Towards Organization–Oriented Software Engineering

Matthias Wester–Ebbinghaus, Daniel Moldt, Christine Reese, Kolja Markwardt
University of Hamburg, Faculty of Mathematics, Informatics and Natural Sciences,
Department of Informatics, Vogt-Kölln-Str. 30, D-22527 Hamburg, Germany
http://www.informatik.uni-hamburg.de/TGI/

Abstract: Software systems are subject to ever increasing complexity and in need of efficient structuring. The concept of organization as an expressive and abstract real-world reference presents a promising starting point. In the field of computer science, organizations have particularly been studied within the multi-agent systems community. However, the individual agent metaphor turns out to be of rather small granularity and somewhat less suited for large-scale software systems. To overcome this problem while preserving the earnings of the agent-oriented approach to software engineering, this paper introduces the new metaphor of organizational unit. This concept allows to summarize a set of units in a manner that complex systems may be regarded and treated as wholes and exhibit corporate agency. According to different levels of analysis drawn from organization theory, different types of organizational units are incorporated into a reference architecture for organization-oriented software engineering.

Keywords: organization-oriented software engineering, software architecture in-the-large, multi-agent-systems, organizational unit

1 Introduction

Modern software systems are subject to ever increasing complexity. They may be comprised of hundreds and thousands of distributed and concurrent computer systems, which is particularly true for business applications ([LMW05]). These systems are connected through different technologies, support different processes and are continuously modified in the course of projects. Multi-agent systems are widely favoured to provide solutions for the challenges of such modern software systems, for example because of their applicability in using the techniques of decomposition, abstraction and organization ([Jen00]) or their prospects concerning self-management ([KC03]).

One characteristic of multi-agent research is to draw inspiration from other scientific fields like sociology, linguistics or cognition research. In this context, organization theory serves to analyse and control phenomena in multi-agent systems that carry a *supra-individual* character. One reason for the existence of organizations in human societies is their ability to overcome limitations of individuals like *cognitive, physical and temporal* ones ([CG00]). Organizations do so in building organizational structures to divide labour (role specialization, departmentalization, multi-divisional forms) and coordinate and control the workflow (formalization, hierarchy, centralization, lateral information flows). Artificial agents analogously suffer from the mentioned limitations. Consequently, agent tech-

nologies have dealt with organizations for quite some time. One can identify hierarchies, matrices, holons, coalitions, teams, federations (see [HL05a] for a summary). However, the corresponding organizational principles have frequently been embodied implicitly and the concept of organization seems to have been considered a special coordination technique rather than deserving a focus of interest ([Gas01]). But over the past few years multi-agent system researchers have begun to comprehend the organizational metaphor as a central instrument to combine agent autonomy with reliability and predictability on the system level ([ZJW03]). A variety of organization-oriented approaches to multi-agent systems engineering have been brought forth including modelling approaches ([FGM03] (AGR), [HSB02] (MOISE), [HL05b] (ODML), [ONL04]), complete development methodologies ([ZJW03] (GAIA), [DeL05] (MaSE), [VSDD05] (OMNI), [BPG$^+$04] (Tropos)) and middleware ([GF00] (MADKIT), [HSB05] (S-MOISE)).

All these approaches have their own distinguishing features. In search of commonalities in order to arrive at an across-the-board conception of organization-oriented multi-agent systems one could possibly best consult the characterization provided by Jennings ([Jen00]) in 2000. This proposal seems to have had substantial influence on following work within the area of multi-agent systems. Different dimensions of software systems are distinguished.

