TOWARDS THE AGENTIFICATION OF A VIRTUAL SITUATED ENVIRONMENT FOR URBAN CROWD SIMULATION

J. Demange*, S. Galland, A. Koukam

*Multi-Agent Simulation Group
Laboratoire Systèmes et Transports
Université de Technologie de Belfort-Montbéliard, France
jonathan.demange@utbm.fr
http://www.multiagent.fr

Keywords: Crowd Simulation, Multi-Agent System, Simulation, Environment Model.

Abstract

Research works in urban and crowd simulation tend to study systems, which are more and more complex, large and realistic. In such a case, the environment of the simulated system is a key concept which represents and manages the virtual world in which the agents are living. This paper starts from a modular object-oriented environment model, JaSIM, and proposes to agentify its components to enhance the modularity of the environmental model and its execution performances.

1. Introduction

This paper is located in the domain of the simulation of urban systems with situated multi-agent systems. A multi-agent system (MAS) is a system composed of multiple interacting intelligent software agents. Multi-agent systems can be used to solve problems that are difficult or impossible for an individual agent or a monolithic system to solve. A multi-agent system is situated when the agents are immersed inside an environment. In the domain of urban system simulation, an agent is assumed to be a pedestrian, a vehicle, or any object that owns an autonomous decision-making process. The environment is then everything that is not an agent.

Three different points of view may be adopted to study the notion of environment in situated MAS: (a) the part of the system which is outside the community of the agents; (b) the medium for coordination among these agents, or (c) the running infrastructure or platform [6]. According to Weyns and al. [7], confusion on the environment's definition is mainly caused by mixing up concepts from points (a) and (b), and the infrastructure defined in point (c). Several authors refer to the environment as the logical entity of MAS in which agents and the other entities and resources are embedded. On another point of view, the notion of environment refers to the software infrastructure on which the multi-agent system is executed. Finally, it may even refer to the underlying hardware infrastructure. Odell and al. [6] distinguish between the *physical environment* and the *communication environment*. The physical environment refers to the laws, rules, constraints and policies that govern and support the physical existence of agents and the other entities. rest of this paper, only this aspect of the environment is taken.

This paper presents a modular and domain-independent agent-oriented model for the simulation of situated urban environment. Here we differ from existing models such as JaSIM [4] in the sense that these models define the environment with an object-oriented point of view, and our model is agentifying the environment. This agentification of the environment model enables load-balancing, fault tolerance and dynamic adaptation to computational resources. Moreover, it allows introducing decision-making algorithms in the environment processes. Finally, it constitutes the first step to a holonic simulation model for situated environments.

Our model is based on an object-oriented (OO) model JaSIM illustrated by Figure 2 [4]. JaSIM provides OO algorithms and data-structures required to model indoor and urban environments. The agent model corresponds to the intelligent agents simulated in the system (vehicle, pedestrian…). Each of them is associated to a physical representation in the environment: its body. This body permits to the agent to perceive its nearest environment and to act on it. Perceptions are computed by the perception generator and actions are gathered by the environment and any conflict among them is solved to obtain a valid reaction to the acts of the agents. The endogenous engine is maintaining the dynamic of the environment, which is not directly controlled by an agent.

This paper is structured as follows. Firstly, agentification of the environment is one way to improve the simulation quality according to computational and simulation constraints, which is presented in the following section. Section 3 is describing several experiments with the agentified environment. Finally, we conclude this paper.

