ENGINEERING MOBILE AGENTS

Arnon Sturm
Department of Information Systems Engineering, Ben Gurion University of the Negev, Beer Sheva, 84105, Israel

Dov Dori
Faculty of Industrial Engineering and Management, Technion – Israel Institute of Technology, Haifa, 32000, Israel

Onn Shehory
IBM Haifa Research Lab, Haifa University Campus, Haifa, 31905, Israel

Keywords: Agent-based system, Agent-Oriented Software Engineering, Mobility.

Abstract: As the mobile agent paradigm becomes of interest to many researchers and industries, it is essential to introduce an engineering approach for designing such systems. Recent studies on agent-oriented modeling languages have recognized the need for modeling mobility aspects such as why a mobile agent moves, where the agent moves to, when it moves and how it reaches its target. These studies extend existing languages to support the modeling of agent mobility. However, these fall short in addressing some modeling needs. They lack in their expressiveness: some of them ignore the notion of location (i.e., the "where") while others do not handle all types of mobility (the "how"). Additionally, they lack in their accessibility, as the handling of the mobility aspects is separated into multiple views and occasionally the mobility aspect is tightly coupled with the functional behavior specification. View multiplicity reduces the comprehensibility and the ease of specification, whereas the coupling with the functional behavior specification reduces the flexibility of deploying a multi-agent systems in different configurations (i.e., without mobility). In this paper, we address these problems by enhancing an expressive and accessible modeling language with capabilities for specifying mobile agents. We provide the details of the extension, then illustrate the use of the extended modeling language, and demonstrate the way in which it overcomes existing problems.

1 INTRODUCTION

Agent mobility has been intensively studied in recent years. Although mobility is not a primary property of an agent, it may enhance agent autonomy (which is definitely a primary attribute of agents (Wooldridge et al., 2000)). Gray et al. (2001) identified several advantages of using mobile agents: conservation of bandwidth, reduction in total processing time, reduced latency, connected operation, mobile computing, load balancing, and dynamic deployment of software. Due to these advantages, mobile agents are sometimes the most suitable paradigm for agent-based systems engineering (Cabri et al., 2001). This paradigm has been demonstrated beneficial in several application areas, e.g., distributed information retrieval, workflow management, and network management (Bellavista et al., 1999; Brewington et al., 1999; Gray et al., 2001).

Because of the increased industrial and research interest in mobile agents, agent-oriented methodologies should support the mobility aspect. In particular, it is necessary to integrate agent mobility into the modeling languages of these methodologies, to provide a comprehensive engineering approach for building agent-based systems. This will help designers of agent-based systems to engineer mobile agents.

OMG (2000) and FIPA (2001; 2003) carried out many activities in order to standardize agent mobility aspects. Those activities led to an agreement on the required infrastructure for agent mobility in terms of general mobility concepts (such as places and regions) within a multi-agent system

Sturm A., Dori D. and Shehory O. (2008).
ENGINEERING MOBILE AGENTS.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 79-84
DOI: 10.5220/0001680300790084

(MAS) and the required functionality of mobile agents (such as agent migration, agent cloning, and agent invocation). To support the standards (to be), it is necessary that agent-oriented modeling languages facilitate the specification of these agent mobility concepts. In this paper, we enhance an existing agent-oriented modeling language with the capabilities of specifying the mobility aspects within an agent-based system. Note that our study is not first in addressing this need. Some of the agent-oriented modeling languages recognized the need for modeling mobility aspects such as why a mobile agent moves, where the agent moves to, when it moves and how it reaches its target (Mouratidis et al., 2002).

In this paper, we address the problem of integrating the mobility aspects of agent-based systems into a modeling language by introducing two key elements that should be supported by a modeling language with respect to mobility: the definition of places and environments and the definition of agent mobility (i.e., migration, cloning and invocation) (FIPA, 2003). We our approach leverage on an exiting method – Object-Process Methodology (OPM) and its MAS extension and suggest a modeling language for specifying mobility aspects of agent-based system. The choice of OPM for modeling agent-based system is discussed in Sturm et al. (2003), in which the authors present the motivation for adopting OPM. They mainly discuss the advantages of OPM with respect to accessibility and expressiveness. We found it useful as well since it provides a single unified model for capturing the various system aspects and enables to view the system as a whole, as we believe, required for modeling mobile agents.