- *System*: The system to be characterized is the multi-agent organization.
- *Components*: The components are the primitive elements from which the system is composed. For multi-agent organizations Jennings identifies *agents, interaction channels*, *dependencies* (due to interacting goals or shared resources) and *organizational relationships* (structures of authority, cooperation or competition).
- *Composition Laws*: The laws of composition specify how the components are assembled to form the system. Jennings names *roles* (abstract characterizations of agent behaviour in specific organizational contexts, including appropriate patterns of interaction) and *organizational rules* (supra-role boundary conditions for the proper instantiation and execution of the organization).[1]
- *Behavioural Laws*: Behavioural Laws describe how the behaviour of the system depends on the composition of its components. It depends on which agents take on which roles, in how far role specifications and organizational rules are respected and in how far agents honour dynamically arising social commitments.
- *Medium*: The medium describes the elements that are processed to achieve the desired system behaviour. According to Jennings, these are *organizational commitments* (that agents enter through roles or social interactions with other agents), *possibilities to influence others* (interactions protocols for negotiation, cooperation, coordination) and *reflexive elements* (ways to alter the organization, for example modifying role specifications and organizational rules).

[1] Ferber criticises that the characterization so far lacks one important feature of organizations, namely partitioning ([FGM03]). For this reason, many organization-oriented approaches to multi-agent systems engineering include group concepts.

Based on this characterization, the merits of organization-oriented multi-agent systems will be pointed out in Sect. 2, followed by a critical review of the state-of-the-art in Sect. 3. Taking this critique as a vantage point, a proposal for the step from agent-oriented to organization-oriented software engineering is submitted in Sect. 4 before the streaks of intended future work will be pointed out in Sect. 5.

2 The Merits of Organization-Oriented Multi-Agent Systems

In this section, a case will be made for organization-oriented multi-agent systems. The advantages will be presented along three lines of argument, namely concerning complex software systems, reliability and a natural approach to software engineering.

Multi-agent systems are qualified for applying the techniques of decomposition, abstraction and organization ([Jen00]). An organization-oriented approach fosters this qualification. When complexity takes on a hierarchical character, a decomposition approach incorporating group concepts is better suited than a single agent approach. The isolated treatment of sub-problems remains possible through the clear distinction between interactions within the group and interactions with the outside. Turning to abstraction, it is up to the designer which components to regard as basic building blocks at a certain stage in the development process of a software system. An organization-oriented approach facilitates to close the gap between early analysis and design steps and the implementation by describing how whole systems and subsystems are successively aggregated from individual agents. Finally, it should be obvious that an organization-oriented approach facilitates the organization of the systems components. It introduces roles, interaction patterns, groups, organizational rules and the like as first-order concepts and thus eases the formation and successive modification of organizational structures.

Without organizational structuring, agents as autonomous entities would pursue their own goals, act according to their own knowledge and abilities and have exclusive responsibility for defining their relationship to others. The behaviour of the system would solely result as a bottom-up process from agent interactions. The number, timing, patterns and results of interactions would depend on a complex interplay between the internal states of the agents and their respective social contexts. Interactive composition results in emergent phenomena that cannot be solely explained in terms of the behaviour of the individual components ([Mil93]). Having to put up with the global system behaviour as emergent is often not acceptable, especially regarding economic applications. Instead, features like stability over time, predictability and commitment towards certain global strategies are called for. To account for these requirements, the currently prevalent conception among the multi-agent systems community is to rely on a mindset that harkens back to social, administrative and economic concepts and principles of human organizations. The (obligatory) bottom-up process is restricted by a top-down design where certain organizational „facts“ are imposed onto the system. These are external to all agents and present directives or restrictions for their local decision making.

The organization-oriented approach to software engineering presents an intuitive one. The

line of argument is analogous to the one brought forth as a justification for the shift from object-oriented to agent-oriented software engineering, namely that key concepts and abstractions for the development of software systems should be rooted in the real world. Exemplary, Jennings argues in [Jen00] that besides passive objects the real world would equally contain active agents, acting and interacting with one another to achieve their goals. Regardless whether a view upon the real world as entirely consisting of passive objects and active agents is strictly speaking correct or not (see Sect. 4), it is not very descriptive. Agent societies of the real world are by no means unorganized. Instead, one can recognize clubs, firms, departments, teams or generally speaking different aggregates of agents and additional relationships between them. In the context of such aggregates, it does not lie within the responsibility of one agent alone to define its relation to other agents. Rather, agents take on roles that endow them with rights and permissions but also duties and commitments. Consequently, to arrive at an intuitive conception of a software system with many active and autonomous agents, the organizational metaphor is omnipresent.

