Collaborative Computational Steering: Interactive collaborative design of ventilation and illumination of operating theatres

A Borrmann[1], P Wenisch[1], M Egger[1], C van Treeck[1], E Rank[1]

[1] Computation in Engineering, Technische Universität München, 80290 Munich, Germany
borrmann@bv.tum.de

Abstract. The layout of the ventilation system of an operating theatre is a crucial matter, because numerous infections occurring in hospitals today are directly related to air flow induced by the air conditioning system. The interactive fluid simulator developed by our group allows the engineer responsible for the HVAC design of an operating theatre to interactively determine the optimum shape and position of the inlets and outlets of the air conditioning system, as well as the ideal position for the operating equipment which acts as an obstacle in the fluid stream. In the case of the operating lamps we are faced with a typical multi-disciplinary optimization task, since their position is also essential for an optimum illumination of the patient undergoing surgery. We have therefore developed a software platform which allows engineers from different domains to come together in a virtual session and collaboratively modify the layout of a shared geometric model. At the same time they can use their specific interactive simulation in order to obtain results which are necessary for their decisions - in the scenario depicted here a fluid simulator and an illumination analysis tool. The paper discusses in detail the interactive fluid simulator and the software platform enabling multi-disciplinary *Collaborative Computational Steering*.

1 Introduction

Computer-based simulation and analysis tools play an important role in today's building engineering workflow. Apart from the structural analysis domain where programs based on the Finite Element Method (FEM) are normally used, numerical simulations are also employed increasingly in other domains such as Heating, Ventilation & Air-Conditioning (HVAC), for example.

The classical method of conducting simulations during the design and engineering stages comprises the steps of pre-processing, simulation and post-processing. The pre-processing step involves preparing all the geometric and non-geometric information for the simulation. This usually means the semi-manual meshing of the simulation domain. After the simulation has been run with this well-prepared input, its output is post-processed, i.e. the generated datasets are analyzed and visualized in a way that makes them easy to interpret for the engineer. If he is not satisfied with the simulation results he will modify the input parameters accordingly and start another run of the simulation. This leads to long optimization cycles and often to a sub-optimal design.

Thanks to high-performance computing resources available today, the realization of interactive simulations, where all three steps are combined in a single tool that allows both the modification of the simulation parameters and the visualization of the simulation results, becomes a feasible prospect. The concept of using an immediately reacting simulation is described by the term "Computational Steering", which was introduced by Liere et al. (1997). Because the design of complex products or buildings is often a multidisciplinary optimization task, the

support of collaborating engineers from different domains is a very important aspect. The combination of these two aspects leads to the concept of "Collaborative Computational Steering", which has been implemented by means of a software platform that is presented in this paper. Further software prototypes for interactive fluid simulations and interactive illumination analysis are also presented as well as a possible application for the design of an operating theatre.

2 The interactive fluid simulator

Over the past few years, our group has developed an interactively steerable fluid simulator which allows the user to modify simulation parameters online during a running simulation which allows for tracking the impact of these modifications on the fluid flow (Kühner, 2003; Wenisch, 2008). A special feature is the possibility of modifying geometric boundary conditions, i.e. to insert, relocate and remove obstacles and to reposition inlets and outlets (van Treeck et al, 2008). The user can also change non-geometric simulation parameters, such as the inlet flow velocity or the outlet pressure, dynamically.
The simulation kernel we have developed is based on the lattice-Boltzmann method (LBM) which has emerged as a complementary technique for the computation of fluid flow phenomena (Zanettti & McNamara 1988, Krafczyk 2001, van Treeck 2004). Traditional simulation methods solve the Navier-Stokes equations numerically by discretizing the non-linear partial differential equations applying finite volume or finite difference techniques, for instance (Ferziger & Peric, 2001). By contrast, the LBM represents a bottom-up approach which starts with a discrete microscopic model preserving the desired quantities, such as mass and momentum, by construction in order to obtain hydrodynamic behavior on a macroscopic scale corresponding to the incompressible Navier-Stokes equations.
The LBM satisfies the demands of Computational Steering perfectly, especially in terms of the possibilities for an interactive modification of geometric boundary conditions, because it uses Cartesian grids, an explicit time-stepping scheme and a marker-and-cell-like approach to define boundaries and obstacles.
However, there is a certain trade-off between interactivity and precision: an interactive simulation requires fast computation and accordingly a coarser grid which results in reduced precision. Nevertheless, the results of the physically simplified interactive simulations are able to show basic trends and thereby serve as a good basis for fundamental design decisions. In our concept, the engineer will perform a detailed non-interactive simulation after the basic decisions have been made for generating results with the desired accuracy (see Section 5).

