# HLA Grid based support for simulation of vascular reconstruction

Katarzyna Rycerz[1], Marian Bubak[1,2], Maciej Malawski[1], Peter M.A. Sloot[3]

[1]Institute of Computer Science, AGH, al. Mickiewicza 30,30-059 Kraków, Poland
[2]Academic Computer Centre – CYFRONET, Nawojki 11,30-950 Kraków, Poland
[3]Faculty of Sciences, Section Computational Science, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
{kzajac |bubak|malawski}@uci.agh.edu.pl, sloot@science.uva.nl
*phone:* (+48 12) 617 39 64, *fax:* (+48 12) 633 80 54

**Abstract.** The Grid paradigm for distributed computation provides an interesting framework for the medical applications. We apply the High Level Architecture (HLA) model to the vascular reconstruction application, using distributed federations on the Grid for the communication among simulation and visualization components. HLA provides advanced features that allow to build collaborative environment where surgeons can exchange their knowledge and plan their operations. To achieve efficient execution of the Grid, we introduce a Grid HLA Management System (G-HLAM) that manages HLA–based simulations running on the Grid. This is done by introducing migration mechanisms for such applications.
**Keywords** HLA, Grid, distributed interactive simulations, federate management, medical simulation

## 1 Introduction

Distributed simulations often require extensive computing resources. The Grid [4, 5] is a promising means of solving this problem as it offers the possibility to use resources which are not centrally controlled and are under different administrative policies. Simulations built with an implementation of the High Level Architecture (HLA) system [6] allow for merging geographically-distributed parts (called *federates*) of simulations (called *federations*) into a coherent entity. The High Level Architecture is explicitly designed as support for interactive distributed simulations, it provides various services required for that specific purpose, such as time management, useful for time-driven or event-driven interactive simulations. It also takes care of data distribution management and enables all application components to see the entire application data space in an efficient way. On the other hand, the HLA standard does not provide automatic setup of HLA distributed applications and there is no mechanism for migrating federates according to the dynamic changes of host loads or failures, which is essential for Grid applications. Therefore, there is a need for a system that would manage HLA-based simulations on the Grid. The Grid Services concept provides a good

starting point for building the Grid HLA Management System (G-HLAM) for that purpose, as described in [10]. In this paper we show how the distributed vascular reconstruction simulation can benefit from the Grid by using G-HLAM. Such simulations remain a great challenge in medicine [12] and there is a need for a computer infrastructure that will significantly improve their performance. We believe that the Grid is a promising environment for such requirements, since it offers the possibility of accessing computational resources that have heretofore been inaccessible.

The paper is organized as follows: in Section 2 we present background of the addressed problem, in Section 3 we describe architecture and functionality of vascular reconstruction application, in Section 4 we describe benefits of using HLA for our purposes, in Section 5 we present G-HLAM system and describe how to use it in the application. In Section 6 experimental results are presented. We conclude in Section 7.

## 2 Background

Computer simulations in medicine are very important aid, especially in surgery [15]. This is because planning vascular operation is difficult task – the physician has to locate affected vessels, analyse them and then predict effect of the operation itself. Therefore, a verification of surgeon's decisions before the operation is very useful and allows for better patients' treatment [12]. These kind of simulations are very complex, require high performance execution and near real time interaction with its results during runtime [8].

Vascular disorders such as stenosis (narrowing of the artery by the accumulation of fat and cholesterol) and aneurysms (weakness of artery's wall which causes its ballooning) seriously influence the blood flow and can cause serious diseases [9]. Therefore, it is important to improve the flow quality in the affected vessels. Vascular reconstruction is a surgical procedure which redirects the blood flow from the affected place using a grafted bridge called a bypass.

Planning such operations requires that surgeon analyse the structure of artery, localize affected areas together with the optimal place to insert a bypass. Using computer simulations surgeons decisions can be verified before the actual operation takes place [7]. In this Section we present virtual reality based medical application [1] developed at Section Computational Science University of Amsterdam. We describe application modules and we outline three scenarios of its execution: a scenario when a single user runs only one simulation, extension for many simulations and finally, the collaborative environment.

## 3 Vascular reconstruction application

The input data for the application are obtained from various medical imaging techniques like X-ray angiography, computer tomography (CT) or MRI (magnetic resonance imaging). After getting digital images of patient vessels, the data is processed by four modules of the application as described below.

