EMBEDDING, ORGANISATION, AND CONTROL OF SIMULATION PROCESSES IN AN OCTREE-BASED CSCW FRAMEWORK

Ralf-Peter Mundani[1], Hans-Joachim Bungartz[1], Andreas Niggl[2], and Ernst Rank[2]

ABSTRACT

In this paper, we present a CSCW framework for simulation processes from the field of civil engineering. The integration of processes into a common scenario helps to ensure consistency of the shared data as well as to easily interchange information among all participants. Thus, side effects can already be detected in early stages, preventing expensive changes in the late stages of design processes.

Furthermore, due to a hierarchical organisation of simulation processes, the embedding of entire processes into the framework and the efficient control of processes are possible. Thus, the framework can be enhanced by a service layer, providing simulation services to be used by other participants. Here, both distributed storage and distributed computations as seen within grid computing can be addressed by this approach.

KEY WORDS

octrees, CSCW, process embedding, simulation services, nested dissection, *p*-version FEM

INTRODUCTION

Current trends within engineering software tend from mere stand-alone applications to complex program packages for solving entire classes of problems. Nevertheless, these packages often have severe drawbacks due to a missing support for concurrent access of the underlying data as well as for sufficient interfaces for data interchange with other processes or packages, resp. Thus, whole parts of design processes are still done in a serial way. This not only increases the risk of errors during the design stage, also impacts/side effects due to changes of the underlying data within one process on other processes often cannot be revealed in early times.

The cooperation of processes in a scenario distributed in time and space based on a common building information model (BIM) arises the necessity for control mechanisms to secure the consistency of the underlying data. This is often referred to as CSCW[3] framework. For the integration of processes, the framework must provide flexible interfaces without the loss of a fast data access as well as flexible data structures for an easy and efficient derivation

[1] Institut für Informatik, Technische Universität München, Boltzmannstraße 3, D-85748 Garching, Germany, Phone +49 89 289 18632, FAX +49 89 289 18607, {mundani, bungartz}@in.tum.de

[2] Lehrstuhl für Bauinformatik, Technische Universität München, Arcisstraße 21, D-80290 München, Germany, Phone +49 89 289 25057, FAX +49 89 289 25051, {niggl, rank}@inf.bv.tum.de

[3] Computer Supported Cooperative Work

of different data representations (surface-oriented and volume-oriented representations of a geometric model, e.g.).

In this paper, we will present an octree-based CSCW framework for the integration of processes from the field of civil engineering. The usage of hierarchical data structures – such as octrees – can further be exploited to embed entire simulation processes into the framework. This also allows the efficient organisation and control of embedded processes, as needed to establish schedule planning tools like in 4D CAD applications. Due to the inherent distribution of the hierarchical data structures even considerations such as parallelisation and distributed computing and distributed storage – as seen within grid computing – can easily be addressed by this approach.

The outline of this paper goes as follows. In chapter 2, an octree-based CSCW framework for the integration of simulation processes is introduced. Chapter 3 briefly deals with the embedding of simulation processes, while in chapter 4 the organisation and control of embedded simulation processes in a service-based context is highlighted. Chapter 5 addresses distributed storage and computation aspects. Finally, chapter 6 gives a short conclusion and outlook on future work.

OCTREE-BASED CSCW FRAMEWORK

We have proposed an octree-based approach for setting up a CSCW framework for processes from the field of engineering. Therefore, a BIM is stored as *vef*-graph (Bungartz et. al. 2004) to a RDBMS. For accessing parts of the BIM, all data base's primary keys are stored to an octree (Mundani et al. 2004a). This allows us to embed SQL queries into a spatial context, thus, even neighbourship relations can be considered. A sample query for parts with material ´`matClass1`´ neighbouring a part labelled ´`object`´ might look like as follows.

```
SELECT geomID FROM modelDatabase
WHERE material = ´matClass1´ and geomID = NEIGHBOUR(object)
```

As the BIM is stored as a surface-oriented geometric model, further processing is necessary to derive a volume-oriented representation as needed for many simulation processes. Here, octrees are again very helpful. If treating the surfaces of a surface-oriented model as half spaces, a volume-oriented representation can be derived in real time and even on-the-fly (Mundani et al. 2003, Bungartz et al. 2003). As octrees have turned out to be advantageous for many kind of simulation tasks, subsequent processes can easily read this volume-oriented models to be used within their computations (Brenk et al. 2005, Bader et al. 2002, e.g.).

