A TRASS-BASED AGENT MODEL FOR TRAFFIC SIMULATION

Ulf Lotzmann
Michael Möhring
Institute of Information Systems Research
University of Koblenz
Universitätsstraße 1, Koblenz 56070, Germany
E-mail: {ulf, moeh}@uni-koblenz.de

KEYWORDS

Simulation framework, traffic simulation, multi-agent based model, normative agents.

ABSTRACT

This paper focuses on traffic simulation generated by the multi-agent simulation framework TRASS (Traffic Simulation System). TRASS-based simulations conceive agents as individual traffic participants moving in an artificial environment and which can be described in a three-layer agent model. There is no restriction on any particular kind of road users such as cars or pedestrians, but a concept is presented which is appropriate to model different kinds of traffic participants and have them interact with each other in one single scenario which may not only include roads, but also stadiums, shopping malls and other situations where pedestrians or vehicles of any kind move around.
Simulation results will be presented by using TRASS as a testbed for analyzing norm emerging processes, ported in the context of a simple traffic scenario (i.e. learn traffic rules by observation and experience).

INTRODUCTION

The design of a simulation model particularly in terms of agent design is affected by multiple factors. An important one is, without doubt, the selection of the simulation software to be used. For almost every serious simulation project, a ready-made, perfect fitting and mature software is not available. In fact a modeller has to search for the most satisfying compromise and will be likely to face three alternatives to choose among:

- The usage of a domain-specific tool. This implies that the simulation model can be adapted to the specifications of the tool. An example from the traffic simulation domain is SUMO, a software system for the microscopic simulation of road traffic, applying a microscopic car-following model (Krajzewicz *et al.*, 2006).
- The usage of a general-purpose simulation tool. In this case most likely a lot of (programming) work has to be done in order to match the tool with the model's requirements. Some important agent-based frameworks are Repast, Mason, Netlogo and Swarm. Evaluations of these tools were published by Railsback *et al.* (2006) and Tobias & Hofmann (2004). In Oliveira & Bazzan (2006) a project that deals with the coordination of traffic light using the Swarm platform is presented.
- The development of the "optimal fitting" software "from scratch".

In this paper we present the multi-agent simulation framework TRASS (Traffic Simulation System). Although the name suggests a domain specific concept, the underlying design makes it applicable for universal agent-based simulation scenarios, where the term "agent" refers to the definition given in Gilbert & Troitzsch (2005). The associated software system was implemented using JAVA with the aim to introduce novel properties like parallel execution incorporating multi-threading techniques and a continuous space. Thus, the approach relies on the third item in the list above.

Furthermore, this paper shows an approach for a broad range of traffic simulation applications. It aims at discussing a concept where agents represent human actors, "playing" different roles of traffic participants in an artificial spatial environment.
The underlying agent model defines a set of different layers for the representation of physical, technical and mental aspects. The physical layer characterizes the appearance of an agent within the environment – defined by a topography model – and comprises properties like shape, facilities for perception and attributes of motion. The technical layer determines the actions an agent is able to perform on the environment by modifying the physical properties. How to use these technical actions strategically in order to achieve a desired behaviour or a predefined goal is the concern for the mental layer.
While the TRASS framework specifies the topography model and the characteristics of the agent physics, appropriate methods have to be employed for the technical and the mental layer. In our implementation, the former is called "robotic layer" as it uses the technique of the finite state machine that is utilised frequently in association with autonomous robots. The latter is called "AI layer" and incorporates methods originating from artificial intelligence research.

To demonstrate how this approach can be applied, an example on simulating the emergence of norms in a simple traffic scenario completes the paper.

