SIF: an Agent-Based Simulation Toolbox using the EMS Paradigm

P. Funk, C. Gerber[1], J. Lind, M. Schillo

German Research Center For Artificial Intelligence (DFKI),

Im Stadtwald, Building 36, 66123 Saarbrücken, Germany,

{funk, gerber, lind, schillo}@dfki.de

Abstract

There are many different approaches to the task of realising a simulation, ranging from using an off-the-shelf simulation tool to implementing a simulation from scratch. In this paper, we propose an approach to simulation based on the use of *multi-agent systems* developed in the discipline of Artificial Intelligence (AI). The field of multi-agent systems is concerned with computational systems composed of several autonomous agents capable of mutual interaction in an artificial or real environment. We present the Social Interaction Framework *SIF* [Funk et al. 1998] and demonstrate its suitability for social simulation as it naturally allows for modelling of processes at both, the individual (micro) and society (macro) level. SIF implements the EMS (Effector-Medium-Sensor) paradigm, which provides a generic agent-world interface.

Motivation: Why Use Multi-Agent Systems?

According to K. Troitsch [Doran et al. 1995] simulation tools seem to be used mainly for two rather conflicting purposes: On the one hand, for the better understanding of some observed real-world phenomenon and on the other hand, for advice and decision support. In order to satisfy these diverging demands, a whole variety of simulation approaches has been developed since the early 1960s.

Pre-built products such as the classic flow diagram system from the 1960s, GPSS (General Purpose Simulation System) of [Gordon 1969] or more recent graphic-oriented Visual Interactive Modelling Systems (VIMS) provide built-in functionalities leading to little modelling effort. Furthermore, these systems require only little programming skills. However, limitations of such tools are the rather low execution speed, the restricted expressivity leading to insufficient scalability as only few entities can be modelled. On the other extreme, the most flexible approach is do-it-yourself-coding, since the developer has all freedom to model entities. But such an approach requires detailed programming skills and rather time consuming implementation and debugging.

Specialised simulation programming languages and pre-written simulation libraries for general-purpose languages are located somewhere between the extremes. The simulation designer still has a high degree of flexibility and can make use of features pre-specified in the system. Such systems allow easy access to simulation modelling for novice users as well as more complex simulation experiments for experts.

A recent development in Artificial Intelligence is *Agent-Oriented Programming* (AOP) [Shoham 1991], derived from Object-Oriented Programming (OOP): the entity in focus is a so-called *agent*, a unit able to autonomously plan and perform actions. Some approaches of AOP see communication between agents as a special case of acting, whereas others regard acting and communication as different forms of interaction. Tasks are performed in a distributed fashion through cooperation between agents. With SIF, we present a

[1] Part of this work has been supported by Siemens AG Germany

JAVA[TM] library which enables agent-oriented programming for quick and platform-independent simulation modelling. SIF provides a multi-threaded framework where basic agent functionalities (such as sensing, performing actions, communication, etc.) are already provided. The simulation designer only needs to specify the agents' decision making procedures. This can be done from scratch or by evoking of reference procedures.

Simulations based on multi-agent systems can be used to model a broad variety of domains because multi-agent systems support simulations of any granularity:

- In *Micro-Level Simulations* agents are used to represent individuals. An agent's decision making procedure can be defined in a primitive way, only depending on a certain state of the agent and on states of *neighbour agents*. In such a case, SIF basically works as a cellular automaton [Axelrod 1984]. However, more complex cognitive processes, such as logic-based inference capabilities or even "illogical" emotions can be modelled leading to a much more realistic simulation of human behaviour.

- In *Macro-Level Simulations* agents represent whole individual groups or societies. Hence, macro-level properties can be modelled effectively. Through combination of micro- and macro-agents the behaviour of individuals in societies can be simulated easily leading to a natural bridge of the micro-macro-gap.

The remainder of the paper is structured in the following way: in the next section we introduce the EMS paradigm. This model is the underlying theory of the SIF simulation system, which is described in the third section. Finally, we give a conclusion.

The EMS Model of Interaction

A very general and broadly accepted definition of an agent is provided by [Russell and Norvig 1996]:

> An *agent* is an entity that can be viewed as *perceiving* its environment through *sensors* and *acting* upon that environment through *effectors*.

The so called *Effector/Medium/Sensor paradigm* (for short EMS) reflects this definition. This paradigm provides an appropriate abstraction of a (human) agent acting and interacting with its environment and other (human) agents. This idea is related to Malsch's work on *generalised media of interaction* in sociology [Malsch and Schulz-Schäfer 1997]. Here action and interaction are transmitted via appropriate media.