**Analysis and segmentation module** The goal of the segmentation process is to automatically find the lumen border between the blood and non-blood in input images [1]. Firstly, a wave front propagator algorithm is used to find an approximation of the centerline of the vessel. Next, the data are divided into a set of 2D slices orthogonal to the centerline. Then, in each slice a contour delineating the lumen border is detected. Finally, the stack of 2D contours is combined to 3D surface model, which is then passed to 3D editing tool.

**3D editing tool** This tool allows for editing stereoscopic images and executing experimental visualisation studies on realistic arteliar geometries [3]. A user can conduct measurements, pick up a position on the artery for creating a bypass and modify its shape and size. The final stage is generation of computational mesh. The prepared arteliar geometry, including aneurysms, bifurcations, bypasses and stents is converted into a coarse or fine (depending on the user) computational mesh that is then passed to the simulation module. The module is implemented using Visualisation Toolkit (Vtk) [13].

**Simulation module** Blood flow simulation is a parallel computational solver [2] that computes pressure, velocities, and shear stresses. The simulator is based on the Lattice–Boltzmann method (LBM) – a mesoscopic approach for simulating fluid flow based on the kinetic Boltzmann equation. The simulation produces intermediate results and sends them to visualisation module. The module is written is C and parallelised using MPI.

**Visualisation and exploration module** The visualization/exploration module uses a Virtual Reality Explorer (VRE) [3] , where the patient's data is visualized as a 3D stereoscopic image together with the graphical interpretation of the simulation results. The user can then manipulate the 3D images of arteries, patient's body and blood flow structures in virtual reality. VRE combines natural input modes of context sensitive interaction by voice, hand gestures and direct manipulation of virtual 3D objects. The interactive measurement component of the VRE provides the possibility to measure quantitatively distances, angles, diameters and some other parameters characterizing 3D objects in a virtual world, where markers are building blocks of distance, angle, and linestrip measurements. A desktop version of VRE which ports the basic functionality of VRE to the normal desktops is implemented using Vtk [13].

**Workflow between application modules** The input data for the application is got from X-ray angiography, computer tomography (CT) or MRI (magnetic resonance imaging). Then, a radiologist uses the module **Analysis and segmentation module** (module A) for analising raw digital images of vessels structures. At the first step, the information of affected vessels are segmented to obtain a 3D geometrical description of the arteries of interest. Next, the segmented artery is prepared for blood flow simulations in a 3D editing tool (module B), allowing to define in- and outlets, to filter and crop part of the artery, to add a by-pass, and to generate computational meshes as input to the blood flow simulators. Then, using module C – dedicated fluid flow solver – the time dependent blood flow in the artery is computed. The resulting flow, pressure and

shear stress fields are then shown to the user using a number of visualization techniques in a Virtual Environment (VE) - module D.

**Need for a Grid** The application modules require different resources: segmentation tool requires quick access to images' database, flow simulation requires computational power, editing and visualisation tools require specific VR hardware. It is quite unlikely to find those resources at one geographical site. Additionally, if more simulations need to be run concurrently, one site with computational power may be not sufficient. The similar problem arise, when many visualisations (users) located in different places want to observe the same simulation. Therefore, the application modules usually have to be located in geographically different places and the Grid concept that facilitates access to computing resources may be a very promising approach here.

**Collaborative environment for vascular reconstruction** The collaborative environment is needed when group of surgeons from different hospitals want to discuss together interesting medical cases and exchange ideas about planning difficult operation. In Fig. 1 we show the situation for two users, but it can be easily extended for more. Each of users starts number of simulations as in previous case and becomes their owner - that means he can fully control them as in previous case. However, the user can also interact with not owned simulations in two ways. Firstly, by requesting the output data of not owned simulation – in this case, there is no problem with concurrent data access, since still only one user can change the data. Secondly, by requesting to control not owned simulation - the concurrent data access has to be controlled by communication bus. The user can: stop/pause the chosen simulation during runtime, switch between different simulations owned by him (receiving output), switch ownership of simulations (ability to stop/pause particular simulation), restart

