JOHANNES KEPLER UNIVERSITY LINZ

Institute of Computational Mathematics

A Grid-enabled Solver for the Fluid-Structure Interaction (FSI) Problem

Ulrich Langer
Institute of Computational Mathematics, Johannes Kepler University
Altenberger Str. 69, 4040 Linz, Austria

Huidong Yang
Institute of Computational Mathematics, Johannes Kepler University
Altenberger Str. 69, 4040 Linz, Austria

Walter Zulehner
Institute of Computational Mathematics, Johannes Kepler University
Altenberger Str. 69, 4040 Linz, Austria

NuMa-Report No. 2009-07 August 2009

Technical Reports before 1998:

1995

95-1	Hedwig Brandstetter *Was ist neu in Fortran 90?*	March 1995
95-2	G. Haase, B. Heise, M. Kuhn, U. Langer *Adaptive Domain Decomposition Methods for Finite and Boundary Element Equations.*	August 1995
95-3	Joachim Schöberl *An Automatic Mesh Generator Using Geometric Rules for Two and Three Space Dimensions.*	August 1995

1996

96-1	Ferdinand Kickinger *Automatic Mesh Generation for 3D Objects.*	February 1996
96-2	Mario Goppold, Gundolf Haase, Bodo Heise und Michael Kuhn *Preprocessing in BE/FE Domain Decomposition Methods.*	February 1996
96-3	Bodo Heise *A Mixed Variational Formulation for 3D Magnetostatics and its Finite Element Discretisation.*	February 1996
96-4	Bodo Heise und Michael Jung *Robust Parallel Newton-Multilevel Methods.*	February 1996
96-5	Ferdinand Kickinger *Algebraic Multigrid for Discrete Elliptic Second Order Problems.*	February 1996
96-6	Bodo Heise *A Mixed Variational Formulation for 3D Magnetostatics and its Finite Element Discretisation.*	May 1996
96-7	Michael Kuhn *Benchmarking for Boundary Element Methods.*	June 1996

1997

97-1	Bodo Heise, Michael Kuhn and Ulrich Langer *A Mixed Variational Formulation for 3D Magnetostatics in the Space $H(rot) \cap H(div)$*	February 1997
97-2	Joachim Schöberl *Robust Multigrid Preconditioning for Parameter Dependent Problems I: The Stokes-type Case.*	June 1997
97-3	Ferdinand Kickinger, Sergei V. Nepomnyaschikh, Ralf Pfau, Joachim Schöberl *Numerical Estimates of Inequalities in $H^{\frac{1}{2}}$.*	August 1997
97-4	Joachim Schöberl *Programmbeschreibung NAOMI 2D und Algebraic Multigrid.*	September 1997

From 1998 to 2008 technical reports were published by SFB013. Please see
`http://www.sfb013.uni-linz.ac.at/index.php?id=reports`
From 2004 on reports were also published by RICAM. Please see
`http://www.ricam.oeaw.ac.at/publications/list/`
For a complete list of NuMa reports see
`http://www.numa.uni-linz.ac.at/Publications/List/`

Numerical Simulations of Fluid-structure Interaction (FSI) Problems On the Grid Environment

Ulrich Langer[*], Huidong Yang[†] and Walter Zulehner[‡]

Abstract. *In this paper, we described a grid-enabled solver for numerical simulations of fluid-structure interaction problems. We use a domain decomposition method such that the whole problem is reduced to an interface equation by requiring solving the structure and the fluid sub-problems independently on each grid node. For realizing this grid-enabled algorithm, a newly designed Client/Server model under the grid environment is discussed.*

1. Introduction

Fluid-structure interaction (FSI) describes a large range of physical problems from aeroelastic problems, such as airflows around rigid structures, to haemodynamics, for instance blood flow in large arteries. More and more interests are arising in this class of problems in many technical applications since a long time (see, e.g. [20, 7, 30, 27, 11]). Recently, FSI simulations have been successfully used in life science as well. For instance, blood flow simulations are among the most interesting and challenging applications in this field (see, e.g. [22, 23, 24, 5, 9, 21, 28]). The numerical simulations for them usually lead to large scale problems which have to be solved using distributed machines. Therefore, as we might see today, using the grid computing resources (see [13, 8, 12]), these large scale numerical simulations can be resolved in a more efficient way because of its huge computational and memory storage resources, e.g. see [29, 17, 15] for our previous work on computational fluid dynamics (CFD), and see [14, 16] on fluid-structure interaction problem.

The fluid-structure interaction problem typically includes the structure (with a state variable: structure domain displacement d^{s}) and fluid sub-problems (with three state variables: velocity u, pressure p and fluid domain displacement d^{f}) in the structure sub-domain Ω^{s} and the fluid sub-domain Ω^{f}, respectively. Additionally, it includes interface condition (equivalence of displacement and normal stress from both sub-problems at the interface Γ), boundary conditions and initial conditions such that the whole problem is uniquely solvable, see Fig.1.