To ensure consistency among all participating processes, direct access to the shared data is forbidden. All processes have to use check-in/out methods (similar to CVS) provided by the framework instead. After a process performs a check-out (and, thus, holds a local copy of parts of the shared data) the local data can freely be manipulated. If the process performs a check-in on the modified data again, a consistency check is initiated by the framework first. Therefore, an octree-based collision detection between the modified parts and the rest of the BIM in the framework's data base is performed. In case no collision could be found, the

modified parts are written to the data base, thus, updating their older instances. Otherwise the parts are rejected (Mundani et al. 2004b, Niggl et al. 2004).

Figure 1 shows a brief sketch of the framework consisting of a data base (storing the BIM), an octree (storing the data base's primary keys), and an access layer (providing check-in/out methods for data access). Furthermore, some sample processes already integrated are shown: CAD, Computational Structure Dynamics (CSD), and Computational Fluid Dynamics (CFD). For a broader analysis of design processes related to security issues, evacuation dynamics (EvacDyn) have also been taken into account and successfully integrated into the framework (Mundani 2005a).

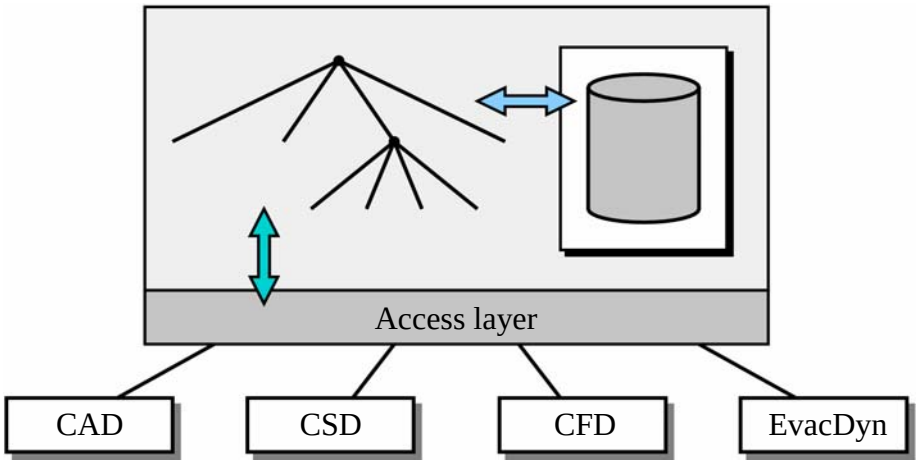

Figure 1: Octree-based CSCW framework with integrated processes

EMBEDDING OF SIMULATION PROCESSES

Further improvement in process control can be achieved by embedding a process into the framework. That is, making a process part of the framework. We have shown this for the statics' simulation as a first example (Mundani et al. 2005b, Niggl et al. 2005).

By organising the finite elements of a FEM discretisation of *p*-version type in an octree, hierarchical solvers of nested dissection type can be applied. These algorithms follow the idea of the divide-and-conquer principle. Instead of solving a large system of equations, it is divided into smaller parts and the global solution is assembled from the parted solutions. A huge advantage of this approach is to avoid redundant computations in case the underlying data changes, as geometric modifications (such as adding an additional window to a wall or moving a pillar aside) have in the most cases only a local influence.

If all parted solutions are known from a previous computation, only modified parts have to be recomputed to get the new global solution (see Figure 2). This not only decreases the amount of computational effort, it also allows a (user driven) steering of the computation where an immediate feedback on modifications comes into reach. For the statics' analysis of an office tower (approx. 250,000 finite elements) the effort of a recomputation could be reduced up to 4% of an entire computation in case some parts have been modified. Thus,

design alternatives can easily be studied online before the final (geometric) setup is written back to the framework. For detailed information on the nested dissection approach see Mundani 2006.