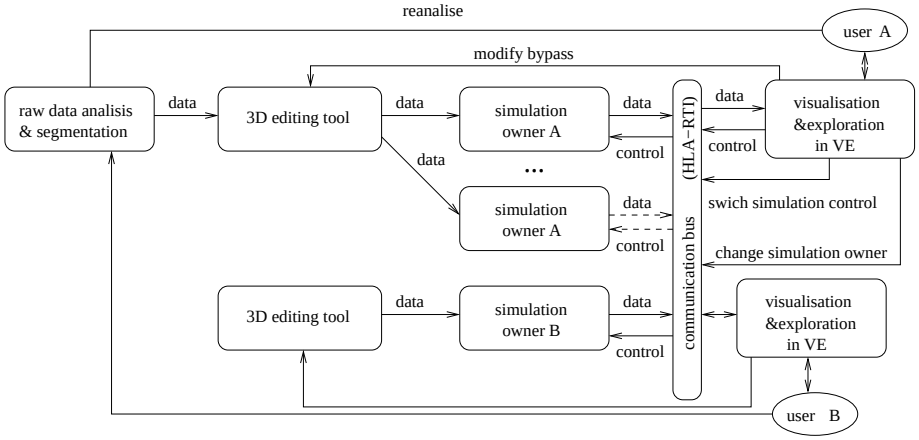


**Fig. 1.** Scenario with multiple user and multiple simulation

with different bypass and reanalyse raw data and remake segmentation.

## 4 HLA–based communication

HLA is very convenient solution for that connection of simulation modules with visualisation modules as it allows for : **building geographically distributed system** - as stated in Section 3, usually there is a need to place simulation in different place then visualisation, especially if there are more than one as in the Fig. 1, **dynamic joining of simulation/visualisation to the application** – useful when a user A in the Fig. 1 needs to start simulations one by one, or the user B joins lately and starts his own simulations, but wants to share results with others, **easy switching** between background (indicated with dashed arrows in the Fig. 1) and foreground (indicated with solid arrows) simulations - useful, when the user decides to get output data from different simulation; using HLA he can unsubscribe from data currently received and subscribe to data of the new simulation, **dynamic resigning of simulation/visualisation from application** – the user can quit the application or stop some of simulations without disturbing others, **notification** - useful for notifying the simulation about the user commands i.e. pause, cancellation, **easy switching between ownership** - useful when a user that created a simulation wants to give its control away to a different user, **concurrency control** – related to ownership – assures that only one user can control the simulation at the time.

According to scenarios presented above and types of interaction within the application, these HLA features are very useful and fill application requirement to communication infrastructure between simulations and visualisations.

## 5 The Grid-based HLA Management System

### 5.1 Overview of the system

The Grid HLA Management System (G-HLAM) supports efficient execution of HLA-based applications on top of the Open Grid Services Infrastructure as presented in [14, 10, 11].

The group of main G-HLAM services consists of: a *Broker Service* which coordinates management of the simulation, a *Performance Decision Service* which decides when the performance of any of the federates is not satisfactory and therefore migration is required, and a *Registry Service* which stores information about the location of local services. On each Grid site, supporting HLA, there are local services for performing migration commands on behalf of the *Broker Service*, as well as for monitoring federates and benchmarking. The *HLA–Speaking Service* is one of the local services interfacing federates with the G-HLAM system, using GRAM for submission of federates. A more detailed description of the *HLA-Speaking Service*, together with the GridHLAController library, which actually interfaces the application code with the system, can be found in [11].

### 5.2 Application within G-HLAM

For purposes of this case study, we have chosen two modules of described application: simulation and visualisation/exploration. G-HLAM usage is depicted in Fig. 2. As described above, and shown in the figure, the case study application

