The Knowledge Engineering Review, Vol. 17:4, 309–316. © 2002, Cambridge University Press
DOI: 10.1017/S0269888903000596 Printed in the United Kingdom

Coordination knowledge engineering

SASCHA OSSOWSKI[1] and ANDREA OMICINI[2]

[1] *ESCET, Universidad Rey Juan Carlos, Campus de Mostoles, Calle Tulipan s/n, E-28933 Madrid, Spain;*
email: sossowski@acm.org, http://www.ia.escet.urjc.es/~sossowski
[2] *DEIS, Università di Bologna, Via Rasi e Spinelli 176, 47023, Cesena, FC, Italy;*
email: andrea.omicini@acm.org, http://lia.deis.unibo.it/~ao

Abstract

By adopting a structured knowledge-level approach, *coordination knowledge* can be ascribed to groups (societies) of system components (agents) as a whole, rather than to individuals, in order to effectively rationalise complex patterns of interaction within intelligent (multi-agent) systems. Be it either explicitly represented at the symbol-level or hard-coded within specific coordination algorithms, coordination knowledge is instrumented by a wide and heterogeneous variety of coordination models, abstractions and technologies. Coordination knowledge engineering is then about eliciting, modelling and instrumenting coordination knowledge in a principled and effective manner.

In this introductory article, we briefly review two well-known frameworks to conceptualise coordination, then we discuss different dimensions along which coordination models can be classified, and analyse their impact on the design of coordination mechanisms and their supporting coordination knowledge. Finally, we sketch our view on coordination knowledge engineering and introduce the different contributions to this special issue along this line.

1 Introduction

The problem of building intelligent systems is rapidly crossing the narrow boundaries of academia to become an industrial issue. Widespread access to information technology by millions of typically low-skilled newcomers has first summoned much attention and work on system usability, then made the request for intelligence almost inescapable. In this context, acceptance of the term "intelligence" is usually quite pragmatic, and basically relates to the human user's perception: a system is intelligent when it behaves in an intelligent way from the viewpoint of the observer/user, independently of the system's inner structure (Omicini & Papadopoulus, 2001).

A similar stance was taken by Newell (1982, 1993) when he introduced the notion of the *knowledge level* almost 20 years ago: from a knowledge-level perspective, "knowledge" is actually *ascribed* to a system by an external observer. Generally speaking, the design of a system aimed at (re)producing intelligent behaviour is conceived in essence as a modelling activity. In this respect, it is obvious that a "what" model of the behaviour itself is mandatory, e.g. in terms of a series of episodes of its (inter)action with the environment. A *symbol-level* model is a "how" model of that behaviour, which aims at its *mechanisation* in terms of symbols and representation. A *knowledge-level* model, instead, is a "why" model that aims at *rationalising* the behaviour in terms of goals and knowledge. From a knowledge-level perspective, a system is treated as a black box that acts "as if" it possessed certain knowledge about the world and used this knowledge in a perfectly rational way towards reaching its goals (van de Velde, 1993). Again, Newell's original concept of the knowledge level makes no assumptions about the structure of knowledge in a system.

Still, in recent years there has been a growing awareness that a constructionist (bottom-up) approach to the construction of intelligent systems may help to reduce the complexity of the engineering process

(Jennings & Campos, 1997). Intelligent systems are built by "glueing together" different *sources* of behaviour which, when conceived as a whole, are perceived as intelligent by the user. The encapsulated computational entities that support these behaviours may come at different levels of complexity, e.g. simple software modules, objects, components or genuine knowledge-based systems. They are often distributed and heterogeneous in nature, and show a certain level of autonomy – but maybe their defining features are the complex patterns of interaction that emerge among them. Systems of this sort are often labelled *multi-agent systems*, and the task of governing their interactions is termed *coordination*.

When approaching the design of such systems from a knowledge engineering perspective, we can apply van de Velde's (1993) *structured knowledge-level* approach to ascribe different *types* of knowledge to the system, in line with its a-priori structure. On the one hand, local knowledge needs to be attributed to the encapsulated computational entities, or *agents*, so as to explain their individual behaviour in relation to a supposed local goal. On the other hand, global goals and *coordination knowledge* must be ascribed to groups (societies) of agents, or to the multi-agent system as a whole, if the complex patterns of interaction are to be rationalised convincingly.

