A Comparison of Three Agent-Oriented Software Development Methodologies: ROADMAP, Prometheus, and MaSE

Ebrahim Al-Hashel, Bala M. Balachandran, and Dharmendra Sharma

School of Information Sciences and Engineering
University of Canberra, ACT 2601, Australia
{ebrahim.al.hashel, bala.balachandran,
dharmendra.sharma}@canberra.edu.au

Abstract. Agent-Oriented Software Development is one of the recent contributions to the field of Software Engineering. To date numerous methodologies for agent-oriented software development have been proposed in the literature. However, their application to real-world problems is still limited due to their lack of maturity. Evaluating their strengths and weaknesses is an important step towards developing better methodologies in the future. This paper presents research results obtained by applying three agent-oriented methodologies, namely ROADMAP, MaSE and Prometheus in the context of an E-commerce system. The results are presented and future work is discussed.

Keywords: Multi-agent system, software engineering, agent-oriented software development, agent-oriented methodology, AOSE.

1 Introduction

Since the 1980s, software agents and multi-agent systems have grown into what is now one of the most active areas of research and development activity in software engineering. Agent-oriented computing offers a new perspective on software design for managing the inherent complexity of software systems [1]. Unlike an object from object-oriented paradigm, an agent serves as a heuristic guideline in the development of software systems [14].

Today agent technology is applied to many domains including robotics, network security, computer games, and e-commerce. However, one of the most fundamental obstacles to large scale take-up of agent technology is the lack of mature agent-oriented software development methodologies [2]. Over the past few years, there have been several attempts at creating tools and methodologies for building agent systems. Tveit [4] presents a comprehensive survey on currently available agent tools and methodologies. The current methodologies can be divided into three categories as follows [10]:

- Agent-oriented methodologies.
- Methodologies that directly extend or adapt object-oriented methodologies.
- Methodologies that adapt knowledge engineering models or other techniques.

B. Apolloni et al. (Eds.): KES 2007/ WIRN 2007, Part III, LNAI 4694, pp. 909–916, 2007.

If the full potential of agents as a software engineering paradigm is to be successful, establishing a systematic methodology is imperative. Currently there exist a number of agent-oriented methodologies such as Gaia [3], ROADMAP [5], Tropos [16], Prometheus [7], and MaSE [8]. So it becomes necessary to analyse and compare these methodologies and help developers to choose the most appropriate one.

Over the past few years, there have been some attempts at comparing agent-oriented tools and methodologies, [11], [12], [17], [18]. For example, Shehory and Sturm performed a feature-based evaluation of several AOSE methodologies. Their evaluation criteria are related to software engineering and agent concepts. Dam and Winikoff presented a comparison of agent methodologies in terms of concepts, modelling language, process and pragmatics.

In this paper, our aim is to present our experience in using three agent-oriented methodologies to engineer a real-world multi-agent system. The methodologies we have chosen are ROADMAP, MaSE and Prometheus. Our experimental application is a Travel Agency System (TAS) [13].

This paper is organised as follows. Section 2 describes our experimental problem domain. Section 3 overviews the three agent-oriented methodologies and presents some of the analysis and design models we created by using each of these methodologies. Section 4 describes our methodology evaluation criteria and presents the comparison results. Section 5 summarises our conclusions and future work.

2 A Case Study in E-Commerce

Today the travel industry is increasingly becoming internationalized. Airlines, accommodators and tour operators control the market, but these retailers face high costs in communication and personnel. The customers adopt the Internet travel sites model as a medium to purchase a travel package. Customers, usually looking for holiday packages, depend on their travel agent to show them what is available according to their taste and budget, or visit specific content providers, such as airlines, accommodators, and car rental agencies. In this environment, intelligent agents have a great potential in helping the customer get the best ‘deal’ on a travel package. The proposed Travel Agency System (TAS) is a multi-agent system designed to obtain travel packages for user depending on their preferences.

3 Agent-Oriented Methodologies

In this section we briefly overview the three agent-oriented methodologies and present some of the analysis and design models we created by using each of these methodologies.

3.1 The ROADMAP Methodology

The ROADMAP (Role Oriented Analysis and Design for Multi-Agent Programming) methodology extends Gaia [6] and focuses on developing open systems. The analysis phase in the original Gaia methodology is extended to cover specification as well. A

use-case model is introduced to support requirements gathering. An environment model and a knowledge model are derived from the use-cases and provide holistic views on the execution environment and the domain knowledge.