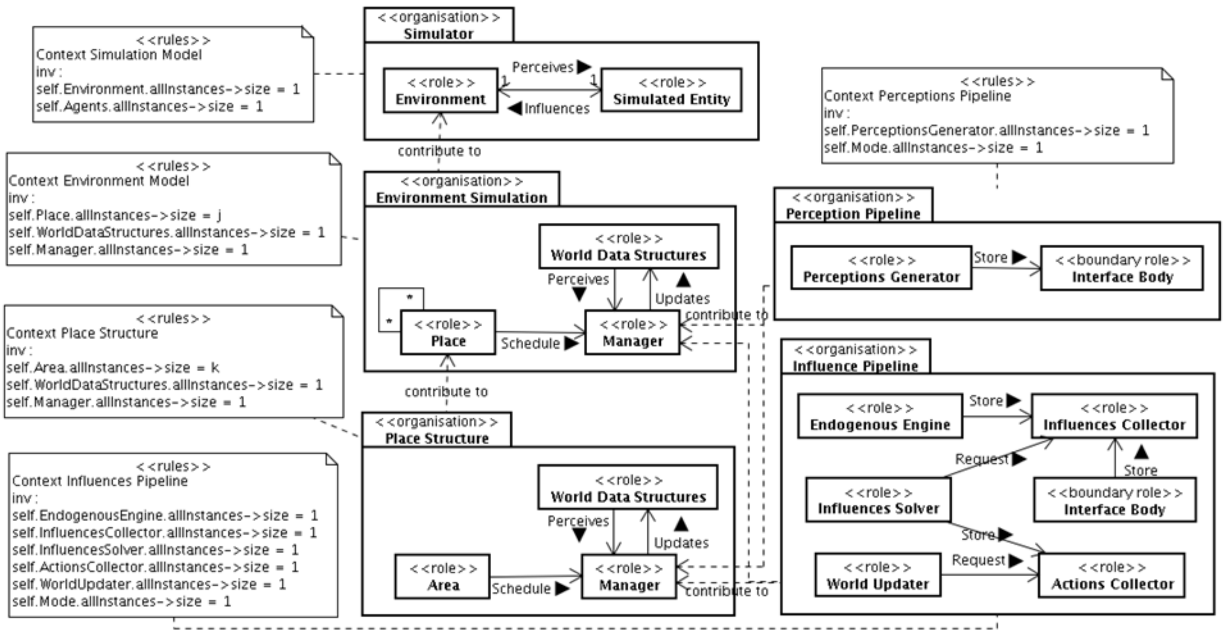

Figure 1: Organizations, Roles and Interactions for the Situated Environment

2. Agentification of the Environment

The JaSIM environment model is primary an object-oriented model. In this paper, it is agentified to provide better performances, modularity, and adaptability to the execution constraints. To help this agentification, the ASPECS [1] software engineering process is used [1]. The ASPECS process is composed of three stages: (a) the system requirements analysis, (b) the agent society design, and (c) the implementation and deployment of the agent society. This section describes the agentified environment model according to these three stages.

2.1. System Requirements Analysis

The system requirements analysis provides a complete description of the environment model. Firstly, the ontology of the system, here of the urban environment, should be defined. Figure 3 illustrates a snippet of this ontology. It focuses on the roles, or processes, in the environment. The environment is structurally decomposed into places, which are portions of the space covered by the environment and linked to the neighbor places. Each place could be in turn decomposed into a hierarchy of areas. An area is also a portion of the environment, but in the opposite of the places, they are dynamically, i.e. during the simulation, defined and built. Places are statically defined a priori. They correspond to the topological hierarchy decomposition of urban systems [3]. Areas are dynamically defined to optimize the computation costs, i.e. keeping a balance of the population density in the decomposition tree. Both places and areas are containing efficient data structures to store and localize the entities in the world: spatial quadtrees.

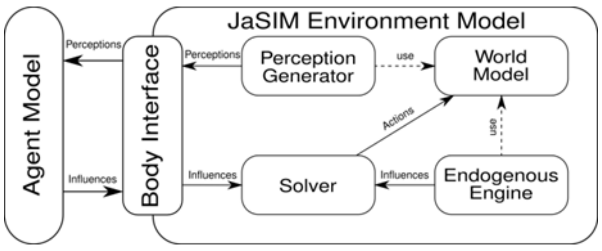

Figure 2: JaSIM Environment Architecture

Another step during the System Requirement Analysis is to determine the organizations, which are composing the environment model. These organizations are illustrated in Figure 1 by a class diagram. Packages are the organizations in the sense of the Organization Theory defined by [2]. Classes are the roles defined in the organizations. Back to our environment model, the *environment* role requires the capacities to generate perceptions for an agent and to gather the agent's influences, i.e. actions. In turn it is possible to refine the *environment model* organization into the following organizations: the *perception pipeline*, the *influence pipeline*, the *environment simulation* and the *place structure*. Perception pipeline organization is responsible of generating the collection of perceived objects for each agent. The influence pipeline collects the agent's actions, a.k.a. influences, and updates the environment's internal data-structures according to the environment reactions to the influences [4]. Finally, the organizations environment structure and place structure correspond respectively to the management of the places and of the areas. Figure 3 also illustrates some constraints on the different organizations expressed in terms of OCL formulas. Moreover, the relationships among the organizations are defined by the