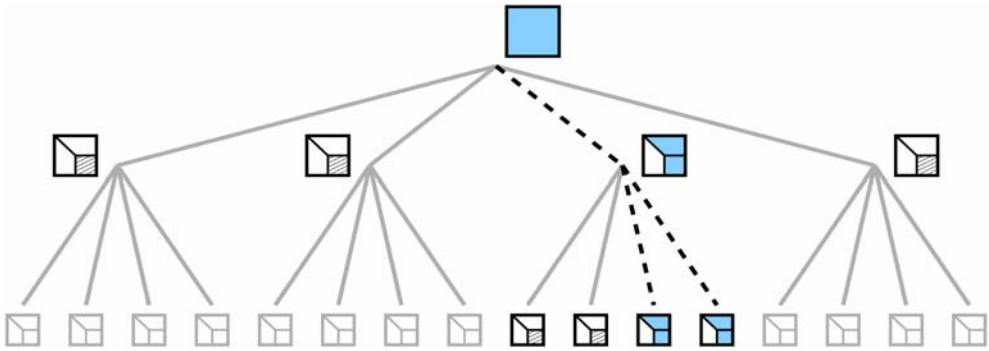

Figure 2: Only the highlighted parts have to be recomputed while all other necessary solutions (hatched) can be taken from a previous computation

ORGANISATION AND CONTROL OF SIMULATION PROCESSES

The hierarchical organisation of simulation processes opens the door to a large amount of new applications. Beside the efficient control of embedded processes, the framework can be enhanced by an additional service layer. Embedded processes are provided as simulation services and, thus, can be invoked by other processes of the framework. Hence, direct access to the (computation) results of a service is possible, to be further considered in another process' own computations.

SIMULATION SERVICES

The services might run on a separate application server (a multi-processor machine, e.g.). Each time a simulation services is invoked, the application server receives all necessary input (via the framework) from the user and loads all missing data from the framework's data base. After a successful completion of the simulation service's computations the application server sends back the results to the calling process for further usage. Figure 3 illustrates this circumstance.

The advantage of a service-based approch is to utilise core simulation tasks for other processes, to outsource expensive (in the sense of memory and time consumption) simulation tasks to more powerful machines, and to organise the workflow of single tasks within design processes. The latter one is a crucial part to foster consistency of the underlying data in a concurrent environment and, thus, to reveal side effects among processes already in early stages.

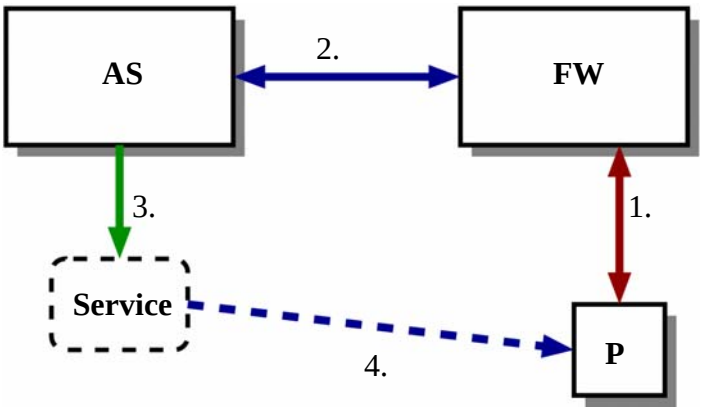

Figure 3: A process (P) wants to use a simulation service provided by the framework (FW). The framework contacts the application server (AS) which loads all necessary data from the framework and launches the service. After the service finished its computations the results are returned to the calling process (via the framework).

Assume a simulation process P_1 checks out parts of the shared data from the framework. Any modifications of these data are not visible to other processes unless P_1 checks in the altered data again. Other processes running have then to update their local copies and, thus, to withdraw their own modifications made so far before any side effects can be revealed. This is counterproductive to the overall workflow and makes it difficult to keep track of a global data consistency. In case of the service-based approach, P_1 only has to invoke the corresponding simulation service on his local modified data. If no side effects have been found, P_1 can check in its modified data to update the global data base (nevertheless, a geometric consistency check is still performed by the framework and the data is rejected if necessary) and to make its changes visible to all participating processes. Otherwise, P_1 has to alter its data and to invoke the simulation service again.

