MULTI-AGENT BASED SOFTWARE ENGINEERING MODELS: A REVIEW

[1]Okoye, C. I, [2]John-Otumu Adetokunbo M, and [3]Ojieabu Clement E.

[1,2]Department of Computer Science, Ebonyi State University, Abakaliki, Nigeria
[3]Department of Electrical/Electronic Engineering, Ambrose Alli University, Ekpoma, Nigeria

ABSTRACT:

This paper critically examined nine different software models for modeling / developing multi-agent based systems. The study revealed that the different models examined have their various advantages / disadvantages and uniqueness in terms of practical development/deployment of the multi-agent based information system in question. The agentology model is one methodology that can be easily adopted or adapted for the development of any multi-agent based software prototype because it was inspired by the best practices and good ideas contained in other agent oriented methodologies like CoMoMas, MASE, GAIA, Prometheus, HIM, MASim and Tropos.

KEYWORDS:

Multi-agent, artificial intelligence, software model, agentology

1. INTRODUCTION

The true concept of agent and multi-agent based technology originated from artificial intelligence (AI) [1, 2]. Thus the roots of the concept extend back to the 1950s when AI was born. Agent concept also has its roots in other research areas like: robotics, distributed systems, and computer graphics. Agent work in robotics and artificial intelligence (AI) was originally strongly interrelated. According to [3] robots such as SHAKEY were programmed to exhibit autonomous behavior in well-defined environments, and laid the groundwork for AI planning systems to this day. Research work by [4] shows that the first software agent was probably "ELIZA", a program which could engage in a conversation with a user. Another influential program called "SHRDLU" according to [5], allowed a person to have a conversation with a simulated robot.

The notion of multi-agent systems was brought to the limelight by Marvin Minsky in his work on the "Society of Mind" [6]. His vision was that a complex system such as the human mind should be understood as a collection of relatively simple agents, each of which was a specialist in certain narrow domains. Through structures called K-lines, agents would activate each other whenever their context became relevant. Though the work of Minsky showed remarkable vision, but was ahead of its time since software complexity had not yet reached the level of where the advantages of such structures would have a practical impact.

DOI: 10.5121/ijsea.2017.8209

Agent can be defined as a computer system situated in an environment with the ability of taking autonomous actions in order to meet its desired goals [7], while multi-agent system according to [8, 9] is concerned with a system or collective behavior of existing autonomous agents in some environment aiming at proffering solutions to a given problem. [7] is also of the opinion that a multi-agent based system can be seen as a system that is a loosely coupled network troubleshooter or solution provider that works together to solve issues that is beyond the capability of an individual agent.

2. LITERATURE REVIEW

Developing agent-based software requires a systematic engineering approach that supports and drives a development team along all the phases of the software production process. A software engineering methodology aims to describe all the elements necessary for the development of a software system. So, a methodology uses a modeling language to capture and describe requirements of the system to be, a modeling language to describe the architectural components and details of their interaction. Various forms of analysis is required to reason about models, a structured process to guide analysts and developers activities, and tools to support and semi-automate the developing process.

Many agent oriented software methodology exist, but for the purpose of this research work, nine of them were carefully selected and reviewed: Agent Unified Modeling Language (AUML), MASE, High-level and Intermediate Models (HIM), CoMoMAS, GAIA, MASim, Prometheus, Tropos and Agentology.

According to [10] agent oriented methodologies adopts several notations for agent software development, while traditional methodologies are inadequate for such development. [10] also used the Agent Unified Modeling Language (AUML) to characterize some activities in their research work on agents; though the AUML is an extension of UML for modeling multi-agent based application behaviors, it is still not a primary methodology that would say what to do; it just offer tools for using other methodologies. [11] acknowledges MASE as a multi-agent methodology that focuses on the development of robotics mainly, [12] also acknowledges that though MASE methodology is multi-agent based, but it has a strong background from Object Oriented Methodology (OOM).

Cassiopeia and High-level and Intermediate Models (HIM) are general-purpose methodologies for designing multi-agent systems; but [13] sees HIM as a methodology that offers five models, but this model is relatively less known for agent development.

According to [14] CoMoMAS is another good methodology and environment for the development of multi-agent based systems; [15] acknowledges that the beauty of CoMoMAS methodology is its focus on knowledge engineering, which is based on CommonKADS. Though CoMoMAS has five distinct phase of development that speaks interestingly well about the methodology, but unfortunately [14] is of the opinion that due to its focus on knowledge, it is not very suitable for the purpose of social simulations.
[16] is of the opinion that GAIA methodology is one of the first proposed agent-oriented software engineering methodologies. GAIA belongs to the most often cited multi-agent methodologies; [17] is of the opinion that GAIA is a universal methodology and its process is in three phases or

stages. Unfortunately, the GAIA methodology is closed and static, i.e. it hardly deals with unusual requirements. The methodology does not deal with designing agent's environment.

