Journal of Computer Engineering Research Vol. 1(2), pp. 14 - 28, April 2010
Available online at http://www.academicjournals.org/JCER

Full Length Research paper

A survey of agent-oriented software engineering paradigm: Towards its industrial acceptance

O. Zohreh Akbari

Department of Information and Communication Technology, Faculty of Engineering, Payame Noor University, Tehran, Iran. E-mail: z.o.akbari@gmail.com.

Accepted 13 January, 2010

Agent-oriented software engineering (AOSE) paradigm represents an interesting means of analyzing, designing and building complex software systems quite suitable to new software development requirements. Many scientific researches have been focused on this paradigm, yet its current state still reports relative lack of industrial acceptance compared to others. As a survey of AOSE paradigm, this paper outlines the overall state of this paradigm; and by identifying its weaknesses in detail, leads to a proposal solution to such shortcoming. This solution, in keeping with the existing approaches that aim to use situational method engineering (SME) in collaborative manner between agent-oriented methodology designers, suggests the use of a methodology evaluation framework in the process as well. This framework is a means to collect the best method fragment and evaluate consecutively the methodology during the development process for possible methodology improvements. The proposed solution is then readjusted to help software development organizations to reach the fifth level of Capability Maturity Model (CMM).

Key words: Agent-oriented software engineering (AOSE), capability maturity model (CMM), evaluation framework, methodology, project-specific, situational method engineering (SME).

INTRODUCTION

The complexity of software development process had caused the development of increasingly powerful and natural abstraction with which to model and develop complex systems. Procedural abstraction, abstract data types, and objects are all examples of such abstractions (Wooldridge et al., 1999). During the past two decades, with the increase in complexity of projects associated with software engineering, agent concepts that originated from Artificial Intelligence (AI) have been considered to devise a new paradigm for handling complex systems (Genesereth and Ketchpel, 1994; Jennings and Wooldridge, 1996, 2000; Shoham, 1990, 1993; Wooldridge, 1997).

Agent-oriented software engineering (AOSE) paradigm represents an interesting means of analyzing, designing and building complex software systems and it is quite suitable to the new software development requirements (agent-oriented methodologies strengths). But although many scientific researches have been fo-cused on this paradigm (existing agent-oriented software engineering), its current state still reports relative lack of industrial acceptance compared to others.

This paper aims to outline the current standing of AOSE paradigm (a survey of agent-oriented software engineering paradigm) and propose a solution to its relative lack of industrial acceptance compared to others, which is then readjusted to present a plan for software development organizations to reach the fifth level of CMM (proposal solution to agent-orientation promotion). Key building blocks of the proposed approach are an evaluation framework for agent-oriented software engineering methodologies (existing approaches for evaluating agent-oriented methodologies) and a project-specific methodology building framework (existing approaches for evaluating agent-oriented methodologies), which both have suitable instances but have never been merged. A practical instance of the proposal plan (agent open method) is also presented in this paper using these suitable frameworks.

A SURVEY OF AGENT-ORIENTED SOFTWARE ENGINEERING PARADIGM

In order to outline the current state of agent-oriented software engineering paradigm, this section starts with

defining AOSE methodologies (The definition of agent-oriented software engineering methodology), then briefly goes over its history (the history of agent-oriented software engineering paradigm), lists existing AOSE methodologies (existing agent-oriented software engineering methodologies) and states their strengths and weaknesses (strengths and weaknesses of agent-oriented methodologies).

The definition of agent-oriented software engineering methodology

To define AOSE methodology, it is first necessary to have a precise definition of methodology itself. Regarding (Brinkkemper, 1996; CMS, 2008; Firesmith, 2002; Lyytinen, 1987; IEEE, 1990; Sturm and Shehory, 2003; Sudeikat et al., 2004) the definition considered for a software engineering methodology in this paper is as follows: A business process equipped with distinct concepts and modeling tools for developing software (Akbari and Faraahi, 2008).

The methodology definition merged with software engineering paradigm concept constitutes the AOSE methodology definition. An agent-based system is a system in which the key abstraction used is that of an agent (Jennings and Wooldridge, 2000; Wooldridge, 1997) and (Wooldridge and Jennings, 1995). Thus by agent-oriented software engineering we mean a software engineering paradigm in which the key abstraction used is that of an agent. Considering this description and the mentioned definition for methodology, an agent-oriented software engineering methodology can be defined as follows: An agent-oriented software engineering methodology is a business process of developing software, equipped with distinct concepts and modeling tools, in which the key abstraction used in its concepts is that of an agent.

The history of agent-oriented software engineering paradigm

AOSE Paradigm, which was first proposed by Yoav Shoham in 1990, is based on a societal view of computation (Shoham, 1990 and 1993). The main source of this paradigm is AI (Debenham and Henderson-Sellers, 2002; Wooldridge, 1997) or precisely, Distributed Artificial Intelligence (DAI) (Bond and Gasser, 1998; Henderson-Sellers and Gorton, 2003). Nevertheless, in agent-orientated software engineering, agents are about computer science and software engineering more than they are about AI (See Wooldridge, 1997 for more description).

Agent-oriented paradigm has multiplied a lot during the past two decades, and although it was first limited to academic researches, it has interested the industry within the last years as well (Debenham and Henderson-Sellers, 2002; Henderson-Sellers and Gorton, 2003). It should be pointed out that after almost a decade of its introduction, the progress of this paradigm has faced a great transformation, which some researches refer to as the entrance to the new generation of software engineering methodologies (Dam and Winikoff, 2003; Henderson-Sellers and Gorton, 2003) (Figure 1) shows the effect of this transition on the number of AOSE methodologies designed per year. The main idea of this transition is based on SME (Harmsen, 1997) and the unification strategy of existing issue (AOSE TFG, 2004), to build a framework for designing project-specific methodologies. The mentioned approach is the researchers' solution to eliminate the relative industry rejection of this paradigm, or eliminate its weaknesses (AOSE TFG, 2004; Henderson-Sellers and Gorton, 2003; Henderson-Sellers et al., 2004). Such issues can be found in (Cossentino and Seidita, 2004), (Henderson-Sellers and Gorton, 2003) and (Juan et al., 2002), which will be described later.

EXISTING AGENT-ORIENTED SOFTWARE ENGINEERING METHODOLOGIES

This section goes through the identification of existing AOSE methodologies. Having the complete list of these methodologies can be a good base to distinguish the current state of AOSE paradigm, yet despite this standing, this list may also be used as a resource reference for the existing project-specific methodology building frameworks to complete their repositories (existing approaches for creating agent-oriented project-specific). The number of existing agent-oriented software engineering methodologies is very high despite their newness. Due to the limited space, examples of existing AOSE methodologies are presented in two separate tables in order of the year of presentation: Table 1 lists the AOSE methodologies introduced before year 2000, and Table 2 lists the AOSE methodologies introduced after year 2000. It should be pointed out that items presented at rows number 37, 43 and 57 are more than just simple methodologies, and are frameworks for creating agent-oriented project-specific methodologies, which will be described in (existing approaches for creating agent-oriented project-specific).

Figure 1 shows the number of AOSE methodologies designed each year from 1990, when AOSE paradigm was first introduced. As it is shown, the number of designed AOSE methodologies has a significant increase in year 2002, but has dropped again the year after. This could have several meanings:

1. AOSE paradigm had dramatically improved till year 2002 that has interested many methodology designers and users at the time: despite some exceptions, the number of methodologies designed has increased each

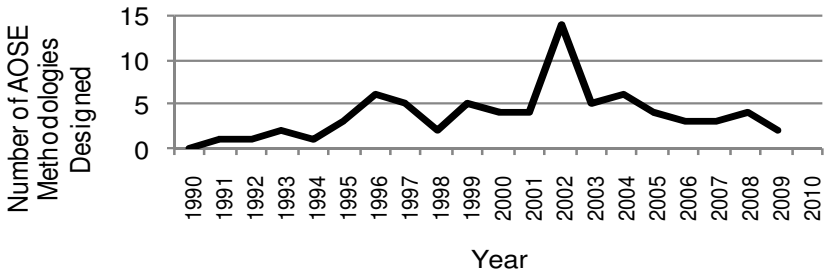

Figure 1. The number of AO methodologies designed each year.

Table 1. List of AOSE methodologies introduced before year 2000.