THE TRASS AGENT MODEL

The usage of agent-based simulation in the social sciences as a method for description, understanding or prediction of social phenomena has become more and more important over the last two decades. The agents involved in the majority of simulation scenarios represent restricted (and sometimes simple) models of human individuals. Although these agent models cover only a few aspects of humans and are driven by minimal sets of explicit goals, the simulation outcomes in terms of emergent structures reproducing complex coherences of real societies seem very impressing and promising. While the typical domains of social simulation generally refer to market analysis and policy, it seems reasonable to transfer the concept into other fields, in particular traffic simulation.

An agent model describing a traffic participant has to consider further aspects besides the deliberation capabilities. Especially the complex interaction between a traffic participant and the environment is of importance. In other words, the AI potential needs a fundament of physical and technical capabilities.
We structure the different aspects of our agent model for traffic participants in three distinct layers: the physical layer, the robotic layer and the AI layer (Figure 1).

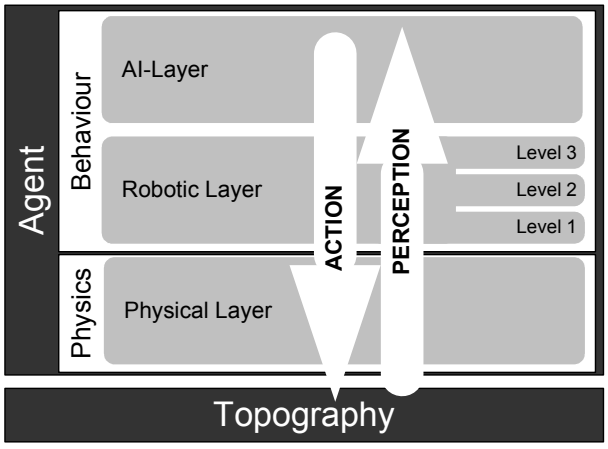

Figure 1: Layers of the TRASS agent model

Physical Layer

The physical layer describes the abstraction of the static properties and the parameters of motion. Thus it defines the physical appearance of the agent within the artificial environment.

This layer constitutes the interface to the spatial environment defined by the topography model. It is obvious that the physical layer and the topography model must be coordinated. For this reason both model descriptions are part of the TRASS core framework.
The topography model is a two-dimensional continuous space that can be split into districts and that is structured by a mesh of polygon-shaped regions. Each region describes one distinct area of the topography, such as a lane of a road section, a crossing or building (see Figure 4). The corresponding physical layer of the agent model is composed of five elements, which are in detail:

- The shape of the agent is constituted by an arbitrary number of circles (with different radii) approximating the outline of the entity to be represented. The usage of circles is due to performance issues when executing necessary calculations during simulation runs (e.g. distance between agents).
- The sensor unit is constituted by an arbitrary number of (various) circle sectors, defining the zones where an agent is able to perceive the environment.
- The communication area is similar to the sensor unit in terms of the geometrical shape involved. It is used to define zones in which an agent can appear for other (observing) agents in different manners. For example, a traffic light agent sends its information (red, yellow or green light) in one distinct direction, affecting only car agents arriving from one distinct way. For any other agent the traffic light is just an obstacle.
- The reference point defines the current position in the two-dimensional world.
- Attributes of motion are velocity and direction of motion.

Robotic Layer

The robotic layer fills the gap between the “low-level” attributes of the physical layer and the abstract strategic decisions treated by the AI layer. Thus, the design of the robotic layer is of utmost importance for the entire system.
The name “robotic layer” is referring to the methods that are used here to implement the transformation process: As in many autonomous robot systems, a form of enhanced finite state machine (Hopcroft & Ullman, 1979) controls the parameter changes depending on the active state and triggered by incidents perceived in the environment (resulting in reactive behaviour) or generated by the AI layer (resulting in proactive behaviour).

The design of the finite state machine concept used in the robotic layer is inspired by an evolutionary classification model of human needs, originated by Max-Neef (1992) and translated into an agent model for

market simulation by Norris & Jager (2004). In this model, a distinction is made between

- survival needs, controlled by the brain stem and cerebellum and resulting in automated decisions (reptile mode);
- social needs, controlled by the limbic system based on social heuristics (mammal mode);
- identity needs, controlled by the neo cortex via individual heuristics (primate mode).