3 Application in operating theatre ventilation

The layout of the ventilation system of an operating theatre is a crucial matter, because numerous infections occurring in hospitals today are directly related to air flow induced by the air conditioning system. Special care has to be taken to ensure that the patient is not exposed to air possibly infected with viruses or bacteria (Carey, 2003). Commercially available, non-interactive CFD software is currently used to simulate the air flow in the projected operating theatre or hospital ward (Chow & Yang, 2003; Tsou et al. 2004), calling for a separation of the engineering steps *pre-processing*, *simulation* and *post-processing*, as discussed in Section 1.
If instead the HVAC engineer is using the interactive fluid simulator discussed above, he is able to interactively determine the optimal layout for the room ventilation by modifying the

inlet(s) and outlets of the air conditioning with respect to position, size and inflow velocity (Wenisch, 2008). The LBM-based fluid simulator running in the background reacts directly to the changed boundary conditions, thus enabling the HVAC engineer to observe the impact of the modifications on the resulting air flow within the operating room. Figure 1 shows a screenshot of the client program used for steering the simulation.

4 Collaborative Computational Steering

Besides the integration of design and simulation facilities, another major research challenge is the development of methods and tools to improve collaboration between the engineers involved in the design process, regardless of whether they are from the same or from different engineering disciplines. Due to on-going specialization in all engineering domains, collaboration will play an even more dominating role in the future.
One important aspect of designing a software platform that shall support collaborating engineers is the fact that specialists from different domains make different demands on the application they are using while taking part in the collaborative session. Technologically simple approaches like desktop or application sharing are therefore often unsuitable.
This is also the case in our demonstration scenario devised for optimizing the design of an operating theatre in terms of (1) the fluid flow resulting from air-conditioning and ventilation, and (2) the illumination of the operating table. Because the different design goals might be prove to be conflicting as regards the positioning of the operating lamps, for example, we envision that the HVAC engineer and the lighting engineer join forces in a virtual session to find a solution that is suitable for both domains. During that session, the HVAC engineer will use an interactive fluid simulator to assess the fluid flow (Fig.1, left-hand side), whereas the lighting engineer uses an interactive rendering software to perform the illumination analysis (Fig.1, right-hand side). To realize such multi-disciplinary Collaborative Steering we have developed the CoCoS platform which will be presented in the next sections.

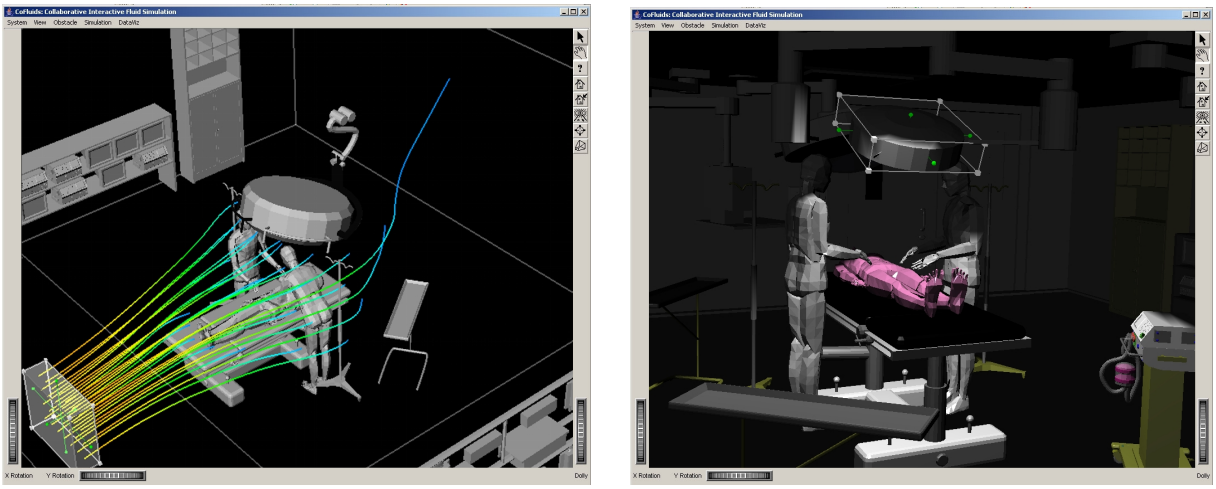

Fig. 1. The client applications of the collaborating engineers from the HVAC and the lighting domain, respectively. While the client of the HVAC engineer (left-hand side) depicts the results of the fluid simulation by means of streamlines, the client of the lighting engineer (right-hand side) shows the illumination analysis obtained from the interactive rendering. Whereas the geometric model is identical for all participants, the view point and the view direction can be chosen independently as well as the source of simulation data. The geometry setup was taken from a commercially distributed VRML scenario available from www.turbosquid.com .