A large variety of coordination models, abstractions and technologies have come up in order to support the (symbol-level) instrumentation of such (knowledge-level) coordination knowledge. Often, there is no explicit declarative representation of that knowledge in the system, as its functionality is hard-coded and hidden in specific coordination algorithms. Still, coordination knowledge is sometimes explicitly represented at the symbol level, be it as part of a coordination infrastructure (Omicini & Denti, 2001), as a knowledge base of a dedicated coordinator agent (Martial, 1992), or as common knowledge of the interacting agents (Ossowski, 1999). Coordination knowledge engineering is all about eliciting, modelling and instrumenting coordination knowledge in a principled and effective manner.

In what follows, we give a brief review of two well-known frameworks for conceptualising coordination and relate them to the knowledge − and the symbol-level perspectives respectively. Then we discuss different dimensions along which coordination models can be classified and analyse their impact on the design of coordination mechanisms and their supporting coordination knowledge. Finally, we sketch our view on coordination knowledge engineering and introduce the different contributions to this special issue along this line.

2 Models of coordination

Maybe the most widely accepted conceptualisation of coordination in AI originates from organisational science. It defines coordination as the *management of dependencies* between organisational activities (Malone & Crowston, 1994). One of the many workflows in an organisation, for instance, may involve a secretary writing a letter, an official signing it and another employee sending it to its final destination. The interrelation of these activities is modelled as a *producer/consumer* dependency, which can be managed by inserting additional *notification* and *transport* actions into the workflow.

It is straightforward to generalise this approach to coordination problems in AI. The subjects whose activities need to be coordinated are sometimes called *coordinables*, but in the context of AI they are commonly conceived of as agents. The entities between which dependencies arise (or *objects of coordination*) are often termed quite differently, but usually come down to things like goals, actions and plans. Depending on the characteristics of the problem at hand, a taxonomy of dependencies can established, and a set of potential coordination actions assigned to each of them (e.g. Martial, 1992). Within this model, the *process* of coordination is to accomplish two major tasks: first, a *detection* of dependencies needs to be performed and, second, a *decision* respecting which coordination action to apply must be taken. A coordination *mechanism* shapes the way that agents perform these tasks (Ossowski, 1999).

The *result* of coordination, and its *quality*, are conceived differently at different levels of granularity. Von Martial's stance on coordination as a *way of adapting to the environment* (Martial, 1992), is quite

well suited to understanding this question from a *micro-level* perspective, in particular if we are concerned with multi-agent settings. If new acquaintances enter an agent's environment, coordination amounts to reassessing its former goals, plans and actions, so as to account for the new (potential) dependencies between itself and other agents. If a STRIPS-like planning agent, for instance, is put into a multi-agent environment, it will definitely have to accommodate its individual plans to the new dependencies between its own prospective actions and potential actions of others, trying to exploit possible synergies (others may free certain relevant blocks for it), and avoiding harmful dependencies (making sure that others do not unstack intentionally constructed stacks and so on). At this level, the result of coordination, the agent's adapted individual plan, is the better the closer it takes the agent to the achievement of its goals in the multi-agent environment.

From a *macro-level* perspective, the outcome of coordination can be conceived of as a "global" plan (or decision, action and so on). This may be a "joint plan" (Rosenschein & Zlotkin, 1994), if the agents reach an explicit agreement on it during the coordination process, or just the sum of the agents' individual plans (or decisions, actions and so on – sometimes called "multi-plan" (Ossowski, 1999)) as perceived by an external observer. Roughly speaking, the quality of the outcome of coordination at the macro level can be evaluated with respect to the system's goal or the desired functionality. If no such notion can be ascribed to the system, other, more basic features can be used instead. A good result of coordination, for instance, is often supposed to be efficiency (Rosenschein & Zlotkin, 1994), which frequently comes down to the notion of pareto-optimality: no agent could have increased the degree of achievements of its goals without any other being worse off in that sense. The amount of resources required for coordination (e.g. the number of messages necessary) is also sometimes used as a measure of efficiency.

