A Hybrid Model for Simulating Room Management Including Vacation Times

S. Tauböck[1], N. Popper[2], M. Bruckner[1,2], D. Wiegand[1], S. Emrich[1], S. Mesic[1]

[1] Vienna University of Technology, Vienna, Austria

[2] "die Drahtwarenhandlung" Simulation Services, Vienna, Austria

SNE Simulation Notes Europe SNE 19(3-4), 2009, 61-66, doi: 10.11128/sne.19.tn.09955

Aim of this project as a part of the simulation system <more space> is to simulate the movement of students between lecture rooms, attending their regular curriculum to implement dynamic vacation times. Major outcome is the calculated time which they need to move from a starting point (for example an auditorium) to another location (arrival point). The program is realized in the object-oriented programming language *Java* and connected to *Enterprise Dynamics*. Modeling approaches are *Cellular Automata* (CA) and discrete simulation because the literature of this approach is widely spread, and after analysis of the project, the cost-benefit calculation for this modeling was the best. One advantage especially shows that the CA can manage the dynamic behavior of the students finer and more efficient. This justifies the cost of planning and implementation of the corresponding interfaces. The goal of the project <*more-space*> is to develop software that shall support the planning phase of *University2015* - a project of the Vienna University of Technology (TU Vienna) to renovate all university buildings and to improve the existing infrastructure and the inherent processes – by determining and evaluating the (spatial) resources required and introducing a model for the room management that can simulate the usage of resources to optimize the planning of the rooms and the future "real-life" usage. The dynamic model implemented in *Enterprise Dynamics* is the main model and simulation system including the data model, process descriptions and dynamic behaviour as using of resources depending on different system or environmental dependencies.

Introduction

The project <*more-space*> was launched to develop software that shall support the planning phase of *University2015*. University2015 is a project of the Vienna University of Technology (TU Vienna) to renovate all university buildings and to improve the existing infrastructure and the inherent processes. This shall also be done by determining and evaluating the (spatial) resources required. In 2008 the working group consisting of the Research Group for Mathematical Modelling and Simulation and the Research Group for Real Estate Development and Management at the Vienna University of Technology were asked by GUT TU, co-responsible for Project Management of *Univercity 2015*, to introduce a model for the room management that can simulate the usage of resources to optimize the planning of the rooms and the future usage.

A classical discrete event simulator was chosen to develop the dynamic simulation including features like different possibilities of room selection, different management of the resources, variability of classes, class structures and number of students – only to mention a few of the features. This discrete dynamic simulator was combined with – and is guided by – methods and procedures of the real estate management like business process models. In the course of developing the first studies the working group identified two main problems that are different to tasks which are normally solved by classical discrete event models. On one hand the interfaces to data collection and booking system had to be improved and standardized. Booking features and also the representation of data was not sufficiently optimized neither to the needs of the simulation tool that has to be implemented nor to the needs of the future booking system that has to be simulated. On the other hand the buildings of the Vienna University of Technology – as the university will not move out of the city of Vienna – remain very scattered over a few districts of the city. Therefore a simulation of room booking and facility management has to implement different kinds of vacation times for students and teachers. Even more the problem of different structured buildings within the university had to be implemented and therefore event a big variety of vacation times even between classrooms within some buildings.