4.1 The CoCoS platform

The CoCoS platform (Borrmann, 2007) was designed as a distributed multi-user application. Thus, everyone participating in a collaborative session can work interactively with this application by means of an individually configurable human-machine interface. This approach has two major advantages: on the one hand, the visual interface can range from desktop monitors to high-end visualization equipment, such as Virtual Reality environments. On the other hand, it enables each participant's viewing and interaction facilities to remain completely independent of one another. In this way, it is possible to avoid typical phenomena which are familiar from collaborative environments based on shared desktop or shared application approaches, like “mouse wars”, or sickness caused by remotely controlled viewing. The basic architecture of the collaboration platform consists of the central collaboration server, an arbitrary number of simulation servers and an arbitrary number of clients. Figure 2 shows these components and the communication paths between them. Each of the components can be run on different machines.
The modularity of the platform provides a great amount of flexibility: First of all, clients can join and leave the collaborative session at any time. This also holds true for the simulation servers: they can be integrated into the platform when a certain simulation facility is required and released when the work is done. So the collaborating engineers can start to discuss a problem within the collaborative environment without using a simulation, for example. Moreover, different clients can receive data from different simulation servers, or not receive any simulation data at all.

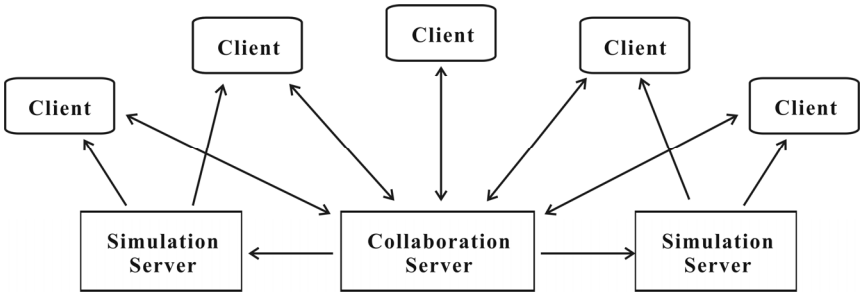

Fig. 2. The CoCoS architecture consists of the central collaboration server, an arbitrary number of simulation servers and an arbitrary number of clients. Not every client has to receive simulation data, and different clients can receive data from different simulation servers.

The CoCoS platform aims at supporting geometry-driven, multi-disciplinary collaboration between engineers. Rather than enforcing the search for an optimum solution, however, the system leaves the responsibility for design decisions in the hands of the engineers. Since formalizing design goals of more general engineering disciplines, such as illumination and ventilation design, is a very complex field, the conflicting decisions made by the designers have to be detected and resolved manually.

4.2 The shared model

The shared model forms the heart of CoCoS. It is managed by the central collaboration server and reflects the current state of the geometrical and non-geometrical boundary conditions, as well as the start-up and steering parameters of the simulations.
In modern data exchange theory and practice, a so-called “product model” is used to structure the information pertaining to the common subject of interest (Eastman, 1999). In the *Architecture Engineering Construction* (AEC) sector, product models are also referred to as Building

Information Models (BIMs). A BIM represents the building in terms of its semantic entities, e.g. a *wall*, a *window*, a *ceiling*, and the relations between them, e.g. *a window fills the void in a wall*. The geometry of the building components is either generated from the objects' attributes and relations ("attribute-driven geometry") or attached to it as an explicit shape description. Accordingly, the core of a product model is formed by the semantics.
Generating geometry from attributes has several advantages, for instance when different representations of a physical object are required in 2D plans and 3D renderings. It is important to keep in mind that major parts of the AEC industry are still highly dependent on 2D plans. However, for fluid simulations and illumination analysis, a precise three-dimensional description of the shape of the building components is usually needed. To this end, we follow a different approach for the shared model used in CoCoS than the one familiar from product modeling. In our approach, the explicit shape description of the physical objects forms the core of the model.
In building engineering, geometry is one of the most important aspects of modeling the product on the computer. Geometry marks the start of the design process: functionality, usability and aesthetics are defined by the shape of the product. For simulation tasks like structural analysis, computational fluid dynamics or illumination analysis, it is the major source for defining boundaries and boundary conditions.