Two main strategies for solving fluid-structure interaction problems have been studied recently. One is to simultaneously solve the fluid and the structure problems with a unique solver. This method is the so-called monolithic method. This monolithic approach typically needs a global solver which is

[*]Institute of Computational Mathematics, Johannes Kepler University Linz, Linz, Austria, e-mail: ulanger@numa.uni-linz.ac.at, http://www.numa.uni-linz.ac.at/∼ulanger/.

[†]Institute of Computational Mathematics, Johannes Kepler University Linz, Linz, Austria, e-mail: huidong@numa.uni-linz.ac.at, http://www.numa.uni-linz.ac.at/∼zulehner/.

[‡]Institute of Computational Mathematics, Johannes Kepler University Linz, Linz, Austria, e-mail: huidong@numa.uni-linz.ac.at, http://www.numa.uni-linz.ac.at/∼huidong/.

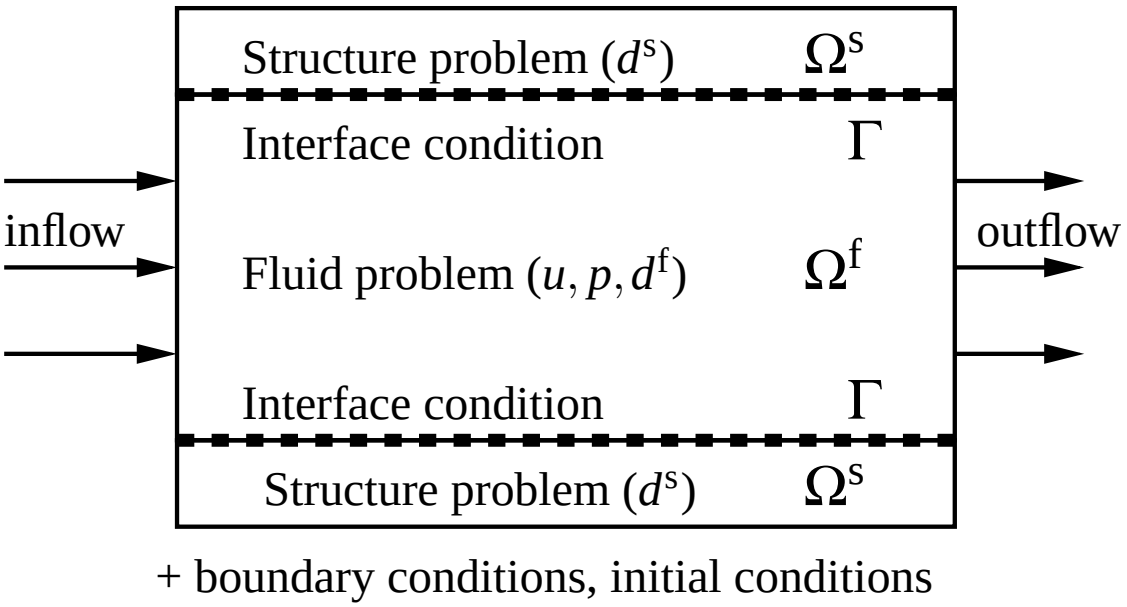


Figure 1. Model problem

less modular than two distinct fluid and structure solvers, see [18]. Another strategy is the so-called partitioned (segregated) approach (see e.g. [19, 9, 5]) which is based on subsequent solutions of the fluid and structure sub-problems and allows the use of existing codes for the fluid and structural fields. Therefore, the second approach is very suitable for grid computing. In order to adapt our FSI solver to the grid computing environment, the method we used in this work for solving such a problem is a partitioned approach based on a reduction to a nonlinear equation at the interface between the structure and the fluid sub-domains. This interface equation is solved by a Newton iteration, see [28, 5, 9].

One main issue on in the seoncond approach for solving this nonlinear coupled FSI problem is to develop algorithms based on efficient, robust and fast solvers for each of the sub-problems (structure and fluid), which are the main costs of this type of algorithms and can be distributed and parallelized to many processors under the grid environment.

We employ three distinct grid nodes. The master node is responsible for the outer nonlinear iteraiton loop, gathering and redistributing data, and synchronizing the process in each nonlinear iteration. The other two slave nodes will solve the fluid and structure sub-problems. See Fig.1. for an illustration. Using this Newton algorithm developed in [5], only a small amount of data (the updated displacement and the normal stresses at the interface Γ_0) will be transferred among the master and slave grid nodes during the process. Hence most of the simulation time is spent in solving the structure and the fluid sub-problems on corresponding slave grid nodes. In principle, each of these sub-problems can be parallelized in each node.