#	Methodology	Year	Reference(s)
1	ARCHON	1991	(Cockburn and Jennings, 1996)
2	MADE	1992	(O'Hare and Wooldridge, 1992)
3	DRM	1993	(Singh et al., 1993)
4	TOGA	1993	(Gadomski, 1993)
5	CIAD	1994	(Verharen and Weigard, 1994; Verharen, 1997)
6	Agent Factory	1995	(Collier, 1996, 2002; Collier and O'Hare, 1999; O'Hare and Collier, 1998)
7	AOMfEM	1995	(Kendall et al., 1996)
8	Cassiopeia	1995	(Collinot and Drogoul; 1998, Collinot et al., 1996)
9	AAII (KGR)	1996	(Kinny and Georgeff, 1996; Kinny et al., 1996)
10	AOAD	1996	(Burmeister, 1996)
11	AWIC	1996	(Muller, 1996)
12	CoMoMas	1996	(Glaser, 1996)
13	MASB	1996	(Moulin and Brassard, 1996)
14	MAS-CommonKADS	1996	(Iglesias et al., 1998)
15	AALAADIN	1997	(Ferber, 1997; Ferber and Gutknecht, 1998)
16	AMBSA	1997	(Neal Reilly, 1997)
17	AOIM	1997	(Kindler et al., 1997)
18	CaseLP	1997	(Martelli et al., 1997)
19	DESIRE	1997	(Brazier et al., 1997)
20	Adept	1998	(Jennings et al., 1998)
21	AMBIA	1998	(Gao and Sterling, 1998)
22	AOAaD	1999	(Wooldridge, 1999)
23	HIM	1999	(Elammari, 1999)
24	MaSE	1999	(Deloach, 1999, 2005)
25	MASSIVE	1999	(Lind, 1999, 2001)
26	ZEUS	1999	(Nwana et al., 1999)
27	ASEfIA	2000	(Zamboneli et al., 2000)
28	Gaia	2000	(Wooldridge et al., 2000; Zamboneli et al., 2005)
29	MESSAGE/UML	2000	(Caire et al., 2000; Evans et al., 2001)
30	SODA	2000	(Omicini, 2000)

year and about 14 methodologies were introduced in year 2002.

2. AOSE paradigm has provided the necessary conditions for creating project-specific methodology building frameworks: two project-specific methodology building frameworks were defined in year 2002, and also one in year 2004.

3. The introduction of project-specific methodology building frameworks has relevantly answered the user willingness to setup project-specific methodology, yet there is still room for improvements: the number of methodologies designed per year has significantly decreased since year 2002, yet there are still some methodologies designed independent from project-specific

Table 2. List of AOSE methodologies introduced after year 2000.

#	Methodology	Year	Reference(s)
31	Agent-SE	2001	(Far, 2001)
32	AOSM	2001	(Shi, 2001)
33	Styx	2001	(Bush, 2001)
34	Tropos	2001	(Bresciani et al., 2001, 2004; Castro et al., 2001, 2002; Mylopoulos et al., 2001)
35	ADELFE	2002	(Bernon et al., 2002)
36	ALCCIG	2002	(Zhang et al., 2002)
37	CAOMF	2002	(Juan et al., 2002a; Juan et al., 2003; Taveter and Sterling, 2008)
38	IEBPM	2002	(Taveter and Wagner, 2002)
39	INGENIAS	2002	(Pavon and Gomez-Sanz, 2003, 2005)
40	MESMA	2002	(Cuesta et al., 2002)
41	Nemo	2002	(Huget, 2002)
42	ODAC	2002	(Gervais, 2002)
43	Agent OPEN	2002	(Debenham and Henderson-Sellers, 2002; Henderson-Sellers and Gorton, 2003; Henderson-Sellers et al., 2005)
44	PASSI	2002	(Cossentino and Potts, 2002; Cossentino, 2005)
45	Prometheus	2002	(Cervenka, 2003; Padgham and Winikoff, 2002a,b)
46	ROADMAP	2002	(Juan et al., 2002b)
47	SABPO	2002	(Dikenelli and Erdur, 2002)
48	SADDE	2002	(Sierra et al., 2002)
49	MAGE	2003	(Shi et al., 2003, Shi et al., 2004)
50	OPM/MAS	2003	(Sturm et al., 2003)
51	RAP/AOR	2003	(Taveter and Wagner, 2005; Wagner, 2003)
52	RoMAS	2003	(Yan et al., 2003)
53	SONIA	2003	(Alonso et al., 2005)
54	AMBTA	2004	(Sardinha et al., 2004)
55	AODM	2004	(Tian et al., 2004)
56	CAMLE	2004	(Shan and Zhu, 2004)
57	FIPA	2004	(Cossentino and Seidita, 2004; Garro et al., 2004)
58	MAOSEM	2004	(Wang and Guo, 2004)
59	RAOM	2004	(Giret and Botti, 2004)
60	MAHIS	2005	(Li and Liu, 2005)
61	MAMfHMS	2005	(Giret, 2005)
62	OMASM	2005	(Villaplana, 2005)
63	OWL-P	2005	(Desai et al., 2005)
64	ADMuJADE	2006	(Nikraz et al., 2006)
65	MOBMAS	2006	(Tran et al., 2007; Tran and Low, 2008)
66	WAiWS	2006	(Lu and Chhabra, 2006)
67	ADEM	2007	(Cervenka and Trencansky, 2007; Whitestein technologies, 2008)
68	ASPECS	2007	(Cossentino et al., 2007)
69	ForMAAD	2007	(Hadj-Kacem et al., 2007)
70	ANEMONA	2008	(Giret, 2008)
71	MASD	2008	(Abdelaziz et al., 2008)
72	MASIM	2008	(Clancey et al., 2008)
73	PerMet	2008	(Grislin-Le Strugeon et al., 2008)
74	AOMEIS	2009	(Athanasiadis and Mitkas; 2009)
75	ODAM	2009	(Mao et al., 2009)

methodology building frameworks.

Strengths and weaknesses of agent-oriented methodologies

In this section the necessity of agent-orientation usage is discussed as the agent-oriented methodologies strengths and its weaknesses, in terms of its relative industrial rejection.

Agent-oriented methodologies strengths

Agent-oriented methodologies strengths can be considered in two different aspects:

1. Inclusion of other paradigms' capabilities and presentation of more abilities: AOSE paradigm includes all the capabilities of other paradigms (e.g. object-oriented, knowledge engineering and service-oriented) and even more abilities.

a) Agent-oriented methodologies versus object-oriented metho-dologies: As stated by Shoham (Shoham, 1993), agents can be considered as active objects with mental states (Iglesias et al.,
1999) which means despite the common characteristics between objects and agents they are not just simple objects but they present more capabilities (Iglesias et al., 1999).
b) Agent-oriented methodologies versus knowledge en-gineering methodologies: Most of the problems subject to knowledge engineering methodologies are also present in designing Multi-Agent Systems (MAS) as knowledge acquisition, modeling, and reuse. Furthermore, these methodologies conceive a knowledge-based system as a centralized one, thus they do not address the distri-buted or social aspects of the agents, or their reflective and goal- oriented attitudes (Iglesias et al., 1999).
c) Agent-oriented methodologies versus service-oriented methodo-logies: Regarding service-oriented methodologies, it should be
pointed out that service is only one of the several concepts presented by an agent, and that agents may not be just service performers, but also predictives – they may volunteer information or services to a user, without being explicitly asked, whenever it is deemed appropriate (Jennings and Wooldridge, 1996).

2. Suitability with new software development requirements: As mentioned before, due to the complexity of software development process, wide range of software engineering paradigms has been devised (e.g. structured programming, object-oriented program-ming, procedural programming and declarative programming) (Jennings and Wooldridge, 2000). But recently, with the high rate of increase in complexity of projects associated with software engi-neering, agent concepts, which originated from artificial intelligence, have been considered to devise a new paradigm for handling complex systems (Genesereth and Ketchpel, 1994; Jennings and Wooldridge, 1996, 2000; Shoham, 1990, 1993; Wooldridge, 1997). Some special applications of this paradigm are presented in (Wooldridge and Ciancarini, 2001).

Agent-oriented methodologies weaknesses

Agent-oriented methodologies weaknesses can be considered in two different aspects:

1. The lack of attraction for methodology user to use the agent-oriented paradigm:
a) Lack of agent-oriented programming languages: Although programming languages are only part of the development story, industry is reticent to adopt a new paradigm at the conceptual level if it is impossible to implement these ideas in a currently acceptable, commercially viable programming language (Henderson-Sellers and Gorton, 2003).
b) Lack of explicit statement of agent-orientation advantages: The benefits of agent technology must be declared by introducing the cases where AOSE paradigm succeeds and other existing paradigms fail (Henderson-Sellers and Gorton, 2003).
c) Relative difficulty of learning concept related to agent-oriented paradigm (AI): As an example the usage of Gaia agent-oriented methodology (Wooldridge et al., 2000) requires learning logic, which decreases the adoption of this methodology, since usually methodology users are not familiar with logic and do not tend to learn it (Sturm and Shehory, 2003).
d) High cost of AO acquisition: The acquisition of this paradigm by software development organizations requires a high cost for training the development team (Henderson-Sellers and Gorton, 2003).