[1] ASPECS: Agent-oriented Software Process for Engineering Complex Systems
http://www.aspecs.org

“contribute to” relations. The “contribute to” relation indicates that an organization can provide the same services as the role it contributes to. Section 2.2 describes the design of the agent society, which is corresponding to the model explained in this current section.

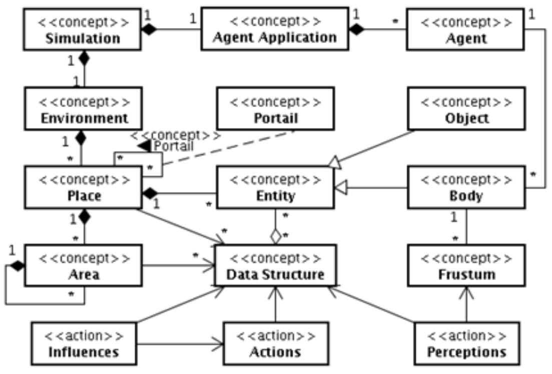

Figure 3: Snippet of the Ontology Class Diagram Centered on the Environment

2.2. Agent Society Design

This phase of ASPECS aims at designing a society of agents, whose global behavior can provide an effective solution to the problem described in the previous phase and to satisfy the associated requirements.

According to the previous section, the environment owns at least two central organizations: the simulation organization, which localizes the objects on statically defined places; and the place organization, which does similar tasks on dynamically defined areas. In this paper, we decide to agentify the places and the areas. The collection of places is fixed, but they make the decision to delegate — or not — the management to their areas. This delegation means that every task, which is responsible for the place, is now realized by the areas. These tasks are: the computation of the perceptions, the gathering and solving of the influences coming from the application agents, and the application of the environment reactions to the agent’s influences. Consequently, areas provide a part of the place behavior. Figure 4 presents the state-charts of both the place and area roles. The activities inside are similar and defined in the same way. Activities *Play places before agents* are running all the tasks mandatory before any agent is run, e.g. the computation of the agent’s perceptions. Activities *Play places after agents* are running the tasks required after all the agents have run, e.g. the solving of the conflicts among the agent’s influences. Activities *Idle* do nothing. Activity *Update area* has responsible to update the decomposition of the areas according to the current location of the objects in the environment. This cycle is based on Weyns’s cycle between agents and environment [6].

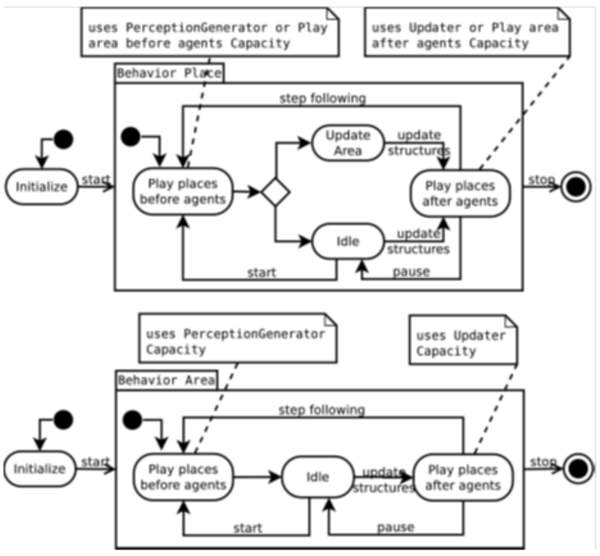

Figure 4: State-chart of place and area roles

2.3. Implementation and Deployment

This section describes required elements to implement and deploy an application in accordance to the previous analysis and design. The agent platform used is Janus[2], a general purpose agent platform commonly used in the ASPECS process. It is used to simulate the pedestrian and the urban environment. Each agent associated to a place is deployed in a thread. This choice enables to distribute the computation of the places on several execution processors. The agents associated to the areas are only deployed in dedicated thread if one is available in the operating system. Otherwise they are executed in the thread of the associated place agent. Finally, application agents are run in a central scheduler provided by the Janus platform. This scheduler ensures that the environment’s agents and the application agents are synchronized and executed in turn.

