MGAIA: EXTENDING THE GAIA METHODOLOGY TO MODEL MOBILE AGENT SYSTEMS

Weanna Sutandiyo, Mohan Baruwal Chhetri, Seng Wai Loke and Shonali Krishnaswamy
School of Computer Science and Software Engineering, Monash University, 900 Dandenong Rd, Caulfield East, VIC 3145, Australia

Keywords: Mobile Agent Systems, Agent Oriented Software Engineering

Abstract: Mobile agents are a class of software agents that have the ability to move from host to host and are particularly relevant for mobile and distributed applications. The development of several mobile agent implementation environments has necessitated conceptual modelling techniques for mobile agent applications. In this paper, we present mGaia, our extension of the Gaia Agent Oriented Software Engineering (AOSE) methodology to model mobile agent systems.

1 INTRODUCTION

Mobile agents are a class of software agents that have the ability to move and are particularly useful in the area of distributed and mobile computing (Kotz, 2002). Even though a sizable number of mobile agent toolkits have been developed to support the implementation and deployment of mobile multiagent applications, the conceptual modelling of mobile agent applications is an area that has not been largely addressed (Krishnaswamy, 2003). The increasing focus on mobile agents as an important technology for developing mobile and distributed applications and the inherent disadvantages of confining the mobility of agents to implementation alone necessitates techniques for modelling such systems prior to embarking on the implementation. Agent Oriented Software Engineering (AOSE) is defined as software engineering for agent based computing (Weiß, 2002). AOSE methodologies aim to provide tools and techniques for modelling, analysing and designing agent systems prior to implementation. Several methodologies have been proposed to model multiagent systems (MAS). However, the focus of these methodologies has been on multiagent systems at a generic level and they do not address the specific modelling issues that pertain to mobile agent systems. We have developed a conceptual modelling methodology for mobile agent systems, mGaia, which is an extension of the Gaia Agent Oriented Software Engineering (AOSE) methodology (Woolridge, 2000). The paper is organised as follows. Section 2 presents mGaia. Section 3 presents an application that was modelled with mGaia and then implemented using Grasshopper. Section 4 concludes the paper. We recommend that extending an existing methodology to model each required property in an agent system is preferred, instead of developing a methodology from scratch for each new property to be considered.

2 mGAIA

We present mGaia as an enhancement of Gaia to facilitate conceptual modelling of mobile agent systems. In order to support conceptual modelling of mobile multiagent systems, mGaia incorporates the existing models of Gaia and adds a new model, namely, the *mobility model*. Figure 1 shows the structure of mGaia's models. The basic ideas of mGaia are borrowed from the existing Gaia methodology. As such, mGaia still consists of the *analysis* and *design phases*. The objective of the analysis phase is to obtain an understanding of the system and its structure. It consists of the *roles model,* which identifies the roles in the system and the *interaction model*, which identifies the interactions between the roles found.
There are four attributes of roles:

Sutandiyo W., Baruwal Chhetri M., Wai Loke S. and Krishnaswamy S. (2004).
MGAIA: EXTENDING THE GAIA METHODOLOGY TO MODEL MOBILE AGENT SYSTEMS.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 515-518
DOI: 10.5220/0002627205150518

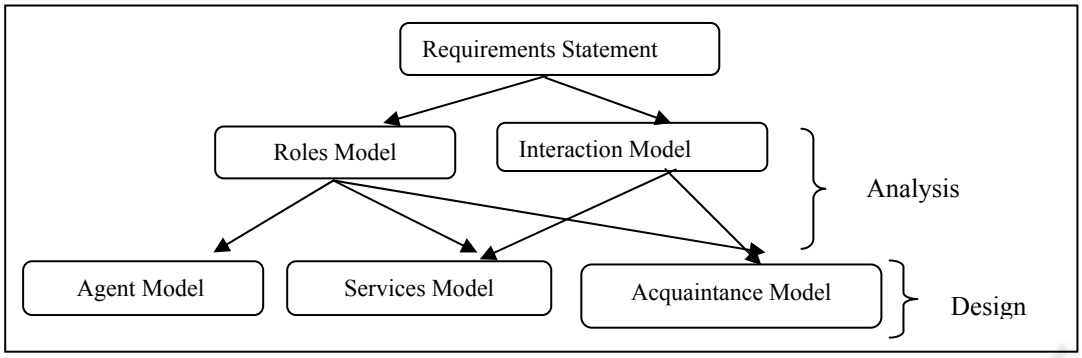

Figure 1: Structure of mGaia's

- Responsibilities that consist of the liveness property, which defines the continual execution of the role within the system, and the safety property, which is a condition that must be maintained to avoid system behaviour that is contrary to its system requirements,
- Permissions, which define the access privileges or rights of roles,
- Activities, which are tasks that need to be performed by roles without interaction with other roles, and
- Protocols, which are activities that involve interactions with other roles.