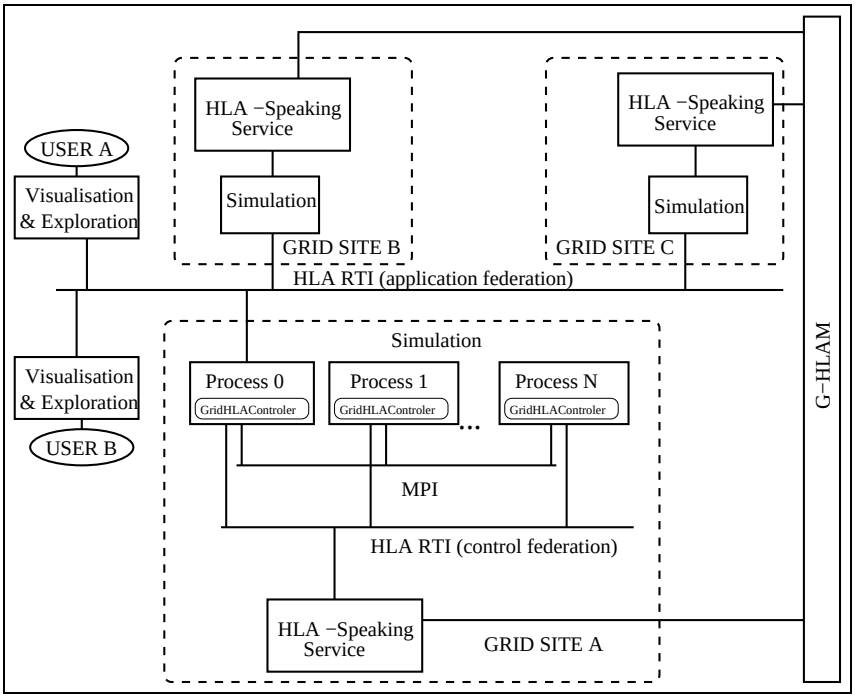


**Fig. 2.** G-HLAM for the medical application

consists of two kinds of modules: an MPI Lattice Boltzman simulation module and an integrated visualization–exploration module. For simplicity, in Fig.2 we have presented an example with three simulations and two visualisations, but it can be easily extended. Moreover, we outline details of simulation only on Grid site A, however the same structure applies to site B and C, where simulation is shown as a one box.

As can be seen, there are two kinds of HLA federations : one application federation and $N$ internal G-HLAM control federations, where $N$ is a number of Grid sites used by simulation modules.

**Application federation.** Communication between application components is performed by HLA, so in the Fig.2 all simulations and visualisations are connected to application federation. Simulations are connected to application federation only by their MPI root processes that are responsible for sending data

to visualisation. This federation is specific to the application and is designed by its developer.

**G-HLAM Control Federations.** On each site, there is a *HLA–Speaking Service* for controlling simulation by G-HLAM The service is an interface between G-HLAM and application federates. It starts the application processes on its site and sends them saving/restoring request from G-HLAM when there is a need for migration. Each site has its own control federation for communicating with the *HLA–Speaking service* residing on this site and all simulation processes join respective control federations by GridHLAController library. The library is an interface between user code and G-HLAM, it passes saving and restoring request to the user code and assures that all user processes behave correctly, when one of them is migrated.

## 6 Results

In this Section we present results from two experiments using G-HLAM for the prototype vascular reconstruction application. The prototype consisted of two types of modules communicating with HLA: *simulation modules* (MPI parallel simulation) and *visualization–receiver modules* (responsible for receiving data from simulation). As above, we use the last scenario of the application (collaborative environment) since other scenarios are just specific cases thereof. First, we show how migration time scales along with the number of simulation and visualization–receiver modules in the collaborative environment. Next, we show how migration improves performance from the point of view of the user - i.e. how sending output data from the simulation changes after migration if the partial simulation results are actually watched by someone. The experiments were performed on the DutchGrid DAS2 testbed infrastructure and at CYFRONET, Krakow, as shown in Tab. 1.

| **Operating System** | Red Hat Enterprise Linux Advanced Server, version 3 | | |
|---|---|---|---|
| **Network** | 10 Gbps (DAS2) + 155 Mbps (DAS2-Cyfronet) | | |
| **Role** | **Name** | **CPU** | **RAM** |
| Migration source | DAS2 Nikhef | Pentium III 1 GHz | 1 GB |
| Migration destination | DAS2 Leiden | Pentium III 1 GHz | 2 GB |
| other visualizations and simulations | DAS2 Delft | Pentium III 2GHz | 2 GB |
| | DAS2 Utrecht | Pentium III 1 GHz | 1 GB |
| | DAS2 Vrije | Pentium III 1 GHz | 2 GB |
| RTIexec | Cyf Krakow | Xeon 2.4 GHz | 1 GB |

**Table 1.** Grid testbed infrastructure

In the first experiment we ran four simulations (each containing 12 MPI processes)in our collaborative environment and the number of visualization–receivers varied from 3 to 22. We assigned visualization–receivers for each simulation in the most balanced way possible, so that each simulation had an equal

number of visualisation-receivers collecting its data (exact to the remainder of dividing the number of visualizations and the number of simulations). We then migrated one of the simulations.