MASim is a methodology that is focused particularly on agent-based simulations, due to the fact that it is worthy of closer exploration, and used in the design of both small and large systems [18]. MASim is about the most appropriate agent-based methodology because it covers not only design phases, but also the development and verification phases, in which a tester is responsible for testing (verification and validation) the application according to the pre-established requirements [18].

Prometheus is another general-purpose and practical oriented agent-based software engineering methodology that covers the overall development process in details [19]. The major phases of the methodology supported are: system specification, architectural design and detailed design, but mainly uses UML and AUML notations for its diagrams. According to [20] Prometheus methodology is tool-supported i.e. Prometheus Design Tool (PDT) and JACK Development Environment (JDE). The PDT allows users to create and elaborate Prometheus design, while JDE is used for the skeleton code generation from the design diagrams.

Another well-known and deeply elaborate multi-agent methodology is the Tropos [21]; the methodology has five main development phases, namely, early requirements, late requirements, architectural design, detailed design and implementation phase. It is a requirement driven methodology based on the concepts used during early requirements analysis [21], and according to [22] it supports a full tool methodology that spans four phases. Some researchers are still of the opinion that the available agent-oriented methodologies are not yet well suited and crafted for real the world application, it is concurrent being observed that research is focusing on moving from basic foundational issues to a more practical software engineering method [23]. It is a known fact that one of the key issues in the transition of agents from research to industrial practice is the need for a mature software engineering methodology for the specification and design of agent systems.

Although there are many other agent methodologies as discussed above, the only relevant methodology that focused specifically on the design and development of agent is the MASim, but its limitations makes it difficult to apply.

Agentology is another sound methodology for the development of agent-based systems that can be easily applicable from requirements to development stage. Agentology is a methodology that was inspired by the best practices and good ideas contained in several other methodologies. The methodology is independent of any agent paradigm, framework, technology and programming language. The design and development process in agentology consists of four phases and nine steps. Each step works with the results of the previous one, so the process has the structure of a successive or iterative approach.

3. FINDINGS AND DISCUSSIONS

The findings and discussions of this study are tabulated in Table 1.

Table 1: Findings of the methodologies reviewed

S/N	Methodology	Advantages	Disadvantages	Comments
1	**AUML**	1. The AUML offer tools for using other methodologies	1. It is not a primary methodology that would say what to do.	1. The AUML is an extension of UML for modeling multi-agent based application behaviors.
2	**MASE**	1. MASE methodology is fully multi-agent based.	1. It has a strong background from Object Oriented Methodology (OOM).	1. MASE as a multi-agent methodology that focuses on the development of mainly robotics.
3	**HIM**	1. HIM is a general-purpose methodology for designing multi-agent.	1. The HIM methodology is relatively less known for agent development.	HIM methodology offers five models
4	**CoMoMAS**	1. It is focused on engineering knowledge based on CommonKADS	1. This methodology is not suitable for social simulations.	1. It is a good methodology for the development of multi-agent based systems
5	**GAIA**	1. Methodology for developing both medium and large multi-agent systems 2. It is situated in an open and dynamic environment 3. It can guarantee predictable and reliable behavior 4. It gives agent a well-defined position with expected behavior.	1. The methodology is closed and static 2. The methodology deals with unusual requirement. 3. The methodology does not deal with designing agent's environment.	1. One of the first proposed agent oriented software methodology 2. It has 3 phases i.e. (a) Analysis phase (b) Architectural phase (c) Detailed design phase
6	**Prometheus**	1. It is a general purpose and practical oriented agent-based methodology. 2. It gives strong		The prometheus methodology is a practical methodology that covers the overall development process

		emphasis to the determination of system's goals and functionalities 3. It is iterative in nature 4. It is tool-supported i.e. Prometheus Design Tool (PDT) and Jack Development Environment (JDE).		in details
7	**Tropos**	1. It is a requirement driven methodology. 2. The methodology supports a full tool that spans four phases.	1. The methodology is not yet well suited and crafted for real the world application, 2. It is still being observed. 3. The methodology is not yet matured for the specification and design of agent systems.	1. It is another well-known and deeply elaborate multi-agent methodology. The methodology has five main development phases, namely, early requirements, late requirements, architectural design, detailed design and implementation phase.
8	**MASim**	1. It is an appropriate agent methodology for practical development 2. It covers the design, development and verification phases 3. It can be used to design both small and large systems.	1. It is particularly focused on agent-based simulations	This methodology is responsible for testing (verification and validation) the application according to the pre-established requirements.
9	**Agentology**	1. It is inspired by the best practices		The agentology model is one

		and good ideas contained in other agent oriented methodologies. 2. It is a methodology for the development of agent-based systems that can be easily applicable from requirements to development stage 3. The methodology is independent of any agent paradigm, framework, technology and programming language. 4. The process has an iterative approach.		methodology that can be easily adopted or adapted for the development any multi-agent based software prototype.