2. The lack of attraction for methodology user to use existing agent-oriented methodologies:

a) Relative immaturity: The AO paradigm immaturity, which is a relative matter compared to other paradigms (Dam and Winikoff, 2003), is clearly because of it newness.
b) Marketing of multiple AO methodologies: As long as the availability and marketing of multiple agent-oriented methodologies are in competitive manner, this feature is an obstacle to their widespread industrial adoption, since it leads to confusion of methodology users (Henderson-Sellers and Gorton, 2003).
c) Lack of confrontation with wrong expectation of one-size-fits-all methodology: No unique specific methodology

can be general enough to be useful to every project without some level of personalization (AOSE TFG, 2004). Users usually think a unique methodology has general usage and ignore the fact that each methodology is designed for some specific goals (e.g. specific domain or different parts of life cycle). Thus when a specific methodology does not fit their requirements and leads to project failure they conceive the problem from the side of methodology whereas the problem is with the wrong methodology selection (Henderson-Sellers and Giorgini, 2005). Agent-oriented paradigm should support its user with the awareness and facilities to find the proper methodology for his project from existing methodologies or to change the existing instances in order to fit the project.

d) Lack of confrontation with user willingness to setup an owned project-specific methodology: The high number of existing AO methodologies can be seen as a proof that methodology users, often prefer to setup an owned methodology specially tailored for their needs instead of reusing existing ones (AOSE TFG, 2004). AO paradigm should support its user with the awareness and facilities to avoid setting up his methodology from the scratch, but to change the existing instances in order to fit the project.

PROPOSAL SOLUTION TO AGENT-ORIENTATION PROMOTION

The progress of AOSE paradigm is dependent to the elimination of its weaknesses as mentioned above. Clearly, when the software development organization becomes justified for using agent-orientation, by its strengths, it will accept its cost and learning effort much easier, since it knows that in long-term this paradigm will not just pay back this cost but that its benefits would be more than others.

With the emergence of industry willingness for agent-orientation, the next problem to be eliminated would be the lack of attraction for agent-oriented methodologies. It is obvious that identifying the strengths and weaknesses of each methodology can be the first step to its progress and wide industrial acceptance as well (Akbari and Faraahi, 2008; Aose TFG, 2004; Dam and Winikoff, 2003). In addition, the availability and marketing of multiple methodologies which is an obstacle to the ease of selection, lack of the presence of a one-size-fits-all methodology and the need of project-specific methodologies, shows the necessity for exploitation of a project-specific building framework.

Thus it is suggested that software development organizations use an evaluation framework for agent-oriented methodologies such as the one described in existing approaches for evaluating agent-oriented methodologies in order to choose the best for their project, and in case of finding no fitting match to exploit the evaluation results for building effective project-specific methodologies. This might be done by completing and thus improving existing methodologies by replacing their weak parts with strong parts from other methodologies, using one of the frameworks for creating agent-oriented project-specific methodologies described in (existing approaches for creating agent-oriented project-specific methodologies). Thus a consolidated approach as also expressed in (Henderson-Sellers and Gorton, 2003) could give a better signal to the industry. With this regard, it is suggested that instead of competing, agent-oriented methodology designers collaborate with each other by evaluating their own methodologies using an appropriate evaluation framework, to collect the method fragments with their rankings in order to use these information for method engineering. This is quite feasible since most of the agent-oriented methodologies are academic and not commercial products.

This approach would: (i) help to improve existing methodologies by identifying their weaknesses, (ii) make the availability of multiple methodologies an advantage (having wide range of method fragment options), (iii) do away with the wrong expectation on one-size-fits-all methodology, and (iv) answer to user willingness to setup an owned project-specific methodology. Clearly this approach will attract methodology users to use agent-oriented methodologies, and in other words results to industrial acceptance of AOSE paradigm. In addition the usage of the frameworks for creating agent-oriented project-specific methodologies will not only make it possible to use programming languages from other paradigms which are suitable for agent-orientation, but the industry willingness for this paradigm will encourage language designers as well.

This solution to AOSE weaknesses may also be readjusted to propose a plan for development organizations to reach the fifth level of CMM. Figure 2 explains this plan. In CMM organizational maturity framework (Humphery, 1990; Paulk et al., 1993), 5 maturity levels are distinguished (Harmsen, 1997): Initial, Repeatable, Defined, Managed and Optimizing. Since the proposed plan exploits the SME in order to build project-specific methodologies, it is clear that it satisfies the third level of CMM. In addition, since the evaluation framework assesses the methodologies for management plans and thus the management plans' method fragments are constructed to methodology, both process and products are regularly evaluated by the project management team to satiate the forth level of CMM. The feedback that is given by the organization while employing the methodology using the evaluation framework causes the methodology correction to take place continuously and concurrent with its exploitation, and satisfies the 5th level of CMM.

What has taken place by now is the growth of repository by adding all the AOSE methodology's components without considering any evaluation (Henderson-Sellers et

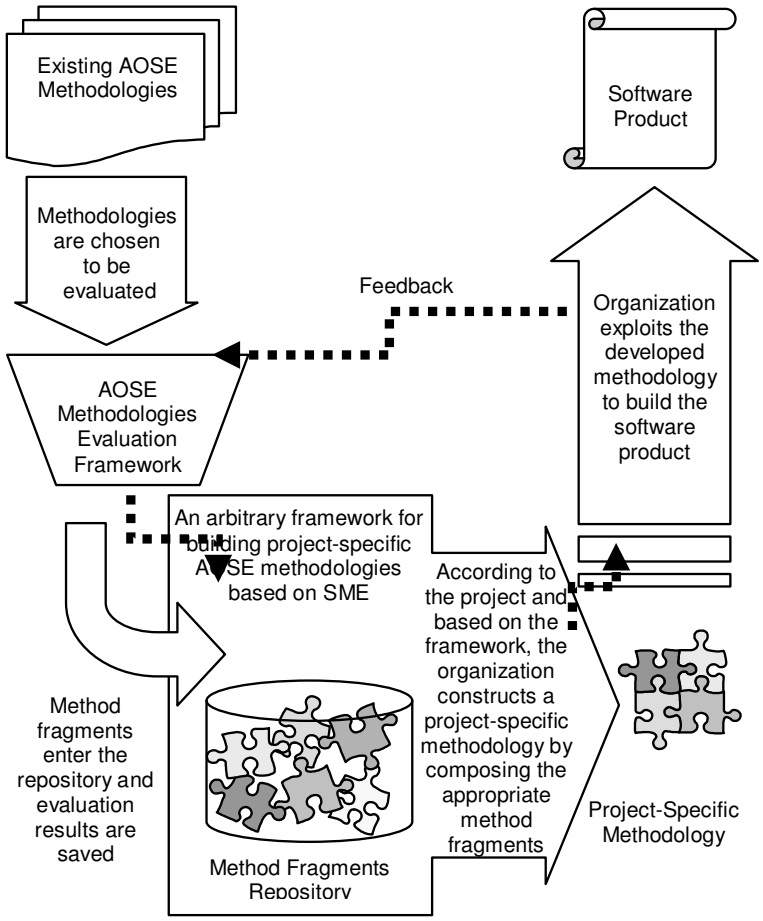

Figure 2. Proposal plan for agent-oriented software development organizations to reach the fifth level of CMM.

al., 2003; Henderson-Sellers et al., 2004; Henderson-Sellers et al., 2006). But the approach presented here is the usage of an evaluation framework and a project-specific methodology building framework simultaneously together. So, each methodology would first be evaluated, and the method fragments with their grades entered in the repository. This makes possible the selection of method fragments with desired grades at the methodology building stage which better implements SME approach. To implement this plan, an evaluation framework and a project-specific methodology building framework are needed. Existing approaches for evaluating agent-oriented methodologies and existing approaches for creating agent-oriented project-specific methodologies describes existing approaches of each of the frameworks.

EXISTING APPROACHES FOR EVALUATING AGENT-ORIENTED METHODOLOGIES

Researches considering the evaluation of agent-oriented methodologies are limited to (Akbari and Faraahi, 2008, 2009; Cernuzzi and Rossi, 2002; Dam and Winikoff, 2003; Henderson-Sellers and Giorgini, 2005; Kumar, 2002; Lin et al., 2007; Sabas et al., 2002; Shehory and Sturm, 2001; Sturm and Shehory, 2003; Sudeikat et al., 2004; Yu and Cysneiros, 2002) and some other studies that compare two or three methodologies, only with respect to the expressiveness and the concepts supported by the methodology (Sturm and Shehory, 2003). Most of the mentioned evaluation frameworks suffer from one or both of the following shortcomings: (1) Lack of coverage for all of the methodology aspects, (2) Lack of definition of a precise evaluation metric. As mentioned above, methodology is referred to as an economical process of developing software, equipped with distinct concepts and modeling tools (Akbari and Faraahi, 2008, 2009). In this regard methodologies can be considered in six major aspects: concepts, notation, process, pragmatics, support for software engineering and marketability. In addition, evaluation metric should be able to present different levels of methodology support for each criterion. The framework presented in (Akbari and Faraahi, 2008) and

completed in (Akbari and Faraahi, 2009) evaluates methodologies from all aspects men-tioned and defines a metric with 7 levels of support; thus it perfectly overcomes the mentioned shortcomings of most evaluation frameworks.