In the second experiment one simulation was migrated. The number of visualization–receivers was fixed and equal 25.

We observed that the type of module (simulation or visualization–receiver) does not have any impact on migration time. This is because only one (master) process of the parallel simulation participates in the application federation and its role is equal to a single visualisation–receiver process. Therefore, we plot migration time as a function of all federates in the application federation regardless of their type.

In our experiments, in each step, the simulation produces 52000 velocity vectors of simulated blood flow in 3D space. For our experiments we used GT v3.2 and HLA RTI 1.3v5. The results were obtained as an average from 10 measurements. The error bars indicate estimated standard deviation.

**Migration Overhead** From our results it can be seen that migration overhead is linear with respect to the number of federates in the application federation (modules in the collaborative environment) when using reliable transport in the HLA RTI implementation. Basing on our other experiments performed with migration of $N$–body simulation [11] we can say that this is probably because the most time consuming operation is the rejoining application federation. The federate has to open TCP connections to all other federates in the federation. In our approximating linear function $A = 2.7$, $\sigma_A = 0.8$ and $B = 87.0$, $\sigma_B = 13$. To check if the approximation is appropriate, we performed a $\chi^2$ test for 11 degrees of freedom. For our data $\chi^2 = 8.7$, which is lower than the critical value 17.2 for significance level 0.1.

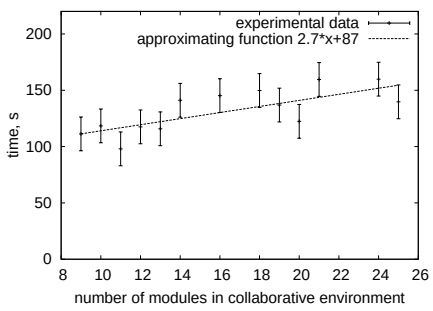


**Fig. 3.** Total migration time as a function of the number of modules in collaborative environment federates

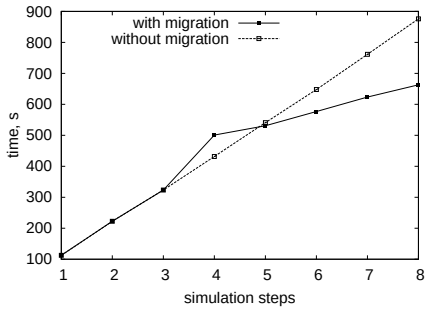


**Fig. 4.** Impact of migration on simulation performance within collaborative environment

**Impact of migration on performance of simulation within the collaborative environment** In this experiment we show how migration can improve the efficiency of simulation execution when its results are sent on–line to many

users. the bandwidth available for testing was broad (10Gbps), so communication did not play an important role and calculations were the most time–consuming part of the execution. In order to create conditions in which migration would be useful, we increased the load of the Grid site where the simulation was executed (cluster in Amsterdam) by submitting non–related, computationally–intensive jobs. Next, we imitated a Resource Broker and migrated the simulation to another site which was not overloaded (cluster in Leiden). The experiments were performed at night in order to avoid interference with other users and repeated 10 times to check if they were reproducible. Fig. 4 shows the time as a function of the number of interactive steps with a human in the loop (for the first 8 steps). At each step, the simulation calculates data and sends it to the 25 visualization–receivers modules using HLA. The dashed line shows the execution time of the simulation steps in the case when the simulation was not migrated. In this situation it is better to spend some time on migration to another site, from where the response time is shorter, as shown by the solid line in the figure. Fig. 4 shows that the human can gain access time between steps 4 and 5, independently of the time lost on migration (performed between steps 3 and 4).

## 7 Summary and Conclusions

This paper presented how collaborative environment for supporting planning vascular reconstruction operations can benefit from both HLA standard and the Grid environment. In the paper we have shown that HLA provides mechanisms to build advanced collaborative environments. The most important features of HLA include synchronisation mechanisms, data distribution management and ownership management. Also, HLA allows for building geographically distributed simulation systems in a relatively easy way.

