AOR-Simulation.org – Cognitive Agent Simulation

Gerd Wagner and Mircea Diaconescu
Brandenburg University of Technology
P.O.Box 101344
03013 Cottbus, Germany
{G.Wagner, M.Diaconescu}@tu-cottbus.de

ABSTRACT

In this paper, we describe an ontologically well-founded agent-based discrete event simulation framework with a high-level rule-based simulation language and an abstract simulator architecture and execution model, which allows both basic discrete event simulations without agents and complex agent-based simulations with (possibly distorted) perceptions and (possibly false) beliefs. The framework includes the simulation language AORSL, a Java simulator with a statistics and a 2D visualization tab, a player for the 3D rendering of simulation logs, and a Web simulation server. It is available on www.AOR-Simulation.org.

Categories and Subject Descriptors

I.2.0 [ARTIFICIAL INTELLIGENCE]: Cognitive simulation

Keywords

Simulation, cognition, beliefs.

1. INTRODUCTION

The *Agent-Object-Relationship (AOR)* simulation framework is intended to be used as a universal multi-purpose simulation framework in science, engineering, education and entertainment. Most parts of the framework are open-source-licensed under the GPL.

The basis of the framework is the XML-based *AOR Simulation Language (AORSL)*, which is used to express high-level declarative simulation models and simulation scenarios independently of a particular programming language. With this language, AOR Simulation is the first agent-based simulation framework that supports truly cognitive agents with a full-fledged model of beliefs (closely related to RDF). For the future, it is planned to provide a syntax-guided text editor and an Eclipse-based visual modeling tool for facilitating the authoring of AOR simulation models.

AOR simulators may be implemented in different programming languages, but they must conform to the AOR simulator architecture and to the AOR simulator execution model as defined in [1]. The reference AOR simulator implementation, called *AOR-JavaSim*, is a standalone desktop program with a graphical user interface based on the Java programming language. It includes a 2D visualization tab that allows observing a simulation run when suitable views have been defined.

When a simulation scenario is executed, the AOR simulator creates a simulation log in an XML format. These logs can be further processed, e.g. by statistical analysis and reporting tools. The framework provides a player that can render a log as a 3D animation, using the OGRE 3D open source graphics engine [2].

The framework also includes a Web simulation server, which allows running AOR simulations on the Web without first downloading and installing any software, implemented with the JBoss Java application server. At present, this server is only accessible on the Intranet of Brandenburg University of Technology. But its public accessibility is planned for the near future.

2. THE SIMULATION LANGUAGE

A simulation scenario file is expressed in the AOR Simulation Language (AORSL) and then first translated to Java source code before it is compiled to Java byte code and executed, as indicated in Figure 1.

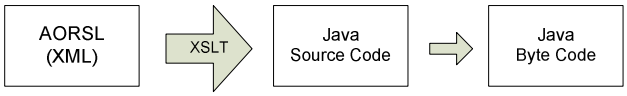

Figure 1: From AORSL to Java byte code.

A *simulation scenario* consists of a simulation model, an initial state definition and zero or more view definitions.

A *simulation model* consists of

- an optional *space model* (needed for physical objects/agents);
- a set of *entity types*, including different categories of event, message, object and agent types;
- a set of *environment rules*, which define causality laws governing the environment state changes.

An agent type is defined by means of

- a set of (objective) properties;
- a set of (subjective) self-belief properties
- a set of (subjective) belief entity types
- a set of agent rules, which define the agent's reactive behavior in response to events

2.1 Space Models

A simulation can use various space models characterized by the parameters *Dimension* (1, 2 or 3); *Discrete/Continuous*; *Geometry* (Euclidean or Toroidal); and space limits (*xMax*, *yMax*, *zMax*)

Cite as: AOR-Simulation.org – Cognitive Agent Simulation, Gerd Wagner and Mircea Diaconescu, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra, and Castelfranchi (eds.), May, 10–15., 2009, Budapest, Hungary, pp. XXX-XXX.

2.2 Ontological Categories

The upper level ontological categories of AOR simulation are messages, events and objects, which include agents, physical objects and physical agents, as depicted in Figure 2.