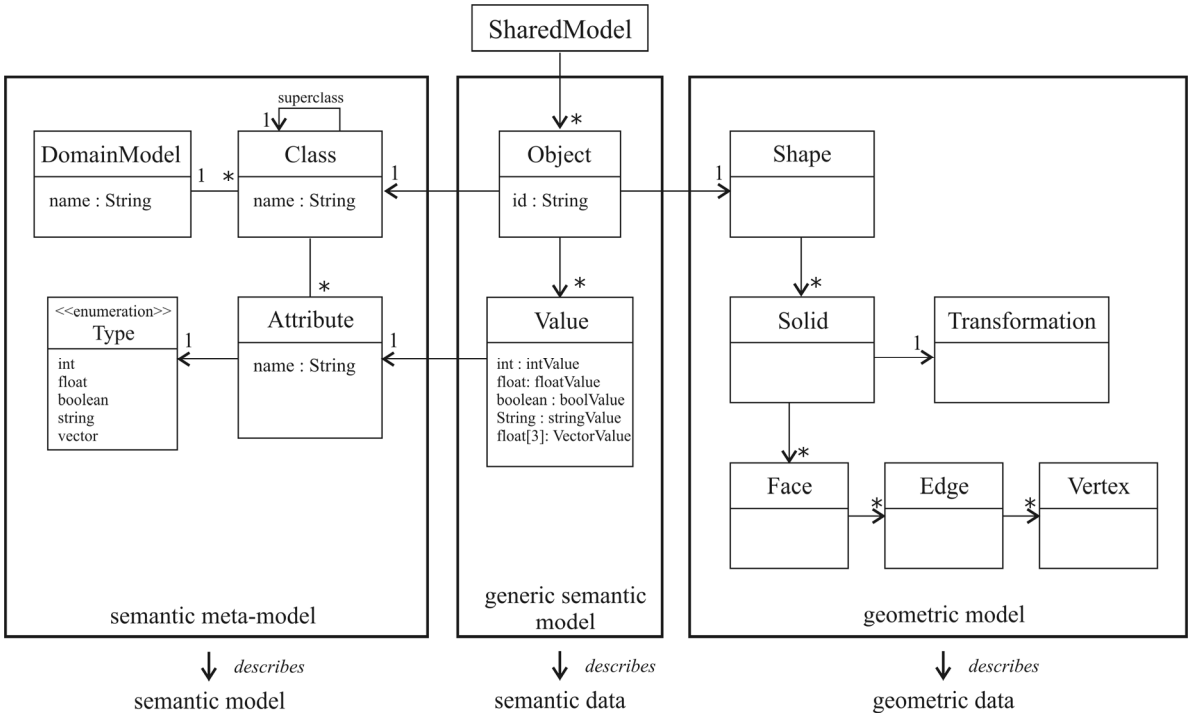

Fig. 3. The centrally managed, shared model combining geometric and semantic data. The explicitly available meta-model on the left describes the structure of the semantic model of a domain (domain model). Because CoCoS was conceived for inter-disciplinary collaboration, it can be used for managing multiple domain models. The instance of a domain model holds the semantic data and is stored and accessed using the generic classes in the middle. The shape and the position of an object is described by means of the classes on the right-hand side.

While it cannot be expected that product models for the various simulation types will become fully standardized in the near future, there are only a limited number of geometric representations employed in practice: the constructive solid geometry (CSG) approach based on the concatenation of Boolean operations on simple shapes, and several boundary-representation (B-rep) models including those which represent the surface of a solid by means of plane facets and those which use parameterized patches (Mäntylä, 1988).

However, there is usually some additional data besides geometry (such as machine and material parameters) to be shared among the participating engineers. Accordingly, a purely geometric model would not fulfill the demands of collaborative engineering. We therefore decided to use a hybrid solution with the ability to attach non-geometric information to geometric objects in a generic way (Figure 3). The emphasis of the hybrid model is placed on the geometric part: the geometry of an object is the information that all client applications must be able to interpret.
The geometry of the shared model is described by means of a classical BRep data structure based on a simple vertex-edge-face graph, as depicted on the right-hand side of Figure 3. To associate non-geometric information with a geometric object, it is possible to assign a class from the domain model to the latter. This class contains the attributes required to hold the respective non-geometric data. By selecting the simulation resource in use, the participant chooses the domain model and the domain classes available to him. By way of an example, Figure 4 shows the domain model of the fluid simulator discussed in Section 2.
All users of the same simulation server share the semantic view on a geometric object, i.e. the class associated with it and the values of its attributes. However, participants using another simulation will assign a class from their own domain model to the same geometric object. The operating lamps in the application scenario, for example, are *light sources* in the context of the illumination analysis but *obstacles* in the context of the fluid simulation (Fig. 5). For this reason we have incorporated the possibility to associate multiple classes from different domain models with a single geometric object.

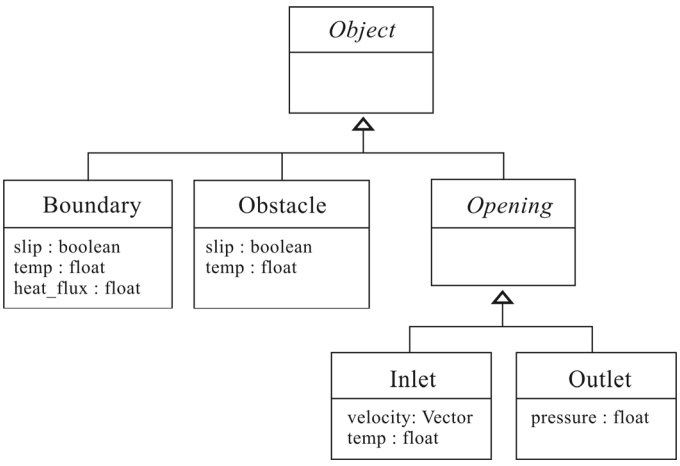

Fig. 4. The domain model CFD which is used by the interactive fluid simulator.