According to EMS, verbal communication – as an instance of action and interaction – is modelled in the following manner: each agent (human) has a *speech effector* (voice, speech apparatus) and an *audio sensor* (ear). The spoken language, respectively the sound, is transmitted through the air. We call this transmitter the *medium*. Figure 1 shows this abstraction.

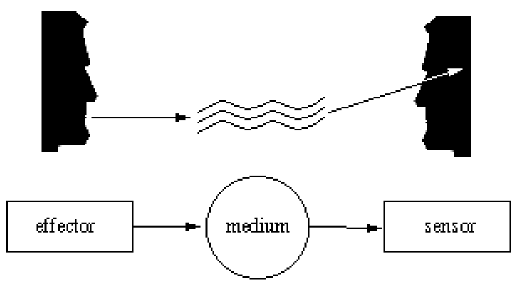

Figure 1 The EMS model of interaction

The EMS paradigm reflects this definition. In general, an agent according to the EMS paradigm is equipped with a number of effectors through which it can act upon its virtual environment and with a number of sensors through which it can perceive the state of the environment (see Figure 2).

The EMS allows to model groups of media and sensors/ effectors belonging to those media. For example, we formalise communication by using an air-like medium comprising message sending effectors and receiving sensors. To simulate e.g. agents moving around in a virtual world, the library provides media which reflect the topology of the environment.

The Simulation Toolbox SIF

In this section we describe the Social Interaction Framework (SIF) that implements the EMS paradigm. SIF is a JAVA™ library which supports agent-oriented programming. It consists of a scenario-independent simulation engine and a number of ready-to-use components. SIF contains a mechanism to specify simulations in a simple script language. These scripts can be loaded, modified and reloaded by using the graphical user interface (GUI) without need to shut down the SIF system. In the following paragraphs we first describe the architecture of the SIF library and then we explain the control and information flow in the system during run-time.

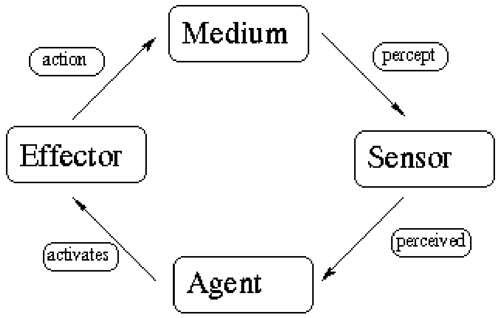

Figure 2 The EMS model of interaction

The SIF Architecture

The EMS-based implementation consists of the following modules: the *world server*, a variety of *agents* situated in the simulation environment and the *graphical user interface*. These parts run independently from each other in a JAVA™ thread of their own.

World Server: The world server updates the world representation and organises the flow of information during the simulation. It starts and stops the simulation and provides debugging and data collection facilities. For creating individual scenarios, it is necessary to specify two parts of the world server: the *action manager* and the *sensor manager*. The *action manager* operationally defines the state transition of the world for every action. The *sensor manager* processes the data in the world representation to compute the input of the sensors according to the specifications of the scenario.

Agents: An agent can possess different types and any number of sensors and effectors. The toolbox offers agents that can move around, pick up resources, communicate, etc. Due to the generic design, any type of agent can be connected to the world server ranging from simple JAVA™ objects to sophisticated InterRRaP [Müller 1996] agents.

Graphical User Interface: The GUI is linked to the core of SIF just like an agent, i.e. it communicates with the core via sensors and effectors. The percepts that it receives tell the GUI e.g. to update certain parts of the visualisation or to display agent information. The GUI offers a control pad which is capable of forcing agents to perform certain actions. It also provides information windows for easy access to internal data of agents.

Control and Information Flow in SIF

The control and information flow in a SIF simulation is implemented according to the EMS paradigm (see Figure 3). To illustrate the information flow we assume that the simulation has been running for a while. We start the description of the control cycle at the point where a certain agent (upper left box in Figure 3) is about to choose its next action to make. As soon as the decision making process of the agent has produced a result, the agent activates the corresponding effector. For instance, if an agent has decided to move forward, it activates the *move effector*.