In the robotic layer a related concept of different "levels of mind" is used where each level corresponds with a stage in a nested hierarchy of automata. The following levels are considered here:

- basic actions which humans execute "automatically" without thinking (e.g. turning the steering wheel);
- elementary activities (composed of basic actions) which are conducted intuitively by humans (e.g. hold the centre of a lane);
- complex activities as sequences of elementary activities where a human's attention is required (e.g. lane change operation).

Each level is represented by a self-contained automaton that operates on the instance below and is controlled by the instance above. These instances can be other automata within the hierarchy, except the bottom-level and the top-level automata. While the former executes parameter changes on the agents' physical layer, the administration of the latter is done by the AI layer.
The administration of an automaton is done by the substitution of state transitions. A state transition is a function that receives the active automata state and the current perception of the environment as input and calculates the successive state (including parameters for configuration of state activities) as output.
The perceptive input of the transition function is generated by the agent sensor. At each automata level perception filter units are attached for categorizing and condensing the perceived objects.
The execution of a state transition is triggered by events:

- every time-step event for the bottom-level automaton (since we use a time discrete simulation concept);
- otherwise, the trigger event for a transition on level n is generated within the transition function on level n-1.

Figure 2 gives an overview on the entire concept for the robotic layer.

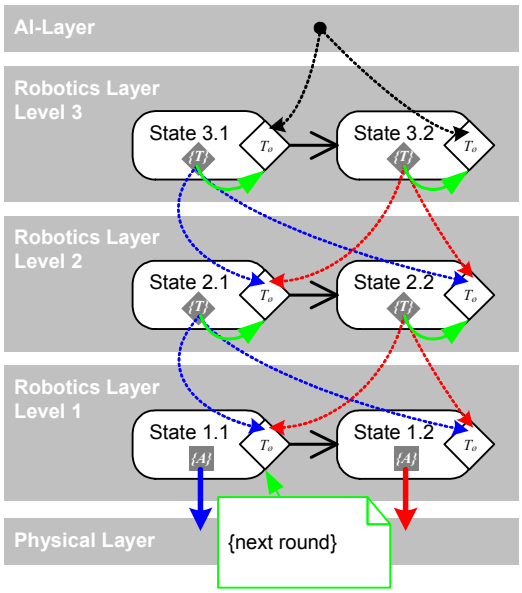

Figure 2: Levels of the robotic layer. $T_{Ø}$ is the abbreviation for "transition template", *{T}* and *{A}* indicate sets of transitions and actions, respectively.

AI Layer

In order to show "intelligent" – and thus realistic – behaviour, the AI part of an agent model for traffic participants requires a control process that is comprised of the following activities:

- Perceive the environment containing geographical elements and neighbouring agents.
- Compose and memorize an agent-individual view of the simulation world based on the perceived environment.
- Communicate with a subset of the agent population.
- Reason about different alternatives to act in order to achieve predetermined goals, considering the current environmental state and the agent memory.
- Execute "physical" action based on the capabilities offered by the robotic layer (e.g. driving a curve or changing a lane).
- Estimate the success of action, leading to revaluation of agent memory.

While most of the issues mentioned in the list above are of a more or less technical nature in terms of processing and memorizing perception data and adjusting of parameters, the subject of "reasoning about alternatives" constitutes the actual AI capabilities.