The grid-enabled fluid-structure interaction solver as illustrated in Fig.1. is realized by extending the idea of constructing a grid-enabled Client/Server model described in [29, 17] where Globus_IO secure channels (see [10]) are employed for creating a flexible and secure data transferring interface on the memory level for each grid node.

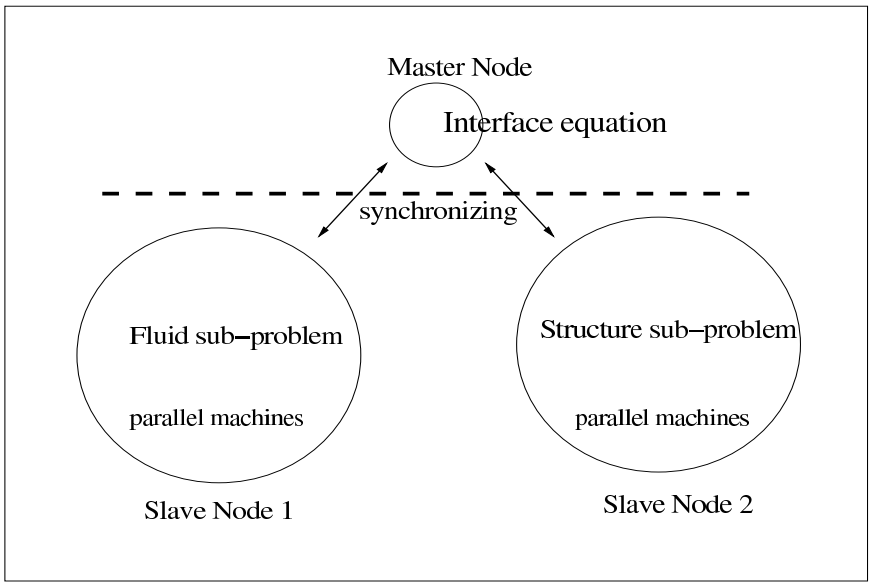


Figure 2. A grid-enabled solver model.

In the following sections, we will firstly describe how to embed the fluid-structure interaction system (systems of partial differential equations (PDEs)) into the the grid computing environment, namely how to distribute the task on grid nodes which play different roles. In addition, we need to adapt the standard Newton method to the grid environment. This is described in the grid-enabled Newton method. Then we will present details concerning how to construct the Client/Server model under the grid computing environment, which is applied to the fluid-structure interaction simulation. Finally, we will report some test results concerning the numerical iterations (the outer nonlinear iteration, the inner iterations for the structure and the fluid sub-problems), the computational complexity of the master and slave nodes, and the comparison of the computational and communicational cost.

However, this is only the first draft implementation which does not want to achieve optimal performance and thus serves as a proof of concept study. Starting from this general framework, we can go further. For instance, although we distribute the structure and the fluid sub-problems on two different grid nodes such that both can start their own job in parallel, each sub-problem itself is not parallelized, i.e. we do not fully utilize the computing and memory resources on slave nodes. Future work will concentrate on improving the performance and investigating the scalability of the method.

2. Coupling System On the Grid

2.1. Models Under the Grid Environment

The system of equations on the structure part will be solved in the reference domain Ω_0^s which does not change with time t. However, the computational fluid domain $\Omega^f(t)$ will change with t. The moving interface between fluid and structure domain is $\Gamma(t)$ (see Fig.3, where $\mathcal{L}(\cdot,t)$ and $\mathcal{A}(\cdot,t)$ are two families of mappings from the structure and fluid reference sub-domains to their current sub-domains, respectively. For details on how to handle the moving domains for the fluid-structure interaction problem, we refer to the dissertation work [28] from one of the authors.

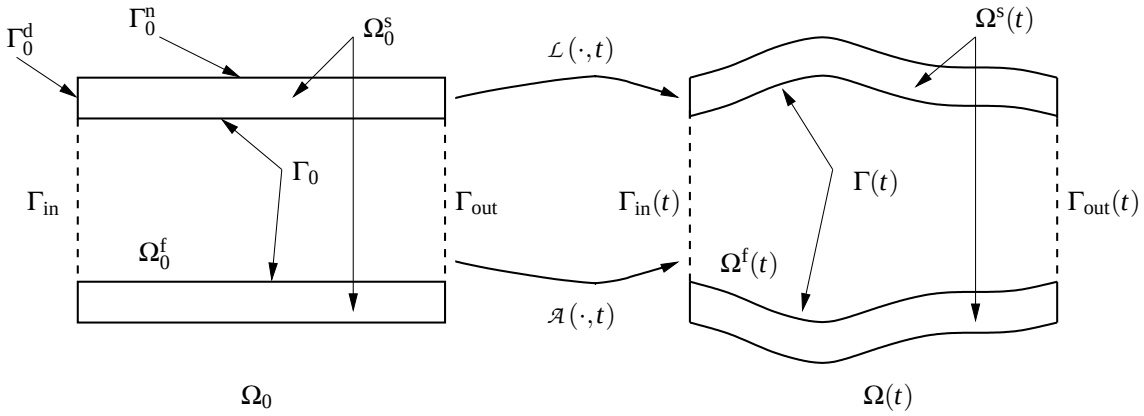


Figure 3. Sub-problems in sub-domains.