As stated in (Akbari, 2010) the most important difference between the mentioned evaluation framework with existing approaches is that this framework is multi-layered (Figure 3); meaning that methodologies are first considered in the six mentioned aspects and in detailed layers base on the criteria and sub-criteria. Actually, each criterion refer to its sub-criteria, thus it increases the preciseness and clarity of the evaluation and helps the evaluator through the process. Furthermore, users will use the evaluation results accordingly to their required level. For example, for software development organization customer, the overall grade of methodology is important; thus average of methodology rating are presented to him (according to the metric of the framework, resulting average should be rounded in each level of evaluation, to fit one of the 7 levels). But on the contrary, for software developer the grade obtained for most detailed criteria are important.

EXISTING APPROACHES FOR CREATING AGENT-ORIENTED PROJECT-SPECIFIC METHODOLOGIES

Existing approaches for creating agent-oriented project-specific methodologies are based on situational method Engineering (SME). The term method engineering (ME) goes back to Maynard, who introduced it as the research area in mechanical engineering, addressing the definition of methods to industrial engineering (Maynard, 1939). In definition, ME approaches do not necessarily take into account the project or situation in which a method will be applied (Harmsen, 1997). SME is the sub-area of ME directed towards the controlled, formal and computer-assisted construction of situational (project-specific) methods out of method fragments (a description of an Information System (IS) engineering method, or any coherent part thereof) (Harmsen, 1997). A well-known synonym for SME is Methodology Engineering, which was first introduced in (Kumar and Walke, 1992).

Despite the strengths of existing approaches for creating agent-oriented project-specific methodologies, they also have some weak points:

1. Lack of methodology evaluation and result saving while storing a methodology in method fragments repository.
2. Lack of consideration of method fragment capability while creating a project-specific methodology

To eliminate mentioned shortcomings, two different approaches may be considered:

1. Screening the method fragments at storing stage, by evaluation and storing strong method fragments with high grades.

2. Evaluating and storing all the method fragments with their corresponding evaluation results, and postponing the selection of method fragments with desired grade to methodology building stage.

Clearly, the second approach is the best one and follows the SME goals. Since SME is not always seeking to assemble the method fragments with high grades, but more precisely, it seeks to assemble the proper method fragments (with proper capabilities). For example, a software development organization that works on large, complex, and business-critical projects, must consider management plans in its methodology (Firesmith and Henderson-Sellers, 2002), and as much as the project is larger, more complex and more business-critical, the management plans method fragments should be stronger with higher grades of evaluation. Yet in opposite way software development organization that works on small, simple, and non-critical projects does not need restricted management plans. In this case, restricted management plans would not even help the progress of software development, but would be an overload to development team by defining unnecessary fruitless tasks. Thus, in such cases, method fragments with average or even low grades would be sufficient for the project-specific methodology.

As a result, weaknesses of existing approaches for creating agent-oriented project-specific methodologies also show the necessity of joining these frameworks with evaluation frameworks in order to build project-specific methodologies and thus improve agent-oriented methodologies acceptance. The existing project-specific methodology building frameworks are briefly introduced in the following sections.

Agent OPEN method

OPEN, which stands for Object-oriented Process, Environment and Notation, was first outlined in (Henderson-Sellers and Graham, 1996) and was published in (Graham et al., 1997) as a full life cycle methodology (Firesmith and Henderson-Sellers, 2002). OPEN Process Framework (OPF) consists of: (i) a process metamodel of framework from which can be generated an organizationally specific process, (instance) created using a method engineering approach from (ii) a repository and (iii) a set of construction guidelines. The major elements in OPF metamodel are Work Units (Activities, Tasks and Techniques), Work Products, Producers and two auxiliary ones (Stages and Languages) (Henderson-Sellers et al., 2003).

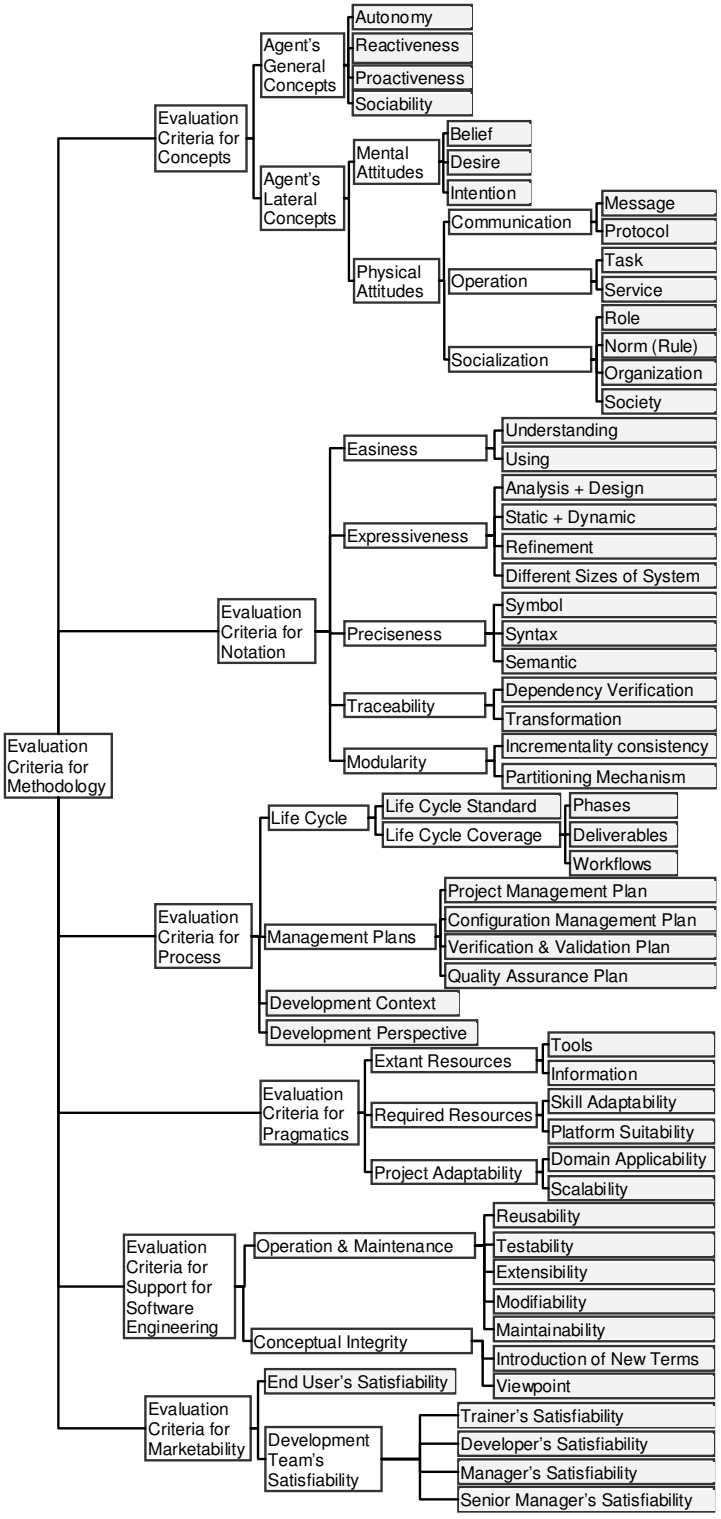

Figure 3. Different levels of criteria in evaluation framework introduced in (Akbari, 2010).

To extend this approach to support agent-oriented information systems, (Debenham and Henderson-Sellers, 2003) analyzes the differences between agent-oriented and object-oriented approaches in order to be able to itemize and outline the necessary additions to the OPF's repository in the standard format provided in (Henderson-

Sellers et al., 1998). A list of method fragments added to OPF from existing agent-oriented methodologies can be found in (Henderson-Sellers, 2005, 2004, 2006 and 2003).

Feature-based method

In (Juan et al., 2002) is proposed a modular approach enabling developers to build customized project-specific methodologies from AOSE features. An AOSE feature is defined in (Juan et al., 2003) to encapsulate software engineering techniques, models, supporting Computer-Aided Software Engineering (CASE) tools and development knowledge such as design patterns. It is considered a stand-alone unit to perform part of a development phase, such as analysis or prototyping, while achieving a quality attribute such as privacy. Comparing to Agent OPEN method, an AOSE feature can be defined in terms of these notions as a Work Unit performed by one or more Producers in support of a specific software engineering stage resulting in one or more Work Products represented in the respective Languages (Taveter and Sterling, 2008). Differing from Agent OPEN approach, this method does not regard it necessary to rely on the formal metamodel of method fragments and has demonstrated in (Juan and Sterling, 2003; Juan et al., 2002, 2003; Sterling and Taveter, 2009) that informal approach to methodology composition works equally well and is more likely to be adopted in industry.

This method identifies and standardizes the common elements of the existing methodologies. The common elements could form a generic agent model on which specialized features might be based. The remaining parts of the methodologies would represent added-value that the methodologies bring to the common elements, and should be componentized into modular features. The small granularity of features allows them to be combined into the common models in a flexible manner. By conforming to the generic agent model in the common elements, it is expected that the semantics of the optional features remain consistent (Juan et al., 2002).