The contribution of this paper is the introduction of a comprehensive modeling language for specifying multi-agent systems including the mobility.

The rest of this paper is organized as follows. Section 2 discusses related work. The Object-Process Methodology for Multi Agent Systems (OPM/MAS) is shortly described in Section 3. Section 4 introduces the enhancements we propose to support the modeling of the mobility aspects of MAS, and Section 5 concludes with a discussion on the advantages of the OPM/MAS approach for specifying mobile agents and with future research directions.

2 RELATED WORK

In the last decade many studies address the notion of modeling mobile MAS. The Multi-agent Software Engineering (MaSE) which is a general-purpose methodology for developing heterogeneous MASs (DeLoach et al., 2001) was extended to enable the specification of mobile agents (Self and DeLoach, 2003). GAIA which is a methodology for agent-oriented analysis and design (Wooldridge et al., 2000) has been enhanced by m-GAIA to model the mobility aspect (Sutandiyo et al., 2003). Several extensions for UML (OMG, 2007), which is the standard de-facto for modeling systems were suggested including Park et al. (2000), Klein et al. (2001), Mouratidis et al. (2002), Poggi et al. (2003), and Kang and Taguchi (2004). The various extensions mainly lacks in their expressiveness and their accessibility. With respect of expressiveness this are lacking in specify the locations related to the mobility operation and the integration of the mobility notion into the system functionality. The accessibility limitation is mainly stems from the model multiplicity problem discussed in Kabeli and Shoval (2001) and Peleg and Dori (2000). The problem is characterized by a lack of integration between the models and the need to maintain consistency across them and the need to gather information from various models for understanding the system specifications.

3 OPM/MAS IN A NUTSHELL

OPM/MAS is based on the Object-Process Methodology (Dori, 2002; OPM, 2007), which is an integrated approach to the study and development of systems in general and information systems in particular.

OPM is a general-purpose methodology, thus using its core symbol set may be too low-level for modeling a domain-specific application. This is so because the pertinent domain uses specialized concepts and building blocks that are at a higher level of abstraction than the basic OPM entities (objects, processes, and states).

OPM extension for MAS, as discussed by Sturm et al. (2003), follows principles of the Meta Object Facility (MOF) concept (OMG, 2002), which enables its flexibility. In that work the authors suggest a three layers architecture as follows: (1) a meta-model, which is the OPM itself; (2) an intermediate meta-model, which describes the specific building blocks within the MAS domain

using OPM; and (3) a model, which is based on both the meta-model and the intermediate meta-model.
That intermediate meta-model, which is the core element of OPM/MAS, was designed based on previous research. The designers of OPM/MAS divide the set of MAS building blocks into two groups. The first group consists of static, declarative building blocks, while the second group consists of building blocks with behavioral, dynamic nature.

In figures 1 and 2 the intermediate meta-model for multi-agent systems using OPM is presented. It also includes mobility-based concepts which were not discussed in Sturm et al. (2003). This intermediate meta-model provides guidelines and constraints for modeling multi-agent systems, including mobile agent systems.

OPM extension for MAS, as discussed by Sturm et al. (2003), follows principles of the Meta Object Facility (MOF) concept (OMG, 2002), which enables its flexibility. In that work the authors suggest a three layers architecture as follows: (1) a meta-model, which is the OPM itself; (2) an intermediate meta-model, which describes the specific building blocks within the MAS domain using OPM; and (3) a model, which is based on both the meta-model and the intermediate meta-model.

That intermediate meta-model, which is the core element of OPM/MAS, was designed based on previous research. The designers of OPM/MAS divide the set of MAS building blocks into two groups. The first group consists of static, declarative building blocks, while the second group consists of building blocks with behavioral, dynamic nature.

