MAS-ML: A Multi-Agent System Modeling Language

Viviane Torres da Silva Carlos J. P. de Lucena
Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Computer Science Department/TecComm Group
Rua Marques de São Vicente, 225, Rio de Janeiro, RJ, Brazil 22.453-900
{viviane, lucena}@inf.puc-rio.br

ABSTRACT
A multi-agent system modeling language (MAS-ML) that extends the UML (Unified Modeling Language) is proposed here based on a conceptual framework (metamodel) called TAO (Taming Agents and Objects). The most important difference between our approach and others presented in the literature is our clear definition and representation of the abstractions and behavioral features that compose MASs. Extensive experimentation is being used to access the expressiveness of models represented in MAS-ML and the ease to implement a multi-agent system from a specification expressed in our notation.

Categories and Subject Descriptors
D.2[**Software Engineering**]: Design – *representation.*

General Terms
Design, Languages

Keywords
Modeling language, multi-agent system, conceptual framework

1. PROBLEM DESCRIPTION
In the present section we justify the need for a new multi-agent system modeling language called MAS-ML (Multi-Agent System Modeling Language).

Multi-agent systems (MASs) are gaining wide acceptance in both industry and academia as a powerful paradigm for designing and developing software systems [1]. Together with this growth, new methodologies, methods, modeling languages, development platforms, tools and programming languages are being proposed. Agent-based systems require adequate techniques that explore their benefits and their peculiar characteristics.

In particular, there is a need for modeling languages for multi-agent systems that explore the use of agents and other abstractions as first order modeling elements. Modeling languages should represent the static and dynamic aspects of such systems by expressing the characteristics of all essential elements of MASs.

Some modeling languages proposed in the literature, such as [2,6], are based on the UML in order to reduce the risk of adopting a new technology. The UML is used as a basis for extension because it is widely accepted as a *de facto* standard for object-oriented modeling; whereas we believe, as in [2], that a new modeling language preferably should be an incremental extension of a known and trusted modeling language. Nevertheless, in its original form UML provides insufficient support for modeling MASs. Among other things, the UML metamodel does not provide support for modeling agents, organizations and agent roles. Current approaches [2,6] propose to extend the UML by expressing agents as a stereotype of objects/classes. This is not satisfactory because stereotypes can only be used to indicate a difference in meaning or usage between two model elements with identical structure. Based on the definition presented in the UML specification [5], stereotypes may not be used to represent two completely different paradigms. As a consequence, the current UML extensions to deal with the fundamental MAS characteristics neither define nor clearly represent the elements they identify, nor their relationships. Therefore, we felt the need for a new approach to extend the UML metamodel so that it could incorporate all features of multi-agent systems.

OOPSLA'03, October 26–30, 2003, Anaheim, California, USA.
ACM 1-58113-751-6/03/0010.

2. GOALS
Our aim is to provide a multi-agent system modeling language that encompasses the essential static and dynamic aspects of MASs by emphasizing a clear representation for their concepts and relationships. Due to the non-existence of a set of commonly accepted abstractions to describe the static and dynamic aspects of MASs, we have developed a conceptual framework (metamodel) that expresses most aspects of MASs. The TAO (Taming Agents and Objects) conceptual framework [3] provides an ontology that makes it possible to understand the abstractions, and their relationships, when used to support the development of large-scale MASs. The ontology associates well accepted abstractions, such as objects and classes, to other abstractions, such as agents, roles and organizations. Together, they establish the foundation for agent and object-based software engineering.

We propose a multi-agent system modeling language (MAS-ML) [4] that extends the UML based on TAO. The UML metamodel was extended by preserving all object-related concepts while including agent-related abstractions. To extend the UML metamodel according to the TAO meta-model concepts, we needed more than the three extensions mechanisms provided by the UML, namely: tag definitions, constraints and stereotypes. Often it was necessary to add new metaclasses to the UML metamodel to represent some new MAS concepts.

3. THE APPROACH

3.1 TAO
In order to understand the difference between MASs and object-oriented systems and to provide foundations to better understand and define MAS elements, we have proposed the TAO conceptual framework. TAO is an evolving innovative conceptual framework based on agent and object abstractions, which are the foundations for modeling distributed software systems.

The TAO metamodel [3] defines the static and dynamic aspects of MASs. The static aspects are related to the definition of the entities (structural and behavioral features) and the relationships

between them. The entities defined in TAO are *object*, *agent*, *organization*, *role* (*agent role* and *object role*), *environment* and *event*. The relationships link two elements and describe how these elements are related to each other. The relationships defined in TAO are *inhabit*, *ownership*, *play*, *control*, *dependency*, *association*, *aggregation* and *specialization*.