The dependency model of coordination appears to be particularly well suited to rationalising observed coordination behaviour in line with Newell's knowledge-level perspective (Newell, 1982; Newell, 1993). Still, when *designing* coordination processes for real-world applications, things are not as simple as the dependency model may suggest. Dependency detection requires an efficient elicitation, representation and enactment of considerable amounts of knowledge (Ossowski *et al.*, 2002). Moreover, incomplete and potentially inconsistent local views of the agents will further complicate this task. In all, making timely decisions that lead to efficient coordination actions is anything but trivial (Lesser, 1998). The problem becomes even more difficult when agents pursuing partially conflicting goals come into play (Ossowski, 1999). In all but the most simple systems, the instrumentation of these tasks gives rise to complex patterns of interaction among agents. The set of possible interactions is often called the *interaction space* of coordination.

At the symbol level, i.e. from the *engineering* perspective of an actual instrumentation of the model, coordination is probably best conceived of as the effort of *governing the space of interaction* (Busi *et al.*, 2001) of a system. When approaching coordination from a *design* stance, the basic challenge amounts to how to make agents converge on an interaction pattern that adequately (i.e. instrumentally with respect to desired system features) solves the dependency detection and decision tasks. A variety of approaches to tackle this problem can be found in literature. Multi-agent planning, negotiation, organisational structures, conventions, norms, reputation management, mechanism design and so on are just some of them. These approaches aim at shaping the interaction space either directly, by making assumptions on agent behaviours and/or knowledge, or indirectly, by modifying the *context* of the agents in their environment. The applicability of these mechanisms depends largely on the characteristics of the coordination problem at hand, as we will outline in the next section.

3 Dimensions of coordination models

It is a commonplace that, the more open an environment in which a multi-agent system has to act, the more difficult it is to instil a sufficient quality of coordination. For instance, in a closed environment, as assumed traditionally by distributed problem-solving systems, agent behaviour is controlled at *design time*. As the agent designer has full control over the agents, she can implement a coordination mechanism of her choice: if certain assumptions on the agents' behaviours are necessary, these can

simply be "hard-coded" into the agent programs. A popular example is the original Contract-Net Protocol (CNP) (Smith, 1980): volunteering services to a contractor relies on the assumption of benevolence from the side of the bidders, which can be easily achieved when agents are designed to follow CNP. Thus the space of interactions is completely determined at the time the system is built. In closed environments, design choices are usually driven by efficiency issues.

At the other extreme, in large-scale open networks like the Internet, agent behaviour is *uncontrolled*, so that very few assumptions can be made about agents' behaviours and their frequencies. In particular, it is almost impossible to globally foresee and influence the space of potential agent interactions. Most probably, agents will behave in a self-interested fashion, which may help to anticipate some of their actions, and may provide some clues on how to design coordination strategies at the micro level. Also, in certain "parts" of the open system it may be possible to influence the frequencies of behaviours by spawning new agents with desired characteristics, so as to improve the quality of coordination at the macro level (e.g. Park *et al.*, 2000). Still, very few things can be done in the general case, so that the main design focus comes to low-level issues such as security and so on.

A promising approach to efficiently instil coordination in open systems is inspired by the notion of agent as a situated entity, and falls in between the two extremes (fully controlled versus uncontrolled), by providing for a sort of *partially controlled* agent behaviour. Since the environment where agents live is partially under human control, agent interaction can be influenced by engineering the agent environment; in particular, agent *infrastructures* are typically exploited to shape the agent environment according to the engineers' needs. For instance, *coordination infrastructures* provide agents with coordination (run-time) abstractions, embodying *coordination as a service* (Viroli & Omicini, 2002), which exert a form of partial *run-time control* over agent behaviour. Coordination infrastructures are not meant to influence agent behaviour directly, but can affect agent actions. As a trivial example, a security infrastructure enforces a set of behavioural restrictions for potential users that implicitly bounds the admissible agent interaction histories; the agent deliberation capability is thus not limited anyhow, but the effects of the agents' actions on the environment are constrained indeed. Given a range of different *coordination services* made available by an infrastructure, agents can freely choose a service based on their self-interest: once they register to a coordination service, however, the infrastructure will enforce their compliance with the behavioural restrictions. This may be achieved by executing mobile agents on specific virtual machines, or by making them interact through "intelligent" communication channels that govern the interaction space by filtering, modifying or generating certain messages. Future agent-based auctions may become an example of such coordination services.

