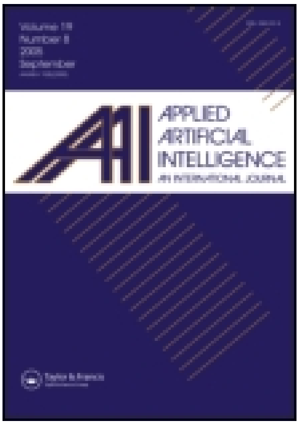

Applied Artificial Intelligence
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/uaai20

Editorial: Why coordination models and languages in ai?

Andrea Omicini [a] & George A. Papadopoulos [b]
[a] DEIS, UniversitÀ di Bologna, Bologna, Italy
[b] Department of Computer Science, University of Cyprus, Nicosia, Cyprus

Available online: 26 Nov 2010

To cite this article: Andrea Omicini & George A. Papadopoulos (2001): Editorial: Why coordination models and languages in ai?, Applied Artificial Intelligence, 15:1, 1-10

To link to this article: http://dx.doi.org/10.1080/08839510150204581

Applied Artificial Intelligence, 15:1–10, 2001
Copyright © 2001 Taylor & Francis
0883-9514/01 $12.00 +.00

EDITORIAL: WHY COORDINATION MODELS AND LANGUAGES IN AI?

ANDREA OMICINI
DEIS, Università di Bologna, Bologna, Italy

GEORGE A. PAPADOPOULOS
Department of Computer Science, University of Cyprus, Nicosia, Cyprus

The pervasive spreading of information technology, along with the continuously growing and almost ubiquitous request for intelligence in systems, is giving new life to AI research. At the same time, this introduces new issues in the AI field, like the engineering of intelligent systems, *which calls for manageable abstractions, methodologies, and technologies effectively supporting engineers in the design, development, and deployment of intelligent systems.*

Interaction *seems to be the most relevant feature of today's complex systems, so that new models and technologies are emerging that focus on engineering as an independent dimension in the modelling and engineering of hardware and software systems. In particular, research on* coordination models *and* languages *has provided computer scientists and engineers with abstractions and tools to model and shape the space of component interaction in multicomponent systems.*

In this article, the impact of coordination models and languages on the process of engineering intelligent systems is discussed, particularly when they are built and organized as multiagent systems. This topic is first speculated on in general, then some specific coordination issues in the context of intelligent systems are introduced, which are subsequently developed and discussed in the four articles constituting the remainder of this special issue.

BUILDING INTELLIGENT SYSTEMS

The problem of building intelligent systems is rapidly crossing the narrow boundaries of academia to become an industrial issue. The widespread access to information technology by millions of typically low-skilled newcomers has first summoned lots of attention and work on system usability, and then has made the request for intelligence almost inescapable.

Address correspondence to Andrea Omicini, DEIS, Universitá di Bologna, Viale Risorgimento, 2 40136 Bologna, Italy. E-mail: aomicini@deis.unibo.it

In this context, the acceptation of the term "intelligence" is 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. This makes acceptable, in principle, the assumptions that the very idea of "building intelligent systems" implies: that there is some notion of nonhuman intelligence, and that one can build systems that embed such a notion.

Historically, the first feature that has labelled computers as intelligent entities is their ability to overcome humans in solving very complex problems within limited and well-defined scopes—like computing primes or playing chess. Even though Brooks (1999) clearly pointed out the limitations of this acceptation of intelligence as pure mechanical manipulation of symbols (the so-called "traditional AI" view), the advantages of computer-based systems in handling huge amounts of data, symbols, and rules with respect to the human brain indeed led to spectacular results, like the automatic proof of the four color theorem (Saaty, 1977) or IBMs Deep Blue beating Garry Kasparov. As a consequence, a plethora of models and systems have been defined and built—like expert systems, inference engines, constraint systems—which typically exploit computers' brute computational power to provide very specialized *quanta* of intelligence within closed and well-delimited domains.