To further increase the flexibility of the CoCoS platform with respect to the domain models available, a meta-model has been integrated into the centrally managed model (Fig. 3, left-hand side). The meta-model supports three major aspects of the object-oriented modeling paradigm (Booch, 1994): encapsulation - by providing classes as containers for attributes, inheritance - by making it possible to derive a class from a super-class, and the option of declaring a class as abstract to prevent it from being instantiated.
Using the meta-model it is possible to adapt the domain model easily to the needs of a specific simulation without any re-compilation of the collaboration server or the clients. This offers considerable advantages especially with respect to the on-going development of numerical simulators and the implied modifications to the structure of their parameters. The same is true for simulators which comply with the basic paradigm of Computational Steering but whose start-up and steering parameters are not known a-priori. They can be integrated into the platform without the need for re-compilation in a plug'n'play-like manner. As soon as

a simulation server joins the platform, it defines the domain model holding its specific start-up and steering parameters. The model is then immediately accessible to all clients within the CoCoS platform. Fig. 5 accordingly shows an instance of the meta-model which is communicated from the CFD simulation server to the collaboration server using the classes of the meta-model. Global parameters of a simulation which are not attached to a specific geometric object such as the viscosity of the fluid are similarly modeled using the meta-model (Fig. 6).

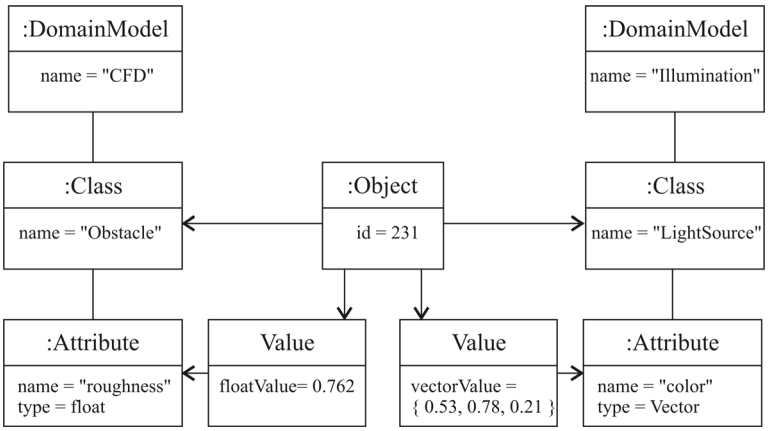

Fig. 5. Using CoCoS, it is possible to assign multiple classes from different domain models to a single geometric object. The example shows object 231, to which the class *Obstacle* from the *CFD* domain as well as the class *LightSource* from the *Illumination* domain have been assigned. Note that all objects in the diagram are instances of classes shown in Fig. 3 and that only parts of the domain models are shown.

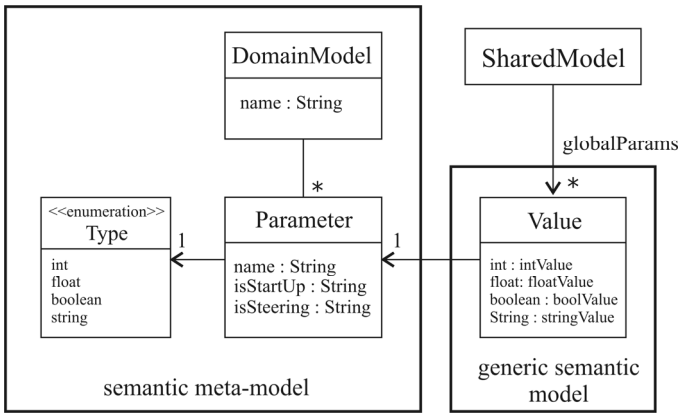

Fig. 6. Global parameters of a simulator which are not attached to a specific geometric object are also managed within the central model. Their structure is defined in the same way as that of semantic attributes of geometric objects using the meta-model on the left.

A CoCoS client uses a generic interface (Fig. 3, middle) to query a geometric object for its non-geometric data. Typically, it will display them in a generic manner and provide means for modifications. In most cases, this will be simple tables with the name of an attribute on the left-hand side, and the editable value on the right-hand side.

Why do we not use simple name-value pairs to describe simulation parameters? Name-value pairs do not involve typing of any kind. Thus, a user can add any pair regardless of whether its name can be interpreted by the simulation facility or the data type of the value is correct. By contrast, strong typing as provided by domain models prevents the user from erroneous

inputs. The explicitly available meta-model allows all components of the CoCoS platform to know which parameters exist, and what type they are.

4.3 The collaboration server

The central collaboration server is the most important component of the CoCoS platform. It has to perform the following tasks:

- management of the shared model,
- management of users, their current view points and directions, their roles and rights,
- concurrency control and
- management of the simulation servers, their locations, their start-up and steering parameters, as well as the structure of the simulation data they are producing.