The objectives of the design phase are to convert the system from an abstract level to a concrete level and to ease implementation. The design phase consists of the *agent model*, the *services model*, and the *acquaintance model* and the new *mobility model*.

- The agent model of mGaia is used to identify the agent types and how many agents are involved within the entire system. It includes constructs to distinguish between agents that possess the characteristic of mobility and those that do not.
- The services model of mGaia is the list of services that each role can provide and be associated with.
- The acquaintance model defines the communication links between each agent.

Abstract Concepts	Concrete Concepts
Roles	Agent types
Permissions	Services
Responsibilities	Acquaintances
Protocols	*Place Types*
Activities	*Atomic movement*
Liveness Properties	*Travel paths*
Safety Properties	

Table 1: Abstract and Concrete Concepts in mGaia

- The mobility model of mGaia defines the mobility characteristics of agents further, such as identifying the movements and travel path of each mobile agent.

Like Gaia, mGaia has abstract concepts and concrete concepts. The abstract concepts are used during the analysis process and they do not necessarily have direct correlations in the run-time system. The concrete concepts are considered during the design process. The concrete concepts have direct correlations in implementation of the run-time system. Table 1 summarises the abstract and concrete concepts of mGaia. It must be noted that in Table 1, the italicised concepts are unique to mGaia and mainly aim to support modelling agent mobility in multiagent system. In summary, the additional features of mGaia involve modifications to the following Gaia's models: role model, agent model and the introduction of a new mobility model. We now present analysis and design phases of mGaia.

2.1 Analysis Phase of mGaia

In mGaia, modifications have been made to roles model so that each role identified is categorized into three different role types - *system*, *interface*, and *user* roles. The purpose of categorising roles is to clarify each role's responsibilities within the system. A system role is defined as a role that interacts with other parts of the system and not the user. An interface role is a role that interacts with the user and the other parts of the system. A user role is a role that represents the human user itself. Despite the modification in the roles model, the remaining components are the same as in Gaia.

2.2 Design Phase of mGaia

In mGaia we have modified Gaia's agent model to specify the mobility characteristic of agents. We have also introduced the new mobility model. The following section will discuss the modified agent model and the mobility model.

2.2.1 The Agent Model

The agent model identifies the number of agents, the agent types, and the relationship between the roles identified (in the role model) and the agent types in the system.

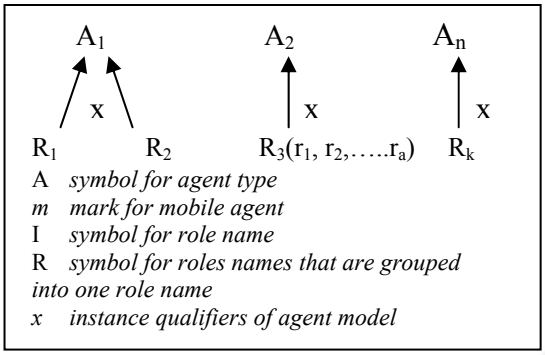

Figure 2: Agent model of mGaia

mGaia's agent model classifies the agents into two different categories - mobile (by adding a notation of "*m*" sign) and stationary. The categorisation of agent types caters for mobility characteristic of agents. Furthermore, we modify the agent model to allow similar behaviour roles to be grouped into one category. This is notational illustrated by grouping the role names between parentheses as shown in Figure 2. This modification is for convenience of presentation.

2.2.2 Mobility Model

The mobility model enhances Gaia to incorporate support for modelling of mobile agents in multiagent systems. While the analysis phase of mGaia involves identifying the roles and the interactions of each role, the design phase of mGaia involves agents. Therefore, the mobility model is best fitted into the design phase rather than in the analysis phase, as mobility is a characteristic of agents and not roles. Furthermore, mobility is not an interaction, as an agent does not need to be mobile to communicate. These considerations motivated the inclusion of the mobility model in the design phase. The mobility model is derived from the agent model. In the agent model, the agent types are categorised into mobile and stationary. In order to model the mobility characteristics of mobile agents, the mobility model identifies *place types*. There are four steps in constructing the mobility model:

Step 1: Place Types

Identify the place types, which are locations that the mobile agent can visit or reside in. Table 2 shows the place types in the mobility model of mGaia. Table 3 identifies the instance operators of place types.

Table 2. Place Types

Place Types	Description	Instances
P1 **P2** **Pn**	***Short English description of place types***	***Instance operators indicates how many place types exist in the system***

Table 3. Cardinality Operators in mGaia

Operator	Description
n	Exactly n instances
m……n	Between m and n instances
*	0 or more instances
+	1 or more instances

Step 2: Agents and Places Specifications

Step 2 of the mobility model is derived from step 1 and the agent model. In this step, we identify the relationship between agent types and place types. It also defines the constraints of the relationship. The agents and places specifications are derived from the place types identified in step 1 of mobility model. Table 4 shows the agent and place specifications for the mobility model.