ROADMAP is supported by a development tool called REBEL (Roadmap Editor Built for Easy Development) which is designed to help the developer to identify the Goal Models and the Role Models during the analysis stage.

3.2 Prometheus Methodology

The Prometheus methodology is a detailed process for specifying, designing, and implementing multi-agent systems. Prometheus consists of three phases: system specification, architectural design and detailed design. The first phase, system specification, deals with determining the system's environment and establishing its goals and functionalities. The environment is defined in terms of percepts, actions, and external data, while the system's functionality is identifies in terms of goals and sub-goals.

The Prometheus method uses the concept of grouping functionalities. The main reason for grouping functionalities together is to identify the data coupling. Prometheus is supported by two tools: The JACK development environment (JDE), and the Prometheus Design Tool (PDT).

3.3 MaSE Methodology

Multi-agent Systems Engineering (MaSE) [8] is a complete-lifecycle development methodology for designing and developing a multi-agent system. It takes an initial set of requirements and analyses, designs and implements a working multi-agent system. MASE is architecture-independent and models a system in terms of goals, roles, agents, tasks and conversations. The MaSE methodology consists of two phases: analysis and design.

The first phase, Analysis, focuses on capturing goals, applying use cases and refining roles. The second phase, Design, focuses on creating agent classes, designing interactions, assembling agent classes, and building deployment diagrams.

MaSE is supported by the agentTool [15] for development which is a graphically based fully interactive software engineering tool. The agentTool also provides automated support for transforming analysis models into design artefacts.

4 The Comparison Criteria and Results

The comparison criteria we have chosen are agent concepts, development phases, artefacts and models, modelling notations, nature of applications, and development tools. To date, there is no evaluation schema or a standard that can be used in the evaluation process [17]. Our comparison results are summarised in the following tables:-

Table 1. Illustrates the scale of the details within each development phase

Phases	ROADMAP	MaSE	Prometheus
System specification	Detailed	Medium	Detailed
Analysis	Detailed	Detailed	Detailed
Architectural design	Abstract, high-level	Detailed	Detailed
Detailed design	Not exists (Architecture)	Not exists (Architecture)	Detailed (BDI agents)

Table 2. Presents the measure of agent concept that each methodology support

Concept	ROADMAP	MaSE	Prometheus
Autonomy	Medium	Medium	High
Mental attitudes	Uses knowledge schema.	Agents do not have to be intelligent	Agents are intelligent agents. BDI agents.

Table 3. Shows the scale of the modelling criteria within each methodology

Criteria	ROADMAP	MaSE	Prometheus
Clear notation	Strongly agree	Strongly agree	Strongly agree
Ease of learning	Strongly agree	Strongly agree	Agree
Ease of use	Agree	Strongly agree	Agree
Adaptability	Strongly agree	Strongly agree	Agree
Traceability	Strongly agree	Agree	Strongly agree

Table 3. (*continued*)

Consistency	Agree	Agree	Agree
Refinement	Agree	Agree	Agree
Scalability	Agree	Do not agree	Do not agree
Concept overload	Medium	Medium	Low

Table 4. Compares the properties of the methodologies

Property	**ROADMAP**	**MaSE**	**Prometheus**
Openness	High	Low	Medium
Environment	High	Medium	Medium
Abstraction	High	High	High
Traceability	High	High	High
Modelling	Medium	High	High
Complexity	Low	Low	Medium
Ease of use	Easy, requires some	Easy	Complicated,
Limitations	Lack of richer notations	Lack of	Highly
Language	Low	Medium	High
Reusability	High	Medium	Medium

Table 5. Illustrates the available activities in each development phase

Phases	ROADMAP	MaSE	Prometheus
System Specification			Stakeholders Scenarios diagram
Analysis	Environment, Knowledge, Goal, Role, Revised role model, Social model	Use cases, Goal hierarchy, Sequence, Concurrent task diagram	Goal overview, Role, Data coupling diagrams
Architectural Design	Agent l, Service, Acquaintance model	Agent classes, Conversations	Agent acquaintance, System Overview Agent Descriptors Protocols
Detailed Design		Agent's internal architectures, Deployment diagram	Process , Agent Overview Diagrams, and Capacity, Capability overview, Event, Data, and Plan

Table 6. Illustrates the type of the system domain that each methodology is suitable for

Methodology	Application
ROADMAP	Coarse-grained computational, complex, open systems
MaSE	Heterogeneous multi agent systems
Prometheus	Intelligent (BDI) agents' systems

Table 7. Summarises the toolkits that are available for each methodology

Methodology	Development Tool
ROADMAP	REBEL is a tool for building Goal Models and Role Models during the analysis stage.
MaSE	AgentTool, able to do printing, verification on developing system, and generating a skeleton code in java.
Prometheus	Prometheus Design Tool (PDT), which is able to do cross checking, saving diagrams as pictures, JDE (JACK Development Environment) generates JACK code.