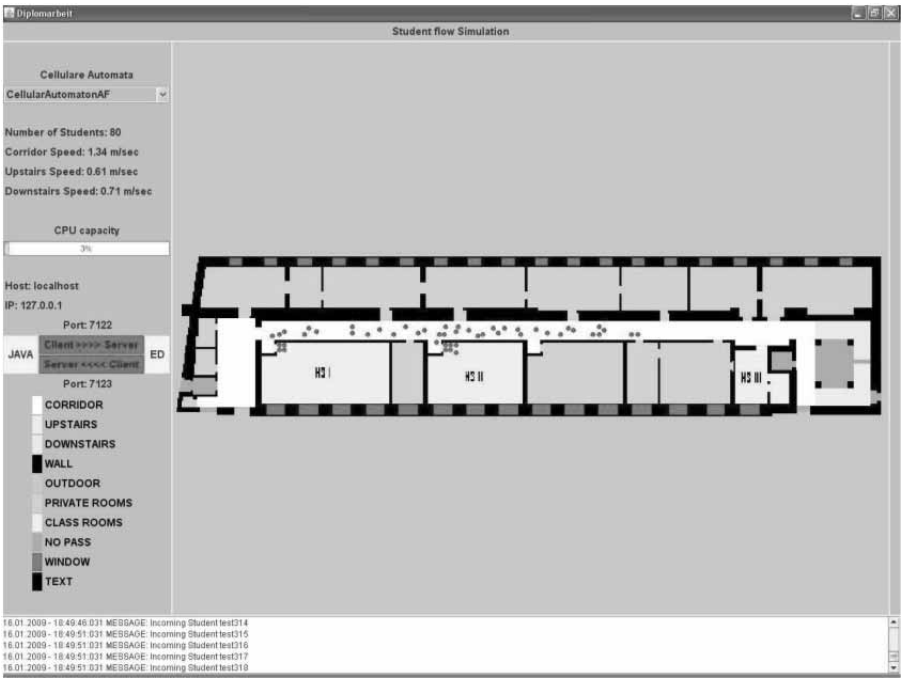

Figure 1. GUI in JAVA with CA of the

The idea was to combine the classical discrete simulation approach (because of its many advantages in questions of room booking, event queues, managing different numbers and kinds of servers) with a classical cellular automata to implement not only vacation times between buildings in different areas but also vacation times within the buildings itself.

This system is furthermore backuped with a dynamic computation of vacation and room clearance times in Taylor ED, as for static values (i.e. areas were the vacation times to not differ influenced by dynamic parameters) values can be computed in the discrete simulator itself. As a result using CAs an added value is given, as such models can – with some modifications – could be used for simulation of escape routes in emergency cases as well. [3]

Implementing the hybrid system different problems occur. For example questions of solving interface problems between Taylor ED and the CA implemented in JAVA had to be solved. Another question was to solve the problem of importing a huge amount of building data - that changes more or less often - into the Cellular Automata.

1 Hybrid model

Main elements of the hybrid model are "Students", these entities are moving through the existing structure of CA i.e. the “Building structure”. The building structure is processed on basis of CAD plans of the buildings. The students are forwarded from the simulation from ED, and processed as long as they do not reach their goal and then send back to ED. In order to make a smooth cooperation and for automation the capture of data, the following approaches have been chosen.

1.1 Representation of building structures

Building structures are based on plans and CAD Data of the University. At the moment those data is transformed manually to the needed structure described in the following part. As planned for future working processes those representation of different areas within the building should be integrated into the specifications for architects and planning offices to improve the speed of integrating different plans into the simulation system.

The building structure and related data are saved as an image file in the Portable Network graphics (.png) format and additional information is stored in two text files (.txt).

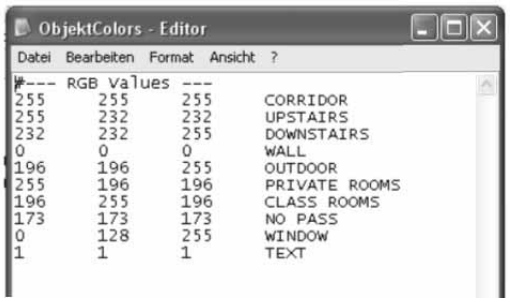

Figure 2. Text File describing the different colours for different areas within the Cellular Automata

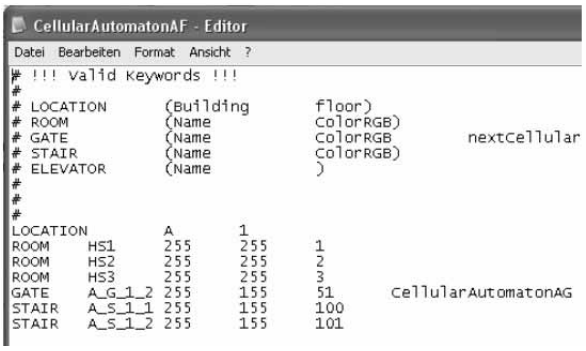

Figure 3. Text File describing special information for every instance of the Cellular Automata.