Such a reasoning process can be expressed in a function where the input parameters contain quantified information about the present state of the perceived environment and the agent-internal memory, and the

output is a set of probabilities for the alternatives to choose.
Since in most cases it seems possible to split a set of three or more options to choose between into the nested execution of different functions, each designed for two concurring alternatives, an approach originated in the social sciences is applicable, namely the Weidlich-Haag opinion forming model (Weidlich & Haag, 1983).
The original version of this simple model includes two levels. At the macro level the current ratio of (the subset of) agents which have chosen alternative *a* in the past (P_a), to agents relying on alternative *b* (P_b) is calculated embracing the entire population of agents *P*, and stored within the real-valued variable *x*:

$$x = \frac{|P_a| - |P_b|}{|P_a| + |P_b|}$$

The range of x is within [-1, 1], where the minimum stands for "all agents rely on alternative *a*", the maximum for "all agents rely on alternative *b*", respectively.
At the micro level each agent "decides" (based on the macro variable x) in every time step whether to change or keep its actual attitude. This is done by calculating the probability for an opinion change. The probability function depends on the present attitude and reads for alternative *b* as

$$\mu = \nu \exp(\pi + \kappa x)$$

and analogously for alternative *a*:

$$\mu = \nu \exp(-(\pi + \kappa x))$$

The meaning of the parameters involved is as follows:

- ν – flexibility (higher values increase the probability for attitude changes);
- π – preference (for alternative *a* when > 0, for alternative *b* when < 0);
- κ – coupling (higher values increase the influence of variable *x*).

Basis for the attitude change decision is the comparison of the calculated probability μ with a uniform random number *m* (in the range [0; 1]).

We use this model in a couple of traffic oriented simulation scenarios one of which is presented in this paper later on.
Besides the Weidlich-Haag model exemplarily described above, the adoption of various other approaches from artificial intelligence research is imaginable as basis for the AI layer.

THE TRASS SOFTWARE

The agent model presented in the previous section was developed with and on the basis of the TRASS framework. While the robotic and AI layers of the agent model are built upon the TRASS software in terms of extensions, the topography model and the physical layer are determined by the software design of the TRASS core. The following list gives an overview on the key properties of the TRASS core:

- continuous space (with a full set of geometric algorithms and tools to handle corresponding topographies);
- discrete time (controlled by a timer component);
- multi-threaded architecture resulting in parallel simulation execution on multi-processor machines;
- message-based inter-agent communication with single and group recipients, enabling the agents to exchange symbolic messages;
- mediator-based communication infrastructure fosters distributed execution.

The TRASS core does not bring any visualization capabilities or graphical user interface. It defines an programming interface that allows for the integration of external components. The interface consists of sections for:

- definition of agent types with specific behaviour by writing Java code;
- XML based configuration of simulations;
- access to the internal data structure of the simulation kernel for collecting attribute values while executing a simulation;
- administration of simulation runs.

In order to offer a complete and concerted interface for user interaction, an Integrated Development Environment has been developed. This software incorporates functionality like topography designer, agent modeller as well as a runtime environment for simulation execution and visualization. Figure 3 shows the component structure of the TRASS system.

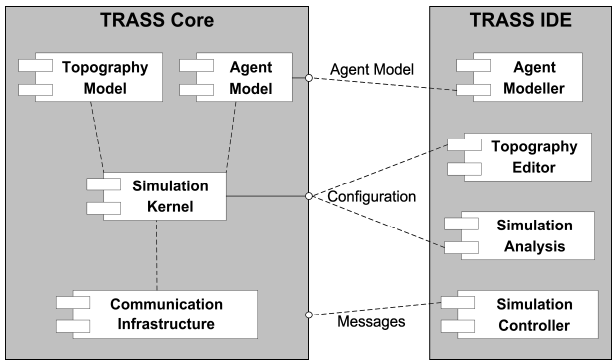

Figure 3: Component diagram of TRASS

EMERGENCE OF NORMS IN A TRAFFIC SCENARIO

The development of concrete model examples plays an important part in the validation process of the TRASS concept and the embedded three-layer agent architecture described in the preceding chapters. This includes typical simulation models of traffic flows in a real topography based on empirical data as well as using TRASS as a *testbed* for the analysis of (more general) dynamic processes between interacting agents, ported in the context of a traffic scenario. The model example presented here follows the latter by introducing simple normative agents, who are able to learn traffic rules by observation and experience in a simple traffic scenario provided by TRASS and save them as explicitly internalized norms.

