# Exposing Agents as Web Services in JADE

Arthur Casals, Amal El Fallah-Seghrouchni, Orso Negroni, Anthony Othmani

# Exposing agents as web services in JADE

Arthur Casals[1,2], Amal El Fallah Seghrouchni[1], Orso Negroni[1], and Anthony Othmani[1]

[1] Sorbonne Université, LIP6- CNRS UMR 7606
4 place Jussieu, 75005 Paris - France
arthur.casals@lip6.fr,amal.elfallah@lip6.fr, orso.negroni@etu.upmc.fr, anthony.othmani@gmail.com
[2] IPSA (Institut Polytechnique des Sciences Avancées)
63 Boulevard de Brandebourg, 94200 Ivry-sur-Seine, France

**Abstract.** The objective of this chapter is to revisit and explore how intelligent agents can be used in conjunction with modern Web technologies. We use JADE and BDI4JADE to expose cognitive agents using a BDI architecture as web services that can be integrated with different modern cloud-based services, such as Amazon AWS services and Google Home.

**Keywords:** Multi-Agents Systems · Web Service Agents · JADE/WSIG.

## 1 Introduction

Agents can be described as relatively independent and autonomous entities that can be used to solve problems of different complexity degrees [16]. These entities can be organized in communities and work together using different interaction mechanisms in order to solve complex problems. Such communities are also known as multiagent system (MAS): a system composed of multiple agents that interact among themselves in a single environment [37].

Due to their autonomous and interactive capabilities, agents can be seen as a paradigm for software engineering. In fact, since software architectures may contain many different components, each one with its own thread of control and attributions, agents can be used as a paradigm when designing such systems [42]. Moreover, such paradigm can be adopted within complex systems, since interaction is one of the most important characteristics of complexity in software

Using agents and MAS in conjunction with web services has been explored for a long time, from simple interactions between web services and agents [3, 15, 17, 23, 27] to using existing software engineering methodologies such as model-driven development (MDD) [7, 33, 43] to support the development of web-based MAS [29, 38, 39].

Using agents exposed as web services to design systems for the Semantic Web [5] was also studied [13, 36]. Designing such systems involves not only creating systems with distributed capabilities, but also systems capable of using existing communication protocols and mechanisms in order to share and reuse knowledge. This is so because the Semantic Web is an extension of the web by adding semantics to the current data format representation in order to turn web information understandable not only for humans but also for software entities [5].

However, despite the extensive research already done in this field, the advent of new paradigms such as the Web of Things (WoT) [18] has encouraged a revisitation on the use of agents in the Web as a viable approach for deploying autonomous systems [9]. This idea revolves around the fact that in the WoT, it is possible for web services to perceive the real world and act on it though the use of physical devices (sensors).

In this context, our research interests reside in the Agent-Oriented Software Engineering (AOSE) [24] domain, focused on agents capable of interacting with the Web and its services. We refer to such agents as *web-based agents*. Our objectives are (i) to review the existing software engineering techniques used in conjunction with web agents and (ii) to explore how web agents and MAS can be extensively used with different new Web-related paradigms. In particular, we would like to understand the difficulties involved in the use of multiagent technologies in real-world situations. This also involves studying the use of well-established multiagent platforms and concepts by developers and development teams already employed in the industry, which are not necessarily familiar with agents. Before exploring these topics, however, it is necessary to better understand the current state-of-the-practice on web-based agents and related frameworks. For that reason we established a process composed of three different steps: (i) reviewing existing work relating agents and the Web; (ii) evaluating existing multiagent frameworks and Web-related capabilities; and (iii) implementing web-capable agents (exposed as web services) to serve as an evolving proof-of-concept, to be used during the course of our work. This process is fundamental to our subsequent research, and it constitutes the basis of the present chapter.

With that in mind, we implemented a Smart Agenda MAS using one of the multiagent platforms of our interest (JADE). At the same time that we understand this platform is fairly mature, we would like to understand how difficult it would be for students recently introduced to multiagent theory to use and apply this platform in the process of solving a problem based in a real-world scenario. For this reason, an agent-based personal assistant was proposed as the evolving proof-of-concept.

This chapter is organized as follows: in Section 2, we provide some background on intelligent agents. Section 3 contains some of the existing work related to using agents in conjunction with Web-related technologies. In Section 4, we illustrate the implementation of the Smart Agenda system, using agents deployed as a web services. Finally, Section 5 presents some discussion and concludes this chapter with some perspectives for future work.