Instead, the more sophisticated idea of intelligence promoted by Brooks' work (the so-called "behavior-based AI" view) is related to the notion of *emergence* (Pfeifer & Scheier, 1999). There *interaction*, rather than computation abilities, is the core in that intelligence is not somehow contained within an artificial *agent*, but is the result of agent interaction with the environment. So agent intelligence is not the mere result of its preprogrammed symbol manipulation skills, but emerges from its ability to get inputs from the outside and elaborate on them—in either a symbolic or nonsymbolic way. In this acceptation, then, the ability to shape the space of agent interaction is an essential precondition to make intelligence emerge.

Overcoming the traditional boundaries of the AI field, the notion of agent (Wooldridge & Jennings, 1995) has nowadays become almost ubiquitous as a key abstraction around which systems can be built (Garijo & Boman, 1999; Ciancarini & Wooldridge, 2000). Agent-based approaches are more and more proving to be powerful and expressive enough to overcome the typical problems of the design and development of today's complex systems. Issues like openness, physical distribution, heterogeneity, decentralization of control, and unpredictability can be better faced by exploiting agents' autonomy, their ability to react to environment changes, and their deliberative capabilities (Wooldridge & Jennings, 1997). More in general, agents are the most natural place to encapsulate intelligence, whatever notion of intelligence is endorsed. As a result, multiagent systems—as societies of intelligent agents—are the typical way in which most intelligent systems are conceived and built (Omicini, Tolksdorf, & Zambonelli, 2000).

So building intelligent systems today is basically a matter of putting many different *sources of intelligence* together. Both reactive (like expert systems, inference engines) and active (like agents) components have to be combined so as to make their *quantum* of intelligence fully available to the overall system. On the one hand, making the components of an intelligent system work together and interact in an effective way is basically a *coordination* issue. On the other hand, building (intelligent) systems by combining (possibly intelligent) subsystems also raises new problems—given that traditional models, tools, and methodologies, like object-oriented ones, fall short in supporting the engineering of intelligent systems (Wooldridge & Jennings, 1999). Thus, the design and development of intelligent systems calls for novel *coordination abstractions* and *tools* enabling engineers to effectively tackle the complexity of the interaction space. The role of *coordination models* and *languages* is to work precisely as the natural sources of such abstractions and tools.

COORDINATION MODELS AND LANGUAGES

According to Malone and Crowstone (1994) and Wegner (1996), the term "coordination" basically means the management of the interaction among the entities of a system—whether they are agents, processes, molecules, individuals, or whatever. Generally speaking, the notion of coordination is a multidisciplinary one, and has been given several specific definitions in the many research areas where it is relevant and commonly used, such as programming languages, parallel and distributed systems, (distributed) artificial intelligence and multiagent systems, Internet technologies, and software engineering.

In particular, a common view in (D)AI sees coordination as the result of the attitude of each individual towards the organization/society it belongs to (Jennings, 1996). The beliefs, goals, and role of each individual, and its subjective perception of the interdependencies among the members of a society determine the coordination of the overall system. Even in the general overview of coordination in artificial agent societies provided by Ossowski (1998), individual and global issues of coordination are intermixed without an easy way to keep them distinct.

However, separation of concerns is typically one of the easiest ways to abate complexity. Indeed, charging an agent of both its individual goals and the burden of handling all the intricacies of interaction in environments that may be open, physically distributed, and unpredictable, does not make agent design and development an easy task (Ciancarini, Omicini, & Zambonelli 2000). Instead, a model of coordination keeping the individual perception of coordination distinct from the global coordination issues would make it possible to model and shape the interaction space somehow independently of the interacting entities. This is what has been called *objective coordination* (Schumacher, 2000), since it prescinds from the subjective view of coordination or also *uncoupled coordination* (Tolksdorf, 2000), since coordination is no longer coupled with the computational issues of the coordinated entities.

The separation between computation and coordination was first promoted by Gelernter and Carriero (1992) in the field of the languages for parallel and distributed systems. More generally, Wegner (1997) clearly showed the conceptual distinction between computation and coordination as two orthogonal dimensions for programming languages, and discussed their different expressive power in terms of abstract machines. In all, this suggests that suitable models of coordination, endorsing the objective/uncoupled acceptation of this term, could have a significant impact over the engineering of complex systems, like intelligent ones.