Figure 5 illustrates an example of airport environment defined in 3D. The picture shows a place of the environment, which is connected to a neighbor place by a gate, located at the background of the picture. The model contains a collection of objects like seats, trashes, trolleys, etc. In Section 3, experiments on this model are briefly described. Figure 7 and Figure 8 illustrate two other examples of experiments, which also demonstrate the application of the model presented in this paper. Figure 7 is the simulation of traffic flow and bus passengers at Belfort town. Figure 8 is a screenshot of the simulation of a metro station at Paris.

[2] http://www.janus-project.org

Figure 5: Representation of the environment in an airport case study. It shows the configuration of a hall.

3. Experiments

This section presents the performance analysis and discusses the agentified model.

3.1. Performance analysis

Benchmarks are run on GNU/Linux with a 1.2GHz dual-core processor and 2Go of RAM. The simulation runs on the Sun's Java Virtual Machine 1.6 with 1.5Go of allocated memory. The used environment is an airport composed of two halls connected by five gates together.

The first bench consists in the comparison of the object-oriented implementation of the environment model and the agent-oriented implementation. It is run with a population of 2000 pedestrians in one hall and 1000 in the other. The average computation time for one simulation step is about 25.94 seconds with the object-oriented version of JaSIM. Two cases are implemented for the agent-oriented model: in one hand the two places are managed by the same place agent; in the other hand, each place is managed by its one place agent. The first implementation takes around 41.5 seconds, and the second one 8.09 seconds for one simulation step. The agent model has a deep impact on the overall simulation performances.

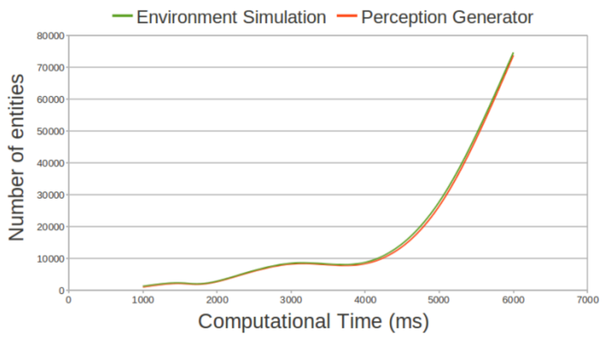

Figure 6: Execution Time of the Perception Generator and the Environment Model

The second bench shows the number of entities in the simulation. As Figure 6 illustrates, the computational time for perception generator increases strongly after 4000 entities. Two reasons explain the shape of the curve.

Firstly, the number of objects per tree node is strongly increasing. Unfortunately, the objects in a node are treated in a linear way by the simulator and by the agents. Secondly, the processor itself is too powerless after 4000 entities. Empirical tests on a faster processor enable to overshoot this limit. Note that the perception generator takes around 90% of the execution time of the environment simulation.

3.2. Discussion

The agentification of the environment model of JaSIM enables load-balancing of the computation and distribution of the data-structures. This is possible due to the intrinsic capabilities of distribution of any multi-agent platform. This agent-oriented model provides a strong advantage than the object-oriented version of JaSIM because it improves the computational performances of the simulation. Moreover, it enables the modularity of the environment model and a dynamic selection the implementation of the environmental tasks during the simulation process.

Unfortunately, the agentification of the JaSIM model still has several drawbacks: (a) the computational cost is too high — mainly, due to the perception generator — and the current model only supports one level of simulation: individuals and not groups of individuals.