4. CONCLUSION

This study reviewed the following multi-agent based models: Agent Unified Modeling Language (AUML), MASE, High-level and Intermediate Models (HIM), CoMoMAS, GAIA, MASim, Prometheus, Tropos and Agentology. The study shows their various strength and weaknesses in terms of modeling and developing multi-agent based systems. Methodologies like CoMoMAS, GAIA, Prometheus, Tropos and MASim strongly supports multi-agent based systems design and development, while MASE methodology focuses on the development of multi-agent system for mainly robotics. Findings shows that Agentology methodology is one of the best methodologies crafted for multi-agent based system due to the fact that it was inspired by the best practices and good ideas contained in other agent oriented methodologies. The agentology methodology is therefore recommended for the development any multi-agent based software prototype.

REFERENCES

[1] Weiss (1999) Multi-Agent Systems, MIT Press

[2]Wooldridge, M. (2002) An Introduction to multi-agent systems. John Wiley & Sons Limited England, ISBN: 0-471-49691-X

[3] Nilsson, N. J. (1984) Shakey the robot. Technical Note 323, SRI International, Menlo Park, California.

[4] Weizenbaum, J. (1965) ELIZA – a computer program for the study of natural language communication between man and machine. Communications of the Association for Computing Machinery. 9(1), pp. 36–45.

[5] Winograd, T. (1973) A procedural model of language understanding. In Computer Models of Thought and Language, Schank, R. and Colby, K. Eds. W.H.Freeman, New York, pp. 152–186.

[6] Minsky, M. (1987) The Society of Mind. Heinemann, London.

[7] Jennings, N. R., and Wooldridge, M. J. (1998) Application of Intelligent Agent: Foundations, Applications and Markets, pp. 3-28, Secaucus, NJ, Springer-Verlag, Berlin.

[8] Lesser, V. R. (1995). Multiagent Systems: An Emerging Subdiscipline of AI, ACM Computing Surveys, 27(3):340-342.

[9] Agre, P. E. and S. J. Rosenschein (eds.) (1996). Computational Theories of Interaction and Agency , Boston, Massachusetts: MIT Press.

[10] Odell, J., Van Dyke Parunak, H. and Bauer, B. (2000) Extending UML for Agents. In Proceedings of Agent Oriented Software Engineering Workshop

[11] DeLoach, S. A., Wood, M. F., and Sparkman, C. H. (2001) Multiagent Systems Engineering. International Journal of Software Engineering and Knowledge Engineering, Vol. 11(3), pp. 231-258

[12] Garcia-Ojeda, J. C., DeLoach, S. A., Oyenan, W., and Valenzuela, J. (2007) O-MaSE: A Customizable Approach to Developing Multiagent system Development Processes. Proceedings of the 8th International Workshop on Agent Oriented Software Engineering

[13] Elammari, M. and Lalonde, W. (1999) An Agent Oriented Methodology: High-Level and Intermediate Models. In Proceedings of AOIS. Heidelberg, Germany.

[14] Glaser, N. (1997) Multi-Agent System Methodologies and Applications. In vol. 1286/1997. Lecture Notes in Computer Science. Springer Berlin/Heidelberg

[15] Schreiber, G. et al (1994) CommonKADS: A Comprehensive Methodology for KBS Development. In IEEE Expert 9.6, pp. 28-37

[16] Zambonelli, F., Jennings, N., and Wooldridge, M. (2003) Developing Multi-agent Systems: the Gaia Methodology, ACM Transactions on Software Engineering and Methodology, Vol. 12, No. 3

[17] Wooldridge, M., Jennings, N. R., and Kinny, D. (2000) The Gaia Methodology for Agent-Oriented Analysis and Design. In Autonomous Agents and Multi-Agents Systems 3.3, pp. 285-312

[18] Campos, Andre, M. C. et al, (2004) MASim: A Methodology for the Development of Agent-based Simulations, In Proceedings 16th European Simulation Symposium. Ed. By Gyorgy lipovzki and Istvan Molnar.

[19] Padgham, L., and Winikoff, M. (2004) Developing Intelligent Agent Systems: A Practical Guide. John Wiley and Sons

[20] Padgham, L., Thangarajah, J., and Winikoff, M. (2008) The Prometheus Design Tool – A Conference Management System Case Study. In Proceedings of 8th International Workshop on Agent Oriented Software Engineering, LNCS 4951 Springer.

[21] Giorgini, P., Bresciani, P., Giunchiglia, F., Mylopoulos, J., and Perini, A. (2004) Tropos: An Agent-Oriented Software Development Methodology, Autonomous Agents and Multi-Agent Systems, Vol. 18, 203-236.

[22] Morandini, M., Nguyen, D. C., Perini, A., and Susi, A. (2008) Tool-Supported Development with Tropos. The Conference Management System Case Study. In the Proceedings of 8th International Workshop on Agent Oriented Software Engineering (AOSE 07). Springer.

[23] Gomez-Sanz, J., and Luck, M. (2008) Proceedings of the 9th International Workshop on Agent-Oriented Software Engineering (AOSE 08)