The image file includes the ground plan of the section that should be read and represents the structure of the Cellular Automata and the room arrangement (subdivision in private rooms and lecture halls) the staircases and the corridors (see Figure 1). Because elevators are not implemented in the simulation at the moment, they are inked in the colour code `NO PASS`, so as an area which is not accessible for students. Staircases are subdivided in `UPSTAIRS` and `DOWNSTAIRS`. The reason for this is because there is a difference between the average speeds a person can have upstairs or downstairs. While the upstairs velocity is approximately 0.61 m/sec you can move downstairs about 0.71 m/sec. With the colour code `CORRIDOR` in the image file the corridors are marked which you can cross with 1.34 m/sec. This is also the last of the three different speeds that are used in the simulation. Colour code `PRIVATE ROOM` are rooms such as offices, toilets, or similar marked which are not of direct benefit for students, while `CLASS ROOM` indicates space for the pure teaching (lecture rooms, laboratories, etc.).

Furthermore we have the area `WALL` which represents the walls, `OUTDOOR` represents the external areal and `TEXT`, indicates text in the graphic file that can give additional information. Plans of the buildings are at the moment separated into different areas; mostly one wing of one level is stored in one .png file. The corresponding levels or wings are connected via stairs (up- or down) or exits which are coded in a relatable way.

The first text file contains general information about the colours. Information is given for all ten different types of ground a particle can reach. Those colours have to be standard for all maps within the whole simulation and cannot be changed.

The second text file is especially for each image file and contains information about the colours that are used only in this image. This information is required to connect the single Cellular Automata, as a lot of different automata are processed on basis of one single plan. So we have to distinguish between exit of a particle between exit of a building or into a lecture room. In these cases the particle returns to the discrete Taylor ED model and leaves the CA model. The other possibility is - as mentioned - exiting one CA model via stairs or an exit to another level or wing of one building.

In this case exit IDs are used as shown in Figure 3 for identifying the following CA which the particle has to enter. For this reason a few “Key Words” are allowed. To mention `LOCATION`, `ROOM`, `GATE`, `STAIR` or `ELEVATOR`. These “key words” can have additional information about the change between the two different Cellular Automatas, for example information about the building number and the level where the CA is located or as well information about gates and stairs and which other CA they are connected to. Last but not least also information about the different lecture rooms is coded for every CA for additional information.

The whole structure of the systems is very simple. This simplification was a main goal in planning and structuring the whole simulation set. On one hand the data interface for getting information is complicated as plans change very often.

For this reason the interface should be very easy to use not only for members of the research group but also for architects or other persons working with the plans.

As mentioned above in future work processes even the producers of such plans should be able to more or less automatically implement a sufficient level of information for exporting data to the simulation system in their CAD software. In this case the best quality of information could be provided.

For this reason a mode was chosen, were data is defined not with classical matrices but with simple color codes, which are defined in one of the text files. On the other hand – as described above – the simulation has to handle a huge amount of data. Vienna University of Technology has to coordinate over 9.000 rooms in about 26 objects spread over the whole inner city of Vienna. A total floor area of about 276.000 square meters would result in a total matrix size for the CA of about 17.700.000 cells.

To handle those sizes two strategies were implemented. First of all the size of the matrices are not constant but variable to reduce the amount of cells. Second for this reason and for future parallelization reasons the plans were cut into a lot of different, small sub-planes to process them in an easy way. So the question of how to connect the different Cellular Automata instances is (and will be) of big importance.

1.2 Cellular automatas

The Cellular Automatas for different levels are connected at certain points, the gates or stairs. Superior structure of the cellular automata is a weighted undirected graph. Nodes are stairs, rooms and gates, Edges are the routes between the nodes with the distance as weight. In order to search for the shortest route across several CAs the Dijkstra algorithm is used.

So subdividing the CA with a size of about 18 Million cells into a number of “Sub-CAs” was described in the chapter above. The reason for this high amount of cells results as the size of a cell is assumed with 0.125×0.125 m – 64 cells per square meter, This assumptions had to be made to make the data connection to CAD possible on the one hand, and on the other hand to reach a sufficient grade of details for implementing the simulation system. Each student occupies 4×4 cells or 0.5×0.5 m value in normal transport areas, but the value is different for students in class rooms for example. In conclusion cell sizes are not constant, and especially the amount of cells a student needs. A reason for this can be found in the different ways the area a student needs is computed in the “real life” system.