The master node will deal with the nonlinear interface condition

$$S_{\mathrm{s}}(\lambda) + S_{\mathrm{f}}(\lambda) = 0, \tag{1}$$

where λ is the displacement field at the interface Γ_0, and S_{s} and S_{f} denote two mappings from the displacement to normal stress at the interface Γ_0 for the structure and the fluid sub-problems, respectively. These two mappings are to be resolved independently and parallelly on the two slave nodes once the displacement λ is sent to them from the master node.

The slave node for calculating $S_{\mathrm{s}}(\lambda)$ requires solving the following system of equations for the Saint-Venant Kirchoff elastic model in the domain Ω_0^{s}:

$$\begin{cases} \rho_{\mathrm{s}} \dfrac{\partial^2 d^{\mathrm{s}}}{\partial t^2} - \mathrm{div}(\sigma_{\mathrm{s}}(d^{\mathrm{s}})) = 0 & \text{in } \Omega_0^{\mathrm{s}}, \\ \sigma_{\mathrm{s}}(d^{\mathrm{s}}) n_{\mathrm{s}} = 0 & \text{on } \Gamma_0^{\mathrm{n}}, \\ d^{\mathrm{s}} = 0 & \text{on } \Gamma_0^{\mathrm{d}}, \\ d^{\mathrm{s}} = \lambda(t) & \text{on } \Gamma_0. \end{cases}$$

where d^{s} is the displacement of the structure domain, $\sigma_{\mathrm{s}}(d^{\mathrm{s}}) = 2\mu^l \varepsilon(d^{\mathrm{s}}) + \lambda^l \mathrm{div}(d^{\mathrm{s}}) I$, the Piola-Kirchoff stress tensor. The other slave node for calculating $S_{\mathrm{f}}(\lambda)$ needs to solve the incompressible Navier-Stokes equations in the domain $\Omega^{\mathrm{f}}(t)$

$$\begin{cases} \rho_{\mathrm{f}} \left.\dfrac{\partial u}{\partial t}\right|_{x_0} + \rho_{\mathrm{f}} \Big((u - w^{\mathrm{f}}) \cdot \nabla \Big) u - 2\mu \mathrm{div}\varepsilon(u) + \nabla p = 0 \quad \text{in } \Omega^{\mathrm{f}}(t) = 0 & \text{in } \Omega^{\mathrm{f}}(t), \\ \mathrm{div} u = 0 & \text{in } \Omega^{\mathrm{f}}(t), \\ \sigma_{\mathrm{f}}(u,p) n_{\mathrm{f}} = g_{\mathrm{f,in}} & \text{on } \Gamma_{\mathrm{in}}(t), \\ \sigma_{\mathrm{f}}(u,p) n_{\mathrm{f}} = g_{\mathrm{f,out}} & \text{on } \Gamma_{\mathrm{out}}(t), \\ u \circ x^{\mathrm{f}} = \dfrac{d\lambda(t)}{dt} & \text{on } \Gamma(t), \end{cases}$$

where u is the velocity, p pressure, ρ_{f} the fluid density, μ its dynamic viscosity, $\sigma_{\mathrm{f}}(u,p) = -pI + 2\mu\varepsilon(u)$ the Cauchy stress tensor, and $\varepsilon(u) = \frac{\nabla u + (\nabla u)^{\mathrm{T}}}{2}$, the strain rate tensor. We refer to [28] for a complete description of a mathematical modeling for the coupled system.

2.2. Grid-enabled Newton Method

As stated in Algorithm 1, a standard Newton method is applied for solving (1). Note that in step 2 of

Algorithm 1 Newton Method for Solving the Interface Equation

1: update the residual $S_s(\lambda^k) + S_f(\lambda^k)$ by solving the structure and fluid sub-problems,
2: solve the linearized problem $\left(S_s'(\lambda^k) + S_f'(\lambda^k)\right)\delta\lambda^k = -\left(S_s(\lambda^k) + S_f(\lambda^k)\right)$ via GMRES method,
3: update the displacement $\lambda^{k+1} = \lambda^k + \delta\lambda^k$, go to step 1 if not accurate enough.

Algorithm 1, we solve the linearized problem via the *generalized minimal residual method* (GMRES) introduced in [26]. It turns out that we need to solve the structure and the fluid sub-problems few times in each GMRES iteration, with a chosen displacement $\delta\lambda^k$ at the interface, see [28] for how to realize the evaluation of $S_s'(\lambda^k)\delta\lambda^k$ and $S_f'(\lambda^k)\delta\lambda^k$. Thus this step can be distributed and parallelized as well.