EXAMPLE: SCHEDULE PLANNING

As described in the previous chapter, the nested dissection approach is perfectly suited for fast computations in case the data changes. If any parts of the underlying geometric model are modified, added, or deleted, only those parts and their region of direct influence have to be recomputed. In case of the statics' simulation, all other results are taken from a previous run and are assembled together with the newly computed results to form the overall solution. Thus, different design stages at different time steps can easily be studied if considering the time (in a three-dimensional space) as fourth dimension—a full 4D simulation (see the accompanying work of Niggl et al. 2006).

The entirety of all design stages defines a schedule that helps to reveal dependencies among processes and, thus, to optimise the workflow. Such a schedule planning can highly profit from the simulation services introduced above. As no knowledge about the core implementation of each service is necessary, a process can use this functionality as long as it

implements the framework's interfaces. To a certain extent, this approach tends into the direction of so called problem solving environments.

DISTRIBUTED STORAGE AND COMPUTATION

The hierarchical organisation of simulation processes defines an inherent distribution in space. The octree storing the finite elements from the statics' simulation, for instance, can be separated into several subtrees, each holding one specific part of the geomtric model.

By now, all computations are done on the application server. Assigning the subtrees to different users (working on different computers) and running local services on each user's machine, all computations take place in a distributed environment. The coordination of all services is done by the application server. As each service does not need to know about all other services (actually it only needs the results from a subtree's root node due to the nested dissection technique), a local recomputation in case of a local modification is possible (see Figure 4). Thus, even complex geometries not fitting into a single computer's memory can be processed by this approach.

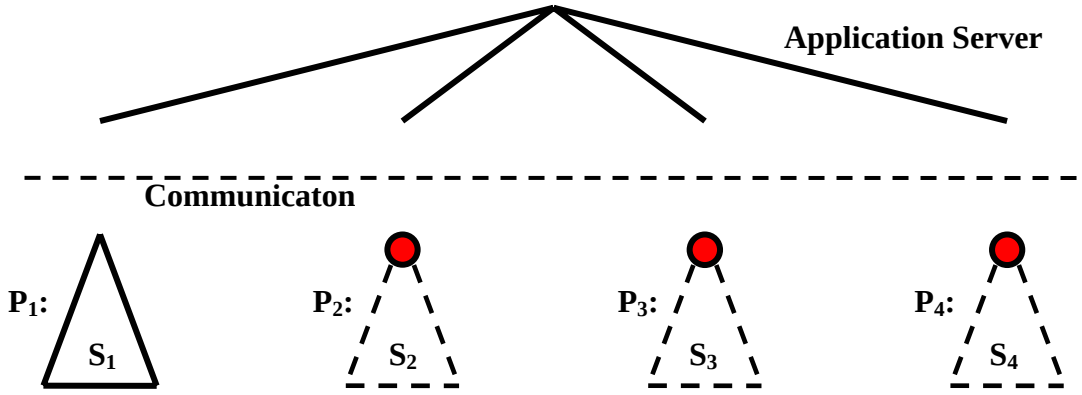

Figure 4: If process P_1 modifies its data – stored within subtree S_1 – a (locally running) simulation service can compute the partial result R_P. Sending R_P to the application server, only the results (already known from a previous computation) of the other subtree's root nodes (highlighted circles) are necessary to compute the global result R_G. This is finally returned to P_1 for further processing.

As concepts of distributed storage and distributed computations are addressed here, the usage of grid services is inevitable. Grid services as provided by the Globus Toolkit 4[4] (GT4) are helpful for the data access in a distributed environment (OGSA-DAI) as well as for the service invocation via standardised interfaces such as web services. In Mundani et al. 2005c we have proposed a schema for using grid services within our octree-based CSCW framework. This is still work in progress, but first results sound very promising.

[4] http://www.globus.org

CONCLUSIONS AND OUTLOOK

The integration of processes into a framework for shared data access is inevitable to foster cooperation and to ensure global consistency of the underlying data. Due to an efficient organisation, these processes can further be embedded into the framework and, thus, be provided as simulation services to other processes. This allows the control of a service's computations (to reduce the computational effort) as well as the direct access to a service's simulation results. Thus, model variants can be studied online and schedule planning tools (4D simulations) can easily be implemented.