FIPA methodology technical committee method

This work refers to the FIPA Methodology Technical Committee activity and it consists in a quite open approach that allows the composition of elements coming from a repository of fragments of existing design processes that could be expressed in terms of a standard notation. Specifically dealing with the methods integration problem in this contribution, two different approaches have been considered to obtain methods integration: (i) guided by a MAS meta-model; (ii) guided by a development process. In the first approach, while building his own methodology, the designer has to preliminary identify the elements that compose the meta-model of the MAS he is going to build; then he has to choose the method fragments that are able to produce the identified meta-model elements. The second approach focuses on the instantiation of some software development process that completely cover the development of MAS. Given a specific problem and/or an application domain, the process will be instantiated by selection, for each phase, suitable method fragments, chosen from agent-oriented methodologies proposed in the literature or ad-hoc defined (AOSE TFG, 2004; Cossentino and Seidita, 2004; Garro et al., 2004).

A PRACTICAL INSTANCE OF PROPOSAL PLAN

As mentioned, to implement the plan proposed in proposal solution to agent-orientation promotion, an evaluation framework and a project-specific methodology building framework are needed. Existing approaches for evaluating agent-oriented methodologies shows that the evaluation framework presented in (Akbari, 2010) perfectly overcomes the shortcoming of most of the existing evaluation frameworks. In addition, since this framework is a feature-based framework, it has the following advantages as well:

1. Previous success (Sturm et al., 2004).
2. The possibility of implementation independent from external resources (e.g. industrial partners) (Sturm et al., 2004).
3. Lack of need of empirical information (Siau and Rossi, 1998)
4. The possibility of direct and detailed identification of methodologies' weaknesses in order to improve them by SME, with stressing on features.

Among existing project-specific methodology building frameworks, Agent OPEN matches the proposed plan best, since:

1. It is more complete and mature compared to others.
2. It has more existing resources compared to others, which facilitates the current research.
3. Method fragment repository of this method is richer compared to others.
4. It is also approved by FIPA (FIPA has some suggestions on merging its own method with Agent OPEN method).

Conclusion

The study of AOSE paradigm strengths shows the necessity of its usage; yet its current state reports relative lack of industrial acceptance compared to others. This paper

proposes a solution to this problem which aims to eliminate the weaknesses of this paradigm by the usage of an evaluation framework and a project-specific methodology building framework, simultaneously in a software development organization. The usage of SME, considerations for project management plans, and continuous improvements in the methodology through a wise combination of these frameworks may also lead the organization to reach the fifth level of CMM. In this regard, following future works are suggested:

1. Activities towards implementation and exploitation of the proposal plan:

a) Enriching the method fragment repository: The list of AOSE methodologies presented in existing agent-oriented software engineering methodologies may be used as a reference of methodologies, in order to extract their method fragments and complete the repositories of project-specific methodology building frameworks.
b) Storing the methodologies' evaluation results: The information stored may be used as a means to select suitable method fragments for building a project-specific methodology.

2. Activities towards completion of proposal plan details:

a) Enforcing the identification of the method fragments related to each criterion while storing a methodology: This will facilitate the selection of suitable method fragments with desired grades (level of property implementation) while building a project-specific methodology.
b) Defining a change management plan for continuous changes that occur in proposal plan structure and data: These changes may occur towards improving the evaluation framework, and/or the methodology in use.
3. Activities towards adding more capabilities to the proposal plan:

a) Preparing possibilities to design Domain-Specific Languages (DSL): The availability of project-specific methodologies is useless if no proper programming languages assure the software implementation. Thus, it is suggested to establish the facilities for designing DSLs along with the building project-specific methodologies as well.
b) Preparing possibilities to determine the proper paradigm for the project and change dominant paradigm of the proposal plan: As the software development organization needs to exploit a project-specific methodology, in case of wide range of projects handled by the organization, there may be the need for different paradigms as well. Thus the proposal plan may be equipped with a framework to select the proper paradigm to handle the project and follow the software development process with this paradigm, which needs suitable evaluation framework and project-specific methodology building framework as well.

REFERENCES

Abdelaziz T, Elammari M, Branki C (2008). MASD: towards a comprehensive multi-agent system development methodology, Meersman R, Tari Z, Herrero P. (Eds.), OTM Workshops, LNCS 5333: 108–117.

Akbari ZO, Faraahi A (2008). Evaluation Framework for Agent-Oriented Methodologies, Proceedings of World Academy of Science, Engineering and Technology, WCSET Paris, France, 35: 419-424, ISSN 2070-3740.

Akbari ZO, Faraahi A (2009). A Feature-Based Framework for Agent-Oriented Methodologies Evaluation, In Proceedings of CCSR, Tehran, Iran, pp 125-133.

Akbari ZO (2010). An Evaluation Framework for Agent-Oriented Methodologies and Its Utilization in Creating an Efficient Agent-Oriented Methodology, M. Sc Thesis, Payame Noor University, Tehran, Iran.

Alonso F, Frutos S, Martinez L, Montes C, Sonia (2004). a methodology for natural agent development, In Gleizes M.P., Omicini A, Zambonelli F (Eds.): ESAW. LNCS 3451, Springer-Verlag Berlin Heidelberg, (2005), pp 245-260.

AOSE Technical Forum Group (2004). AL3-TF1 Report, The first AgentLink III Technical Forum (AL3-TF1), Rome, Italy.

Athanasiadis IN, Mitkas PA (2009). A methodology for developing Environmental Information Systems with Software Agents, Whitestein Series in Software Agent Technologies and Autonomic Computing, Birkhauser Verlag Basel/Switzerland, pp. 119-137.

Bernon C, Gleizes MP, Picard G, Glize P (2002). The ADELFE methodology for an intranet system design, In Giorgini P, Lespérance Y, Wagner G and Yu E (Eds.). Proceedings of Agent-Oriented Information Systems (AOIS). pp. 1-15.

Bond AH, Gasser L (1988). A Survey of Distributed Artificial Intelligence, Readings in Distributed Artificial Intelligence, Morgan Kaufmann Publishers: San Mateo, CA.

Brazier FMT, Dunin-Keplicz BM, Jennings NR, Treur J (1997). DESIRE: modelling multi-agent systems in a compositional formal framework, Int J. Cooperative Inf. Syst. 6(1), 67-94.

Bresciani P, Giorgini P, Giunchiglia F, Mylopolous J, Perini A, (2004). Tropos: An agent-oriented software development methodology, Auton. Agents Multi Agent Syst. 8(3): 203-236.

Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J, (2001). A Knowledge Level Software Engineering Methodology for Agent-Oriented Programming, Proceedings of the 5th International Conference on Autonomous Agents, Agents'01, Montreal, Canada, pp. 648-655.

Brinkkemper S (1996). Method engineering: engineering of information systems development methods and tools, Inf. Software Technol. 38: 275-280.

Burmeister B (1996). Models and methodology for agent-oriented analysis and design, In Fischer K. editor, Working Notes of the KI'96 Workshop on Agent Oriented Programming and Distributed Artificial Intelligence, Dresden.

Bush G, Cranefield S, Purvis M (2001). The Styx agent methodology, The Information Science Discussion Paper Series, Number (2001/02).ISSN 1172-6024, Department of Information Science, University of Otago, Dunedin, New Zealand.

Caire G, Leal F, Chainho P, Evans R, Garijo F, Gomez J, Pavon J, Kearney P, Stark J, Massonet P (2000). Agent Oriented Analysis using MESSAGE/UML, In Wooldridge M., Weiss G, Ciancarini P, (Eds.). AOSE II, 119-135, LNCS 2222, Berlin: Springer-Verlag.

Castro J, Kolp M, Mylopoulos J (2001). A Requirements-Driven Development Methodology, In: Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01) pp. 108-123.

Castro J, Kolp M, Mylopoulos J (2002). Towards requirements-driven information systems engineering: The Tropos project, Inf. Syst. 27(6): 365-389.

Cernuzzi L, Rossi G (2002). On the Evaluation of Agent Oriented Methodologies, in Proc. of the OOPSLA (2002) Workshop on Agent-Oriented Methodologies.

Cervenka R, Trencansky I (2007). The Agent Modeling Language – AML, A Comprehensive Approach to Modeling Multi-Agent Systems, Whitestein Series in Software Agent Technologies and Autonomic Computing, ISBN: 978-3-7643-8395-4.

Cervenka R (2003). Modeling Notation Source: Prometheus, Version: 03-04-02, Foundation for Intelligent Physical Agents.

Clancey WJ, Sierhuis M, Seah C, Buckley C, Reynolds F, Hall T, Scott M (2007). Multi-agent simulation to implementation: a practical engineering methodology for designing space flight operations, In Artikis A et al. (Eds.): ESAW. LNCS 4995, Springer-Verlag Berlin Heidelberg, (2008) pp. 108-123.

CMS (2008), Selecting a development approach, Centers for Medicare and Medicaid Services (CMS), Original Issuance: February (2005). Revalidated: March.