3 State-of-the-Art Criticism

Following [CSB05], two interpretations concerning the term of organization can be distinguished within the field of multi-agent systems. In the first case, the term refers to an explicitly identifiable entity that is represented by (though not identical to) a set of agents. In the second case, it rather refers to a set of boundary conditions that shape the agents' actions and interactions in specific social contexts. In [CSB05], the denomination *social organization* is used for the second conception whereas the first one is continued to be simply called organization. It should be noted that the second case is the more general one. An organization entails a social organization, but not vice versa. Generally speaking, any organization-oriented approach to multi-agent systems design defines a social organization. Organizations following the first conception are frequently used in form of group concepts for several reasons like (iterated) decomposition, modularity, security, framework-component approaches ([FGM03]). Group concepts are extraordinarily applied in holonic multi-agent systems ([FSS03]) that embody nested, self-similar structures. Characteristically, each grouping within such systems exhibits features that are derived but distinct from its members while at the same time it contributes to the features of the grouping whose member it itself is.

Some of the methodologies and modelling approaches mentioned in the introduction make use of the abstractional power of group concepts by regarding groups as holistic entities and independent actors during *early phases* of system development. However, in later phases of development and especially in the implementation these groups are broken down to the agent level. Activities are divided into operational and managerial parts and corresponding agents are placed into a network that is maintained through the definition of corresponding agent interactions. Frequently used types of such interactions are for example explicated in [CKM02] as *social patterns*. The groups of the early development stages are now only identifiable through the inspection of the interactions and the roles that the agents play in their context. Especially for the realization of large-scale organization-

oriented systems this reveals considerable problems.

- *Design*: The design of the system has to be successively refined until the granularity of the involved entities is small enough to allow the identification of individual agents. This may lead to a cumbersome waterfall model of the development process and finally to large, unwieldy system designs.
- *Implementation*: All characteristics of the organization have to be captured within the interactions between the agents and the corresponding roles. This work is both tedious and extremely error-prone as each involved agent has its own specific view on an interaction (depending on the role it plays and its social context).
- *Attendance, adaptation*: Modifications to the design of a running system may in some cases affect many components. Even more severe problems arise when the system itself is modified without immediately working them into the design of the system (roundtrip engineering). At a later point in time it might become extremely difficult or even impossible to trace the sources of the modification back due to the complete distribution of the organizational information.

It appears that these problems all result from the fact that the „classical“ individual agent is too small as a basic building block of software systems for a pure agent-oriented approach to comfortably realize large-scale organization-oriented system designs.

4 A Reified Software Organizations View

This section will relate organization-oriented multi-agent systems engineering to organization theory. It will be pointed out that the concept of reifying organizations has been largely neglected in the area of agent-oriented software engineering. The prospects along with the difficulties of such an approach will be discussed in order to derive a preliminary reference model of organization-oriented software engineering.

4.1 Organization Theory and its Relation to Multi-Agent Systems

In [Pfe82], organization theories are classified along the two dimensions *analysis level* and *action perspective*. The analysis level relates to whether organizations are just regarded as systems composed of individuals and interactions between them or whether they are regarded as holistic entities that embody corporate agents and are thus reified. In [Sco03], an analogous identification of analysis levels is brought forth by separating the *organizational structure level* (explaining structural features and social processes that characterize organizations and their subdivisions) from the *ecological level* (focusing on the characteristics or actions of the organization viewed as a collective entity operating in a larger system of relations). The action perspective relates to whether action is controlled from

the outside, whether behaviour is rational, goal-oriented and autonomous or whether the bi-directional relationship between the first two perspectives is emphasized. These three perspectives are projected onto the two analysis levels. Thus, six classes of action result.