Future researches will focus on embedding further simulation processes (CFD or evacuation dynamics, e.g.) into the framework. Another topic is the usage of grid services and how they can help to improve the service-based concept of this framework.

ACKNOWLEDGEMENTS

We would like to thank the Deutsche Forschungsgemeinschaft (DFG) for the financial support within the Schwerpunktprogramm 1103 "Vernetzt-kooperative Planungsprozesse im konstruktiven Ingenieurbau".

REFERENCES

Bader, M., Bungartz, H.-J., Frank, A.C., and Mundani, R.-P. (2002). "Space Tree Structures for PDE Software" *Proc. of the Int. Conf. on Comp. Science (3),* LNCS Vol. 2331, Springer, pp. 662-671.

Bungartz, H.-J., Griebel, M., and Zenger, C. (2004). "*Introduction to Computer Graphics, 2nd edition*", Charles River Media.

Bungartz, H.-J. and Mundani, R.-P. (2003). "Simulation Technology – the role of concepts and algorithms" *Selçuk Journal of Applied Mathematics*, 4 (2) 33-52.

Brenk, M., Bungartz, H.-J., Mehl, M., Mundani, R.-P., Düster, A., and Scholz, D. (2005). "Efficient Interface Treatment for Fluid-Structure Interaction on Cartesian Grids" *Proc. of the ECCOMAS Thematic Conf. on Comp. Methods for Coupled Problems in Science and Engineering*, Int. Center for Num. Methods in Engineering.

Mundani, R.-P. (2006). "*Hierarchische Geometriemodelle zur Einbettung verteilter Simulationsaufgaben*", will be published by Shaker Verlag.

Mundani, R.-P. and Bungartz, H.-J. (2004a). "An Octree-Based Framework for Process Integration in Structural Engineering" *Proc. of the 8th World Multi-Conf. on Systemics, Cybernetics, and Informatics (2), Int. Inst. of Informatics and Systemics*, pp. 197-202.

Mundani, R.-P. and Bungartz, H.-J. (2004b). "Octrees for Cooperative Work in a Network-Based Environment" *Proc. of the 10th Int. Conf. on Comp. in Civil and Build. Engineering*, VDG Weimar.

Mundani, R.-P., Bungartz, H.-J., and Giesecke, S. (2005a). "Integrating Evacuation Planning into an Octree-Based CSCW Framework for Structural Engineering" *Proc. of the 22nd Conf. on Information Technology in Constr.*, Inst. for Constr. Informatics, pp. 435-440.

Mundani, R.-P., Bungartz, H.-J., Rank, E., Niggl, A., and Romberg, R. (2005b). "Extending the *p*-Version of Finite Elements by an Octree-Based Hierarchy" *will be published in Proc. of the 16th Int. Conf. on Domain Decomposition Methods*.

Mundani, R.-P., Bungartz, H.-J., Rank, E., Romberg, R., and Niggl, A. (2003). "Efficient Algorithms for Octree-Based Geometric Modelling", *Proc. of the 9th Int. Conf. on Civil and Structural Engineering Comp.*, Civil-Comp Press.

Mundani, R.-P., Muntean, I.L., Bungartz, H.-J., Niggl, A., and Rank, E. (2005c). "Applying Grid Techniques to an Octree-Based CSCW Framework" *Proc. of the 12th European Parallel Virtual Machine and Message Passing Interface Conf.*, LNCS Vol. 3666, Springer, pp. 504-511.

Niggl, A., Rank, E., Mundani, R.-P., and Bungartz, H.-J. (2005). "Organizing a *p*-Version Finite Element Computation by an Octree-Based Hierarchy" *Proc. of the Int. Conf. on Adaptive Modeling and Simulation*, Int. Center for Num. Methods in Engineering.

Niggl, A., Rank, E., Mundani, R.-P., and Bungartz, H.-J. (2006). "A Framework for Embedded Structural Simulation: Benefits in Building Design", *to be published in Proc. of the 11th Int. Conf. on Comp. in Civil and Build. Engineering.*

Niggl, A., Romberg, R., Rank, E., Mundani, R.-P., and Bungartz, H.-J. (2004). "A Framework for Concurrent Structure Analysis in Building Industry" *Proc. of the 5th European Conf. on Product and Process Modelling in the AEC Industry*, A.A. Balkema.