## 2 Background

In this section we present some concepts that will be used along this chapter. First we provide a brief description of web services, followed by an overview on intelligent agents (with emphasis on one particular architecture). After that we provide an overview on languages used by the agents to interact between themselves, referred to as *agent communication languages*.

## 2.1 Web services

Web services can be generically described as "a software system designed to support inter-operable machine-to-machine interaction over a network, providing a standard means of inter-operating between different software applications, running on a variety of platforms and/or frameworks"[1]. It possesses an interface that allows other systems (or web services) to interact with it using structured messages [2].

In practical terms, a web service can be described as a mean to interact with existing applications using Web-related technologies. A web service can be *discoverable* if it is published in a registration or directory service, which acts as an yellow-pages equivalent. Its specification also includes a *Web Service description* defining all message formats, protocols, and data types used by the web service. From this description, valid messages can be formed and used to communicate with the web service in question. When a web service is published in a directory service, its description is also included.

From a distributed systems perspective, web services enable multiple applications to interact between themselves using Web technologies and protocols [1]. A web service can be used to remotely access a database, or to execute specific processes in a remote machine with a certain set of parameters. They embody the concepts of a service-oriented architecture (SOA)[2] [14]. This allows for different web services to be used together in order to obtain the functionalities of an equivalent larger system.

## 2.2 Intelligent agents

An agent is a computer system with autonomous capabilities, which are able to take decisions on what is needed to do in order to satisfy their objectives [42]. While this definition is not consensual, it revolves around the concept of autonomy.

Agents are also capable of interacting with other agents, exchanging data and engaging in complex interactions such as cooperation, coordination, and negotiation. A multiagent system is a system composed of multiple agents that interact among themselves in a single environment [37]. The main purpose of a MAS is to autonomously solve complex problems that would otherwise be impossible (or very difficult) to solve by using a monolithic system.

Intelligent agents are capable of reason and decide which action to perform according to available information. This information usually is related to the conditions or consequences of the actions to be executed [42]. Taking advantage

[1] https://www.w3.org/TR/ws-arch/
[2] http://www.opengroup.org/subjectareas/soa

of the agents' inherent capabilities allow MAS to adapt to changes in the environment. Also, the use of multiple agents allow for cooperation mechanisms to take place in situations where solving the problem at hand is beyond the capabilities of any single agent. Applications of MAS include manufacturing control, production planning, logistics, transportation, among others [35].

BDI is one of the software architectures used to model and implement intelligent agents. It is based on the human practical reasoning model [6] and any BDI agent uses the concepts of belief, desire, and intention in its reasoning process. These concepts are used by agents in a means-ends reasoning process. In this sense, the agent's actions are organized in an execution plan built on top of what it believes to be true, and what it desires to achieve as a goal. This reasoning process is usually referred to as "planning" [26]. By using this separation between information, planning, and execution, the agent is capable of adapting to different conditions presented by the environment it is situated in.

## 3 Related work

The idea of integrating agents and the web services has been proposed (and experimented on) since the late 1990s [3, 15, 17, 23, 27]. The main idea at that time was to use agents as assistants for web-based services, such as web stores [3], or even for web browsing [27].

In particular, the idea of using agents *as* web services and using the Semantic Web appeared in early 2000s [20, 22]. Specifically, the use of BDI agents as web services was also proposed around that time by Dickinson and Wooldridge [13]. Their work was later referenced in different works using agent as web services, including collaborative testing [4] and e-learning [21].

Part of the subsequent work on agents as web services, however, was also related to web-based MAS [29, 38, 39] and agent architectures [34, 40]. Software engineering methodologies related to this topic were also being developed and studied [19, 25, 45]. In 2007, a survey on existing agent methodologies based on the agent-oriented paradigm was published [8] with the aim of evaluating in what extent existing agent-oriented methodologies were also oriented to the development of services. From the AOSE perspective, the study served to demonstrate that all evaluated methodologies could be used to model service-oriented agents. At this point, the focus was on modeling agents as services, or generally using the concepts around services in order to facilitate the interaction between agents.