The classification in [Pfe82] relates to human societies. In [vdB01], the different classes of action are examined according to their suitability to be utilized in multi-agent systems. Generally speaking, the three classes for the individual level have been studied and applied quite thoroughly. The perspective on action controlled externally can be found within the area of cooperative problem solving.[2] The rational action perspective has especially been applied within the area of artificial intelligence where agents weigh their goals and action alternatives to decide on execution paths. The bi-directional perspective is exactly the one taken on for organization-oriented approaches discussed in the introduction of this paper. The three action classes for the collective level however, have, if at all, only been applied tentatively. As described in the former section, organizations and groups are *approximated* as actors in early phases of system development. But when it comes to later phases of design and especially to implementing the system, this view is abandoned. Some of the approaches mentioned in the introduction maintain concepts like *agentified groups* (e.g. [FGM03, ONL04]) and especially holons in holonic multi-agent systems shall appear as holistic actors. But either the concrete realization is left open or a federative solution is proposed with one or more agents acting as mediators, managers, heads and the like in order to represent a group to the outside. Thus, these approaches remain purely agent-oriented and do not solve the problems identified in the former section.

Within the area of organization theory, there exist arguments for and against the reification of organizations to explain certain phenomena of human societies (see [vdB01] for a discussion). However, as Scott points out, the importance of organizations cannot be fully acknowledged when they are merely regarded as contexts that shape the actions and interactions of individuals ([Sco03]). Organizations are actors in itself. They act, utilize resources and enter commitments. The paper at hand is in line with this conception and in the next subsection, it will be carried forward to software engineering.

4.2 Reference Architecture for Organization-Oriented Software Engineering

As a starting point for the approach to reify software organizations, four general difficulties are cited from [vdB01].

- *Epistemological problem*: Reified organizations shall appear as corporate actors. However, knowledge and goals of an organization as central elements of deliberation are distributed among its members. So it is not trivial to satisfyingly involve them into the corporate decision making procedures.
- *Reification problem*: Reifying organizations bears the risk to conceal the fact that the mechanisms of decision making and the corresponding processes result from the

[2] Due to the lack of a considerable amount of autonomy, the term *agent* is not entirely suited in this case. So, for example in [DLC89] the term *problem-solver* is rather used.

interactions of the organization's members. In this respect the organization might prove misleading as an explanatory model.

- *Responsibility problem*: The question arises whom to hold responsible for the goals, actions and commitments of an organization as corporate actor.
- *Realization problem*: It seems unclear how to realize reified software organizations with multi-agent systems.

The key problem seems to be the last one. Classical multi-agent systems are indeed not suited to realize reified organizations. Consequently, a partial renunciation of the agent-oriented paradigm is propagated at this point. Instead of the agent the concept of *organizational unit* shall serve as the central metaphor and building block for software systems. This does not mean to completely replace the agent-oriented paradigm. Rather, the reciprocity of the two metaphors agent and organization is to be stressed. Two general requirements are imposed onto organizational units. First of all, they must take on platform functionality for their members and place them into a certain organizational context. Secondly, this context has to allow the organizational unit to appear as a holistic, autonomous actor to the outside. By this means, organizational units may themselves take on roles within other organizational units. Thus, the reification of organizations is accomplished and the scalability of large software systems through recursive nesting of organizational units is made possible. Within the field of multi-agent systems, sufficient concepts exist to pave the way for this form of reification. The platform concept is widely applied for FIPA-compliant multi-agent systems. Rölke introduces the idea of agentifying whole multi-agent systems, where agents exhibit platform functionality for contained agents and platforms reside as agents on other platforms ([Röl04]).