From these steps in Algorithm 1, it is easy to realize a grid-enabled Newton method for solving the interface equation (see Algorithm 2). Note that if a preconditioner, e.g. S_s^{-1}, is applied to the

Algorithm 2 Grid-enabled Newton Method for Solving the Interface Equation

1: distribute displacement λ^k at the interface from the master node to slave nodes, update the residual $S_s(\lambda^k)$ and $S_f(\lambda^k)$ by solving the structure and fluid sub-problems independently on corresponding slave nodes, send back the results to the master node, and calculate $S_s(\lambda^k) + S_f(\lambda^k)$ on the master node,
2: solve the linearized problem $\left(S_s'(\lambda^k) + S_f'(\lambda^k)\right)\delta\lambda^k = -\left(S_s(\lambda^k) + S_f(\lambda^k)\right)$ via GMRES method on the master node, i.e. update $S_s'(\lambda^k)\delta\lambda$ and $S_f'(\lambda^k)\delta\lambda$ independently on corresponding slave nodes, send them back to the master node, and update $\left(S_s'(\lambda^k) + S_f'(\lambda^k)\right)\delta\lambda^k$ for each GMRES iteration,
3: update the displacement $\lambda^{k+1} = \lambda^k + \delta\lambda^k$ on the master node, and go to step 1 if not accurate enough

linearized problem in Step 2 of Algorithm 2 (preconditioned GMRES (PreGMRES)), some additional communication among nodes are needed. We also mention that in Step 2 of Algorithm 2, the GMRES iteration needs the operations of $S_s'(\lambda^k)\delta\lambda^k$ and $S_f'(\lambda^k)\delta\lambda^k$ which are the main cost of the GMRES iteration and done on the slave nodes. The main task for the master node is to gather and redistribute a small amount of displacement and stress data between the master and the slave nodes.

3. Client/Server Model On the Grid Environment

Because of its differences to usual Client/Server models, in this section, a grid-enabled Client/Server model under the Austrian grid environment (see [4]) is discussed in detail. Note that we assume the master node will take the server role, while the two slave nodes play the client role. In our previous work (see [29, 17]), we already constructed this model and apply it to the 3D incompressible Stokes/Navier-Stokes problems in computational fluid dynamics under the Austrian grid environment. Here we extend this idea to the fluid-structure interaction problem which will involve more grid nodes and are more potentially suitable for grid computing.

3.1. The Secure Grid Environment

One important point concerning grid computing is how to realize secure data transfer among client and server nodes through Internet/Intranet. The Globus Toolkit 4.0.4 includes the open source software MyProxy 3.7 for managing security credentials (certificates and private keys) [1]. One highlight of this package is to combine an online credential repository with an online certificate authority which allow users to obtain credentials when and where needed. Under the Austrian grid environment, the user would use *myproxy-init* command to upload a credential to the myproxy-server *hydra.gup.uni-linz.ac.at* for later retrievals by Client/Server nodes, e.g., under the Austrian Grid environment, we employ *altix1.jku.austriangrid.at* and *Schafberg.coma.sbg.ac.at* as client and server nodes, respectively. The credential is then delegated to the myproxy-server and stored with the given MyProxy passphrase. Proxy credentials with default lifetime of 12 hours can then be retrieved via *myproxy-get-delegation* with the MyProxy passphrase. Once the Client/Server nodes obtain the proxy credentials, the authentication and authorization on the Client/Server are done. The secure mode is verified via setting Globus_IO secure mode parameters as input arguments of these functions *globus_io_attr_set_secure_authentication/channel_mode*. See Fig. 4 for an illustration.

