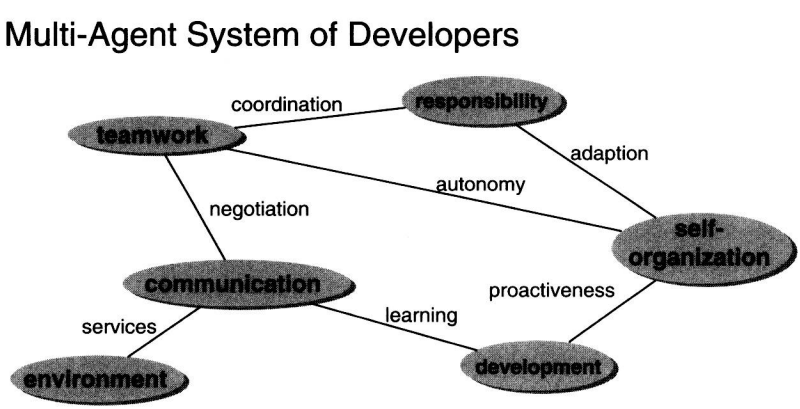

Fig. 1. Agent concepts used in the context of team organization (selection)

[4] In the following many agent concepts are used to describe behavior or attributes of members of the development team. These are used for the metaphorical power.

Figure 1 shows a selection of typical multi-agent concepts and their interrelationships that are utilized in the development project context as metaphors. Lippert et al. [8] identify a selection of key metaphors as shown in the figure as a metaphor design space.

The agent metaphor leads to dynamic and flexible structures in the team's organization. All members can form (sub-)teams with other members during the development process. This is not only encouraged but also a main aspect of the self-responsible and autonomous actions of team members. The structure of a team is not static. Sub-teams are able to decide their own dissolution and to proactively decide on new alliances. From this point of view concurrent and distributed work is a natural phenomenon.

According to the *multi-agent system of developers* metaphor control, project management and organizational matters in the development process are managed through mechanisms typically owned by social agents [14]. Thus social norms, conventions and motivation become important forces in the team's behavioral patterns.

At first glance it seems odd to re-transfer the concept (metaphor) of a multi-agent system, which has been used to define and organize software systems in the manner of (sociological) organizations, back to an organizational structure of people. However, the metaphor of a multi-agent system has grown so strong in recent years that many developers are well acquainted with the notions and key elements of agent concepts. Therefore, the multi-agent system is a reasonable, well-established and powerful guiding metaphor. But even for participants of the team that do not share the concepts of multi-agent systems as paradigm – e.g. users with no technical background – still all the concepts are well known, since they are rooted in social organizations.

In the following two sections we elaborate on two main aspects of agent-oriented development. These are the communication of agents and the concurrency and distribution. Through the guiding metaphor both aspects take a leading role in our vision on the project organization.

2.3 Matrix Organization

In a multi-agent application development project the organizational structure has to be defined, such that responsibilities for certain aspects can be assumed by team members or sub-teams. The general perspectives in the area of a multi-agent system and – therefore also here – for the development process are *structure*, *behavior* and *terminology*. These perspectives are orthogonal with connecting points at some intersections (compare Fig. 2).

The structure of a multi-agent system is given by the agents, their roles, knowledge bases and decision components [15,16]. The behavior of a multi-agent system is given by the interactions of the agents, their communicative acts and the internal actions related to the interactions [17]. The terminology of a multi-agent system is given as a domain-specific ontology that enables agents to refer to the same objects, actions and facts. The agents' common ontology is crucial for their successful interactions.

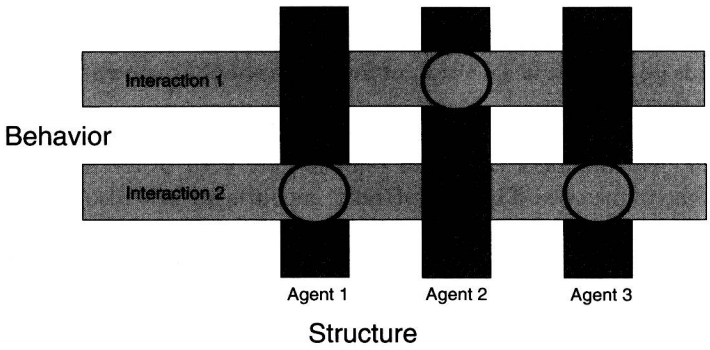

Fig. 2. Two dimensional matrix showing perspectives (*behavior, structure*)

A schematic two dimensional matrix is depicted in Figure 2 showing the independence and interconnections of agents and interactions. Neither is there any direct relationship between any pair of agents, nor between any pair of interactions. Thus these architectural elements are independent and drawn in parallel to each other. Agents and interactions are orthogonal because each agent is involved in some interactions and the same holds the other way around. When an agent and an interaction are coupled, a circle marks the interconnection point.

The general case for any two structural and/or behavioral elements is independence. In the diagram interconnections are explicitly marked. The ontology, which is omitted in the diagram, is the third dimension of perspectives. This perspective is orthogonal to the other two perspectives, but it tends to have many interconnection points because each interaction and each agent needs parts of the ontology definition to fulfill its purpose.

Since the three perspectives are orthogonal and independent within each perspective, it is easily possible to divide the tasks of design and implementation into independent perspectives and independent parts. This means that different interactions can be developed by independent sub-teams and different agents can be designed by other independent sub-teams. Between agent teams and interaction teams, coordination is needed for the crucial parts only (circles).

Following this method, the different parts of the system can be developed independently and concurrently as long as there is enough coordination / synchronization between intersecting groups.

In general it is not a good idea to assign tasks of orthogonal dimensions to the same sub-team because then the responsibilities of the different dimensions might become blurred. However, developers are well advised to look for similarities between independent elements of the same dimension, like for example a set of similar interactions. In such a situation, code reuse becomes possible if a sub-team is responsible for multiple parallel elements.

The (agent-based) software system imposes its matrix structure onto the team organization. In the metaphor of *multi-agent system of developers* this is naturally supported.