In 2010, however, Ricci *et al* introduced a platform called CArtAgO-WS [36], intended to allow the development of service-oriented applications (SOA) populated by agents. This was a more embracing approach from the MAS perspective, since it took into account both the concepts of artifacts - "objects" or services that could be used by the agents - and the use of heterogeneous agent architectures (including BDI). This idea was further explored and used in conjunction with other agent frameworks, such as JaCaMo [1] (we will briefly introduce this

[1] http://jacamo.sourceforge.net/

framework later in the text). More recently, another step in the same direction was proposed by Ciortea *et al* [10]: they proposed that the World Wide Web (WWW), in its current state, could be suitable as a middleware for Internet-scale multiagent systems. This would be achieved through the use of a resource-oriented layer that would allow the application environment to be decoupled from its deployment context. In order to demonstrate this proposal, an agent environment was developed using JaCaMo. This environment was specifically aimed at the Internet of Things (IoT), emphasizing the separation between the application environment and the deployment context.

It is interesting to notice, however, that while different research has been performed in web-based MAS, most of this work perceives the Internet environment as a means of communication. Even larger proposals such as Agentcities [41] and distributed platforms for data processing [32] rely on web protocols and agent communication standards to establish and coordinate the communication processes, while the agents - although exposed as services and sharing resources - are reasoning blocks taking advantage of such structure. In addition, the lower-level communication mechanisms are still based in procedure calls, not taking advantage of newer mechanisms. There is very recent work on this field that aims at addressing such issues [9, 10].

In the current state of the Web, using it as a middleware for MAS is certainly appealing. At the same time, modeling MAS while taking full advantage of the web environment (not only as a communication-enabled environment) is partially subject to the constant evolution of the Internet itself. From a higher-level perspective, it is possible to model web-based agents and MAS using existing AOSE methodologies (such as GAIA [44] and O-MASE [11, 12]) while abstracting the Internet environment. From a lower-level perspective, however, different communication protocols, web-specific components and technologies can influence the way web-based agents and MAS can be build and deployed into the Internet environment.

Our motivation comes from this context. From the modeling perspective, we would like to explore how we could structure web-based agents that could take full advantage from the Internet environment. Also, from the implementation perspective, we would like to explore how the newest mechanisms and protocols could be used by agents and MAS when deployed into the Internet.

## 4 Smart Agenda

As part of our study process, we modeled a simple MAS that used agents deployed as web services. The objective of this step was to verify how this could be built and deployed using the current technologies and tools. As mentioned before, we chose JADE as the target framework. Part of the reason behind this choice was related to the framework's web-related functionalities previously described. While the modeled MAS is not complex, we intend to explore and evolve this implementation as our research advances.

Integrating Web Services and JADE agents has already been done in the past [28, 30, 31]. Nevertheless, we decided to design a simple MAS completely based in agents deployed as web services. For that purpose, we adopted a JADE add-on aimed at providing support for exposing agents as web services called WSIG[1]. This add-on acts as a relaying gateway, handling all requests coming from the Web and sending them to the MAS.

In order to be exposed as a web service, a JADE agent must possess an *Ontology* and a set of *Actions*. Ontologies are structures used to define the vocabulary and semantics used in the communication between JADE agents. The WSIG add-on also uses the ontologies to define which agent operations will be exposed as web services. Moreover, an action is an object corresponding to a request. Once a request is made from the Web, the resulting action object containing the request elements is sent to the JADE agent.

In this first phase of our project, our objective was to implement an MAS that would be able to handle all the tasks associated with an specific event. For this purpose, we modeled a Smart Agenda MAS, which is built to function as an agent-based personal assistant. For example, if the user updates his agenda with a meeting in Brussels and he is in Paris, the Smart Agenda system is supposed to detect that the user needs a train to go from Paris to Brussels, and use a text-to-speech service in order to ask Google Home to book a train ticket. The text-to-speech service is named AWS Polly [2].

## 4.1 Requirements and Architecture

The Smart Agenda possesses a web page used as an interface between the user and the MAS. After creating an account, the user can set personal preferences regarding transportation, hotel stars, and hours. The web page is used to schedule new events and to modify or view existing events. The system also allows the user to create events involving other users, or to join existing events created by other users. Events are classified either as "individual events" or "group events".

Individual events are scheduled solely according to the user's preferences. They possess two distinct properties, named "Automatic" and "Movable". An automatic event can be scheduled by the agent arbitrarily along the day, according to the preset user preferences. A movable event can be rescheduled without the user's confirmation if it's necessary. Thus, if an automatic event is created, the agent will try to find an available time slot in the agenda according to the user's preferences. If there is not a free slot long enough to accommodate it, the agent will try to reorganize other movable events to optimize the day.

Group events are events shared by two or more users. They are non-movable by default. When a user creates an event, he can assign it to an existing users group. Group events can be marked as "Optional", meaning that the attendance for all users is not mandatory. If at least two users within the group are available at the scheduled time, the event will be created and added to these users' agenda.