The EMIL Project

The development of this simulation model is part of the EMIL (EMergence In the Loop: Simulating the two-way dynamics of norm innovation) project funded by the EU (FP6) in the framework of the initiative "Simulating Emergent properties in Complex Systems". This project is especially focussed on modelling norm innovation processes in societies by introducing intelligent agents on different levels, which allows both modelling and observing the emergence of properties at a macro-level and their immergence into the micro-level (Castelfranchi, 1998). Furthermore, the development of a simulator for exploring and experimenting norm-innovation is part of the EMIL project (Andrighetto *et al.*, 2007).

Model Framework

Following the described modelling concept of TRASS, the static structure of a TRASS model requires a topography and the description of model agents.

Topography

The topography consists of a single *one-way road* heading from "north" to "south", possibly embedded in a simple road network.
On both sides of the lane *rectangular areas* exclusively *allowed for pedestrians* are joined, which may represent playgrounds as well as shopping zones. The three edges of each pedestrian zone that are not aligned to the road are bordered by regions marked as "wall".
The road features one to three (configurable) sections marked as *crosswalks*.

Agents

In a first version, the scenario includes the following types of agents:

- *Pedestrians* represent humans with the capability to walk with random direction and velocity, but are also able to head for a distinct position (some "point of attraction"). They are able to recognize and (if desired) to avoid collisions with other pedestrians and cars, and to change from one pedestrian zone to another by crossing the road (using a crosswalk or not).
- *Cars* are container agents which define the physical properties of (real) cars and are controlled by driver agents (one per car). *Drivers* represent humans in the role of car drivers. They are (technically) capable at least to drive on a road with variable velocity, to keep a safety distance to a car driving ahead, to stop in front of or to run over pedestrians, respectively, to recognise collisions, to observe agents (pedestrians) walking close to the road and to distinguish between "normal" road sections and crosswalks
  (Figure 4 shows the described traffic scenario including pedestrians and car/drivers with their defined perception area).
- One *supervisor* observes the complete scenario, keeps track of the (normative) behaviour of pedestrians and reacts to it (e.g. hands out sanctions)

Future versions might involve further types of agents:

- *Sign posts* to communicate deontics (e.g. permissions, obligations, forbearances) like "crosswalk ahead" for example.
- *Observers* ("cops") to replace part of the supervisor agent's job by an internal model agent

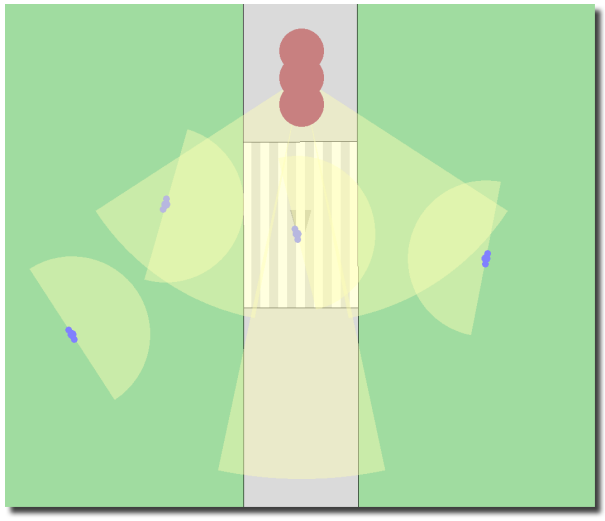

Figure 4: Screenshot of a running simulation. A car is waiting at a crosswalk while a pedestrian is crossing the road and three other pedestrians are approaching at the crosswalk. The circular sectors in front of the agents are visualizations of the perception sensors.

Model Dynamics

The dynamics of the simulation is constituted by the pedestrian agents, car driver agents and the supervisor.