5 Conclusions and Future Work

In this paper we have presented an overview of three popular agent methodologies, namely ROADMAP, MaSE, and Prometheus and discussed how they handle the analysis and design of multi-agent systems. Generally the above three methodologies provide some support for the analysis and design of agent-oriented systems. In addition, each methodology also provides a toolkit for the development.

Each methodology is focused on different type of agent applications, matching the methodology with the system requirement need further investigation. For instance, if we need intelligent (BDI) agents, Prometheus is good; if the system has heterogeneous agents, then MaSE is the most suitable method; but for coarse grained computational agents, ROADMAP is recommended.

We are currently implementing the TAS system using JADE [9]. We will be reporting our research progress in the subsequent papers.

References

1. James, O.: Object and Agents Compared. Journal of Object Technology 1(1), 41–53 (2002)
2. Luck, M., Ashri, R., D'Inverno, M.: Agent-Based Software Development. Artech House, London (2004)
3. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design (2000) (retrieved, February 15, 2006), http://www.ecs.soton. ac.uk/ nrj/download-files/jaamas2000.pdf
4. Tveit, A.: Survey of Agent-Oriented Software Engineering. First NTNU CSGSC (2001)
5. Juan, T., Sterling, L.: The ROADMAP Meta-Model for Intelligent Adaptive Multi-Agent Systems in Open Environments. In: Giorgini, P., Müller, J.P., Odell, J.J. (eds.) Agent-Oriented Software Engineering IV. LNCS, vol. 2935, Springer, Heidelberg (2004)
6. Juan, T., Pearce, A., Sterling, L.: ROADMAP: Extending the Gaia Methodology for Complex Open Systems. In: Falcone, R., Barber, S., Korba, L., Singh, M.P. (eds.) AAMAS 2002. LNCS (LNAI), vol. 2631, Springer, Heidelberg (2003)
7. Padgham, L., Winikoff, M.: Developing intelligent Agent Systems: A Practical Guide. John Wiley & Sons, Chichester (2004)
8. Mark, F., Wood, Scott, A., DeLoach, L.: An Overview of the Multiagent Systems Engineering Methodology, pp. 207–221. Springer, Heidelberg (2001)
9. http://www.cis.ksu.edu/ sdeloach/publications/Conference/mase-aose2000.pdf
10. Bellifemine, F., Caire, G., Trucco, T.: JADE Adminstirator,s Guide. GmbH (2006)
11. Fasli, M.: Agent Technology for e-Commerce. John Wiley and Sons, UK (2007)
12. Shehory, O., Sturm, A.: Evaluation of modelling techniques for agent-oriented systems. In: Fifth International Conference on Autonomous Agents, pp. 624–631 (2001)
13. Dam, K.H., Winikoff, M.: Comparing Agent-Oriented Methodologies. In: Giorgini, P., Henderson-Sellers, B., Winikoff, M. (eds.) AOIS 2003. LNCS (LNAI), vol. 3030, Springer, Heidelberg (2004)
14. Far, B.H.: Sample Project: Travel Agency System (TAS) (2004) (retrieved February 2, 2006) http://www.enel.ucalgary.ca/People/far/

15. Wooldridge, M.: Introduction to Multiagent Systems. John Wiley and Sons, UK (2002)
16. AgentTool 1.8: User Manual http://www.cis.ksu.edu/ sdeloach/ai/software/agentTool-1.8.3.exe
17. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An Agent-Oriented Software Development Methodology. In: AAMAAS- 8, pp. 203–236 (2004)
18. Numi, Q., Low, G., Williams, M.: A Feature Analysis Framework for Evaluating Multi-agent System Development Methodologies. In: Zhong, N., Raś, Z.W., Tsumoto, S., Suzuki, E. (eds.) ISMIS 2003. LNCS (LNAI), vol. 2871, pp. 613–617. Springer, Heidelberg (2003)
19. Sudeikat, J., Braubach, L., Pokahr, A., Lamersdorf, W.: Evaluation of Agent-Oriented Software Methodologies - Examination of the Gap Between Modeling and Platform. In: Müller, J.P., Zambonelli, F. (eds.) AOSE 2005. LNCS, vol. 3950, pp. 126–141. Springer, Heidelberg (2006)