In figures 1 and 2 the intermediate meta-model for multi-agent systems using OPM is presented. It also includes mobility-based concepts which were not discussed in Sturm et al. (2003). This intermediate meta-model provides guidelines and constraints for modeling multi-agent systems, including mobile agent systems.

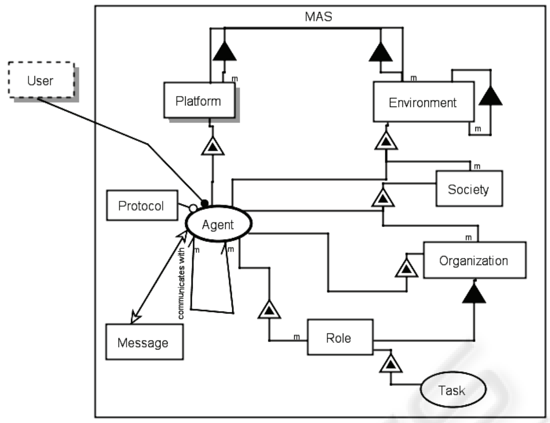

Figure 1: System Diagram of the OPM/MAS meta-model.

4 DESIGNING MOBILE AGENTS USING OPM/MAS

In this section we present the OPM/MAS enhancements to support agent mobility specification. The first sub-section introduces the new building blocks, whereas the second sub-section demonstrates the use of the intermediate metamodel for specifying agent mobility within the context of MAS.

4.1 Mobility in OPM/MAS

In this section we introduce the proposed changes to OPM/MAS in order to make it suitable for specifying agent mobility. As stated in Sturm et al. (2003), the OPM/MAS meta-model can be changed according to application needs. Following that principle, we propose an enhancement to the suggested set of building blocks.

The new the building blocks are the following:

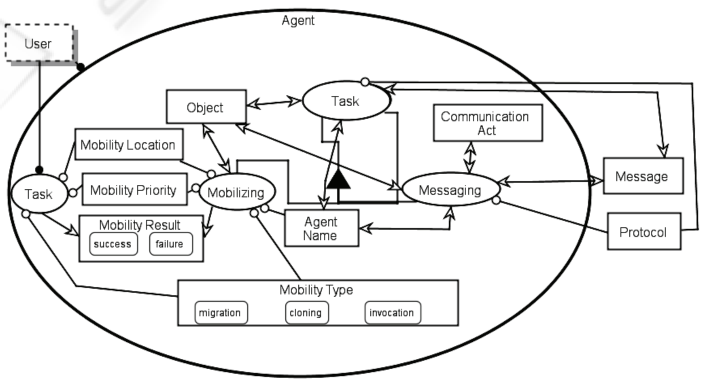

Figure 2: OPM/MAS meta-model - Agent in-zoomed.

- Mobilizing: A set of methods by which an agent performs a mobility activity such as migration, cloning, and agent invocation.
- Type: The mobility type - migration, cloning, or invocation.
- Location: The location to which the agent should migrate.
- Priority: The importance level of the mobility. This parameter is defined in order to define priorities in case of several possibilities for mobility within an agent.
- Result: The indication of the mobilizing process result.

The enhancement of the set of building blocks proposed in this paper refers to the system structure and to the system flow. The system structure is separated into the logical architecture, which is represented by the environment building block and the physical architecture, which is represented by the platform building block. Further, the relationships between the architectures are also specified. Since we are dealing with structural aspects of the system, the above building blocks were chosen to be OPM objects. The system flow enhancement consists of adding the mobilizing process and its parameters. Since we view mobility as part of the agent functionality (as in other studies, e.g., Self and DeLoach (2003) and Baumeister et al. (2003)), we define it as an OPM process in order to integrate the agent mobility into the regular flow of the agent process and yet differentiate it from regular tasks by its role (i.e., mobilizing). However, in contrast to existing languages the mobility specification can be easily removed by filtering out all mobility building blocks mentioned above, allowing the specification of a system without introducing mobility constraints.