Each of these tasks corresponds to a dedicated module in the collaboration server, as shown in Figure 7.

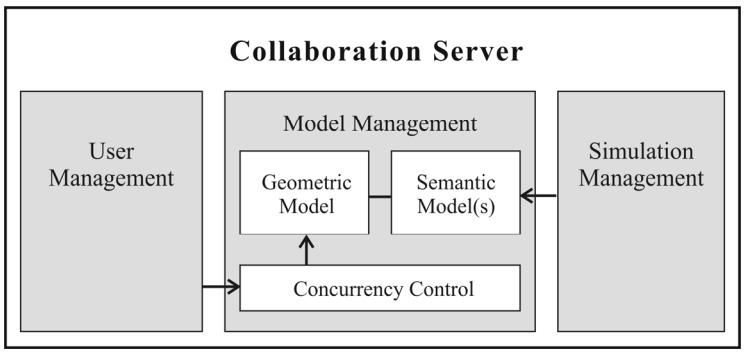

Fig. 7. The modules of the collaboration server.

The core of the collaboration server is the model management module. As discussed in Section 4.2, a hybrid model joining the geometric and the semantic model is used. In the HVAC domain, for example, the model represents the obstacles in the fluid domain and the fluid domain hull. Boundary conditions are managed as semantic data attached to geometric objects.
Modifications like adding, removing or transforming obstacles are communicated from the performing client to the collaboration server. In order to avoid conflicts between the participants, the collaborative work is coordinated by means of locking mechanisms. As long as an object is locked by a certain user it cannot be modified by any other user.
The simulation management module keeps track of the simulators available, their locations and current state. When a simulation server joins the CoCoS platform, it registers with the collaboration server and provides all the required data. A client queries the collaboration server in order to connect to a specific simulation facility.
Each Simulation Server announces the structure of the numerical results it produces. This involves the type of grid, the denomination of each single field variable and the identification of scalar and vector fields. Up to now, only grids (uniform, recti-linear and structured) are supported, but the concept is open for unstructured meshes to be added in the future. The information is used by the clients to provide suitable visualization techniques and to display the results accordingly. For vector fields, for example, streamline and vector plane visualization are provided whereas scalar fields can be depicted by simple value planes or isometric surfaces.
The user management module provides information about the users currently participating in the collaborative session. This is necessary to support the several different levels of awareness, as discussed in (Borrmann et al., 2006). Besides information on the current view point

and direction, the visualization method currently utilized by the participating engineers is stored on the Collaboration Server. There are three visualization methods supported by CoCoS: vector planes, value planes, streamlines and isometric surfaces. The parameters of each visualization object are stored on the Collaboration Server and changes are passed to the clients by means of a notification mechanism.

4.4 The clients

The CoCoS clients serve as visualization and interaction interfaces for the engineers taking part in the collaborative session. The following basic services have to be provided by each client application:

- logging into / logging out of the collaboration server,
- visualization of the geometric objects, interaction facilities for transforming them, support for locking mechanisms,
- displaying semantic data attached to the geometric object (at least via attribute-value tables),
- displaying the current participants, notification of participants entering and leaving the session.

For the HVAC usage scenario, the prototype client CoFluids was implemented. It enables a user to transform the obstacles inside an office and to depict the CFD simulation data in the form of vector planes, iso-surfaces or streamlines. The client can be run in single-window mode capable of stereoscopic rendering - for use in Virtual-Reality environments, or in multi-viewer mode - for use on desktop computers. Obstacles that are modified by another participant are shown in a different color and cannot be modified, i.e. they are locked.
In CoFluids, awareness of the activities of the other participants is provided by a list of currently registered users, and by means of messages that signal whenever a participant joins or leaves the collaborative session. Furthermore, CoFluids can display the viewpoint and the view direction of the other participants using avatars. Audio communication between the participants is provided by integrating a third-party Voice-over-IP solution.

4.5 The simulation servers

The main task of a simulation server is to build a bridge between the distributed collaborative system and a particular simulation kernel. Like the clients, the simulation server is listening to the event channels provided by the collaboration server. It is accordingly notified of any changes in the geometry and the corresponding boundary conditions and forwards this information to the simulation kernel.
At the start-up of the simulation server, it registers with the collaboration server and defines the model it uses for the start-up and steering parameters by means of the meta-model (see Section 4.2). In addition, it announces the structure of the numerical results that it produces. This information is retrieved by the client via the collaboration server in order to connect to the simulation server and to display the simulation results together with the available steering parameters accordingly.
The minimum service that a simulation server must provide is an interface to start and stop the simulation and to pass on steering parameters. It has to notify the connected clients when the simulation has been started or stopped, and when a steering parameter has been changed.
For the transmission of the simulation data from the simulation server to the clients, we experimented with two different configurations: a *push* and a *pull* communication mode.