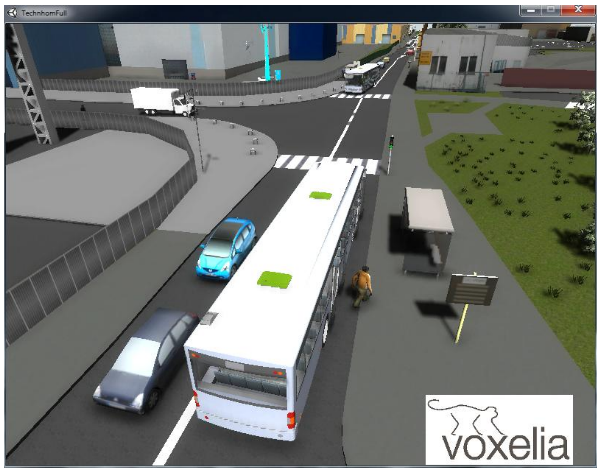

Figure 7: Simulation of Traffic Flow and Bus Passengers

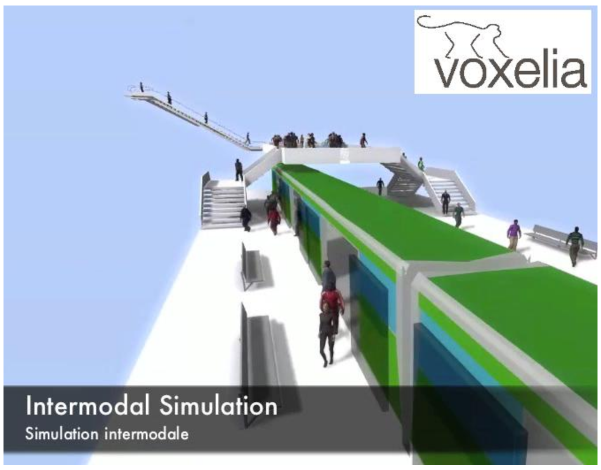

Figure 8: Simulation a Metro Station

4. Conclusions and Perspectives

This paper describes the agentification of the JaSIM components to improve the overall performances of an urban simulation. The proposed model is well adapted to distribute the simulation model. This model is constituted of places and areas. They are directly managed by "environment" agents. The area assignment to agents enables load-balancing of the simulation according to the available computational and memory resources.

To improve the support of large-scale simulations by our platform, multi-level models will be introduced both in the application agent and environment models. The multi-level should enable fast computation, but it may decrease the accuracy of simulation. These points will be studied in future works on the JaSIM platform.

Acknowledgments

The JaSIM platform and related tools are distributed as part of the SIMULATE commercial offer of the Voxelia S.A.S.[3] company, France. The authors thank Mikael GONCALVES, Renan ZEO, Olivier LAMOTTE, and Nicolas GAUD for their contributions and support.

5. References

[1] M. Cossentino, N. Gaud, V. Hilaire, S. Galland, and A. Koukam, ASPECS: an Agent-oriented Software Process for Engineering Complex Systems, *Journal of Autonomous Agents and Multi-Agent Systems*, pages 1387–2532, June 2009.

[2] M. Cossentino, N. Gaud, S. Galland, V. Hilaire, and A. Koukam. "A Holonic Metamodel for Agent-Oriented Analysis and Design." LNAI 4659, pp. 237-246, 2007.

[3] N. Farenc, An informed environment for inhabited city simulation, PhD thesis, Lausanne, 2001.

[4] S. Galland, N. Gaud, J. Demange, and A. Koukam, Environment Model for Multiagent-Based Simulation of 3D Urban Systems, In the 7th *European Workshop on Multi-Agent Systems*, Ayia Napa, Cyprus, Dec. 2009.

[5] J. Odell, H. Parunak, M. Fleisher, and S. Brueckner. Modeling Agents and their Environment. In *Agent-Oriented Software Engineering III*, volume 2585. Springer-Verlag, 2002.

[6] D. Weyns, A. Omicini, and J. Odell, Environment as a first-class abstraction in multi-agent systems. *Journal on Autonomous Agents and Multi-Agent Systems*, 14(1):5–30, Feb. 2007.

[7] D. Weyns, H. V. D. Parunak, F. Michel, T. Holvoet, and J. Ferber, Environments for Multiagent Systems State-of-the-Art and Research Challenges. In Third International Workshop *E4MAS*, volume 4389, pages 1–47. Springer, may 2006.

[3] http://www.voxelia.com