In the simulation, the effector wraps the information of what is intended to happen into an *action object*. The world server stores all incoming actions in a first-in-first-out queue. In each step of the simulation, the action manager (as part of the world server) picks the next action from the queue and accesses the world representation to find out whether the action is legal or not. If it is legal the action manager will update the world representation accordingly. Furthermore, the action manager commits the action and sends a percept to the GUI, informing it that a part of the world has to be redrawn before it passes control to the sensor manager. If the action is not legal in a particular situation the action manager will directly pass control to the sensor manager.

The sensor manager then computes the input for all sensors that are affected by the change of the world. The input is sent in form of percepts (reflecting the local point of view of the agents) to the sensors in question. Now one cycle for the agent is completed and it can perform the next action. As a side effect of the asynchronous implementation of SIF the agent does not have to wait for this feedback from the world.

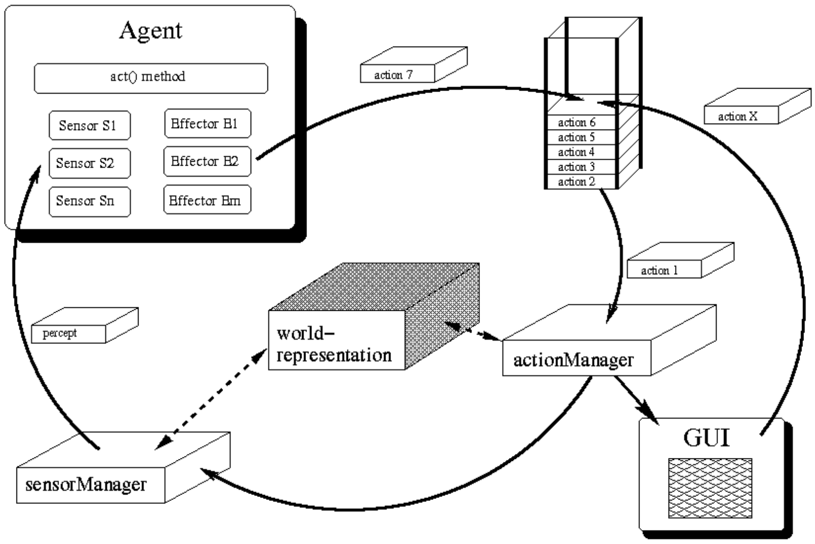

Figure 3 The information flow in SIF

The connection of the GUI to the queue of actions illustrates an information flow that occurs when either the experimenter uses a control pad (which is part of the GUI) to force an agent to execute a particular action or when the experimenter wants to start or stop the experiment. In the first case, an action containing the command is posted to the queue. These commands are passed to the agent and force it to execute the respective action. In the second case a command action is sent to the action manager to start or stop the control flow in the simulation.

Conclusion

In this paper we have outlined a general model for agent interaction (EMS paradigm) and presented a simulation library (SIF) based on this model. The library supports agent-oriented programming (AOP) which facilitates rapid prototyping of asynchronous multi-threaded simulations. Further work will focus on extending SIF to a distributed testbed using TCP/IP and enhancing the implementation of time restrictions of actions.

References

[Axelrod 1984] Axelrod, R. *The Evolution of Cooperation*. Basic Books, New York, 1984.

[Doran et al. 1995] Doran, J. et al. *Social Science Microsimulation: A Challenge to Computer Science.* Dagstuhl-Seminar-Report 112. D-66687 Wadern, Germany, 1995.

[Funk et al. 1998] Funk, P., Gerber, C., Lind, J., Schillo, M. *Social Interaction Framework – A Generic Testbed for Social Agents*. Research Report, German Research Center for Artificial Intelligence. To appear, 1998.

[Gordon 1969] Gordon, G. *System Simulation.* Prentice-Hall, Englewoods Cliffs, NJ, 1969

[Malsch and Schulz-Schäfer 1997] Malsch, T. and Schulz-Schäfer, I. Generalized Media of Interaction and Interagent Coordination. In: *Socially Intelligent Agents: Papers from the 1997 AAAI Fall Symposium*. Technical Report FS-97-02, Cambridge, Massachusetts, 1997.

[Müller 1996] Müller, J. P. *An Architecture for Dynamically Interacting Agents*. PhD thesis, Universität des Saarlandes, Saarbrücken, 1996.

[Russell and Norvig 1996] Russell, S. and Norvig, P. *Artificial Intelligence, A Modern Approach.* Prentice-Hall, 1996.

[Shoham 1991] Shoham, Y. Agent-oriented programming. In *Proceedings of the 11th International Workshop on DAI*, pages 345-353, 1991.