There are two main criteria for estimating the efficiency of the configuration: First, the simulation should not be slowed down by the communication overhead. Secondly, the transmission should be as fast as possible and as flexible as required. As opposed to streaming-based communication solutions, as applied in video and audio broadcasting, for a Computational Steering client it is not necessary to receive every single 'frame' of the simulation results, but it is more important that it receives the most current simulation data.
In the first configuration, we used a CORBA Event Service implementation in order to distribute the simulation data. In this case, the data is pushed from the simulation server to the event service, which in turn pushes the data to the listening clients. The advantage of this configuration is that the distribution of the simulation results is completely decoupled from the simulation process. The major disadvantage is that each and every result set is transmitted, even if there are more up-to-date sets of result available on the simulation server. So occasionally the event channel becomes choked by obsolete simulation data, if one of the clients is not fast enough to receive and process it.
The use of pull communication is more appropriate here. In this mode, the client pulls the data whenever it is ready to process it. This method ensures that a client always receives the latest simulation data available. This should be regarded as the highest priority for a Computational Steering application.

5 Non-interactive simulation runs

After basic decisions have been made about the size and position of the HVAC inlets and the operating lamps during the collaborative computational steering session, non-interactive simulations with higher spatial resolutions and thus longer computation times have to be performed to obtain reliable results with a suitable precision.

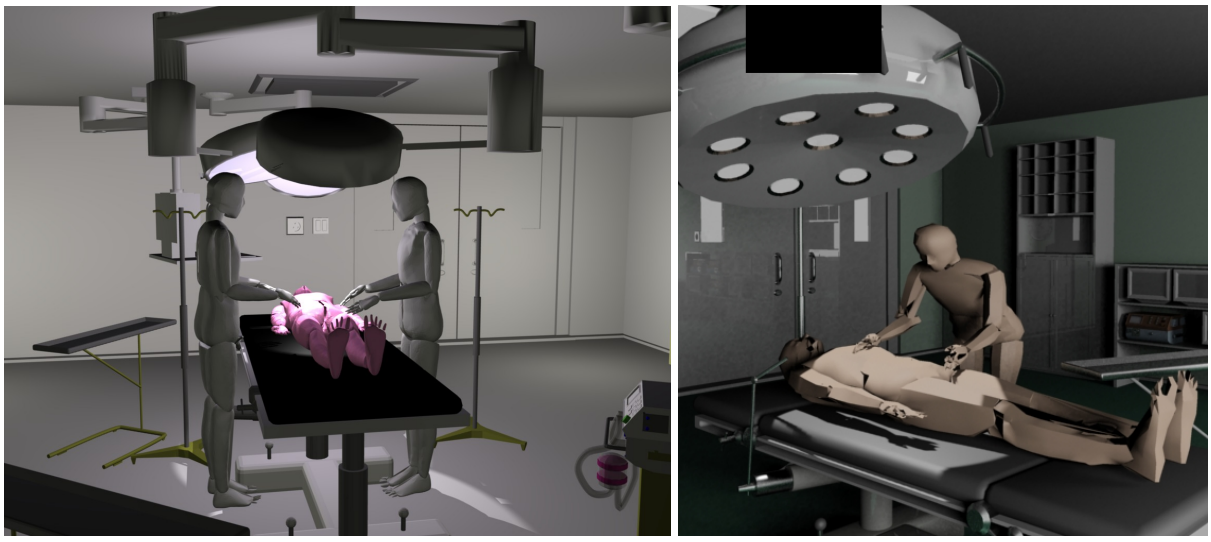

Fig. 7. Results of the non-interactive illumination analysis using the open-source renderer *Blender* and the commercial tool *3DS MAX,* respectively.

Figure 7 shows the results of the non-interactive illumination analysis using the Open-Source renderer *Blender* and the commercial tool *3DS MAX*, which are based on advanced ray-tracing algorithms. Figure 8 shows the results of the non-interactive CFD simulation realized with the help of the LBM simulator developed by our group (van Treeck et al., 2006).

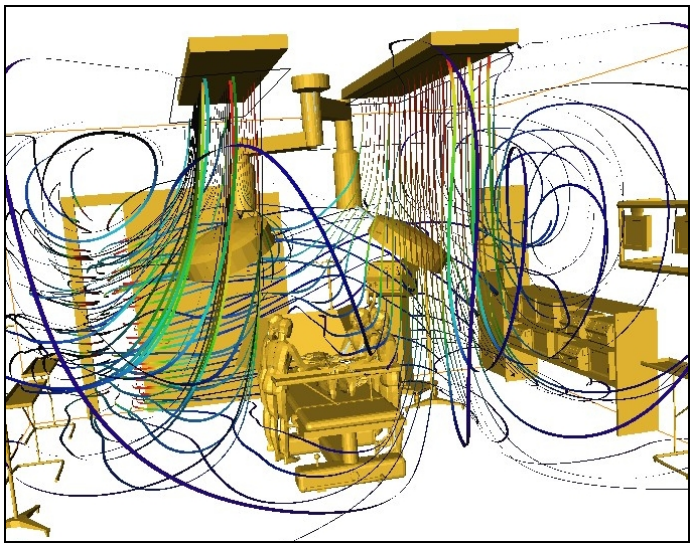

Fig. 8. Results of the non-interactive CFD simulation realized through the LBM simulator developed by our group