4.2 The Technical Report Searcher Case Study

In the following, we present an example of using OPM/MAS for modeling mobile agents related to distributed information retrieval via the technical report (TR) searcher case study (Gray et al., 2001). In this case study technical reports are distributed across several machines, each executing a stationary information retrieval agent. When these agents are executed, they register with a virtual yellow pages agent. In search for technical reports, a client agent queries the yellow pages agent for the location of the stationary information retrieval agents. It then sends its "child" agents to these locations. These child agents interact with the stationary agents, which in turn reply to the child agents with the search results. In addition, the client agent checks for the network quality and determines whether it requires migrating to a proxy site in order to communicate with the machines hosting the stationary agents.

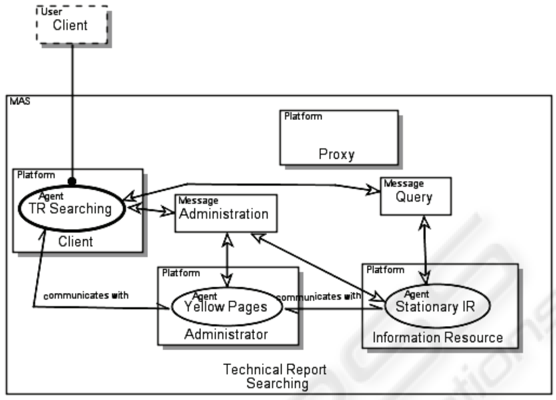

Figure 3: Technical Report Searching System – System Diagram.

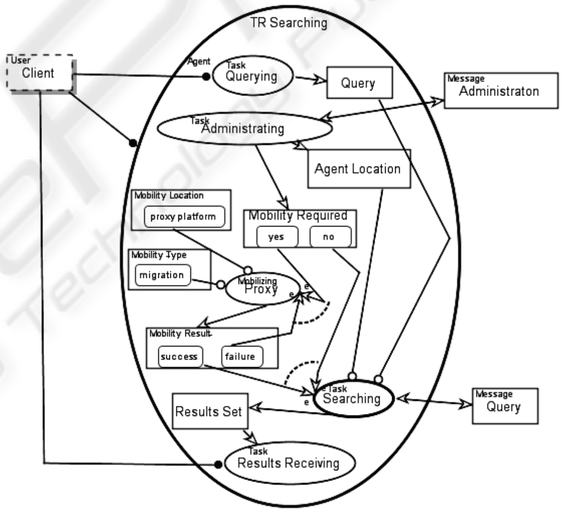

Figure 4: Technical Report Searching System – TR Searching Agent in-zoomed.

Figure 3 presents the system diagram of the Technical Report Searcher system. It consists of four platform types: **Client** Platform, which hosts the **TR Searching** Agent; **Administrator** Platform, which hosts the **Yellow Pages** Agent; the **Information Resource** Platform, which hosts the **Stationary IR** (Information Retrieval) Agents; and the **Proxy** Platform, which is capable of hosting the **TR Searching** Agent in case of faulty communication between the **Client** Platform and the **Information Resource** Platform. The OPD in Figure 3 also describes the communication paths (which are the logical routes among the agents) and messages.

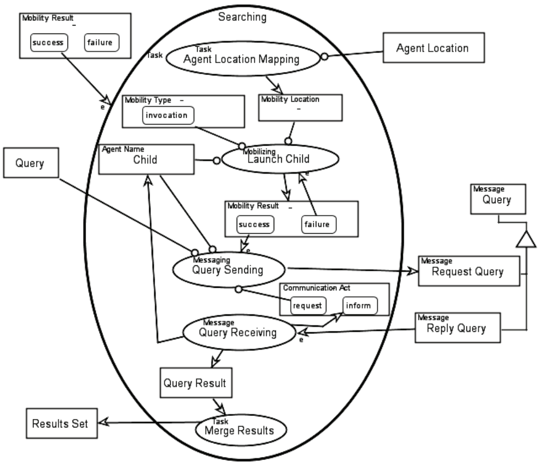

Figure 5: Technical Report Searching System – Searching Task in-zoomed.