However, vascular reconstruction application needs also to take advantage from the Grid and distributed resources provided by it. Therefore, we show how G-HLAM system can be used to manage efficient execution of HLA–based application on the Grid environment. This is done by migration of badly performing parts of simulations to a better location in order to reduce computation and communication time and effectively improving the overall performance.

**Acknowledgments** The authors would like to thank Piotr Nowakowski for useful remarks. This research is partly funded by the EU IST Project CoreGRID, the Polish State Committee for Scientific Research SPUB-M grant, and by the Dutch Virtual Laboratory for e-Science project (www.vl-e.nl).

## References

1. L. Abrahamyan, J.A. Schaap, A.G. Hoekstra, D.P. Shamonin, F.M.A. Box, R.J. van der Geest, J.H.C. Reiber, and P.M.A. Sloot. A Problem Solving Environment for Image-Based Computational Hemodynamics. In V.S. Sunderam, G.D. van Albada, P.M.A. Sloot, and J.J. Dongarra, editors, *Computational Science - ICCS*

*2005: 5th International Conference, Atlanta, GA, USA, Proceedings, Part I*, volume 3514 of *Lecture Notes in Computer Science*, pages 287–294, Berlin, Heidelberg, May 2005. Springer.

2. A.G. Artoli, A.M. Hoekstra and P.M.A. Sloot. Simulation of a systolic cycle in a realistic artery with the Lattice Boltzmann BGK method. *Int. J. Mod. Phys. B*, 17:95–98, 2003.
3. R.G. Belleman. *Interactive Exploration in Virtual Environments.* PhD thesis, University of Amsterdam, Amsterdam, The Netherlands, April 2003. Promotor: Prof. Dr. P.M.A. Sloot.
4. I. Foster. What is the Grid? A three checkpoints list. *GridToday Daily News And Information For The Global Grid Community*, 1(6), July 2002.
5. I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. *Open Grid Service Infrastructure WG, Global Grid Forum*, June 2002. `http://www.globus.org/research/papers.html`.
6. HLA specification. `http://www.sisostds.org/stdsdev/hla/`.
7. Kevin Montgomery, Michael Stephanides, Stephen Schendel, and Muriel Ross. A case study using the virtual environment for reconstructive surgery. In *VIS '98: Proceedings of the conference on Visualization '98*, pages 431–434, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.
8. Steven Pieper, Joseph Rosen, and David Zeltzer. Interactive graphics for plastic surgery: a task-level analysis and implementation. In *SI3D '92: Proceedings of the 1992 symposium on Interactive 3D graphics*, pages 127–134, New York, NY, USA, 1992. ACM Press.
9. P.M. Rothwell, J. Slattery, and C.P. Warlow. A Systematic Comparison of the Risks of Stroke and Death Due to Carotid Endarterectomy for Symptomatic and Asymptomatic Stenosis. *Stroke*, 27(2):266–269, 1996.
10. K. Rycerz, M. Bubak, M. Malawski, and P.M.A. Sloot. A Framework for HLA-Based Interactive Simulations on the Grid. *SIMULATION*, 81(1):67–76, 2005.
11. K. Rycerz, M. Bubak, M. Malawski and P. Sloot. A Grid Service for Management of Multiple HLA Federate Processes. submitted to PPAM conference, Poznan, 2005.
12. P.M.A. Sloot, A. Tirado-Ramos, A.G. Hoekstra, and M. Bubak. Interactive Grid Environment for Non-Invasive Vascular Reconstruction. In *2nd International Workshop on Biomedical Computations on the Grid (BioGrid'04), in conjunction with Fourth IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid2004).* IEEE, April 2004.
13. Visualisation toolkit home page. http://public.kitware.com/VTK/.
14. K. Zając, M. Bubak, M. Malawski, and P.M.A Sloot. Towards a Grid Management System for HLA-Based Interactive Simulations. In S.J. Turner and S.J.E. Taylor, editor, *Proceedings Seventh IEEE International Symposium on Distributed Simulation and Real Time Applications (DS-RT 2003)*, pages 4–11, Delft, The Netherlands, October 2003. IEEE Computer Society.
15. Z. Zhao. *An agent based architecture for constructing Interactive Simulation Systems.* PhD thesis, University of Amsterdam, Amsterdam, The Netherlands, December 2004. Promotor: Prof. Dr. P.M.A. Sloot, Co-promotor: Dr. G.D. van Albada.