Agents

- *Pedestrian*: *Initially* the pedestrians walk within the pedestrian zones randomly. From time to time a desire to reach some target within the opposite pedestrian zone arises within a pedestrian. This results in goal-oriented movement towards this target, either on the shortest path or including a detour to one of the crosswalks. In any case the road traffic is ignored. This leads to incidental collisions with cars (with the effect of sanctions for both the pedestrian and the car driver involved, depending on using a crosswalk or not).
- *Car Driver*: *Initially* the car drivers steer their cars on the road trying to avoid crashes with other cars, but ignore pedestrians. This leads to incidental collisions with pedestrians (with the effect of sanctions for both the pedestrian and the car driver involved, depending on using a crosswalk or not).

During simulation the decisions of both *pedestrians* and *car drivers* are influenced by sanctions they got and by observing the behaviour of other traffic participants (via *supervisor*). Basically, the dynamics between the micro level (individual agents) and the macro level (agent populations) follow the Weidlich/Haag approach, described in section "AI Layer".
Speaking in EMIL terms, each agent includes an ability called *norm recognizer* (CNR-ISTC, 2007), in which each agent checks whether the individual decision would comply or violate a decision, more or less accepted by the whole agent community.

Supervisor

The supervisor keeps track of the (normative) behaviour of pedestrians and reacts on the following incidents:

- Collision between pedestrian and car within a crosswalk region leads to a *hard* sanction (sanctions are scaled between 0 and 1) against the car driver. In more advanced model versions a sanction against the pedestrian regarding additional parameters (e.g. car velocity, velocity of pedestrian as gauge for attention) might be reasonable as well.
- Collision between pedestrian and car outside crosswalk regions leads to an *average* sanction against the car driver and the pedestrian as well with a grade depending on the agents velocity.
- Driving with undamped velocity within a crosswalk region and pedestrians waiting for crossing the road at the same time leads to a *mild* sanction against the car driver.

Therefore, a supervisor acts as an (normative) authority with the ability to sanction defective behaviour (according to predefined or emerged norms) of pedestrians and car drivers. This includes also the conversion from a widely accepted behaviour (*pre-norm*), emerged from the interactions between *pedestrians* and *car drivers*, into an official prescribed behaviour standard (*norm*):
When the supervisor agent notices that a number of agents beyond a threshold comply with a pre-norm, the pre-norm is converted into a norm by the supervisor. This results in harder sanctions against norm-violating behaviour of pedestrian and driver agents.

Preliminary Findings

As a result of the interactions between *pedestrians* and *car drivers* some stable behaviour of agents, which represent (emerged) *pre-norms/norms*, is observable (e.g.):

- *Pedestrians*:

  "Use crosswalks to cross the road instead of walking the shortest/direct path!"

- *Car Drivers:*

  "Do not run over pedestrians even when they do not use a crosswalk!"

  "Slow down and stop before crosswalks when pedestrians are waiting!"

Although, the development of this simple traffic scenario is in a very early state and more work on this model is necessary, experience with the modelling process and also first simulation experiments show that the universal multi-agent framework TRASS combined with the described layer-based agent modelling concept, supports a fast and structured development of simulation models in the context of a traffic scenario (e.g. description of modeller-defined robotics and AI capabilities built upon already provided agent capability on the physical layer).
Additionally, this layer-based agent modelling concept explicitly supports a step-by-step modelling strategy starting with very simple agent architectures and interactions. With reference to the introduced model example, the developed prototype and first simulation experiments support a reliable (and promising) estimation how much work to invest in modelling more complex norm emergence/immergence processes (e.g. agent architecture with */norm recognition, norm adoption, decision making/* modules) described in the EMIL proposal.

CONCLUSION

In this chapter, an approach to traffic simulation is presented that incorporates an agent model built upon mental and cognitive processes of humans. These

properties in combination with the overall agent autonomy allow for interaction between agents representing human traffic participants in multiple roles within an artificial environment.
The distinct aspects of the agent model are represented by different layers:

- Physical layer for agent appearance within the topography model,
- Robotic layer covering technical aspects of agent behaviour and
- AI layer for representing mental properties of human traffic participants.