In Figure 4, the **TR Searching** Agent is in-zoomed. The agent is activated by the **Client** User, as shown by the agent link from the object **Client** User (a human) to the process **TR Searching** Agent. The **TR Searching** Agent performs its tasks sequentially, as determined by the vertical order within the OPD. The **Querying** Task accepts the user's input and yields a **Query** object. The **Administrating** Task follows the **Querying** Task and yields an object indicating whether mobility is required and an object representing the required Agent Location. If mobility is required, the **Proxy** Mobilizing process occurs and causes the **TR Searching** Agent to migrate to the **Proxy** Platform. If the **Proxy** Mobilizing process fails, then it is triggered again. This is indicated by the "e", for "event", attached to the arrowhead of the link connecting the failure state within the Mobility Result object and the **Proxy** Mobilizing process. When the mobility result is a success (or if it was not required in the first place), the **Searching** Task is performed, followed by the **Results Receiving** Task.

In Figure 5, the **Searching** Task from the OPD of Figure 4 is specified by zoom into its specification. It begins with the **Agent Location Mapping** Task, which determines Mobility Location, in which the **Child** Agent has to be invoked. This task is followed by launching the **Child** Agent in the appropriate platform. In case the invocation fails, the agent tries to re-launch the **Child** Agent until it succeeds. When the **Child** Agent is running, the **TR Searching** Agent sends a **Request Query** Message and waits for a reply. Upon receiving the query results, **Reply Query** Message, the results received from its child agents are merged to obtain the **Results Set**, which the **Client** User ultimately receives, as Figure 4 shows.

5 CONCLUSIONS

In this paper we leverage on the object-process methodology to facilitate the modeling of agent mobility. We show how the OPM/MAS intermediate meta-model can be enhanced in order to support agent mobility. Following that enhancement, we demonstrate the use of that intermediate meta-model to specify a MAS application. In particular, we exemplify the way according to which the OPM/MAS addresses the four questions of mobility:

1. Why a mobile agent performs a mobility action?
   In OPM/MAS the reason of the agent mobility is encapsulated within the task flow. This means that the agent mobility is determined according to the task flow and decisions that are made during its process. The mobility is represented as an OPM process (Mobilizing) thus easily integrated within the task flow. In the TR case study, the Client agent moves in order to improve its communication, where as a Child agent is invoked in order to search information in another location.
2. When the agent performs a mobility action?
   The timing in which the agent moves is specified in OPM/MAS via the process sequence. It may move due to other task termination, it may move due to new information, or it may move due to a user request. In the TR case study, the Client agent moves upon determining Administrating problems.
3. Where the agent moves to?
   The destination of the agent in OPM/MAS is determined by the mobilizing process parameter – Location. The locations within a system according to OPM/MAS could be platforms (could be referred to as places) or environments (could be referred to as regions). These are usually specified within the top level OPDs.
4. How the agent reaches its target?
   The path according to which the agent reaches its target is specified by the order of the mobilizing processes. In TR case study, the path is a straight forward way, from the Client platform to the Proxy platform.

The weaknesses of the existing agent-oriented methods with regards to mobility are addressed within the proposed solution. We refer to the location notion with its various level of abstraction by providing the environment building block and its relationships with the platform building block. The proposed solution refers to all mobility types by defining the mobility type object. We integrate all of the mobility aspects within a single-unified framework, whereas in the other methods the integration of some of the mobility aspects is not clear, difficult to understand, or non-exist.

Further research is required to examine the accessibility and adherence of the OPM/MAS approach to build agent-based systems and in particular, mobile agents.

ACKNOWLEDGEMENTS

This work was partially funded by Deutsche Telekom Laboratories at Ben-Gurion University.

REFERENCES

Bellavista, P., Corradi, A., and Stefanelli, C., 1999. An Open Secure Mobile Agent Framework for Systems Management. J*ournal of Network and Systems Management (JNSM), Special Issue on Mobile Agent-based Network and Service Management 7(3)*, 323-339.