This form of coordination is often termed *mediated coordination* – which in general could rely on either a distinguished middle agent (Klusch & Sycara, 2001) or a coordination abstraction provided by an infrastructure. In the literature, mediated coordination is often confused with *centralised coordination*. In fact, another important dimension of coordination models amounts to whether they can be designed in a *centralised* fashion, or need a *decentralised* instrumentation. While centralised mechanisms fit closed environments with design-time coordination well, decentralised mechanisms (like peer-to-peer models) better satisfy the needs of open environments with run-time coordination. However, mediated coordination is often *multicentric* – so neither centralised nor fully decentralised – thus achieving a sort of welcome compromise between the engineering urges (pushing towards controlled and predictable systems) and the typical features of the systems of today (emphasising openness, dynamics and unpredictability).

Closely related to the above discussion is a concept recently introduced by Tolksdorf (2000): a coordination mechanism is said to be *coupled* if the effectiveness of an agent's coordination behaviour is based on assumptions about some (other) agent's behaviour. This is the case, for instance, for coordination mechanisms that rely on distributed constraint satisfaction (Ossowski, 2001). By contrast, *uncoupled* mechanisms impose no assumptions on agent behaviour. A truly decentralised coordination can only be achieved by a coupled mechanism, so it bears the additional burden of ensuring that all involved agents will behave as expected.

When shifting our attention to the micro level, the distinction between *quantitative* and *qualitative* models of coordination comes into play (Ossowski, 1999). Qualitative approaches basically follow the

dependency model outlined in the previous sections, by directly representing the different "reasons" for preferring or not certain objects of coordination to others. An agent's coordination behaviour is guided by whether a certain local action (plan, goal and so on) depends positively or negatively on the actions of others: it will choose its local and communicative actions based on the "structure" of dependencies that it shares with its acquaintances. So in cooperative environments it is straightforward to conceive of coordination as a kind of *constraint satisfaction problem* (Ossowski, 1999). In quantitative models, by contrast, the structure of the coordination problem is hidden in the shape of a multi-attribute utility function. An agent has control over only some of the function's attributes (i.e. some of the objects of coordination), and its utility may increase (or decrease) in cases where there is a positive (or negative) dependency with an attribute governed by another agent, but these dependencies are not explicitly modelled. Its local coordination decision problem then corresponds to a special type of *optimisation problem*: to determine a local action (plan, goal and so on), and to induce others to choose local actions (plans, goals and so on), so as to maximise its local utility. The quantitative approach may draw upon a well-founded theoretical framework for both cooperative settings (operations research) and non-cooperative settings (game theory), but suffers from the fact that, due to the uncertainties intrinsic to many AI domains, the utility function is only an approximation, so that its optimum need not coincide with an agent's best choice in the real environment. On the other hand, a coordination mechanism based on qualitative dependencies is less prone to such modelling inaccuracies, but its foundations go back to theories from social sciences (e.g. social psychology), that do not provide a sound formal framework to guide local decision-making.

Finally, the distinction between *subjective* and *objective* coordination, introduced by Schumacher (2001), is closely related to the micro/macro duality discussed in the previous section. Subjective coordination looks at interactions from the agent's own point of view, so it roughly amounts to (i) monitoring all interactions that are perceivable and relevant to the agent, as well as their evolution over time, and (ii) devising which actions the agent could perform that could bring the overall state of the system (or, more generally, of the world) to coincide with one of the agent's goals. Objective coordination conceives of interaction within a multi-agent system from outside the interacting agents, as an external observer; coordination means affecting agent interactions so as to make the resulting evolution of the system accomplish one or more of the designer's goals. It is important to note that affecting agent interactions does not mean totally determining their behaviour, just biasing it in a desired direction, or modifying the "shape" of the interaction space. That is why objective coordination by no means contrasts with the fundamental notion of agent *autonomy* (Wooldridge & Jennings, 1995). Both approaches play a fundamental role in the engineering of agent systems, and any methodology for the design and development of agent systems has to exploit both objective and subjective coordination models and technologies (Omicini & Ossowski, 2003).