This agent model allows for the usage within a wide range of different application fields. Besides the pure research-oriented scenario presented in this paper, several ongoing projects related to practice make use of the TRASS concept. Subject of one important project is the simulation of traffic flows within an urban area in maximum load situations, where different types of traffic participants have to be considered. This simulation – developed in cooperation with the local city administration – is set up in the context of traffic planning activities within a major reconstruction project in the city centre of Koblenz, Germany.

REFERENCES

Andrighetto, G., Conte, R., Turrini, P., & Paolucci, M. 2007. Emergence In the Loop: Simulating the two way dynamics of norm innovation. *In: Dagstuhl Seminar on Normative Multi-agent Systems, Dagstuhl, Germany*.

Castelfranchi, C. 1998. Simulating with cognitive agents. *In:* Sichman, J.S., Conte, R., & Gilbert, N. (eds), *Multi-Agent Systems and Agent-Based Simulation*. Springer, Berlin.

CNR-ISTC. 2007. *Deliverable 1.2 EMIL-M: Report of models of norms emergence, norms immergence and the 2-way dynamic*. Tech. rept.

Gilbert, Nigel, & Troitzsch, Klaus G. 2005. *Simulation for the Social Scientist. 2nd edition.* Open University Press, McGraw-Hill, Maidenhead.

Hopcroft, John, & Ullman, Jeffrey. 1979. *Introduction to Automata Theory, Languages and Computation*. 1st edn. Addison-Wesley.

Krajzewicz, D., Bonert, M., & Wagner, P. 2006. The Open Source Traffic Simulation Package SUMO. *In: RoboCup 2006 Infrastructure Simulation Competition, RoboCup, Bremen*.

Max-Neef, M. 1992. Development and human needs. *In:* Ekins, P., & Max-Neef, M. (eds), *Real-life economics: Understanding wealth creation*. Routledge, London, New York.

Norris, Gregory A., & Jager, Wander. 2004 (October). Household-Level Modeling for Sustainable Consumption. *In: Third International Workshop on Sustainable Consumption, Tokyo*.

Oliveira, D., & Bazzan, A. L. C. 2006. Traffic Lights Control with Adaptive Group Formation Based on Swarm Intelligence. *In: 5th International Workshop on Ant Colony Optimization and Swarm Intelligence, ANTS, Brussels*.

Railsback, S. F., Lytinen, S. L., & Jackson, S. K. 2006. Agent-based Simulation Platforms: Review and Development Recommendations. *SIMULATION*, **82**(9), 609–623.

Tobias, R., & Hofmann, C. 2004. Evaluation of free Java-libraries for social-scientific agent based simulation. *Journal of Artificial Societies and Social Simulation*, **7**(1).

Weidlich, W., & Haag, G. 1983. Concepts and Models of a Quantitative Sociology. The Dynamics of Interacting Populations. *Springer Series in Synergetics*, **14**.

AUTHOR BIOGRAPHIES

ULF LOTZMANN obtained his diploma degree in Computer Science from the University of Koblenz-Landau in 2006. Already during his studies he has participated in development of several simulation tools. Since 2005 he has specialized in agent-based systems mainly within the domain of traffic simulation and is developer of the TRASS system. He is also involved in several other recent projects of the research group. His e-mail address is ulf@uni-koblenz.de.

Dr. MICHAEL MÖHRING received his first and his PhD degree from the computer science faculty of the University of Koblenz-Landau and has worked in the research group for about 20 years. He developed a multi-level simulation tool called MIMOSE and participated in all the research projects of the group, currently specialising in both multi-agent simulation and in data mining; he has been teaching both information society and data mining for more than a decade. His e-mail address is moeh@uni-koblenz.de.