Table 4. Agent and Place Specifications

Agent Types	Mobile	Place Types	Constraints
A1 A2 An	A tick sign to identify if the specific agent is mobile or not	P1, P2 P3 Pn	The constraints of agents and place types relationship

Step 3: Cardinality of Agents and Places

The cardinality between agent types and place types shows how many agents of an agent type can reside in a place of a place type. The cardinality of agents and places (step 3) is based on the agents and places specifications (step 2).

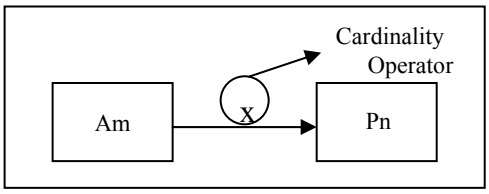

Figure 3. Cardinality of Agents and Places

Step4: Travel Schema of Mobile Agent Types

The travel schema of each mobile agent type includes *origin*, *final destination*, *list of atomic movements*, and *paths*. The origin is the place type where the mobile agent starts the movement to accomplish the tasks assigned. The final destination is the place type where mobile agent will reside after it completed the tasks assigned. The atomic movement is the smallest granularity movement required to accomplish the tasks assigned. The paths are the list of atomic movements that the mobile agent may travel in order to accomplish the tasks assigned.

3 OVERVIEW OF THE SMART LECTURE THEATRE SYSTEM

The Smart Lecture Theatre system is a multiagent system, where a different type of agent manages each type of interaction. Within the Smart Lecture Theatre architecture, agents will be stationary and/or mobile. Each agent will have a task assigned and it will seek to perform and fulfil the task assigned. For example, a student needs to find out the contact details of lecturer A. This student will query from his/her user device. First the student is required to login into Smart Lecture Theatre system and a unique user agent handles this operation. Once the query has been triggered, the user agent creates a query agent automatically. The query agent with the assigned task (i.e. "What are the contact details of lecturer A?") will migrate to the lecture theatre and attempt to answer the student's query. The design of Smart Lecture Theatre requires the mobile agent to move from the user device to the lecture theatre to accomplish the task assigned. There are three types of users in Smart Lecture Theatre system namely student, lecturer and administrator. Each particular user has different types of services available. Student can query lecturer details, such as lecturer's URL address, lecturer's room number and email address, and query unit details, such as unit's name and unit's URL address. The administrator performs update of the lecture theatre features, such as disabled access, capacity, speakers, OH projectors and LAN connections and also the user details, such as username, real name, URL and user type. The Lecturer can list lecture theatres based on the features such as disabled access, capacity, speakers, OH projectors and LAN connection, query lecture theatres on available times and book the specified lecture theatre at a particular campus and negotiate with another user when the other user has booked the lecture theatre for a particular time slot. The analysis and design of the Smart Lecture Theatre system was done using mGaia and the implementation was performed using the Grasshopper mobile agent toolkit.

4 CONCLUSION AND FUTURE WORK

We have presented a conceptual modelling methodology for mobile agent systems. We presented our experiences in mapping the mGaia models to Grasshopper mobile agent toolkit (http://www.grasshopper.de) in (Sutandiyo *et al*., 2004), which showed mGaia to be effective. These experiences have indicated that there are several open issues that need to be addressed which are the focus of our current work. The key issues include identification of additional constructs for mobile agent systems, formalization of the constructs, specification of mobility of agent contexts/places and addressing the mobility of *roles* – or mobility in the analysis phase of the modeling.

REFERENCES

Kindberg, T. and Barton, J., *A Web-based Nomadic Computing System,* ACM: Elsevier North-Holland, Inc., pages 443-456, New York, 2001

Kotz, D., Gray, R., and Rus, D. *Future Directions for Mobile-Agent Research. IEEE Distributed Systems Online,* 3(8) August 2002. Based on a conversation with Jeff Bradshaw, Colin Harrison, Guenter Karjoth, Amy Murphy, Gian Pietro Picco, M. Ranganathan, Niranjan Suri and Christian Tschudin

Krishnaswamy, S., and Loke, S. W., *On Modelling Agent Mobility in Multiagent Methodologies,* Position paper, Workshop on Agent Oriented Information Systems (AOIS 2003) held in conjunction with the Second International Joint Conference on Autonomous Agents and Multiagent Systems, Melbourne, 2003.

Sutandiyo, W., Chhetri, M, B., Krishnaswamy, S., and Loke, S.W. *Experiences with Software Engineering of Mobile Agent Applications*. To appear in the 2004 Australian Software Engineering Conference.

Weiß, G. *Agent Orientation in Software Engineering,* Knowledge Engineering Review, Vol 16, No. 4, pages 349-373, 2002

Wooldridge, M., Jennings, N. R., and Kinny, D. *The Gaia Methodology for Agent-Oriented Analysis and Design,* Journal of Autonomous Agents and Multi-Agent Systems, Vol.3, No.3, pages 285-312, 2000.