Cockburn D, Jennings NR (1996). ARCHON: A distributed artificial intelligence system for industrial applications, In O'Hare GMP, Jennings NR, (editors), Foundations of Distributed Artificial Intelligence, pp. 319–344, JohnWiley & Sons.

Collier R (1996). The Realisation of Agent Factory: An Environment for the Rapid Prototyping of Intelligent Agents, M. Phil Thesis, UMIST, UK.

Collier RW (2002). Agent Factory: a framework for the engineering of agent-oriented applications, PHD Thesis, National University of Ireland.

Collier RW, O'Hare GMP (1999). Agent Factory: A Revised Agent Prototyping Environment, in 10th Irish Conference on Artificial Intelligence and Cognitive Science (AICS).

Collinot A, Drogoul A (1998). Using the Cassiopeia method to design a soccer robot team, Appl. Artif. Intell. (AAI) J. 12(2-3): 127-147.

Collinot A, Drogoul A, Benhamou P (1996). Agent-oriented design of a soccer robot team, In Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS'96), 41-57, Menlo Park, CA: American Association for Artificial Intelligence.

Cossentino M, Potts C (2002). A CASE tool supported methodology for the design of multi-agent systems. In Ababnia HR and Mun Y (Eds.), Proceedings of the International Conference on Software Engineering Research and Practice (SERP'02), Las Vegas pp. 315-321.

Cossentino M, Seidita V (2004). Composition of a New Process to Meet Agile Needs Using Method Engineering, Software Engineering for Large Multi-Agent Systems, vol. III, LNCS Series, Elsivier Ed.

Cossentino M (2005). From requirements to code with the PASSI methodology, In Henderson-Sellers B, Giorgini P (Eds.), Agent-oriented methodologies (Chapter 4). Hershey, PA: Idea Group.

Cossentino M, Gaud N, Hilaire V, Galland S, Koukam A (2007), ASPECS: an Agent-oriented Software Process for Engineering Complex Systems, In Proc. of the Fifth Agent Oriented Software Engineering Technical Forum (AOSE-TF5), Hammameth, Tunisia.

Cuesta P, Gomez A, Gonzalez JC, Rodriguez FJ (2002). The MESMA approach for AOSE, 4th Iberoamerican Workshop on Multi-Agent Systems (Iberagents'2002). a workshop of IBERAMIA'2002, The VIII Iberoamerican Conference on Artificial Intelligence.

Dam KH, Winikoff M (2003). Comparing agent-oriented methodologies, Proceedings of the 5th Int Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS), Melbourne, Australia.

Debenham J, Henderson-Sellers B (2003). Designing agent-based process systems - Extending the OPEN process framework, In Plekhanova V (Ed.), Intelligent agent software engineering, Chapter VIII pp. 160-190, Hershey, PA: Idea Group Publishing.

Debenham JK, Henderson-Sellers B (2002). Full lifecycle methodologies for agent-oriented systems – the extended OPEN process framework, In Proceedings of Agent-Oriented Information Systems (Eds. Giorini P, Lespreance Y, Wagner G, Yu E), Toronto pp. 87-101.

DeLoach SA (1999). Multiagent systems engineering: A methodology and language for designing agent systems, In Proceedings of the First International Bi-conference Workshop on Agent-Oriented Information Systems (AOIS '99), In the Third International Conference on Autonomous Agents, Seattle, USA.

DeLoach SA (2005). Multi-Agent Systems Engineering: An Overview and Case Study, In Henderson-Sellers B, Giorgini P (Eds.), Agent-oriented methodologies (Chapter 11), Hershey, PA: Idea Group.

Desai N, Mallya AU, Chopra AK, Singh MP (2005). OWL-P: a methodology for business process development, Kolp M, Bresciani P, Henderson-Sellers B, Winikoff M (Eds.), Agent-Oriented Information Systems III, 7th International Bi-Conference Workshop, AOIS. pp. 79-94.

Dikenelli O, Erdur RC (2002). SABPO: A Standards Based and Pattern Oriented Multi-agent Development Methodology, ESAW, 213-226.

Elammari M, Lalonde W (1999). An agent-oriented methodology: high-level and intermediate models, In the proceedings of AOIS (Agent-Oriented Information Systems), In the Third International Conference on Autonomous Agents, Seattle, USA.

Evans R, Kearney P, Stark J, Caire G, Garijo F , Gomez Sanz J, Pavon J, Leal F, Chainho P, Massonet P (2001). MESSAGE: Methodology for engineering systems of software agents, EURESCOM Technical Information.

Far BH (2001). Agent-SE: A Methodology for Agent Oriented Software Engineering, In Jin Q, Li J, Zhang N, Cheng J, Yu C, Noguchi S, (Eds.), Enabling Society with Information Technology, Springer pp. 357-366.

Ferber J, Gutknecht O (1997). Aalaadin: a meta-model for the analysis and design of organizations in multi-agent systems, rapport de recherche, Lirmm, univ. de Montpellier.

Ferber J, Gutknecht O (1998). A Meta-Model for the Analysis and Design of Organizations in Multi-Agent Systems, In Proceedings of the Third International Conference on Multi Agent Systems (ICMAS98), Paris, France.

Firesmith DG, Henderson-Sellers B (2002). The OPEN Process Framework: An Introduction, Addison-Wesley, UK, ISBN 0-201-67510-2.

Gadomski AM (1993). TOGA: A Methodological and Conceptual Pattern for modeling of Abstract Intelligent Agent, Proceedings of the First International Round-Table on Abstract Intelligent Agent, Gadomski AM, (editor) pp. 25-27.

Gao X, Sterling L (1998). A Methodology for Building Information Agents, In Yang Y, Li M, Ellis A, (editors), Web Technologies and Applications, Chapter 5, pp. 43-52, International Academic Pulishers.

Garro A, Fortino G, Russo W (2004). Using Method Engineering for the Construction of Agent-Oriented Methodologies, In Proc. of WOA 04 – Dagli Oggetti Agli Agenti, Sistemi Complessi e Agenti razionali, pp. 51-54, Torino, Italy.

Genesereth MR, Ketchpel SP (1994). Software agents, Communications of the ACM, 37, 7, pp. 48-53.

Gervais MP (2002). ODAC: An Agent-Oriented Methodology Based on ODP, J. Auton. Agents Multi Agent Syst. 7: 199-228.

Giret A, Botti V (2004). Towards a Recursive Agent Oriented Methodology for Large-Scale MAS, In Giorgini P, Muller JP, Odell J, (Eds.): AOSE (2003). LNCS 2935, 25–35, Springer-Verlag, Berlin, Heidelberg.

Giret A (2008). ANEMONA: a multi-agent methodology for holonic manufacturing systems, Springer Series in Advanced Manufacturing, First edition, ISBN-13: 978-1848003095,

Giret A, Botti V, Valero S (2005). MAS methodology for HMS, In Marik V, Brennan RW, Pechoucek M (Eds.): HoloMAS. LNCS 3593, Springer-Verlag Berlin Heidelberg pp. 39-49.

Glaser N (1996). The CoMoMAS methodology and environment for multiagent system development, In Zhang C, Lukose D (Eds.), Multi-agent systems methodologies and applications, pp. 1-16, Second Australian Workshop on Distributed Artificial Intelligence, LNAI 1286, Berlin: Springer-Verlag.

Graham I, Henderson-Sellers B, Younessi H (1997). The OPEN Process Specification, Addison-Wesley.

Grislin-Le Strugeon E, Anli A, Adam E (2006). A methodology to bring MAS to information systems, In Kolp M et al. (Eds.): AOIS. LNCS 4898, Springer-Verlag Berlin Heidelberg, (2008) pp. 90-104.

Hadj-Kacem A, Regayeg A, Jmaiel M (2007). ForMAAD: a formal method for agent-oriented application design, Web Intell. Agent Syst. 5(4): 435-454, IOS Press, Amesterdam, Netherland, ISSN: 1570-1263.

Harmsen AF (1997). Situational Method Engineering, Doctoral dissertation University of Twente, With ref., index and summary in Dutch, ISBN: 90-75498-10-1.

Henderson-Sellers B, Giorgini P (2005). Agent-Oriented Methodologies, Idea Group Publishing, ISBN 1-59140-587-4.

Henderson-Sellers B, Gorton I (2003). Agent-based Software Development Methodologies, White Paper on OOPSLA 2002 Workshop on Agent-Oriented Methodologies, COTAR, Sydney.

Henderson-Sellers B, Graham IM (1996). OPEN: Toward Method Convergence, IEEE Comput. 29(4): 86–89.

Henderson-Sellers B (2005). Evaluating the Feasibility of Method Engineering for the Creating of Agent-Oriented Methodologies, In Pechoucek M, Petta P, Varga LZ, (Eds.), CEEMAS pp. 142–152.

Henderson-Sellers B, Debenham J, Tran QNN (2004). Adding agent-oriented concepts derived from GAIA to Agent OPEN, In Advanced Information Systems Engineering: 16th International Conference, CAiSE. Riga, Latvia pp. 98-111, Berlin: Springer-Verlag.

