Extending the Gaia Methodology with Agent-UML

Juan C. García-Ojeda,
José de J. Pérez-Alcázar
Laboratorio de Cómputo Especializado
Universidad Autónoma de Bucaramanga
Calle 48 No 39-234. Santander-Colombia
{jgarciao,jperez}@unab.edu.co

Alvaro E. Arenas
Business and Information Technology Department
CCLRC Rutherford Appleton Laboratory
Oxfordshire OX11 0QX, United Kingdom.
a.e.arenas@rl.ac.uk

Abstract

This paper describes how to integrate a design produced by Gaia with the AIPs layers of AUML, taking into consideration Bauer's extensions to UML Class diagrams in the context of agent-oriented development. We take as a case study the development of an agent-based system for conference management.

1. Introduction

The *agent* paradigm introduces a number of new abstractions and design/development issues when compared with more traditional approaches to software development. For that reason, successful industrial development of agent technology requires techniques that reduce the risk inherent in any new technology. Two ways that reduce risk in the eyes of potential adopters are: to present new technology as an incremental extension of known and trusted methods, and to provide explicit engineering tools that support industry-accepted methods for technology deployment[2].

This paper realizes the first way by extending Gaia [5], a methodology for developing multiagent systems, with the first recommended extension to UML for the agents case, called the Agent Interaction Protocol (AIP)[3].

2. The Gaia Methodology

Gaia is founded on the view of a multiagent system as a computational organization [5]. Gaia's authors suggest the application of object-oriented techniques to the generated design in order to obtain a more concrete one [5]. However, we have found that some agent-oriented properties from Gaia's models could be improved, and that such improvements are not possible by using traditional object-oriented techniques. This has motivated the integration of Gaia with AUML.

3. Putting All Together

The work presented here reformulates our previous work [1], taking into consideration the new proposal of Gaia [5], the extended UML class diagram for agents [2] and the extension of the Aalaadin Model proposed by Parunak in [4]. The relation among all these models is illustrated in Figure 1.

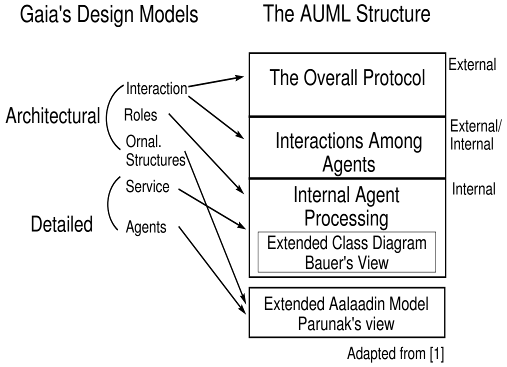

Figure 1. Combining Gaia and AUML

3.1. Towards a More Concrete Model with AUML

AIP specification allows us to model both internal and external aspects of a multiagent system. These characteristics bring us the possibility to integrate Gaia with AUML. A method to carry this out is to use a top-down approach, beginning from the more general models of the AIP specification to the most specific, as expressed in Figure 1.

3.1.1. Representing the Overall Protocol Taking as example the *ReceivePaper* protocol ([5], p 348), we have represented that protocol from the point of view of Gaia and AUML (see Figure 2).

AAMAS'04, July 19-23, 2004, New York, New York, USA.
Copyright 2004 ACM 1-58113-864-4/04/0007...$5.00

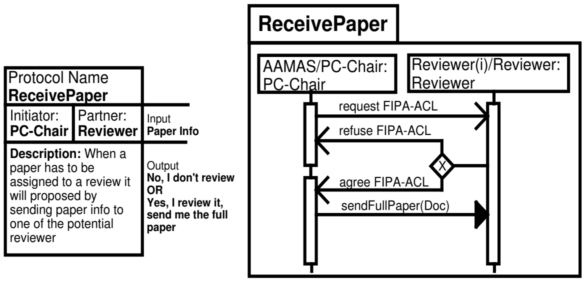

Figure 2. Protocol Definition in Gaia and AUML

3.1.2. Representing Interaction between Agents Other possibility to represent the interaction between agents is by using the second layer of AIP (see [1]).