With this conception of organizational unit, the solutions to the two above mentioned problems of reification and responsibility are outlined as well. In spite of reifying, it always remains clear that the behaviour of an organizational unit results from the actions and interactions of its members. The question of responsibility can only be answered based on the respective organizational context of an organizational unit. The first mentioned problem remains critical and presents an essential challenge for the realization of the described approach of organization-oriented software engineering. It is crucial to provide principles and concrete guidelines for how the behaviour of complex organizational units arises from the actions and interactions of its members in a deliberate way. This includes reactive and proactive behaviour, the ability to adapt the behaviour, the participation in social interactions with other organizational units and to enter commitments.

Figure 1 displays an abstract architecture for organization-oriented software systems that captures the ideas and demands identified in this and the former sections.[3] Four levels of organizational units are distinguished according to different degrees of abstraction. These levels are analogous to the levels of analysis that Scott proposes in [Sco03] for organizations in human societies. Thus, the related concepts are rooted in organization theory

[3]The architecture is modelled as a reference net (a higher order Petri net formalism) due to its suitability to model dynamic composition of components. However, no familiarity with Petri net theory is required to read this rather abstract model. The round icons represent *places* that may contain complex *tokens*. Tokens may itself be reference nets. The rectangle icons are *transitions* and model events that involve tokens from adjacent places.

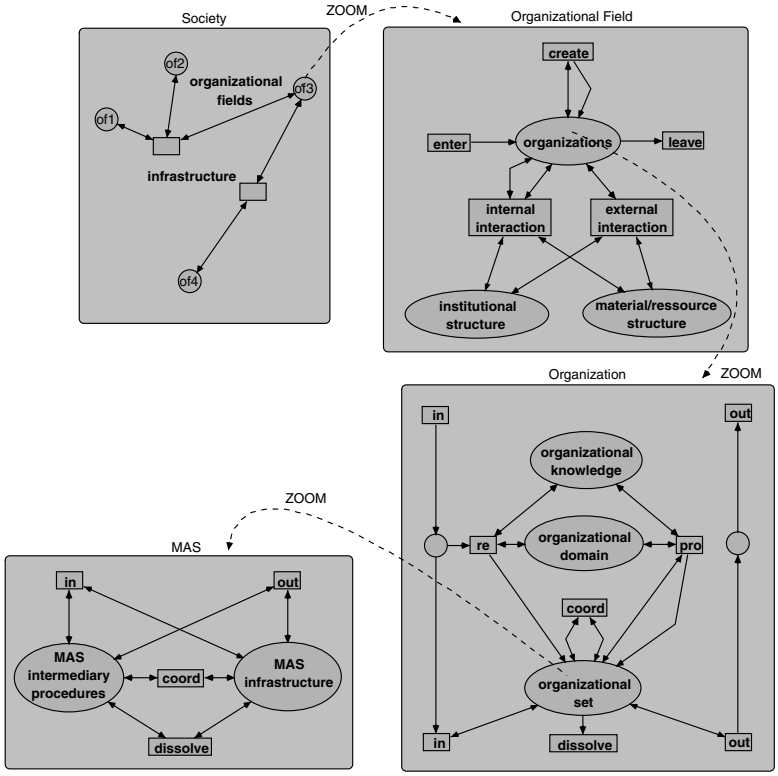

Figure 1: Reference Architecture for Organization-Oriented Software Engineering

and are re-considered within the more technical context of software architectures. At the lowest level are the internal components of a particular organization analogous to the *organizational structure level* identified by Scott. One could think of departments and offices that are embodied as distinct multi-agent systems in the software architecture. The three super-ordinate levels are analogous to Scott's further separation of the *ecological level*. The second level captures *organizations* as holistic, corporate actors. Collections of interrelated organizations are captured at the level of *organizational fields*. Different organizational fields are integrated into an even wider context at the level of the *society*.

