Agent Architecture based on Semantic Knowledge Model*

Michal Laclavik[1], Zoltan Balogh[1], Emil Gatial[1], Ladislav Hluchy[1]

[1] Institute of Informatics, Slovak Academy of Sciences, Dubravska cesta 9
845 07 Bratislava, Slovakia
laclavik.ui@savba.sk

Abstract. In this paper we describe a semantic knowledge model of an agent as well as implementation and use of such model using the Jena semantic library and the JADE agent system. The solution has been used and evaluated in the Pellucid and K-Wf Grid projects.

1 Introduction

Multi-Agent Systems or Agent Oriented Programming is a highly powerful method in distributed heterogeneous information systems requiring representation and reasoning with knowledge. At present, MAS lack interconnection with current commercial technological standards and the results of the semantic web research [6]. In the MAS knowledge is usually represented by states, rules or predicate logic. This is extremely powerful but it is hard to capture knowledge from a person or from current information systems in such a way, furthermore it is difficult to present knowledge information expressed in e.g. predicate logic to the end user. Ontology as understood in the Semantic Web is closer to current information systems because it is based on XML/RDF and enables an easier capture or return from/to a person or an existing information system.

For this reason we have decided to integrate semantic web results into MAS and create architecture, methodology and software for such integration.

Agent architectures are the fundamental engines underlying the autonomous components that support effective behavior in the real-world, dynamic and open environments. There are several basic types of agent architectures: Reactive Architecture; Belief Desire Intention Architecture – BDI; Behavioral Architecture. In literature, the main focus is on the externals of the agents, their communication with environment and other agents. The internal knowledge model is left for an agent creator. The most advanced, but not sufficient approach is in the JADE [1] agent system. JADE support for ontologies or agent knowledge modeling [2] is based on Java classes. The JADE agent model is not sufficient in several ways. Such model cannot support features of semantic ontology representations such as OWL, does not have a query engine for the

* This work is supported by the project K-Wf Grid, EU RTD IST FP6-511385, VEGA No. 2/3132/23, APVT-51-024604 and SPVV 1025/2004.

FIPA-SL language [3] and a model based on the predicate logic is hard to be communicated with a user and existing commercial systems. Thus we have created an RDF/OWL based model. Agent communication based on RDF/OWL was presented in other works such as [4], but this work lacks a generic internal model, it only offers a theory how RDF/OWL can be used for agent communication. In this work we considered parts of those two models and extended it with the event based memory model. The event based model was previously used in many areas but in the agent field only work of Anderson [5] deals with the event model as a typical reactive architecture. On the other hand in our model we use events to take action and to have a history of an action allowing us to have history of an environment in any moment of its history which can be preprocessed at any time and different results can be achieved with the same knowledge model.

2 Agent Knowledge Model

Our architecture is based on behavioral architecture, where an agent memory model based on OWL-DL is used. First we will describe agent memory formally by description logic (DL) compatible with OWL-DL and graph description and subsequently we will move forward to modeling and implementation using RDF/OWL. A model is based on events. The idea is taken partially from the JADE ontology model – predicates and concept. As we described, the JADE ontology model [2] has limitations.

Our model is based on five main elements: Resources, Actions, Actors, Contexts and Events. Figure 1 shows formal graph representation of a model. The proposed model is described using DL too[7].

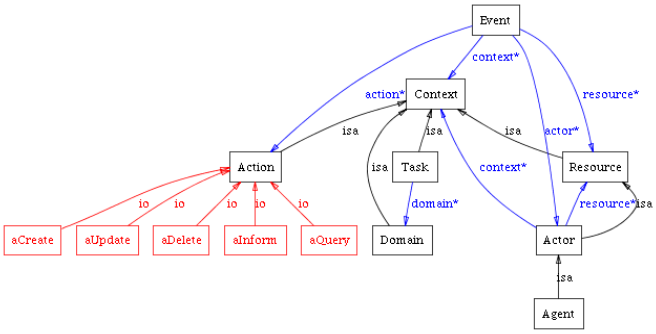

Fig. 1. Basic Ontology for Knowledge Modeling

Here we describe only the Actor class using DL, which consists of important properties: context.Actor, resource.Actor. The context.Actor represents actor current context. The system or application environment can be found based on stored events. Events model environment state. The current state of the environment or the actor related environment/context is thus affected by relevant new events. The resource.Actor property stands for all current resources of the actor. This resources are results of actors intentions or objectives and thus they are dependable on current actors environment state/context (context.Actor). Functions/algorithms for context and resource updating were specified above.

$$\text{Actor} \subseteq \text{resource.Actor}(\{\text{resource}\}) \cap \quad (1)$$

$$\text{context.Actor}(\{\text{actor}\}) \cap \text{Resource}$$

$$\text{contextActor}(\{\text{actor}\}) = f_C(\forall \{\text{event}\}; \text{actor.Event}(\{\text{actor}\}) \in \{\text{event}\}) \quad (2)$$

$$\text{resource.Actor}(\{\text{resource}\}) = f_R(\text{contextActor}(\{\text{actor}\})) \quad (3)$$

This is where the brain of the system is located. An advantage of such model is that it enables to achieve better results when such algorithms are changed in the future, using the same model and data. Due to storing all events we can model the environment in any moment from past and process it later from any starting point with improved algorithms for context and resource updating.