3.1.3. Representing Internal Agent Processing Once the interactive part between roles and agents of the system is represented, we proceed to combine the model of roles with the model of services associated to the agent. In order to do that, we make use of the extended vision of Bauer's Class Diagrams.

We model each one of the main characteristics of a role. The extended vision of Bauer splits a conventional class diagram of UML in seven major sections. In the first section we associate a role to the agent class to which it belongs. The second section makes reference to the description of the state (like the liveness properties in Gaia). The third section defines the actions that an agent or role must take (i.e. reactive or proactive). In the fourth section the methods are defined. In the fifth section the capabilities of the role are specified (similar to protocols and activities in Gaia's roles schema). The sixth section defines the restrictions by which the role will be subdued (like the security properties of Gaia's roles schema). The last section is the most important of the extended class diagram proposed by Bauer, since it allows us to model each one of the communication acts in which the role or agent participates. An important aspect when modelling this section is the Agent-Head-Automata, which can be modelled with the the third layer of the AIP model. It is worth noticing that these activities or internal processing derive from Gaia's model of services.

Finally, we refine the agents and organizational structure model (see [5], p. 362) using the extended Aalaadin model proposed by Parunak. His proposal lies in representing an organizational model of agents expressed in UML class diagrams (see Figure 3).

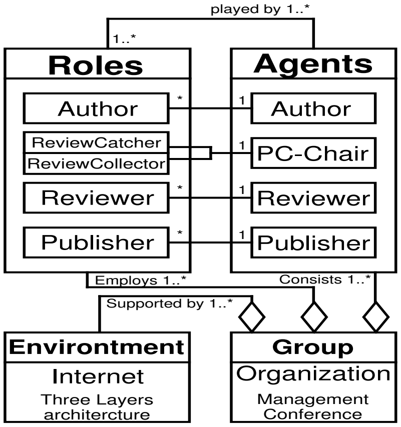

Figure 3. The Agent Model (extended)

4. Conclusions

We have applied the Agent Interaction Protocol (AIP) of AUML to the Gaia design models by following a top-down approach. The interaction model is refined in the first two layers of the AIP. Next, we refine the roles and service model by means of the integration of the layer three of AIP and the extend Class Diagrams proposed by Bauer. Finally, the agent and structural organizational models are refined with the extended Aalaadin model proposed by Parunak, resulting in a detailed description of the agent organization structure.

Acknowledgments

This work was partially supported by the Colombian Research Council (Colciencias) under Grant 1241-14-11080.

References

[1] A. E. Arenas, J. C. García-Ojeda, and J. de J. Pérez Alcázar. On combining organisational modelling and graphical languages for the development of multiagent systems. *Integrated Computer-Aided Engineering*, 11(2):151–163, Mar. 2004.

[2] B. Bauer. Uml class diagrams revisited in the context of agent-based systems. In *Agent-Oriented Software Engineering II (LNCS Volume 2222)*, pages 101–118. Springer-Verlag, 2002.

[3] J. Odell, V. D. Parunak, and B. Bauer. Extending uml for agents. In *AOIS Workshop at AAAI*, pages 3–17, 2000.

[4] H. V. D. Parunak and J. Odell. Representing social structures in uml. In *Agent-Oriented Software Engineering II (LNCS Volume 2222)*, pages 1–16. Springer-Verlag, 2002.

[5] F. Zambonelli, N. R. Jennings, and M. Wooldridge. Developing multiagent systems: The gaia methodology. *ACM Trans. Softw. Eng. Methodol.*, 12(3):317–370, Dec. 2003.

AAMAS'04, July 19-23, 2004, New York, New York, USA.