At the organizational level one can identify the concepts *organizational domain* and *organizational set* from organization theory. An organization's domain is a description of the range of products and services it provides and the types of clients and consumers served. For the software architecture, an organization's domain is identified as a set of multi-agent system templates that can be launched as new subunits of the organization. These templates embody *formal organizational models*. According to Scott, organizational models are formalized to the extent that the rules governing behaviour are precisely and explicitly

formulated and to the extent that they are independent from any individuals (agents) that occupy organizational positions. Several approaches to specify organizations that can be considered formal in this sense have been brought forth by the multi-agent systems community (consult Sect. 1 for references). They may capture multiple organizational features among which the most prominent are structural, functional and interactional ones.

The organizational set comprises all the roles, services and relationships an organization is engaged in at a given point in time. Turning to the software architecture, the organizational set is made up of an organization's running multi-agent systems. The agents of these multi-agent systems utilize the corresponding formal organizational models in order to carry out organizational services, realize organizational processes, form teams and the like. As the agents bring in their own individual characteristics, the multi-agent systems constitute *informal organizational units*. Being constituent parts of an organization they also have to implement intermediary procedures to connect them with the next higher level of organizational unit. Taking into respect the organizational knowledge, new multi-agent systems are launched reactively as a response to some influence from the outside or proactively as a response to an internal request by some already existing multi-agent system. The organizational knowledge is continuously altered bottom-up by the running multi-agent systems as well as top-down when information from the super-ordinate level of organizational unit is incorporated.

An organization structures and frames its internals and provides an interface to the outside. Note, that the organizational level deals with the environment from the viewpoint of one *focal organization*. The level of organizational fields shifts the view of the environment to collections of interdependent organizations. Here, one can find market places, virtual organizations, strategic alliances, coalitions and the like. Organizations may enter and leave fields and engage in field-internal or field-external interactions with one another. In addition, new organizations might be created (either entirely new or as a fusion or splitting of existing organizations). The features of the environment are divided into *material-resource* and *institutional* ones. Material-resource conceptions characterize the environment as stocks of resources and sources of information and thus capture which organization owns which resources, offers which services and can provide which information. Institutional conceptions capture the regulative and normative forces of the field that shape the behaviour of the organizations and their members.

Organizational fields may define their own material-resource and institutional context but at the same time exist within the wider context of society. The society defines supra-field rules and laws and provides an infrastructure for the communication und migration between fields and corresponding restrictions when and under what bounding conditions these are permitted.

Recalling the criticism from Sect. 3, the described problems were traced back to the fact that the individual software agent was too small as a building block for large-scale organization-oriented software engineering. The reference architecture presented here provides the system designer with different reified organizational units of ever increasing abstraction level. At any level of abstraction, the next lower organizational units can be treated as wholes throughout the entire development process and remain identifiable entities in the implemented system.

5 Conclusion and Future Work

In this paper a case was made for an organization-oriented approach to the engineering of large software systems. As a first step, the advantages for multi-agent system engineering stemming from the incorporation of organizational concepts were identified. A critical analysis of the current multi-agent systems state-of-the-art together with a consideration of organization theories that view organizations as corporate actors led to a proposal for the advancement from an agent-oriented to a truly organization-oriented view on software engineering. The proposal was accompanied by the presentation of an abstract reference architecture for the reification of software organizations. This architecture especially emphasizes the means of abstraction introduced by the metaphor of organizational unit. It allows to summarize a set of entities in a way that complex systems may be regarded and treated as wholes. In this respect, the organization-oriented paradigm formulated here does not only serve the engineering of software systems. Rather, it might serve as a general approach to the comprehension of complex systems and thus help to close the conceptual gap between information technology and other fields like economics or sociology whose applications are to be supported with computer systems.

Turning to future work, the most pressing issue is the elaboration of the reference architecture. Concerning the multi-agent system and the organization level, it is expected to be able to build upon recent work of the Theoretical Foundations of Computer Science Group at the University of Hamburg. In [Köh06], an approach to formalize the organization of multi-agent systems in order that they fulfil specific purposes by carrying out well-defined team processes is presented. These processes rely heavily on role-based delegation. In so far, the agent-based distributed workflow management system presented in [ROM+05] might serve as an adequate target platform for the execution.