This is precisely the specific notion of *coordination model* as it comes out from the research on coordination models and languages (SAC, 1998, 1999, 2000; Ciancarini & Hankin, 1996; Garlan & Le Métayer, 1997; Ciancarini & Wolf 1999; Omicini, Zambonelli, Klusch, & Tolksdorf, 2000). There, a coordination model is, first of all, a conceptual framework to model the space of interaction. More precisely, one can think of a coordination model as consisting of three elements (Ciancarini et al., 2000):

- the *coordinables*—that is, the entities whose mutual interaction is ruled by the model,
- the *coordination media*—that is, the abstractions enabling the interaction among coordinables (e.g., semaphores, monitors, channels, blackboards, etc.), and
- the *coordination laws*—that is, the rules governing the interaction among coordinables and coordination media, as well as the behavior of the coordination media in response to interaction events.

In turn, a *coordination language* is a linguistic reification of a coordination model. For a comprehensive outlook over the field of coordination models and languages, the interested reader is forwarded to Papadopoulos and Arbab (1998), Busi, Ciancarini, Gorrieri, and Zavattaro (2000), and Papadopoulos (2000).

Coordination media, like *tuple spaces* (Gelernter, 1985) or *manifolds* (Arbab, Herman, & Spilling, 1993), can be exploited as the core around which the component of an intelligent system can be organized. In particular, by working as the natural place where the laws of coordination can be embodied, a coordination medium allows the designer of an intelligent system to define coordination rules separately from the interacting entities, and to encapsulate them into a dedicated abstraction.

This suggests how coordination models and languages could impact on the design and development of intelligent systems. In particular, according to Ciancarini et al. (2000), this provides the engineers of intelligent systems with fundamental properties like ***modularity*** and ***reusability***. A component encapsulating some form of intelligence, like an expert system or an intelligent agent, could in fact work as an "intelligent module" and be reused wherever its capabilities are needed by defining coordination rules according to the component interaction pattern, and by encapsulating them into a coordination medium. Even more, since they are encapsulated into different components, individual abilities and coordination rules could be refined and modified independently, thus providing for ***incremental design*** and ***development***. For a simple example of an intelligent multiagent system designed around a suitably expressive coordination abstraction the interested reader is forwarded to Denti and Omicini (1999).

From an engineering viewpoint, a coordination model can then be exploited as a source for the design metaphors, abstractions, and mechanisms that are required to support the definition of the architecture of an intelligent system, as well as its development and deployment. That is why coordination models and languages are likely to become a core notion also in the methodologies and processes for the engineering of intelligent systems: Zambonelli, Jennings, and Wooldridge (2000), Omicini (2000), and Zambonelli, Jennings, Omicini & Wooldridge (2000) report some preliminary results on this topic.

COORDINATION ISSUES IN INTELLIGENT SYSTEMS

The origin of the literature on coordination models and languages dates back to Linda (Gelernter, 1985)—perhaps the only coordination model widely known also outside the boundaries of this research field. While Linda was conceived in the field of parallel programming, with closed implementation and compiler-based optimization techniques (Rowstron, 2000), its features made it viable for open systems, too, and led to the definition of a plethora of derived ***tuple-based models*** for the coordination of open, heterogeneous, and distributed multicomponent systems (Rossi, Cabri, & Denti, 2000). In particular, several tuple-based coordination models were defined for multiagent systems and also used for the coordination of intelligent components/agents.

However, before they can be effectively exploited in the engineering of intelligent systems, coordination models have to address some key issues like ***security***. In particular, ***authentication*** and ***authorization*** are particularly relevant issues when one considers that intelligent systems are likely to gain more and more acceptance, become an everyday technology we all rely on, and act on one's behalf. In this context, *Safe Tuplespace-Based Coordination in Multi Agent Systems*, by Naftaly H. Minsky, Yaron M. Minsky, and Victoria Ungureanu (2001), discusses a coordination model that extends the basic tuple space model of Linda with the notion of *law-governed interaction* towards the definition of *safe tuple spaces*. Since they both concern the ruling of interaction, coordination and security are clearly related issues, and should be addressed as such in the same phase of the engineering of systems (Bryce & Cremonini, 2000). Correspondingly, this article provides the reader with a framework where both coordination and security can be expressed in a uniform way in the context of open and distributed systems, in general, and of multiagent systems, in particular.