4 Coordination and knowledge engineering: the contributions

Coordination is a key aspect in the design of distributed intelligent (multi-agent) systems and, in turn, the capability of coordinating with others is essential to ensure globally coherent reasoning processes within the system: from a multi-agent point of view, it constitutes a centrepiece of agenthood. Roughly speaking, coordination knowledge shapes the mechanisms through which coordination is achieved in a particular domain, be it by characterising and structuring the coordination task at the knowledge level (e.g. in terms of the dependency model), or as an actual symbolic representation upon which inference is performed (e.g. within coordination media that govern interaction). On the other hand, as we have seen in the many dimensions of coordination models, different characteristics of a system's environment may have a deep impact on the type of mechanism that coordination knowledge is to support.

So it is no surprise that many important questions arise, and at quite different levels of abstraction, when the question of coordination knowledge *engineering* comes into play. What is an adequate terminological (and formal) framework to express coordination knowledge and its structure? How far

can such a framework be general, or should it be sensitive to the characteristics of a particular class of application? If coordination knowledge is explicitly modelled, what are adequate formalisms, not just for the representation but also for the instrumentation of that knowledge? How far can and should the properties of coordination infrastructures influence the whole design process? The contributions to this special issue individually focus on quite specific parts of the general problem. Still, altogether they may provide the reader with some clues respecting the lines along which a principled solution to the above questions might develop.

From a (structured) knowledge-level perspective, *organisations* can be seen as a means to rationalise system behaviour. Typically, the concept of *role* is used to characterise observed individual behaviour within a multi-agent system by relating the individuals' interaction with their environment and their acquaintances to the goals that they pursue (or tasks that they are to perform) (Zambonelli *et al.*, 2000). Usually, additional characteristics such as authorisations and permissions (or even communicative competence (Serrano & Ossowski, 2003)) are ascribed to roles. These characteristics can then be used to define the organisation's coordination knowledge – specific rules that shape the space of interaction among roles. In their article "RAMASD: a semi-automatic method for designing agent organisations", Karageorgos, Mehandjiev and Thompson are concerned with principled design methods for multi-agent systems that are driven by the notion of organisation. Roles are proposed as first-class constructs, and a role algebra is outlined that allows the representation of the interdependencies between roles which, among other characteristics, will shape the space of interaction once agents actually play these roles in a concrete implementation. In turn, Davidsson and Wernstedt set out from a particular class of problem, supply-chain management, and discuss the impact of their requirements on the structure of the multi-agent systems in general and the coordination model together with its interaction space in particular. In their article "A multi-agent system architecture for coordination of just-in-time production and distribution", they evaluate their findings in the real-world domain of district heating management.

The contribution by Ciancarini, Tolksdorf and Zambonelli focuses on different types of middleware to support the *instrumentation* of XML-based representations of coordination knowledge for active documents. Their article, "A survey of coordination middleware for XML-centric applications", first describes how active documents can be conceived of as mobile agents (document agents), and sketch the structural requirements that call for the use of coordination models as first-class entities. An extensive review of coordination middleware architectures, which support the instrumentation of coordination knowledge in societies of document agents, is presented, and their strengths and weaknesses are evaluated. Finally, Fiege, Mühl and Gärtner report on advances on event-based systems that serve as the basis for many coordination middleware implementations (besides rule-based systems, active databases and so on). Their article, "Modular event-based systems", puts forward the idea of adding scopes to this kind of system, so as to cope with the complexities of engineering distributed systems and platforms to enact coordination knowledge.

5 Conclusions

The notion of coordination is today relevant in several different research and application areas – AI, among many others – where it has been given a number of different definitions. In this article, we have given a brief overview of current conceptualisations of coordination. Some dimensions along which to organise coordination models have been discussed, and related to the characteristics of the supporting coordination infrastructure.