The following set of hypotheses is derived from the presented reference architecture. To prove these justified (or not) presents a further essential venture point for future work.

- *Application closeness:* In drawing explicitly on organization theory for real-world organizations like firms, companies and the like, the organization-oriented paradigm that is embodied by the reference architecture is particularly suited for realizing the corresponding supporting information technology. This applies for single organizations as well as for the bridging between different companies.
- *Concept of autarchy:* Organizations own and utilize resources. Autarchy describes the degree to which an organization can support itself with all the resources it needs. Autarchy arises as an important concept of the organization-oriented paradigm, parallel to that of autonomy inherited from the agent-oriented paradigm.
- *Physical and logical platforms:* Organizational units are complex entities and inherently distributed. Their components may reside on different physical platforms. The coherence derives from the combination of physical and logical platform functionality of an organizational unit.
- *Multi-perspective approach:* The four levels of the reference architecture basically present abstraction levels. *Each* level models organizational units. The four types

are distinguished according to their special characteristics and features that they possess with respect to their organizational embedding. But in fact, the modelling should be regarded from a multi-perspective viewpoint. An organizational unit may take on multiple roles of those identified in the reference architecture in multiple instances, depending on context.

References

[BPG+04] Bresciani, P.; Perini, A.; Giorgini, P.; Giunchiglia, F.; Mylopoulos, J.: Tropos: An Agent–Oriented Software Development Methodology. In: Autonomous Agents and Multi-Agent Systems, 8. 2004; Pages 203–236.

[CG00] Carley, K.; Gasser, L.: Computational Organization Theory. In: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, Cambridge, MA, 2000; Pages 299–330.

[CKM02] Castro, J.; Kolp, M.; Mylopoulos, J.: Towards Requirements-Driven Information Systems Engineering: The Tropos Project. In: Proceedings of the 13th International Conference on Advanced Systems Engineering (CAISE '01). Elsevier Science Ltd., 2002; Pages 365–389.

[CSB05] Coutinho, L.; Sichman, J.; Boissier, O.: Modeling Organization in MAS: A Comparison of Models. In: Proceedings of the 1st International Workshop on Software Engineering for Agent-Oriented Systems (SEAS '05). 2005; Pages 1–10.

[DeL05] DeLoach, S.: Engineering Organization-Based Multiagent Systems. In: Software Engineering for Large-Scale Multi-Agent Systems (SELMAS), volume 3914 of Lecture Notes in Computer Science. Springer Verlag, 2005; Pages 109–125.

[DLC89] Durfee, E.; Lesser, V.; Corkill, D.: Trends in Cooperative Distributed Problem Solving. In: IEEE Transactions on Knowledge and Data Enginieering, 1(1). 1989; Pages 63–83.

[FGM03] Ferber, J.; Gutknecht, O.; Michel, F.: From Agents to Organizations: an Organizational View of Multi-Agent Systems. In: Agent-Oriented Software Engineering IV, 4th International Workshop (AOSE 2003), volume 2935 of Lecture Notes in Computer Science. Springer Verlag, 2003; Pages 214–230.

[FSS03] Fischer, K.; Schillo, M.; Siekmann, J.: Holonic Multiagent Systems: A Foundation for the Organization of Multiagent Systems. In: Holonic and Multi-Agent Systems for Manufacturing, First International Conference on Industrial Applications of Holonic and Multi-Agent Systems (HoloMAS), volume 2744 of Lecture Notes in Computer Science. Springer Verlag, 2003; Pages 71–80.

[Gas01] Gasser, L.: Perspectives on Organizations in Multi-agent Systems. In: EASSS '01: Selected Tutorial Papers from the 9th ECCAI Advanced Course ACAI 2001 and Agent Link's 3rd European Agent Systems Summer School on Multi-Agent Systems and Applications. Springer Verlag, 2001; Pages 1–16.