XML Dataspaces for the Coordination of Internet Agents by Giacomo Cabri, Letizia Leonardi, and Franco Zambonelli (2001), presents the MARS-X architecture for the coordination of mobile and intelligent Internet agents, based on *programmable XML dataspaces*. In the MARS-X model, tuple-based coordination is extended in two directions: programmability of the coordination abstraction (Denti, Natali, & Omicini, 1998), which is then enabled to encapsulate any required (computable) coordination rule, and XML tuples, which allow XML (World Wide Web Consortium, 2000) to be exploited as the standard communication language in a tuple-based coordination model. This approach shows how the issues of agent communication and knowledge-sharing can be expressed within the scope of a coordination model, and sets a viable path towards the integration of coordination models and agent communication languages (Finin, Labrou, & Cost, 2000). This article also suggests that *intelligence* and *mobility* are not dual issues in the context of multiagent systems—instead, they are orthogonal design dimensions that could find an effective and uniform support in a suitably defined coordination architecture.

The intrinsic complexity of handling multiagent systems promotes the adoption of declarative approaches, which are supposed to make the specification of complex systems easier. In *A Coordination Language for Collective Agent Based Systems: GroupLog*, Fernanda Barbosa and José C. Cunha (2001) describe a logic-based coordination framework, and define the GroupLog language for the specification of multiagent systems. There, the organizational view of coordination is captured by the notion of *group*, and extended Horn clauses are used to specify both individual agents and groups. Several interagent communication patterns coexist in GroupLog, so that, for instance, interagent conversations may occur through point-to-point communication at the same time as tuple-based coordination is exploited to define agent cooperative behaviors.

Finally, *Control-Driven Constraint Propagation*, by Eric Monfroy (2001), shows that coordination model is not synonymous with Linda, and that coordination of intelligent systems is a more general issue than the coordination of multiagent systems. Papadopoulos and Arbab (1998) classify coordination models as either *data-driven* or *control-driven* models. Roughly speaking, while data-driven models (like Linda and all the tuple-based models) make coordination depend on the data exchanged among interacting entities, control-driven models (like MANIFOLD (Arbab et al. 1993)) focus on the acts of communication, rather than on data.

In this article, the author shows how a coordination language (namely, MANIFOLD) can be exploited for coordinating constraint propagation within a constraint solver. Even though it addresses quite a specific problem, this article also gives a new perspective on the issue of coordinating intelligent systems, by implicitly suggesting that the conceptual support provided by a coordination model can be fruitfully exploited to effectively combine distinct, possibly distributed, and heterogeneous "sources of intelligence", like constraint solvers and inference engines (see also Monfroy and Arbab (2000) for a comprehensive outlook on this subject).

CONCLUSIONS

The notion of coordination is relevant today in several different research and application areas—AI, among many others—where it has been given a number of different definitions. It is suggested that the acceptation coming from the field of coordination models and languages could be relevant for AI, in particular in the engineering of intelligent systems. In fact, it promotes an uncoupled notion of coordination model, which enables engineers to handle computation and coordination independently, thus keeping elaboration both conceptually and practically distinct from interaction in the design of intelligent components and systems.

In this paper, this topic was discussed, in general, then some specific coordination issues were introduced in the context of intelligent systems, which are subsequently developed and discussed in the four articles constituting the remainder of this Special Issue. There, different coordination models and languages are successfully exploited in the specification and construction of complex systems including intelligent components.

ACKNOWLEDGMENTS

This Special Issue is an outcome of the *Special Track on Coordination Model, Languages and Applications* of the ACM 2000 Symposium on Applied Computing (SAC, 2000). This year, 52 high-quality papers were submitted to the Special Track, resulting in 17 accepted papers (plus two short

papers). Among the accepted ones, four papers were further selected for the project *Coordination Models and Languages in AI*. These papers were extended and reviewed according to the results of the track discussion, the reviewers' remarks, and the guest editors' recommendations, and were finally included in this Special Issue.

As Guest Editors of this Special Issue, we would like to thank first, all the SAC submitting authors and reviewers, as well as the track's attendants, for their contributions to the track success that made this project possible. In particular, we would like to thank all the authors who agreed to participate in this project—they devoted lots of their time and energies to fulfill the project goals. Then, thanks to Barrett R. Bryant, Gary B. Lamont, Jan Carroll, Hisham (Al) Haddad, and all the ACM SAC people for their work and support.