We suggest that the notion of *coordination knowledge* is essential to the design of coordination models and that, roughly speaking, depending on whether we take a knowledge-level or a symbol-level stance, it relates to the structure or to the actual enactment of the model. From this perspective, *coordination knowledge engineering* comes to be a key task in the design of a heterogeneous distributed intelligent system: the different repercussions of this consideration are further developed and discussed in the four articles constituting the remainder of this special issue.

References

Busi, N, Ciancarini, P, Gorrieri, R and Zavattaro, G, 2001, "Coordination models: a guided tour" in A Omicini, F Zambonelli, M Klusch, and R Tolksdorf, (eds) *Coordination of Internet Agents: Models, Technologies, and Applications* Springer-Verlag.

Casati, F, Castano, S, Fugini, M, Mirabel, I and Pernici, B, 2000, "Using Patterns to Design Rules in Workflows", *IEEE Transactions on Software Engineering* **26**(8) 760–785.

Ciancarini, P and Wooldridge, MJ (eds), 2001, *Agent-Oriented Software Engineering* Lecture Notes on Computer Science, Springer-Verlag.

Cuena, José and Ossowski, S, 1999, "Distributed models for decision support" in Gerhard Weiss (ed.) *Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence* MIT Press.

Decker, K, 1996, "TAEMS: a framework for environment centered analysis and design of coordination mechanisms" in G O'Hare and Nicholas R Jennings (eds) *Foundations of Distributed Artificial Intelligence* John Wiley and Sons.

Denti, E, Omicini, A and Ricci, A, 2002, "Coordination tools for MAS development and deployment" *Applied Artificial Intelligence* **16**(10) 721–752.

Durfee, EH, 2001, "Scaling up agent coordination strategies" *IEEE Computer* **34**(7) 39–46.

Gelernter, D, 1985, "Generative communication in Linda" *ACM Transactions on Programming Languages and Systems* **7**(1) 80–112.

Jennings, NR and Campos, JR "Towards a Social level characterisation of socially responsible agents" *IEE Proceedings on Software Engineering* **144**(1) 11–25.

Klusch, M and Sycara, K, 2001, "Brokering and matchmaking for coordination of agent societies: a survey" in A Omicini, F Zambonelli, M Klusch, and R Tolksdorf (eds) *Coordination of Internet Agents: Models, Technologies, and Applications* Springer-Verlag.

Lesser, VR, 1998, "Reflections on the Nature of Multi-Agent Coordination and Its Implications for an Agent Architecture *Autonomous Agents and Multi-Agent Systems* **1**(1) 89–111.

Malone, T and Crowston, K, 1994, "The interdisciplinary study of coordination" *ACM Computing Surveys* **26**(1) 87–119.

Martial, F von, 1992, *Co-ordinating Plans of Autonomous Agents* Lecture Notes on Artificial Intelligence, Springer-Verlag.

Nash, JF, 1950, "The bargaining problem" *Econometrica* **28** 152–155.

Newell, A, 1982, "The knowledge level" *Artificial Intelligence* **18** 87–127.

Newell, A, 1993, "Reflections on the knowledge level" *Artificial Intelligence* **59** 31–38.

Omicini, A, 2001, "Societies and infrastructures in the analysis and design of agent-based systems" in P Ciancarini and MJ Wooldridge (eds) *Agent-Oriented Software Engineering* Lecture Notes on Computer Science, Springer-Verlag.

Omicini, A and Denti, E, 2001, "Formal ReSpecT" in Agostino Dovier, Maria Chiara Meo and Andrea Omicini, *Declarative Programming – Selected Papers from AGP'00* Electronic Notes in Theoretical Computer Science, Elsevier Science B. V.

Omicini, A and Denti, E, 2001, "From tuple spaces to tuple centres" *Science of Computer Programming* **41**(3) 277–294.

Omicini, A and Ossowski, S, 2003, "Objective versus subjective coordination in the engineering of agent systems" in M Klusch, S Bergamaschi, P Edwards and P Petta (eds) *Intelligent Information Agents – An Agentlink Perspective* Springer-Verlag.

Omicini, A and Papadopoulos, GA, 2001, "Why coordination models and languages in AI?" *Applied Artificial Intelligence* **15**(1) 1–10.

Omicini, A, Zambonelli, F, Klusch, M and Tolksdorf, R (eds), 2001, *Coordination of Internet Agents: Models, Technologies, and Applications* Springer-Verlag.