[1] http://jade.tilab.com/doc/tutorials/WSIG_Guide.pdf
[2] https://aws.amazon.com/polly/

In the case the event is not marked as “Optional”, the attendance is considered mandatory for all participants - meaning that if at least one user is not available at the scheduled time the event will not be created.

The Smart Agenda MAS is composed by four different agents: Coordinator, Manager, Agenda, and Assistant. The Coordinator is responsible for handling all requests from the Web. When a new user account is created, the request is forwarded to an available Manager agent, which creates two new agents linked to this user: the Agenda agent, and the Assistant agent. In order to provide scalability to the system each Manager agent is responsible for a limited number of users. The Agenda agent is responsible for processing all future event operations for the new user, and the Assistant is the BDI agent responsible for processing complementary actions related to an event (such as booking a ticket) and generating the final sentence with the help of AWS Polly. Consequently, there are two agents (Assistant and Agenda) for each user registered in the system.

When a new event is created in the user’s calendar, the Coordinator forwards it to the corresponding Manager, which then sends it to user’s Agenda agent. This event is added to the agenda and, if this is an individual event, the Assistant agent creates its related Goals. A plan will be selected from an existing plan library according to the user’s preferences and context-dependent feasibility. If it fails, the next one will be tried. If the event contains the keywords “rdv”, “rendez vous” or “meeting”, and “@SomeTown”, the resulting plan will be a sentence containing all the event’s elements: the date, the time, the starting and arrival towns. This sentence is then translated into audio (speech) through the use of the AWS Polly service, and finally played to Google Home with the help of speakers. This architecture is shown in Figure 1, and its demonstration can be found at: http://bit.ly/Emas18Demo

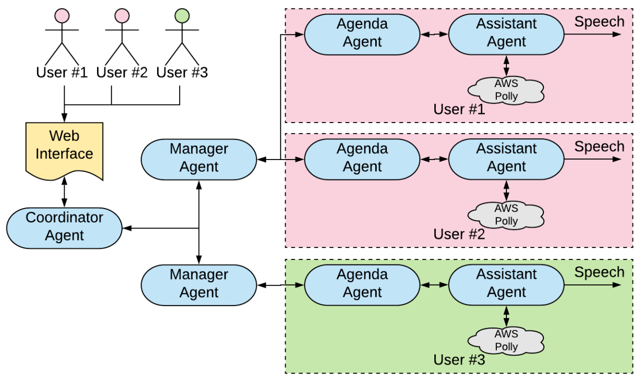

**Fig. 1.** Implementation architecture for the Smart Agenda agents

## 5 Discussion

In this chapter, we revisited some of the existing work on web-based agents and MAS. We also implemented a simple MAS using JADE and agents deployed as web services. Although not being complex, the idea behind the implementation was to create an MAS that could be further evolved and explored during the course of our research. Thus, the implemented system is meant to be an evolving proof-of-concept. We found a few difficulties related to the implementation (mostly related to WSIG), but the MAS architecture was simple enough to be modeled using a traditional BDI architecture. At the moment, the BDI model is limited to individual events and sequential goals (context dependency and AWS integration).

From the development process perspective, we found a few difficulties related to using the JADE platform. Most of these difficulties were related to the WSIG plugin, used to expose JADE agents as web services. Despite being a mature platform, we identified a few discrepancies in the framework documentation, and apparently the platform itself possesses some outdated dependencies (considering the state-of-the-art of web-related technologies and base frameworks, such as the http server and the graphical user interface). Also, the initial use of the platform raised questions related to its scalability, and how it could be used by a truly distributed system. Applying multiagent concepts to the proposed problem, however, was not difficult - most of the development effort was concentrated on learning and running the JADE platform according to the desired conditions.

This work is meant to be a stepping stone to our research interests in the AOSE domain. Our long-term objective is to study and explore web-based agents and MAS from both the modeling and the implementation perspective, considering AOSE methodologies and new Web-related paradigms. The next steps in this research topic involve (*i*) studying architectural patterns for web-based agents and related interaction protocols, as well as their relationship with existing agent modeling methods; and (*ii*) exploring new Web-related paradigms and studying how web-based agents could be used in conjunction with such paradigms. From the development perspective, we intend not only to evaluate existing multiagent frameworks and Web-related capabilities, but also to explore the existing platforms considering the current state-of-the-art scenario in terms of technologies and tools. In particular, we intend to address the problem of implementing distributed MAS using the available platforms, and which are the current limitations in this sense.