[GF00] Gutknecht, O.; Ferber, J.: The MADKIT Agent Platform Architecture. In: International Workshop on Infrastructure for Multi-Agent Systems: Infrastructure for Agents, Multi-Agent Systems and Scalable Multi-Agent Systems, volume 1887 of Lecture Notes in Computer Science. Springer Verlag, 2000; Pages 48–55.

[HL05a] Horling, B.; Lesser, V.: A Survey of Multi-Agent Organizational Paradigms. In: The Knowledge Engineering Review, 19(4). 2005; Pages 281–316.

[HL05b] Horling, B.; Lesser, V.: Using ODML to Model Organizations for Multi-Agent Systems. In: Proceedings of the 2005 International Conference on Intelligent Agent Technology (IAT 2005). IEEE Computer Society, 2005; , Pages 72–80.

[HSB02] Hübner, J.; Sichman, J.; Boissier, O.: A Model for the Structural, Functional and Deontic Specification of Organizations in Multiagent Systems. In: Proceedings of the 16th Brazilian Symposium on Artificial Intelligence (SBIA'02), volume 2507 of Lecture Notes in Artificial Intelligence. Springer Verlag, 2002; Pages 118–128.

[HSB05] Hübner, J.; Sichman, J.; Boissier, O.: S-MOISE: A Middleware for Developing Organised Multi-Agent Systems. In: International Workshop on Organizations in Multi-Agent Systems: From Organizations to Organization-Oriented Programming (OOOP 2005). 2005; Pages 107–120.

[Jen00] Jennings, N.: On agent-based software engineering. In: Artificial Intelligence, 177(2). 2000; Pages 277–296.

[KC03] Kephart, J.; Chess, D.: The Vision of Autonomic Computing. In: IEEE Computer, 36(1). 2003; Pages 41–50.

[Köh06] Köhler, M.: Formalizing Multi-Agent Organizations. In: Proceedings of the International Workshop on Concurrency, Specification and Programming (CS& P 2006). 2006.

[LMW05] Lankes, J.; Matthes, F.; Wittenburg, A.: Softwarekartographie: Systematische Darstellung von Anwendungslandschaften. In: Wirtschaftsinformatik 2005. 2005.

[Mil93] Milner, R.: Elements of Interaction. In: Communications of the ACM 36, 1. 1993; Pages 78–89.

[ONL04] Odell, J.; Nodine, M.; Levy, R.: A Metamodel for Agents, Groups and Roles. In: Agent-Oriented Software Engineering (AOSE) 5, volume 3382 of Lecture Notes in Computer Science. Springer Verlag, 2004; Pages 78–92.

[Pfe82] Pfeffer, J.: Organizations and Organization Theory. Pitman Publishing Inc., 1982.

[Röl04] Rölke, H.:. Modellierung von Agenten und Multiagentensystemen. Logos Verlag Berlin, 2004.

[ROM+05] Reese, C.; Ortmann, J.; Moldt, D.; Offermann, S.; Lehmann, K.; Carl, T.: Architecture for Distributed Agent-Based Workflows. In: Proceedings of the Seventh International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2005). 2005; Pages 42–49.

[Sco03] Scott, W.: Organizations: Rational, Natural and Open Systems. Prentice Hall, 2003.

[vdB01] van den Broek, J.: On Agent Cooperation: The Relevance of Cognitive Plausibility for Multiagent Simulation Models of Organizations. Labyrinth Publications, 2001.

[VSDD05] Vasquez-Salceda, .J; Dignum, V.; Dignum, F.: Organizing Multiagent Systems. In: Autonomous Agents and Multi-Agent Systems, 11. 2005; Pages 307–360.

[ZJW03] Zambonelli, F.; Jennings, N.; Wooldridge, M.: Developing Multiagent Systems: The Gaia Methodology. In: ACM Transactions on Software Engineering and Methodology, 12(3). 2003; Pages 317–370.