Omicini, A, Zambonelli, F and Tolksdorf, R (eds), 2000, *Engineering Societies in an Agent World* Springer-Verlag.

Ossowski, S, 1999, *Coordination in Artificial Agent Societies – Social Structure and its Implications for Autonomus Problem-solving Agents* Springer.

Ossowski, S, 2001, "Constraint-based coordination of autonomous agents" *Electronic Notes in Theoretical Computer Science* **48** Elsevier.

Ossowski, S and Garcia Serrano, AM, 1996, "A knowledge level model of co-ordination" in Chengqi Zhang and Dickson Lukose (eds) *Distributed Artificial Intelligence – Architectures and Modelling* Springer-Verlag".

Ossowski, S, Hernández, JZ, Iglesias, CA and Fernández, A, 2002, "Engineering agent systems for decision support" in Paolo Petta, Robert Tolksdorf and Franco Zambonelli (eds) *Engineering Societies in an Agent World III* Springer-Verlag.

Papadopoulos, GA, 2001, "Models and technologies for the coordination of Internet agents: a survey" in A Omicini, F Zambonelli, M Klusch, and R Tolksdorf (eds) *Coordination of Internet Agents: Models, Technologies, and Applications* Springer-Verlag.

Park, S, Durfee, EH and Birmingham, WP, 2000, "Emergent properties of a market-based digital library with strategic agents" *Autonomous Agents and Multiagent Systems* **5** 33–51.

Ricci, A, Omicini, A and Denti, E, 2002, "Virtual enterprises and workflow management as agent coordination issues" *International Journal of Cooperative Information Systems* **11**(3–4) 355–380.

Rosenschein, JS and Zlotkin, G, 1994, *Rules of Encounter – Designing Conventions for Automated Negotiation Among Computers* MIT Press.

Schumacher, M, 2001, *Objective Coordination in Multi-Agent System Engineering – Design and Implementation* Springer-Verlag.

Serrano, JM and Ossowski, S, 2003, "The pragmatics of software agents: analysis and design of agent communication languages" in M Klusch, S Bergamaschi, P Edwards and P Petta (eds) *Intelligent Information Agents – An Agentlink Perspective* Springer-Verlag.

Smith, RG, 1980, "The Contract Net Protocol: high level communication and control in distributed problem solver", *IEEE Transactions on Computers*, **12**(29) 1104–1113.

Thomson, WL, 1994, "Cooperative models of bargaining" in R Aumann and S Hart (eds) *Handbook of Game Theory* North-Holland.

Tolksdorf, R, 2000, "Models of coordination" in A Omicini, F Zambonelli and R Tolksdorf *Engineering Societies in an Agent World* Springer-Verlag.

Van de Velde, W, 1993, "Issues in knowledge level modelling" in R Simmons (ed.) *Second Generation Expert Systems* Springer-Verlag.

Viroli, M and Omicini, A, 2002, "Coordination as a service: ontological and formal foundation" *First International Workshop on Foundations of Coordination Languages and Software Architectures* (FOCLASA'02) Volume 68 of Electronic Notes in Theoretical Computer Science, Elsevier.

Wegner, P, 1997, "Why interaction is more powerful than computing "*Communications of the ACM* **40**(5) 80–91.

WfMC, "Workflow Management Coalition Home Page, `http://www.wfmc.org/`.

Wooldridge, MJ and Jennings, NR, 1995, "Intelligent agents: theory and practice" *The Knowledge Engineering Review* **10**,(2) 115–152.

Zambonelli, F, Jennings, NR and Wooldridge, MJ, 2001, "Organisational abstractions for the analysis and design of multi-agent systems" in P Ciancarini and MJ Wooldridge (eds) *Agent-Oriented Software Engineering* Lecture Notes on Computer Science, Springer-Verlag.

Zambonelli, F, Jennings, NR, Omicini, A and Wooldridge, MJ, 2001, "Agent-oriented software engineering for Internet applications" in A Omicini, F Zambonelli, M Klusch, and R Tolksdorf (eds) *Coordination of Internet Agents: Models, Technologies, and Applications* Springer-Verlag.