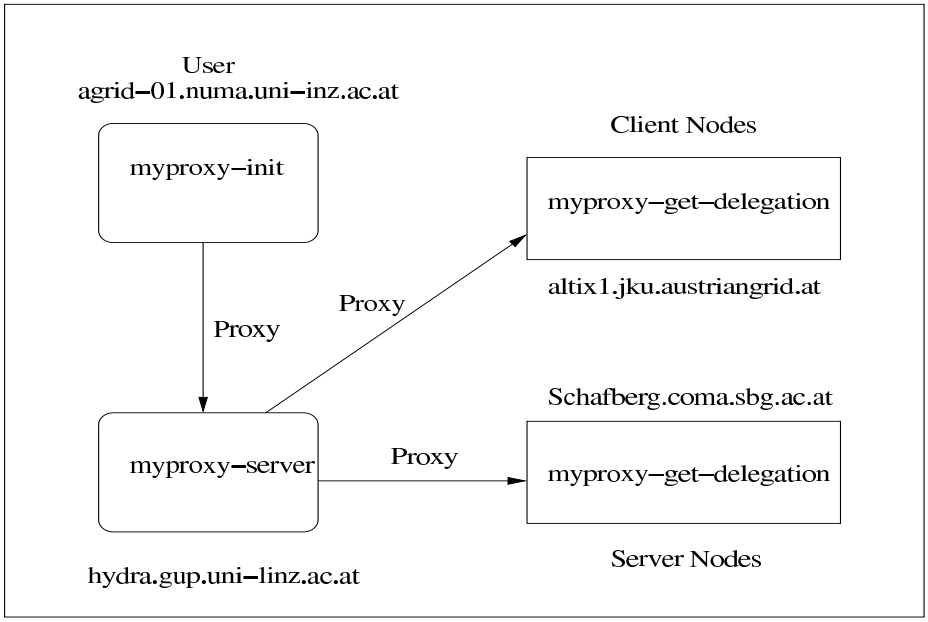


Figure 4. MyProxy process on the Austrian grid

Using the previous authentication and authorization, a secure channel connecting Client and Server nodes through the Internet/Intranet is guaranteed by using the Globus_IO secure channel. It provides high-performance I/O with integrated security and a socket-like interface for users, see [10]. Normal users holding no powerful machines can also succeed in doing such numerical simulations. As shown in Fig. 5, once their identities are certified by a Certification Authority and recognized by the requested resources, users can submit the job to nodes and control the data flow between nodes. For instance, *globus-url-copy* [2], can realize the data files transfer among nodes, and by RSL (resource specification language), users can control the job running schedule on the nodes. Under the Austrian grid environment, using *glogin* (see [25]), the identified user can realize the interactive usage of grid

[1] http://grid.ncsa.uiuc.edu/myproxy/
[2] http://www.globus.org/toolkit/

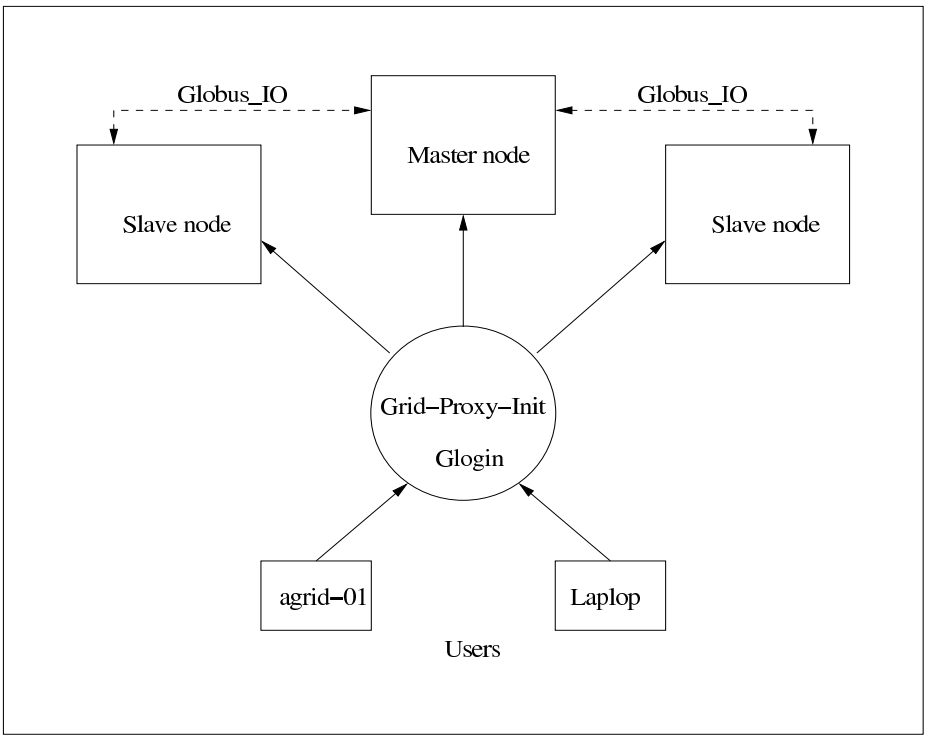


Figure 5. A secure channel in the Internet/Intranet

resources. Via the Globus_IO secure and efficient channel mode, we can distribute our task into different nodes such that they are able to cooperate with each other. The communication between nodes is guaranteed in this channel created by calling *globus_io_tcp_connect*.

3.2. The Grid-enabled Client/Server Model

An additional feature compared to usual Client/Server models on TCP/IP protocols is the authentication part on both client and server nodes by employing Globus_IO operations. The TCP connecting is mainly implemented via functions of *globus_io_tcp_listen*, and *globus_io_tcp_connect/accept*. The master node firstly creates two TCP listeners at two ports and keeps listening at these ports. Once it gets a notification from the two slave nodes, it will try to establish TCP connections. If this connecting is successful, they will continue executing the next jobs. Otherwise, the master node still listens at the opening ports until it gets next notifications from the slave nodes. The data transfer and redistribution among nodes are hidden in the modulus of FSI solver (Algorithm 2). Since this is a time dependent problem, this modulus has to be called at each time step. When the iteration ends at the final time step, all I/O operations and TCP connections on the nodes are closed. See Fig. 6 for an illustration.