As for our initial proof-of-concept, we intend to evolve it by (*i*) implementing the same MAS in different frameworks, in order to establish a common baseline for future performance and scalability comparisons; and (*ii*) expanding the BDI model used in the implementation, increasing the complexity of the agents and their associated reasoning processes. The next steps reside in (*i*) expanding the BDI model (taking geographical constraints and group preferences into consideration), (*ii*) the original related research (architectural patterns for web-capable agents and related interaction protocols, as well as their relationship with existing agent modelling methods) and (*ii*) the implementation itself (comparing non-agent and agent-based approaches, automating the ticket booking process, improve shared events).

## References

1. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Distributed information systems. In: Web Services: Concepts, Architectures and Applications. pp. 3–27. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
2. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web services. In: Web Services: Concepts, Architectures and Applications. pp. 123–149. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
3. Ardissono, L., Barbero, C., Goy, A., Petrone, G.: An agent architecture for personalized web stores. In: Proceedings of the third annual conference on Autonomous Agents. pp. 182–189. ACM (1999)
4. Bai, X., Dai, G., Xu, D., Tsai, W.T.: A multi-agent based framework for collaborative testing on web services. In: Software Technologies for Future Embedded and Ubiquitous Systems, 2006 and the 2006 Second International Workshop on Collaborative Computing, Integration, and Assurance. SEUS 2006/WCCIA 2006. The Fourth IEEE Workshop on. pp. 6–pp. IEEE (2006)
5. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific american **284**(5), 34–43 (2001)
6. Bratman, M.: Intention, plans, and practical reason (1987)
7. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems **8**(3), 203–236 (2004)
8. Cabri, G., Leonardi, L., Puviani, M.: Service-oriented agent methodologies. In: Enabling Technologies: Infrastructure for Collaborative Enterprises, 2007. WETICE 2007. 16th IEEE International Workshops on. pp. 24–29. IEEE (2007)
9. Ciortea, A., Boissier, O., Ricci, A.: Beyond physical mashups: Autonomous systems for the web of things. In: Proceedings of the Eighth International Workshop on the Web of Things. pp. 16–20. ACM (2017)
10. Ciortea, A., Boissier, O., Zimmermann, A., Florea, A.M.: Give agents some rest: A resource-oriented abstraction layer for internet-scale agent environments. In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems. pp. 1502–1504. AAMAS '17, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2017), http://dl.acm.org/citation.cfm?id=3091125.3091342
11. DeLoach, S.A.: The mase methodology. In: Methodologies and software engineering for agent systems, pp. 107–125. Springer (2004)
12. DeLoach, S.A., Garcia-Ojeda, J.C.: The o-mase methodology. In: Handbook on Agent-Oriented Design Processes, pp. 253–285. Springer (2014)
13. Dickinson, I., Wooldridge, M.: Agents are not (just) web services: considering bdi agents and web services. In: Proceedings of the 2005 Workshop on Service-Oriented Computing and Agent-Based Engineering (SOCABE'2005), Utrecht, The Netherlands (2005)
14. Erl, T.: Service-Oriented Architecture – Concepts, Technology, and Design. Prentice Hall, 1 edn. (2005)
15. Etzioni, O.: Moving up the information food chain: Deploying softbots on the world wide web. In: Proceedings of the national conference on artificial intelligence. pp. 1322–1326 (1996)