Henderson-Sellers B, Debenham J, Tran QN, Cossentino M, Low G (2006). Identification of Reusable Method Fragments from the PASSI Agent-Oriented Methodology, In Agent Oriented Information Systems III, Lecture Notes in Computer Science, 3529, Springer-Verlag GmbH, 95-110.

Henderson-Sellers B, Giorgini P, Bresciani P (2003). Enhancing Agent OPEN with concepts used in the Tropos methodology, in Proceedings of the Fourth International Workshop Engineering Societies in the Agents World, Imperial College London, UK.

Henderson-Sellers B, Simons AJH, Younessi H (1998). The OPEN Toolbox of Techniques, Addison-Wesley, UK, 426 pp.

Henderson-Sellers B, Tran Q, Debenham J, Gonzalez-Perez C (2005). Agent-oriented information systems development using OPEN and the agent factory, Information Systems Development Advances in Theory, Practice and Education: 13th International Conference on Information Systems Development, ISD (2004). Vilnius, Lithuania, 149-160, New York: Kluwer Acadmic / Plenum Publishers.

Huget M (2002). Nemo: an agent-oriented software engineering methodology, In Proceedings of OOPSLA Workshop on Agent-Oriented Methodologies, Debenham J, Henderson-Sellers B, Jennings NR, Odell J, Seattle, USA.

Iglesias CA, Garijo M, Gonzalez JC (1999). A Survey of Agent-Oriented Methodologies, in Intelligents Agents IV: Agent Theories, Architectures, and Languages, 1555 of LNAI, Springer-Verlag pp. 317-330.

Iglesias CA, Garijo M, Gonzalez JC, Velasco JR (1998). Analysis and design of multiagent systems using MAS-CommonKADS. In Singh, MP, Rao A, Wooldridge MJ (eds.), Intelligent Agents IV (LNAI 1365), Springer-Verlag: Berlin Germany pp. 313-326.

Jennings N, Wooldridge M (1996). Software Agents, IEEE Rev. 17-20.

Jennings NR, Wooldridge M (2000). Agent-Oriented Software Engineering, In Handbook of Agent Technology (ed. Bradshaw J.), AAAI/MIT Press.

Jennings NR, Norman TJ, Faratin P (1998).ADEPT: An agent-based approach to business process management, SIGMOD Record 27(4): 32-39,

Juan T, Sterling L (2003). The ROADMAP meta-model for intelligent adaptive multiagent systems in open environments, In Giorgini P, Muller J, Odell J, (Eds.), Agent-Oriented Software Engineering IV, 4th International Workshop, AOSE. Melbourne, Australia, Revised Papers (LNCS 2935, 53–68), Berlin, Germany: Springer-Verlag.

Juan T, Pearce A, Sterling L (2002).ROADMAP: Extending the Gaia methodology for Complex Open Systems, Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Bologna, Italy.

Juan T, Sterling L, Winikoff M (2002). Assembling agent oriented software engineering methodologies from features, In Giunchiglia F, Odel J, Weiss G, (Eds.), Agent-Oriented Software Engineering III, Third International Workshop, AOSE. Bologna, Italy, Revised Papers and Invited Contributions (LNCS 2585, 198–209). Berlin, Germany: Springer-Verlag.

Juan T, Sterling L, Martelli M, Mascardi V (2003). Customizing AOSE methodologies by reusing AOSE features, In Rosenschein JS, Sandholm T, Wooldridge M, Yokoo M, (Eds.), Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Melbourne, Australia, 113–120.

Kendall EA, Malkoun MT, Jiang C (1996). A methodology for developing agent based systems for enterprise integration. In Lukose D, Zhang C, (editors), Distributed Artificial Intelligence Architecture and Modelling: Proceedings of the First Australian Workshop on DAI, LNCS 1087, Springer-Verlag: Heidelberg, Germany pp. 85-99.

Kindler C, DeLuke R, Rhea J, Kunz JC (1997). Development and Demonstration of an Agent-Oriented Integration Methodology, Kaman Sciences Corporation, Rome, NY, Contract Number F30602-94-C-0216.

Kinny D, Georgeff M (1996). Modelling and design of multi-agent systems, In Intelligent Agents III: Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages (ATAL-96), LNAI 1193, Berlin: Springer-Verlag.

Kinny D, Georgeff M, Rao A (1996). A methodology and modelling technique for systems of BDI agents, In Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW-96), Eindhoven, The Netherlands, pp. 56-71, Springer.

Kumar K, Welke RJ (1992). Method Engineering, a Proposal for Situation-Specific Methodology Construction, In Systems Analysis and Design: A Research Agenda, Cotterman and Senn (Eds.), Wiley pp. 257-268.

Kumar M (2002). Contrast and comparison of five major Agent Oriented Software Engineering (AOSE) methodologies, Available at http://students.jmc.ksu.edu/grad/madhukar/www/professional /aose paper.pdf.

Li C, Liu L (2005). MAHIS: An Agent-Oriented Methodology for Constructing Dynamic Platform-Based HIS, Australian Conference on Artificial Intelligence, 705-714.

Lin C, Kavi KM, Sheldon FT, Daley KM, Abercrombie RK, (2007). A Methodology to Evaluate Agent Oriented Software Engineering Techniques, Software Agents and Semantic Web Technologies Minitrack, IEEE Proc. HICSS-40, Big Island HI.

Lind J (1999). MASSIVE: Software Engineering for Multiagent Systems, PhD Thesis, University of Saarland, Saarbrucken.

Lind J (2001). Iterative software engineering for multiagent systems, The MASSIVE Method, (LNAI 1994), Berlin: Springer-Verlag.

Lu H, Chhabra M (2006). A methodology for agent oriented web service engineering, In Shi Z, Sadananda R (Eds.): PRIMA LNCS 4088, Springer-Verlag Berlin Heidelberg, 2006 pp. 650-655.

Lyytinen K (1987). A Taxonomic Perspective of Information Systems Development: Theoretical Constructs and Recommendations, In Critical Issues in Information Systems Research, Jr, R J B, Hirschheim RA, (eds.) John Wiley & Sons Ltd., 3-41.

Mao X, Zhao J, Wang J(2009). Engineering adaptive multi-agent systems with ODAM methodology, In Ghose A, Governatori G, Sadananda R, (Eds.): PRIMA 2007, LNCS 5044, Springer-Verlag Berlin Heidelberg pp. 380-385.

Martelli M, Mascardi V, Zini F (1997). CaseLP: a Complex Application Specification Environment based on Logic Programming, In Proc. of ICLP'97 Post Conference Workshop on Logic Programming and Multi-Agents pp. 35-50.

Maynard HB, Stegemerten GJ (1939). Operation Analysis, McGraw-Hill, New York.

Moulin B, Brassard M (1996). A Scenario-Based Design Method and an Environment for the Development of Multiagent systems, In Lukose D, Zhang C, (editors), Proceedings of the First Australian workshop on Distributed Artificial Intelligence, Lecture Notes in Artificial Intelligence, No. 1087, pp. 216-231, Springer-Verlag.

Muller HJ (1996). Towards agent systems engineering, , Special Issue on Distributed Expertise, Int. J. Data Knowledge Eng. (23): 217–245.

Mylopoulos J, Kolp M, Castro J (2001). UML for agent-oriented software development: The Tropos proposal, Proceedings of the 4th