Also, thanks to Marco Cremonini, Michela Milano, Antony Rowstron, Christophe Sibertin-Blanc, Robert Tolksdorf, Paolo Torroni, and Gianluigi Zavattaro, who contributed to the quality of the articles of this Special Issue by providing authors with helpful and in-depth reviews. Then, thanks to Enrico Denti and Franco Zambonelli, whose bright remarks and strong criticisms helped a lot to improve this article.

Finally, we would like to thank Paolo Petta, who first conceived this Special Issue, and the Editor of this journal, Robert Trappl, who promptly supported the project.

REFERENCES

Arbab, F., I. Herman, and P. Spilling. 1993. An overview of MANIFOLD and its implementation. *Concurrency: Practice and Experience* 5(1)23–70.

Barbosa, F., and J. C. Cunha. 2001. A coordination language for collective agent based systems: GroupLog.

Brooks, R. A. 1999. *Cambrian Intelligence: The early history of the new AI*. Cambridge MA: The MIT Press. In this Issue.

Bryce, C., and M. Cremonini. 2001. *Coordination and security on the internet*. In *Coordination of Internet agents: Model technologies and applications*, Chapter 9. Springer-Verlag.

Busi, N., P. Ciancarini, R. Gorrieri, and G. Zavattaro. 2001. *Coordination models: A guided tour*. In *Coordination of Internet agents: Model technologies and applications*, Chapter 1. Springer-Verlag.

Cabri, G., L. Leonardi, and F. Zambonelli. 2001. XML dataspaces for the coordination of Internet agents. In this Issue.

Ciancarini, P., and C. Hankin, eds. 1996. Coordination Languages and Models. (*COORDINATION'96*), 1061, *LNCS*, Springer-Verlag.

Ciancarini, P., and A. L. Wolf, eds. 1999. Coordination Languages and Models. (*COORDINATION'99*), 1594, *LNCS*, Springer-Verlag.

Ciancarini, P., and M. J. Wooldridge eds. 2000. Agent Oriented Software Engineering. (*AOSE 2000*), 1957, *LNCS*, Springer-Verlag.

Ciancarini, P., A. Omicini, and F. Zambonelli. 2000. Multiagent system engineering: The coordination viewpoint. In N. R. Jennings and Y. Lespérance, eds., *Intelligent Agents VI—Agent Theories, Architectures, and Languages*, 1767, *LNAI*, Springer-Verlag, pp. 250–259.

Denti, E., and A. Omicini. 1999. Designing multi-agent systems around an extensible communication abstraction. In J.-J. C. Meyer and P.-Y. Schobbens, eds., *Formal Models of Agents—ESPRIT Project ModelAge Final Report*, 1760, *LNAI*, Springer-Verlag, pp. 90–102.

Denti, E., A. Natali, and A. Omicini. 1998. On the expressive power of a language for programming coordination media. In *SAC*, 1999, pp. 169–177.

Finin, T., Y. Labrou, and R. Scott Cost. 2001. Coordinating agents using agent communication languages conservations. In *Coordination of Internet agents: Models, technologies and applications*, Chapter 7. New York: Springer-Verlag.

Garijo, F. J., and M. Boman, eds. 1999. Multi-Agent Systems Engineering. (*MAMAAW'99*), 1647, *LNAI*, Valencia (E). Springer-Verlag.

Garlan, D., and D. Le Métayer, eds. 1997. Coordination languages and Models. (*COORDINATION'97*) 1282, *LNCS*, Springer-Verlag, Berlin (D).

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

Gelernter, D., and N. Carriero. 1992. Coordination languages and their significance. *Communications of the ACM* 35(2):97–107.

Jennings, N. R. 1996. Coordination techniques for distributed artificial intelligence. In G. M. P. O'Hare and N. R. Jennings, eds., *Foundations of distributed artificial intelligence.* New York: John Wiley & Sons, pp. 187–210.

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

Minsky, N. H., Y. M. Minsky, and V. Ungureanu. 2001. Safe tuplespace-based coordination in multi agent systems. In this Issue.

Monfroy, E. 2001. Control-driven constraint propagation. In this Issue.