As student space in lecture rooms is computed directly from the floor area in the specific lecture room, in laboratories the situation is of course totally different. A fixed number of working places are here to be used, no matter how big the room is on the plan or in “real life”. This is only a small example of how different the systems, measures and real situations are handled - and so have to be handled and computed within the simulation system.

For speed values of individuals in the system some information is described in Chapter 1.1 as the speed results of the different grounds particles can walk over. While on stairs the upstairs velocity is approximately 0.61 m/sec an individual can move downstairs with about 0.71 m/sec. On corridors particles can cross with 1.34 m/sec. But the ground is not the only influence on speed of individuals. Another influence is the density of particles in an area. This is well known from simulation tools for escape routes in emergency situations and was also taken and integrated in the model.

At the moment two additional points are examined. On one hand the question whether the areas between the different buildings should be implemented via CAs or not. As mentioned above in the outlines of the simulation structure this idea was integrated. At the moment results are not significant, whether the amount of computational time is reasonable. Results of tests have to be validated, at the moment the idea of combining an Agent Based simulation as it is used in [2] is examined. On the other hand interfaces to integrate clearance time for different types of lectures in different types of rooms have to be integrated into the system as well.

2 Interface between Java and TaylorED

Because the Java program for Cellular Automatas is of course only a part of the simulation system, it is necessary for data exchange to establish a connection between the two simulations. This interface was implemented with the aid of the Transmission Control Protocol and the Internet Protocol, or short TCP/IP.

2.1 Java cellular automata

Basically, the communication is divided into two different parts sending and the receiving data. Differences between these two states are the allocation of server and client functions.

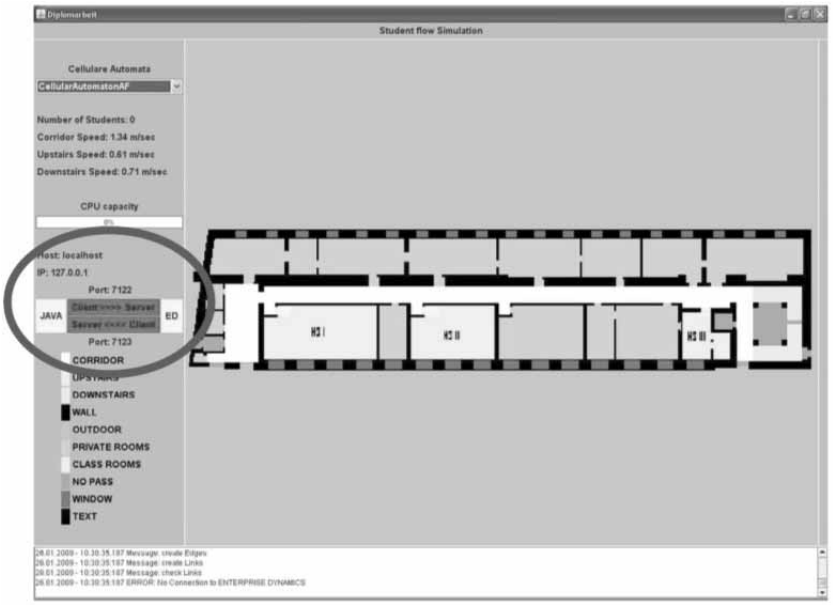

Figure 5. Representation of the Interface in JAVA GUI

In the implementation of this interface it is only a unidirectional traffic towards server possible, however, for this simulation bidirectional data traffic is required, so both Java and ED act as server and client at the same time.

The red area in the picture shows the most important information for communicating. From starting above the first three lines contains information about the Java client. In our case, both programs are running on the same computer so Java can send data to the ED server while it is sending to the host `localhost`, the loopback address `127.0.0.1` and the port 7122. Conversely Java acts as a server and receives data on port 7123. The underlying two yellow boxes with the content "JAVA" and "ED" represent the two programs, which are connected via a network. The status of the connection (red is a disturbed connection) will be symbolize by the two bars in the middle. Faultless connection is signaled with green color. The status check is realized with the aid of a well-defined message ("`PING`") by sending from Java in a parametric time interval to ED who must respond with the message ("`ALIVE`") in also a parametric time.