3.3. Shared Data Transferring Interface

The communication between master and slave nodes are mainly focus on delivering a small amount of data (vector values of the displacement and the stress at the interface). A secure, stable and efficient data transferring interface has to be constructed on both master and slave nodes. As we mentioned before, the Globus_IO offers such desirable operations. By calling the *globus_io_write/read* pair, one can realize the data sending and receiving through the established Globus_IO channel. Usually, hierarchical data structures of vectors are used for storing the vector data. The vector themselves

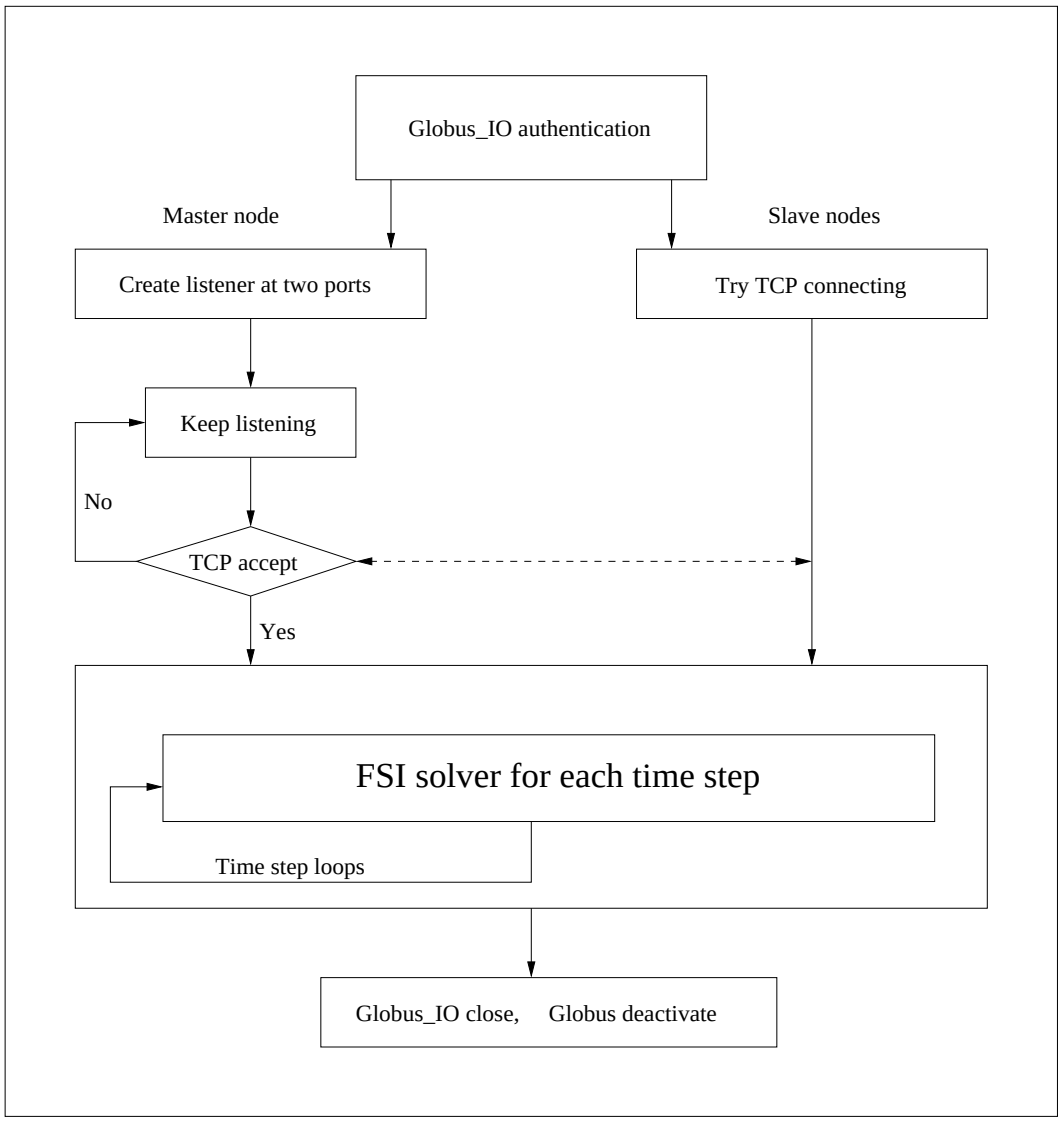


Figure 6. Grid-enabled Client/Server model for the FSI simulation

could also contain simple data structures. The size of these hierarchical data structures has to be measured carefully since the *globus_io_write/read* calling needs to know the exact block size of the message the nodes will send and receive. The *globus_io_write/read* callings are encapsulated inside such that the synchronizing process will be guaranteed for each sending and receiving pair.