The dynamic aspects of the TAO metamodel describe the interactions between its static elements. They can be classified as primitive (elementary) dynamic processes and high-level dynamic processes. Primitive processes describe the most basic domain-independent interactions that exist between elements. The processes of creating and destroying MAS elements are characterized as primitive processes. High-level dynamic processes are more complex domain-independent behavior that are described based on primitive dynamic processes and other high-level dynamic processes.

3.2 MAS-ML

Our approach involved extending the UML metamodel to incorporate concepts related to the MAS theory. TAO defines three main concepts – *entities*, *properties* (*structural* and *behavioral features*) and *relationships* – that have been mapped to the UML metamodel. In order to extend UML according to TAO non-object concepts, new metaclasses and stereotypes have been created and associated with the UML metamodel. When introducing new abstraction in the UML metamodel, it is necessary to create new diagram elements to represent the new elements and relationships.

Because of the set of different elements and relationships defined in the TAO metamodel that have been incorporated in the UML metamodel, new diagrams – Organization and Role Diagram – have been created and UML diagrams that already exist – Class and Sequence Diagram – have been adapted. The three structural diagrams – Class, Organization and Role diagrams – show all elements and all relationships defined in TAO. The Sequence diagram represents the dynamic interaction between the elements that compose a MAS — i.e., between objects, agents, organizations and environments.

The extended Class diagram also represents agents, organizations and the relationships between agents, organizations and classes as defined in TAO. The Organization Diagram models the system organizations identifying their habitats, the roles that they define and the elements – objects, agents and sub-organizations – that play these roles. This diagram shows the TAO's ownership, play and inhabits relationships. The Role Diagram is responsible for clarifying the relationships between the agent roles and object roles. This diagram shows the TAO's control, dependency, association, aggregation and specialization relationships.

3.3 From Design Models to Code

Software development based on the agent-oriented paradigm depends on modeling and programming languages that provide the traceability between the requirement, analysis, design and the implementation code [2]. The proposed modeling language defines agent, organization and environments as first order abstractions. It would be easier to use a programming language that also considers these elements as first order abstractions to implement MAS-ML models . An object-oriented programming language could be used as an alternative; however, the elements presented in the models need to be mapped to objects/classes.

This work includes an analysis of existing agent and object-oriented programming languages to support MAS implementation. Our aim is to evaluate the mapping of agent-oriented design models and concepts expressed in MAS-ML to object-oriented and agent-oriented programming languages.

4. SUMMARY

The most important difference between our approach and others presented in the literature is our clear definition and representation of the elements that compose MASs and their behavior. Our proposal is based on a conceptual framework (TAO) that describes the structural and dynamic properties of MASs.

The main contributions of our work can be summarized in three aspects: i) the proposal of a conceptual framework that defines the structural and dynamic aspects of MASs; ii) the proposal of a modeling language that extends the UML, based on the conceptual framework; and iii) the mapping of the design elements in the agent level of abstraction to a programming language.

Our goal will be achieved through extensive experimentation with MAS-ML in multi-agent systems published as benchmarks in the literature and other realistic MAS applications. Specialists other than the author are conducting experiments. It is important to experiment with different application situations that explore a large spectrum of multi-agent system characteristics. The set of projects has been chosen to illustrate the structural and dynamic aspects proposed in TAO and expressed by the MAS-ML.

The contributions of creating an MAS modeling language will be evaluated by expressing, in MAS-ML, several modeling situations that are difficult and/or impossible to represent in existing MAS modeling languages. These situations will be selected from the characteristics of MASs proposed in the literature. The MAS-ML designs will also be compared to designs provided by other multi-agent system modeling languages that extend the UML.

5. REFERENCES

[1] Jennings, N.: On agent-based software engineering, In Artificial Intelligence, 2000, 11, pp. 277-296.

[2] Odell, J., Parunak, H. and Bauer, B.: Extending UML for Agents, In: Proceedings of the Agent-Oriented Information Systems Workshop, Odell, J., et. al, Eds., 17th National Conference on Artificial Intelligence,2000, pp.3-17

[3] Silva, V. et.al: Taming Agents and Objects in Software Engineering, In Software Engineering for Large-Scale Multi-Agent System, Garcia, A. et. al Eds., LNCS, Springer, 2003.

[4] Silva, V., Lucena, C.: From a Conceptual Framework for Agents and Objects to a Multi-Agent System Modeling Language, Technical Report CS2003-03, School of Computer Science, University of Waterloo, Canada, 2003.

[5] UML, Unified Modeling Language Specification, Version 1.5, http://www.omg.org/uml/, 2002.

[6] Wagner, G.: The Agent-Object-Relationship Metamodel, In Proceedings of the 2nd International Symposium: From Agent Theory to Agent Implementation, 2000.