Monfroy, E., and F. Arbab. 2000. Coordination of inference engines. In *Coordination of Internet agents: Models, technologies and applications*, Chapter 16. New York: Springer-Verlag.

Omicini, A. 2000. SODA: Societies and infrastructures in the analysis and design of agent-based systems. In *Agent-oriented Software engineering*. Springer-Verlag.

Omicini, A., R. Tolksdorf, and F. Zambonelli. eds. 2000. *Engineering Societies in the Agents' World* (*ESAW'00*), *LNCS*. Springer-Verlag.

Omicini, A., F. Zambonelli, M. Klusch, and R. Tolksdorf. eds. 2001. *Coordination of Internet agents: Models, technologies and applications*. Springer-Verlag.

Ossowski, S. 1998. *Co-ordination in artificial agent societies*. 1535, *LNAI*. Springer-Verlag.

Papadopoulos, G. A. 2001. Models and technologies for the coordination of internet agents: A survey. In *Coordination of Internet agents: Models, technologies and applications*. Chapter 2.

Papadopoulos, G. A., and F. Arbab. 1998. Coordination models and languages. In M. V. Zelkowitz, ed., The engineering of large systems, Vol. 46, *Advances in Computers*. New York: Academic Press, 329–400.

Pfeifer, R., and C. Scheier. 1999. *Understanding intelligence*. Cambridge, MA: The MIT Press.

Rossi, D., G. Cabri, and E. Denti. 2001. *Tuple-based technologies for coordination*. In *Coordination of Internet agents: Models, technologies and applications*, Chapter 4. Springer-Verlag.

Rowstron, A. 2001. *Run-time systems for coordination*. In *Coordination of Internet agents: Models, technologies and applications*, Chapter 3. Springer-Verlag.

Saaty, T. 1977. *The four-color problem: Assaults and conquest*. New York: McGraw Hill, Intl.

SAC. 1998. *Proceedings of the 1998 ACM Symposium on Applied Computing* (*SAC'98*), ACM, Atlanta, GA. Track on Coordination Models, Languages and Applications. http://www.cs.ucy.ac.cy/SAC98.html

SAC. 1999. *Proceedings of the 1999 ACM Symposium on Applied Computing* (*SAC'99*), ACM, San Antonio, TX. Track on Coordination Models, Languages and Applications. http://www.cs.ucy.ac.cy/SAC99.html

SAC. 2000. *Proceedings of the 2000 ACM Symposium on Applied Computing* (*SAC 2000*), ACM, Como, Italy. Track on Coordination Models, Languages and Applications. http://lia.deis.unibo.it/confs/SAC00

Schumacher, M. 2000. *Designing and implementing objective coordination in multi-agent systems*, Ph.D. thesis, Université de Fribourg, Switzerland.

Tolksdorf, R. 2000. Models of coordination. In *Engineering Societies in the Agents' World* (*ESAW'00*)—*Proc. 1st International Workshop*, pp. 75–89. *LNCS*. Berlin: Springer-Verlag.

Wegner, P. 1996. Coordination as constrained interaction. In *Coordination Languages and Models* (*COORDINATION '96*), pp. 28–33.1061, *LNCS*, Springer-Verlag.

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

Wooldridge, M. J., and N. R. Jennings. 1995. Intelligent agents: Theory and practice. *The Knowledge Engineering Review* 10(2): 115–152.

Wooldridge, M. J. 1997. Agent-based software engineering. *IEE Proceedings on Software Engineering* 144(1):26–37.

Wooldridge, M. J., and N. R. Jennings. 1999. Software engineering with agents: Pitfalls and pratfalls. *IEEE Internet Computing* 3(3):20–27.

World Wide Web Consortium. 2000. XML—Extensible Markup Language. http://www.w3.org/XML

Zambonelli, F., N. R. Jennings, and M. J. Wooldridge. 2000. Organisational abstractions for the analysis and design of multi-agent systems. In *Agent-Oriented Software Engineering* (*AOSE 2000*), *LNCS*. Springer-Verlag.

Zambonelli, F., N. R. Jennings, A. Omicini, and M. J. Wooldridge. 2001. Agent-oriented software engineering for Internet applications. In. *Coordination of Internet agents: Models, technologies, and applications*. Springer-Verlag.