6 Summary

This article presents the CoCoS platform, a highly flexible distributed system for multi-disciplinary collaboration enabling the integration of interactive simulators. In order to provide the collaborating engineers with suitable analysis and simulation capabilities, we have illustrated the flexible integration of simulation servers into the distributed system. The suitability of the concept has been proved by the implementation of clients and servers for a collaborative HVAC engineering scenario. It demonstrates the integration of a CFD simulation server whose kernel is based on the lattice-Boltzmann method, thus providing an interactive fluid simulation.
The basis of the collaboration platform is formed by a centrally managed, shared model joining an explicit geometric model and a variable semantic model. Because of its vital importance for analysis and simulation tools and its domain-independent validity, its emphasis lies on the geometric model. By separating the functionality of the collaboration server and the simulation servers, it possible to use different simulation facilities without the need to re-implement the generic functionality of geometry-focused, collaborative engineering. In addition, it is possible to use simulators for different physical phenomena at the same time, thus providing the basis for multi-disciplinary synchronous collaboration.

Acknowledgements

The presented project has been supported by Siemens Corporate Technology and the Bavarian Competence Network of Scientific High Performance Computing (KonWiHR).

7 References

Booch, G. (1994). Object-oriented Analysis and Design with Applications, Benjamin-Cummings Publishing.

Borrmann, A. (2007). Computerunterstützung verteilt-kooperativer Bauplanung durch Integration interaktiver Simulationen und räumlicher Datenbanken, Ph.D. thesis, Lehrstuhl für Bauinformatik, Technische Universität München.

Borrmann, A., Wenisch, P., van Treeck, C. and Rank, E. (2006). "Collaborative computational steering: Principles and application in HVAC layout." Integrated Computer-Aided Engineering Vol. 13, No. 4, pp. 361–376.

Carey, A. (2003). "My air conditioner gave me SARS." EcoLibrium Vol. 7, pp. 5–6.

Chow, T.-T. and Yang, X.-Y. (2003). "Performance of ventilation system in a non-standard operating room." Building and environment Vol. 38, No. 12, pp. 1401–1411.

Eastman, C. (1999). Building Product Models: Computer Environments Supporting Design and Construction, CRC Press.

Ferziger, J. H. and Peric, M. (2001). Computational Methods for Fluid Dynamics, Springer.

Kühner, S. (2003). Virtual Reality - basierte Analyse und interaktive Steuerung von Strömungssimulationen im Bauingenieurwesen, Ph.D. thesis, Lehrstuhl für Bauinformatik, Technische Universität München.

Krafczyk, M. (2001). Gitter-Boltzmann-Methoden: Von der Theorie zur Anwendung, Habilitation, Lehrstuhl für Bauinformatik, Technische Universität München.

Liere, R., Mulder, J. and Wijk, J. (1997). "Computational Steering." Future Generation Computer Systems Vol. 12, No. 5, pp. 441–450.

Mäntylä, M. (1988). An Introduction to Solid Modelling, Computer Science Press.

van Treeck, C. (2004). Gebäudemodell-basierte Simulation von Raumluftströmungen, Ph.D. thesis, Technische Universität München.

van Treeck, C., Rank, E., Krafczyk, M., Tölke, J. and Nachtwey, B. (2006). "Extension of a hybrid thermal LBE scheme for Large-Eddy simulations of turbulent convective flows." Computers and Fluids Vol. 35, No. 8-9, pp. 863–871.

van Treeck, C., Wenisch, P., Pfaffinger, M., Egger, M. and Rank, E. (2008). "Computational Steering of Thermal Comfort Assessment." Proc. of IndoorAir 2008.

Tsou, J. Y., Wong, G. W. K. and Chow, B. (2004). "Design of Rapidly Assembled Isolation Patient Ward – IT-Supported Collaborative Design Process between Architects and Medical Officers." Proc. of the 10th Int. Conf. on Computing in Civil and Building Engineering (ICCCBE-X).

Wenisch, P. (2008). Computational Steering of CFD Simulations on Terraflop-Supercomputers, Ph.D. thesis, Lehrstuhl für Bauinformatik, Technische Universität München.

Zanetti, G. and McNamara, G. R. (1988). "Use of the Boltzmann equation to simulate lattice-gas automata." Phys. Rev. Lett. Vol. 61, pp. 2332–2335