2.2 Discrete Event Simulation in Taylor ED

Taylor ED offers several Interfaces to communicate with external programs, for example DDE and ADO for communication with databases:

- *4d script commands* for execution of DDL functions
- Exchange of socket messages to allow communication via the *TCP/IP ports*
- *Sending and receiving emails* to communicate worldwide

For this particular simulation the communication using socket messages is used. It allows exchanging ASCII messages via a TCP/IP port. These messages are sent and received by a specified atom in ED; incoming messages trigger the so called `ONMESSAGE`-Event Handler of this atom, that contains the code sequence to be executed.

Enterprise Dynamics sends a message containing the students ID, the current position and the room the student has to move to, to the agent based model. A confirmation message ensures that the message was received; after processing this student the agent based model sends a return message that contains the ID of the student as well as the description of their movement: the current position as well as the time needed to cover the distance between former and current position.

So the interface is implemented with a few lines of code. As the system has to be called for every change for every student at the moment a kind of pulsing was integrated to collect requests.

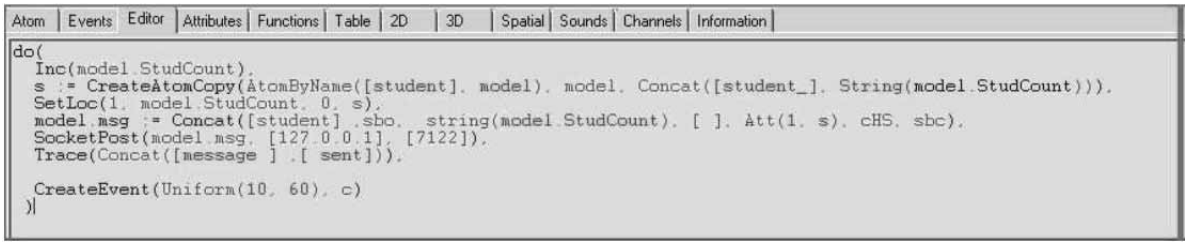

Figure 6. Screen Shot of Code sending a socket message at a student's (individuals) query for changing room

3 Conclusions

Combining different model types to implement hybrid simulation systems can solve some classical problems. On the other hand some other problems may occur. In this paper some aspects of possible advantages of such implementations were mentioned and combining positive characteristics of different model types like different possibilities of data identification, implementations of data interfaces or structural advantages were shown.

As Cellular Automatas were introduced in this simulation system to represent a model computing vacation times it was no choice in the setting of the whole project to re-implement the whole discrete system. A standard software for implementing parts like queuing, servers and so on would be a huge amount of work. So on the other handsome disadvantages like implementing interfaces for connecting the different "sub models" had to be accepted. As work on this project is in progress the quantitative results for the whole system with about 9.000 rooms and – in future – maybe about 30.000 students has to be validated.

References

[1] Breitenecker, F. and Solar, D. *Models, Methods, Experiments – Modern aspects of simulation languages*. In: Proc. 2nd European Simulation Conference, Antwerpen, 1986, SCS, San Diego, 1986, 195 - 199.

[2] Emrich S. et al., *Simulation of Influenza Epidemics with a Hybrid Model – Combining Cellular Automata and Agent Based Features*, In: Proc. ITI 2008 30th Int. Conf. on Information Technology Interfaces, 2008

[3] Tauböck S. and Breitenecker F. *Features of Discrete Event Simulation Systems for Spatial Pedestrian and Evacuation Dynamics*. In: Proc. PED 2005, 3rd Int. Conf. on Pedestrian and Evacuation Dynamics 2005, Vienna, Austria

Corresponding author: S. Tauböck,
Vienna University of Technology
Dept. for Analysis and Scientific Computing
Wiedner Hauptstrasse 8 - 10, 1040 Wien, Austria
shabnam.tauboeck+e101@tuwien.ac.at

Received & Accepted: MATHMOD 2009
Revised: September 21, 2009
Accepted: October 10, 2009