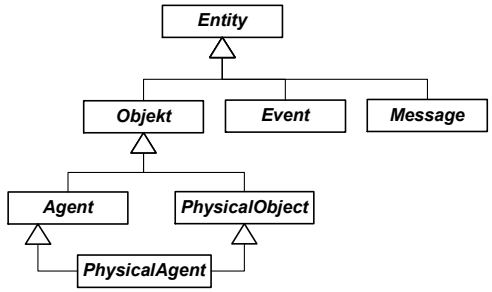

Figure 2: Upper-level ontological categories

An elaborate ontology of event types, shown in Figure 3, has proven to be fundamental in AOR simulation. Internal events are those events that happen "in the mind" of the agent. For modeling distorted perceptions, both a perception event type and the corresponding actual perception event type can be defined and related with each other via actual perception mapping rules.

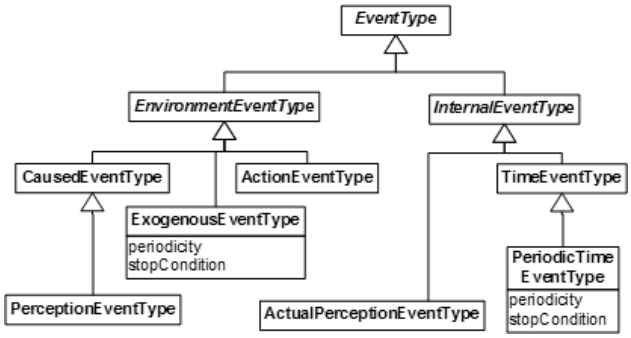

Figure 3: Categories of event types

2.3 Rule-Based Modeling

Both the behavior of the environment (its causality laws) and the behavior of agents are modeled with the help of rules, thus supporting high-level declarative behavior modeling.

An environment rule is a 5-tuple < EvtT, Var, Cond, UpdExpr, ResEvtExpr >, where

- *EvtT* denotes the type of event that triggers the rule
- *Var* is a set of variable declarations, such that each variable is bound either to a specific object or to a set of objects
- *Cond* is a logical condition formula, allowing for variables
- *UpdExpr* specifies an update of the environment state
- *ResEvtExpr* is a list of resulting events, which will be created when the rule is fired

2.4 Beliefs and Query Answering

AOR simulation supports the distinction between *facts* and *beliefs*, including *self-beliefs* (the agent's beliefs about itself). The W3C RDF query language *SPARQL* is used for expressing queries that an agent can ask another agent about its beliefs.

3. ARCHITECTURE AND COMPARISON

As depicted in Figure 4 , an AOR simulator consists of

- an environment simulator, which creates perception events and passes them to the agent simulators concerned
- a possibly empty set of agent simulators, which process any perception events received and, in response, create action events that are passed back to the environment simulator

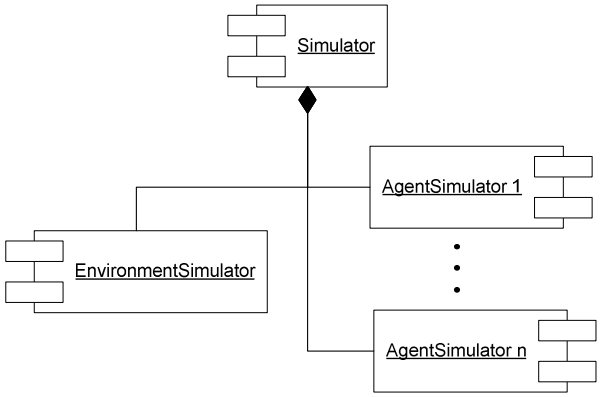

Figure 4: The AOR simulator architecture

This architecture supports the distribution and parallel execution of simulations, as the environment simulator and all agent simulators can be executed in parallel. The AOR simulation server prototype is running on an 8 node cluster with dual quadcore processors, thus using 64 processor cores.

Comparison Table	*RePast*	*NetLogo*	*SeSAm*	*AORS*
Distinction between object and agent	–	√	√	√
Rule-based behavior modeling	–	–	–	√
Cognitive model of perception	–	–	–	√
Distinction between facts and beliefs	–	–	–	√
Beliefs about other entities (belief triples)	–	–	–	√

Acknowledgements Thanks to the AOR Simulation team: Jens Werner, Marco Pehla, Stefan Böcker, Xia Keqiu, Adrian Giurca and Emilian Pascalau.

4. REFERENCES

[1] Gerd Wagner: AOR Modeling and Simulation – Towards a General Architecture for Agent-Based Discrete Event Simulation. In P. Giorgini et al. (Eds.): *Agent-Oriented Information Systems*, Springer LNAI **3030**, pp. 174–188, 2004.

[2] OGRE 3D: Open source graphics engine. Accessed at 12-Jan-2009. [http://www.ogre3d.org]