16. Ferber, J.: Multi-agent systems: an introduction to distributed artificial intelligence, vol. 1. Addison-Wesley Reading (1999)
17. Greenwood, D., Calisti, M.: Engineering web service-agent integration. In: Systems, Man and Cybernetics, 2004 IEEE International Conference on. vol. 2, pp. 1918–1925. IEEE (2004)
18. Guinard, D., Trifa, V., Wilde, E.: A resource oriented architecture for the web of things. In: Internet of Things (IOT), 2010. pp. 1–8. IEEE (2010)
19. Hahn, C., Jacobi, S., Raber, D.: Enhancing the interoperability between multiagent systems and service-oriented architectures through a model-driven approach. In: Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on. vol. 2, pp. 415–422. IEEE (2010)
20. Hendler, J.: Agents and the semantic web. IEEE Intelligent systems **16**(2), 30–37 (2001)
21. Hirsch, B., Ng, J.W.: Education beyond the cloud: anytime-anywhere learning in a smart campus environment. In: Internet Technology and Secured Transactions (ICITST), 2011 International Conference for. pp. 718–723. IEEE (2011)
22. Huhns, M.N.: Agents as web services. IEEE Internet computing **6**(4), 93–95 (2002)
23. Jennings, N.R., Sycara, K., Wooldridge, M.: A roadmap of agent research and development. Autonomous agents and multi-agent systems **1**(1), 7–38 (1998)
24. Jennings, N.R., Wooldridge, M.: Agent-oriented software engineering. Handbook of agent technology **18** (2001)
25. Kardas, G., Goknil, A., Dikenelli, O., Topaloglu, N.Y.: Model driven development of semantic web enabled multi-agent systems. International Journal of Cooperative Information Systems **18**(02), 261–308 (2009)
26. Konolige, K., Nilsson, N.J.: Multiple-agent planning systems. In: AAAI. vol. 80, pp. 138–142 (1980)
27. Lieberman, H., et al.: Letizia: An agent that assists web browsing. IJCAI (1) **1995**, 924–929 (1995)
28. Liu, S., Küngas, P., Matskin, M.: Agent-based web service composition with jade and jxta. In: SWWS. vol. 6, pp. 110–116 (2006)
29. Muldoon, C.: An agent framework for ubiquitous services. Ph.D. thesis, Citeseer (2007)
30. Nguyen, X.T., Kowalczyk, R.: Ws2jade: Integrating web service with jade agents. In: Huang, J., Kowalczyk, R., Maamar, Z., Martin, D., Müller, I., Stoutenburg, S., Sycara, K.P. (eds.) Service-Oriented Computing: Agents, Semantics, and Engineering. pp. 147–159. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
31. Nguyen, X.T., Kowalczyk, R.: Ws2jade: Integrating web service with jade agents. In: International Workshop on Service-Oriented Computing: Agents, Semantics, and Engineering. pp. 147–159. Springer (2007)
32. O'Reilly, R.D.: A distributed architecture for the monitoring and analysis of time series data (2015)
33. Padgham, L., Winikoff, M.: Prometheus: A methodology for developing intelligent agents. In: International Workshop on Agent-Oriented Software Engineering. pp. 174–185. Springer (2002)
34. Paletta, M.: A scouting-based multi-agent system model to deal with service collaboration in cloud computing. Systems and Software Development, Modeling, and Analysis: New Perspectives and Methodologies: New Perspectives and Methodologies p. 282 (2014)
35. Pěchouček, M., Mařík, V.: Industrial deployment of multi-agent technologies: review and selected case studies. Autonomous Agents and Multi-Agent

Systems **17**(3), 397–431 (Dec 2008). https://doi.org/10.1007/s10458-008-9050-0, https://doi.org/10.1007/s10458-008-9050-0

36. Ricci, A., Denti, E., Piunti, M.: A platform for developing soa/ws applications as open and heterogeneous multi-agent systems. Multiagent and Grid Systems **6**(2), 105–132 (2010)
37. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Pearson Education, 2 edn. (2003)
38. Tapia, D.I., Rodríguez, S., Bajo, J., Corchado, J.M.: Fusion@, a soa-based multi-agent architecture. In: International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008). pp. 99–107. Springer (2009)
39. Thiele, A., Kaiser, S., Konnerth, T., Hirsch, B.: Mams service framework. In: International Workshop on Service-Oriented Computing: Agents, Semantics, and Engineering. pp. 126–142. Springer (2009)
40. Tolk, A., Uhrmacher, A.M.: Agents: Agenthood, agent architectures, and agent taxonomies. Agent-directed simulation and systems engineering pp. 75–109 (2009)
41. Willmott, S., Dale, J., Burg, B., Charlton, P., O'Brien, P.: Agentcities: a worldwide open agent network. Agentlink News **8**(LIA-ARTICLE-2001-002) (2001)
42. Wooldridge, M.: An introduction to multiagent systems. John Wiley & Sons (2009)
43. Wooldridge, M., Jennings, N.R., Kinny, D.: The gaia methodology for agent-oriented analysis and design. Autonomous Agents and multi-agent systems **3**(3), 285–312 (2000)
44. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Developing multiagent systems: The gaia methodology. ACM Transactions on Software Engineering and Methodology (TOSEM) **12**(3), 317–370 (2003)
45. Zinnikus, I., Hahn, C., Fischer, K.: A model-driven, agent-based approach for the integration of services into a collaborative business process. In: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1. pp. 241–248. International Foundation for Autonomous Agents and Multiagent Systems (2008)