2.1 Modeling Methodology for Agent Design

The developed methodology integrates several parts of different methodologies and follows CommonKADS methodology. However CommonKADS is not tied with any modeling tool, knowledge representation or ontologies. We present knowledge based on the OWL ontology and we model it in the Protégé ontology editor. Thus defined methodology for knowledge model is similar to [8]. Ontology modeled with Protégé reflects CommonKADS models and it has several commonalities with the JADE ontology model. Design of the system is based on UML, AUML and MAScommonKADS. When using this methodology, good results can be archived only after several iterations of the process and remodeling after the first developing, use and evaluation of the first system version.

Modeling a knowledge model for an application we follow first three CommonKADS models: Organizational or in our case the Environment Model; the Task Model; the Agent or the Actor Model.

Modeling the knowledge model we have to extend the generic agent model (Figure 1) with new elements and relations.

Results of models is the knowledge model which consists of :

- Ontology developed in Protégé which can be exported in the OWL format.
- Algorithms for each actor (algorithms often are similar or same), which updates actors' context f_C (2) and resources f_R (3).

The design methodology of a system is based on three UML diagram types in a similar way as known in object oriented programming: Use Case Diagram; Sequence Diagram; Class Diagram.

2.2 Design and Specification of Agent Software Library

The developed library is based on the JADE agent system and the Jena library. It covers functionalities which we identified as missing in current agent architectures such as JADE, the Agent Knowledge Model based on RDF/OWL; Action resource as basics for communication and incorporated in the OWL ontology; Sending ACL based RDF/OWL messages; Receiving ACL based RDF/OWL messages; Incorpora-

tion of received information into a model; XMLRPC receiving messages; XMLRPC returning RDF and XML; An inference Engine or RDQL messages handling.

The JADE based agent can be developed using this library to support the Jena OWL model as the Agent memory and furthermore it is possible to include XML-RPC based functionalities for presenting knowledge or receiving events from external systems as RDF messages based on a used ontology. Moreover it allows communication among agents based on RDF/OWL. The developed library is published on JADE website as the 3rd party software [9] as a way to bring together Jena RDF/OWL and JADE Multi-agent system features.

6 Conclusion and Future Work

The paper describes how semantic web results can be used in multi-agent systems. An agent knowledge model has been created; it can model the agent environment, agent context and its results. The agent library to support such a model has been developed extending the JADE agent system. It has been proved that such a model and agent architecture are implementable and can be used in different knowledge management applications. The approach has been successfully used and evaluated in several R&D projects, mainly in the Pellucid IST project. The developed library for using such a semantic model has been published on the JADE agent website and it is available to the worldwide JADE community. Presented work will be further developed and extended. More detailed information on modeling, software library and concrete scenarios and use can be found in [10].

References

1. JADE (Java Agent DEvelopment Framework) Website, 2004, http://jade.cselt.it/
2. Giovani Caire, JADE Tutorial Application-defined Content Languages and Ontology, 2002
3. FIPA, FIPA SL Content Language Specification, 2000
4. Marek Obitko, Vladimir Maríik, OWL Ontology Agent based on FIPA proposal, Znalosti 2004, Brno, Czech Republic,2004
5. JAMES J. ANDERSON, An Agent-based Event Driven Foraging Model, Natural Resource Modeling, Volume 15, Number 1, 2002
6. Michael Luck, Peter McBurney, Chris Preist, Agent Technology: Enabling Next Generation Computing, A Roadmap for Agent Based Computing, 2003
7. Franz Baader, Deborah McGuinness, Daniele Nardi, The Description Logic Handbook, ISBN:0521781760, January 9, 2003
8. Guus Schreiber, Monica Crubezy, Mark Musen, A Case Study in Using Protege-2000 as a Tool for CommonKADS, ,2001
9. Michal Laclavik, AgentOWL; http://jade.tilab.com/community-3rdpartysw.htm#AgentOWL
10. Michal Laclavik: Ontology and Agent based Approach for Knowledge Management; PhD Thesis; Institute of Informatics, SAS, field: Applied Informatics, June 2005