Brewington, B., Gray, R., Moizumi, K., Kotz, D., Cybenko, G., and Rus, D., 1999. Mobile Agents for Distributed Information Retrieval. *In Intelligent Information Agents, chapter 15*.

Cabri, G., Leonardi, L., Mamei, M., and Zambonelli, F., 2001. Mobile Agent Organizations. *In WOA* 2001.

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

DeLoach, S. A. and Wood, M., 2001. Developing Multiagent Systems with agentTool. *In Proceedings of ATAL 2000, LNCS 1986.*

Dori, D., 2002. *Object-Process Methodology - A Holistic Systems Paradigm*, Springer, Heidelberg.

FIPA, 2001. *FIPA Agent Management Support for Mobility Specification*. http://www.fipa.org/specs/fipa00087/PC00087B.pdf.

FIPA, 2003. *FIPA Modeling Area: Deployment and Mobility*. http://www.auml.org/auml/documents/DeploymentMobility.zip.

Gray, R.S., Cybenko, G., Kotz, D., and Rus, D., 2001. Mobile Agents: Motivations and State of the Art. *Handbook of Agent Technology*.

Kabeli, J., and Shoval, P., 2001. FOOM: functional- and object-oriented analysis & design of information systems - an integrated methodology. *Journal of Database Management, 12 (1)*, 15-25.

Kang, M. and Taguchi, K., (2004) Modelling Mobile Agent Applications by Extended UML Activity Diagram. *ICEIS 2004*, 519-522.

Klein, C., Rausch, A., Shiling, M., and Wen, Z., 2001. Extension of the Unified Modeling Language for Mobile Agents. *In The Unified Modeling Language: Systems Analysis, Design and Development Issues,* 1-25, Idea Group Publishing.

Mouratidis, H., Odell, J., and Manson, G., 2002. Extending the Unified Modeling Language to Model Mobile Agents. *In Proceedings of the Agent Oriented Methodologies Workshop (at the OOPSLA 2002)*.

OMG, 2000. Mobile Agent Facility Specification. http://www.omg.org/docs/formal/00-01-02.pdf.

OMG, 2002. Meta Object Facility (MOF) v1.4. http://www.omg.org/technology/documents/formal/mof.htm.

OMG, 2007. Unified Modeling Language (UML) 2.0, http://www.omg.org/technology/documents/formal/uml.htm,.

OPM, 2007. http://www.objectprocess.org.

Park, S., Kim, J., and Lee, S., 2000. Agent-Oriented Software Modeling with UML Approach. *IEICE Transaction on Information & Systems, E83-D (8)*, 1631-1641.

Peleg, M. and Dori, D., 2000. The Model Multiplicity Problem: Experimenting with Real-Time Specification Methods. *IEEE Transaction on Software Engineering, 26(8)*, 742-759.

Poggi, A., Rimassa, G. , Turci, P., Odell, J., Mouratidis, H., and Manson, G., 2003. Modeling Deployment and Mobility Issues in Multiagent Systems Using AUML *In Proceeding of Agent-Oriented Software Engineering (AOSE)*, 69-84.

Self, A. and DeLoach, S. A., 2003. Designing and Specifying Mobility within the Multiagent Systems Engineering Methodology. *Special Track on Agents, Interactions, In Mobility, and Systems (AIMS) at The 18th ACM Symposium on Applied Computing*.

Sturm, A., Dori, D., and Shehory, O., 2003. Single-Model Method for Specifying Multi-Agent Systems. *In Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, 121-128.

Sutandiyo, W., Chhetri, M. B., Krishnaswamy, S., and Loke, S. W., 2003. From m-GAIA to Grasshopper: Engineering Mobile Agent Applications. In Proceedings of iiWAS2003.

Wooldridge, M., Jennings, N. R., and Kinny, D., 2000. The Gaia Methodology for Agent-Oriented Analysis and Design. *Journal of Autonomous Agents and Multi-Agent Systems 3 (3*), 285-312.