International Conference on the Unified Modeling Language, UML'01, Toronto, Canada, October 1-5, Springer pp. 422-442.
Neal Reilly WS (1997). A Methodology for Building Believable Social Agents, Proceedings of the First International Conference on Autonomous Agents (Agents '97), Marina del Rey, CA, USA, 114-121, ACM Press, New York, ISBN 0-89791-877-0, ACM Order Number 605971.
Nikraz M, Caire G, Bahri PA (2006). A methodology for the development of multi-agent systems using the JADE platform, Computer Systems Science and Engineering, 21(2), 99–116.
Nwana H, Ndumu D, Lee L, Collis J (1999). ZEUS: A Tool-Kit for Building Distributed Multi-Agent Systems, Appl. Artif. Intell. J. 13(1): 129-186.
O'Hare GMP, Wooldridge MJ (1992). A software engineering perspective on multi-agent system design: Experience in the development of MADE, In Avouris NM, Gasser L, (Eds), Distributed Artificial Intelligence: Theory and Praxis pp. 109–127, Kluwer Academic Publishers: Boston, MA.
O'Hare GMP, Collier RW, Conlon J, Abbas S (1998). Agent Factory: An Environment for Constructing and Visualising Agent Communities, 9th Irish Conference on Artificial Intelligence and Cognitive Science (AICS).
Omicini A (2000). SODA: societies and infrastructures in the analysis and design of agent-based systems, In Agent-Oriented Software Engineering, LNCS, 1957 pp. 185-193, Berlin: Springer-Verlag.
Padgham L, Winikoff M (2002). Prometheus: A methodology for developing intelligent agents, In Giunchiglia F, Odell J, Weiß G, (Eds.), Agent-Oriented Software Engineering III Proceedings of the Third International Workshop on Agent-Oriented Software Engineering (AAMAS'02) pp. 174-185, LNCS 2585.
Padgham L, Winikoff M (2002). Prometheus: A pragmatic methodology for engineering intelligent agents, In Debenham J, Henderson-Sellers B, Jennings N, Odell JJ, (Eds.), Agent-oriented Software Engineering III Proceedings of the Workshop on Agent-oriented Methodologies at OOPSLA. Seattle pp. 97-108, Sydney: Centre for Object Technology Applications and Research.
Pavon J, Gomez-Sanz J (2003). Agent Oriented Software Engineering with INGENIAS, Proc. 3rd International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS). Marik V, Muller J, Pechoucek M, (Eds.), Multi-Agent Systems and Applications II, LNAI 2691, Spring-Verlag pp. 394-403.
Pavon J, Gomez-Sanz JJ, Fuentes R (2005). The INGENIAS methodology and tools, In Henderson-Sellers B, Giorgini P, (Eds.), Agent-oriented methodologies (Chapter 9), Hershey, PA: Idea Group.
Sabas A, Badri M, Delisle S (2002). A Multidimentional Framework for the Evaluation of Multiagent System Methodologies, 6th World MultiConf on Systemics, Cybernetics and Informatics (SCI) pp. 211-216.
Sardinha J, Milidiu R, Lucena C, Paranhos P (2004). A Methodology for Building Trading Agents in Electronic Markets, Technical Report, Computer Science Department, PUC-Rio, Brazil, PUC-RioInf.MCC36/04.
Shan L, Zhu H (2004). Software engineering for multi-agent systems III: Research issues and practical applications, In Choren R, Garcia A, Lucena C, Romanovsky A, (Eds.), Proceedings of the Third International Workshop on Software Engineering for Large-Scale Multi-Agent Systems pp. 144-161, Berlin: Springer-Verlag.
Shehory O, Sturm A (2001). Evaluation of modeling techniques for agent-based systems, Agents pp. 624-631.
Shi Z, Jiao W, Sheng Q (2001). Agent-oriented software methodology, CEEMAS Cracow, Poland.
Shi Z, Zhang H, Cheng Y, Jiang Y, Sheng Q, Zhao Z (2004). MAGE: An Agent-Oriented Programming Environment, IEEE ICCI pp. 250-257.
Shi Z, Zhang H, Dong M, Zhao Z (2003). MAGE: Multi-Agent Environment, Proc. of the Int. Conference on Compt. Networks and Mobile Computing (ICCNMC'03) pp. 181-188.
Shoham Y (1990). Agent-Oriented Programming, Technical Report STAN-CS-1335-90, Computer Science Department, Stanford University, Stanford, CA 94305.
Shoham Y (1993).Agent-Oriented Programming, Artif. Intell. 60(1):51-92
Siau K, Rossi M (1998).Evaluation of Information Modeling Methods – A Review, In Proc. 31 Annual Hawaii International Conference on System Science pp. 314-322.
Sierra C, Sabater J, Agusti J, Garcia P (2002). Evolutionary Programming in SADDE, AAMAS'02, ACM, Bologna, Italy pp. 1270-1271.
Singh MP, Huhns MN, Stephens LM, (1993). Declarative representations of multiagent systems, IEEE Trans. Knowledge Data Eng. 5(5): 721–739.
Standards Coordinating Committee of the Computer Society of the IEEE, (1990). IEEE Standard Glossary of Software Engineering Terminology, IEEE Standards Board, IEEE Std 610.12.
Sterling L, Taveter K (2009). The art of agentoriented modeling, Cambridge, MA, London, England: The MIT Press.
Sturm A, Shehory O (2003). A Framework for evaluating agent-oriented methodologies, In Giorgini P, Winikoff M (Eds.), Proceedings of the Fifth Int. Bi-Conference Workshop on Agent-Oriented Information Systems pp. 60-67, Melbourne, Australia.
Sturm A, Dori D, Shehory O (2003). Single-Model Method for Specifying Multi-Agent Systems, Proceeding of Second Int. Joint Conference on Autonomous Agents and Multi Agent Systems pp. 121-128.
Sturm A, Shehory O, Dori D (2004). Evaluation of Agent-Oriented Methodologies, In AgentLink AOSE TFG1.
Sudeikat J, Braubach L, Pokahr A, Lamersdorf W (2004). Evaluation of agent-oriented software methodologies: Examination of the gap between modeling and platform, Proceedings of the Workshop on Agent-Oriented Software Engineering (AOSE), New York, USA.
Taveter K, Wagner G, (2005). Towards radical agent-oriented software engineering processes based on AOR modelling, In Henderson-Sellers B, Giorgini P, (Eds.), Agent-oriented methodologies (Chapter 10), Hershey PA: Idea Group.
Taveter K, Wagner G (2002). A multi-perspective methodology for modelling inter-enterprise business processes, In Arisawa H, Kambayashi Y, (Eds.): ER (2001). Workshops, LNCS 2465, Springer-Verlag Berlin Heidelberg pp. 403-416.
Taveter K, Sterling L (2008), Features as Loosely Defined Method Fragments, AOSE TFG08.
Tian J, Foley R, Tianfield H (2004). A new agent-oriented development methodology, Proceedings of the Intelligent Agent Tech., IEEE/WIC/ACM Int. Conference pp. 373–376.
Tran QN, Low G (2008). MOBMAS: A Methodology for Ontology-Based Multi-Agent Systems Development, Inf. Software Technol. 50: 697–722.
Tran QNN, Beydoun G, Low G (2007). Design of a peer-to-peer information sharing MAS using MOBMAS (ontology-centric agent-oriented methodology, In Advances in Information Systems Development, Springer pp. 63-76.
Verharen E, Weigard H (1994). Agent-Oriented Information Systems Design, In Ras Z, Zemankova M, editors, Poster Proceedings of the International Symposium on Methodologies for Intelligent Systems (ISMIS'94), Amsterdam, 378- 392.
Verharen EM (1997). A Language-Action Perspective on the Design of Cooperative Information Agents, PhD thesis, Katholieke Universieit Brabant, the Netherlands.
Villaplana EA (2005). Proposal for an organizational MAS methodology, AAMAS'05, 1370.
Wagner G (2003). The agent-object-relationship meta-model: Towards a unified view of state and behavior, Inf. Syst. 28(5): 475-504.
Wang L, Guo Q (2004). Mobile Agent Oriented Software Engineering (MAOSE), In Karmouch A, Korba L, Madeira E (Eds.): MATA LNCS 3284, Springer-Verlag Berlin Heidelberg pp. 168-177.
Whitestein Technologies (2008). LS/TS Product Brochure, Available at http://www.whitestein.com/library/Whitestein Technologies_LS-TS_ProductBrochure.pdf.
Wooldridge M, Ciancarini P (2001). Agent-Oriented Software Engineering: The State of the Art, In Ciancarini P. and Wooldridge M. (editors), Agent-Oriented Software Engineering, Springer-Verlag Lecture Notes in AI (1957).
Wooldridge M (1997). Agent-based software engineering, IEE Proc. Software Eng. 144(1): 26–37.

Wooldridge M, Jennings NR, Kinny D (1999). A methodology for agent-oriented analysis and design, In Proceedings of the Third International Conference on Autonomous Agents (Agents 99), 69–76, Seattle, WA.

Wooldridge M, Jennings NR, Kinny D (2000). The Gaia methodology for agent-oriented analysis and design, J. Autonomous Agents Multi Agent Syst. 3(3): 285-312.

Wooldridge M, Jennings NR (1995). Intelligent agents: theory and practice, Knowl. Eng. Rev. 10(2) 115–152.

Yan Q, Shan L, Mao X, Qi Z (2003). RoMAS: a role-based modeling method for multi-agent systems, Proceedings of International Conference on Active Media Technology pp. 156-161.

Yan E, Cysneiros LM (2002). Agent-Oriented Methodologies – Towards A Challenge Exemplar, 4th Intl. Workshop on Agent-Oriented Information Systems (AOIS'02).

Zamboneli F, Jennings NR, Omicini A, Wooldridge M (2000). Agent-Oriented Software Engineering for Internet Applications, Published as chapter 13 in the book: Coordination of Internet Agents: Models, Technologies and Applications, Omicini A, Zambonelli F, Klusch M, Tolksdorf R (Eds.), Springer.

Zambonelli F, Jennings NR, Wooldridge M (2005). Multi-Agent Systems as Computational Organizations: The Gaia Methodology, In Henderson-Sellers B, Giorgini P (Eds.) Agent-oriented methodologies (Chapter 6), Hershey, PA: Idea Group.

Zhang T, Kendall EA, Jiang H (2002). An Agent-Oriented Software Engineering Methodology with Application of Information Gathering Systems for LCC, Proceedings of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002 at CAiSE*02) pp. 1-15.