3.4. Client/Server Configuration Files

Using the high-performance and secure data transferring protocol GridFTP (see [3]), the executable binary files and necessary configuration files are transferred from a user machine (*agrid-01*) to client and server grid nodes (*altix1.uibk.ac.at, alex.jku.austriangrid.at, lilli.edvz.uni-linz.ac.at*). Once the binary files installed on the grid nodes, one can specify the grid node roles with the help of configuration files. See Fig.7.

For the configuration on the server node (*alex.jku.austriangrid.at*), we specify the grid node type (*GRIDTYPE Gserver*), the name of the grid node (*GRIDNODE alex.jku.austriangrid.at*), and the port

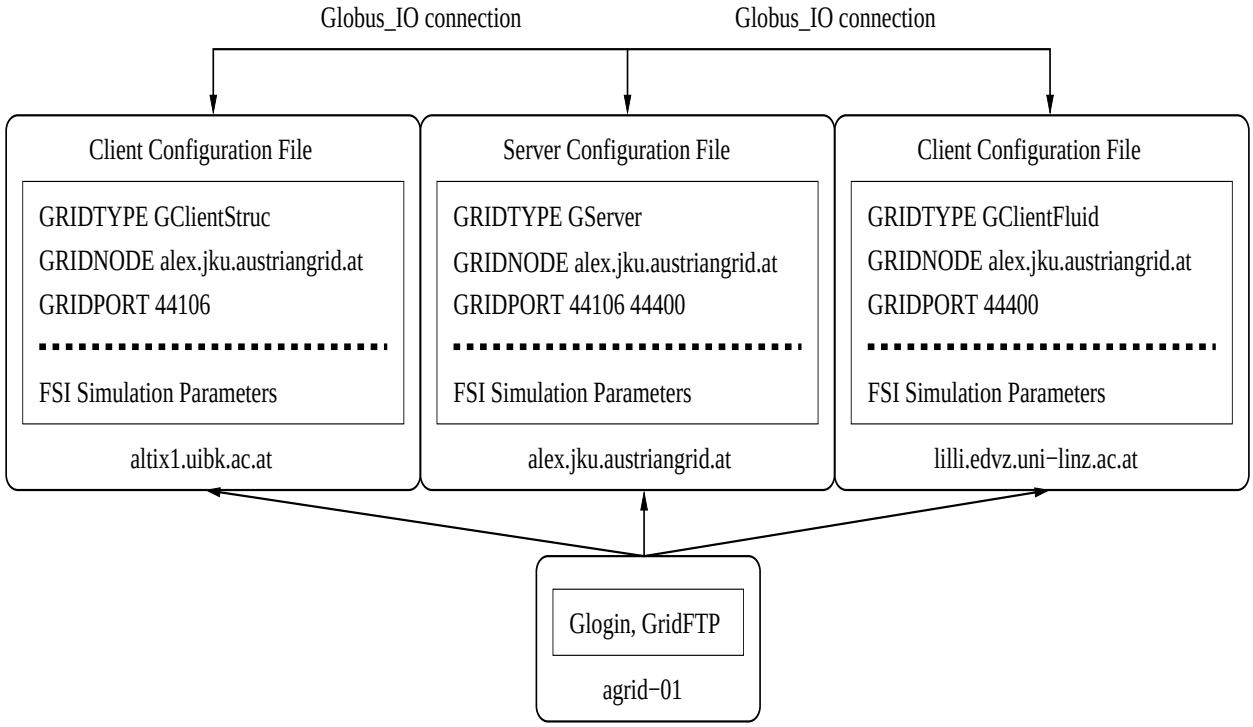


Figure 7. Client/Server configuration files

numbers opening for two client nodes((*GRIDPORT* 44106, 44400). These parameters tell the node it should play the server role. Some additional parameters have to be specified in order to distribute and gather data from client nodes.

Two client nodes are used in this model. For the fluid part, the grid node *lilli.edvz.uni-linz.ac.at* is chosen as a client node. We set the grid node type (*GRIDTYPE GClientFluid*), the server node which we want to connect to (*GRIDNODE alex.jku.austriangrid.at*), and the port number as one of port numbers specified on the server node (*GRIDPORT 44400*). In analogous way, for the structure part, the grid node (*altix1.uibk.ac.at*) is chosen as a client node. We set the grid node type (*GRIDTYPE GClientStruc*), the server node which we want to connect to (*GRIDNODE alex.jku.austriangrid.at*), and the port number as one of port numbers specified on the server node (*GRIDPORT 44106*). For both client nodes, we need to set some FSI simulation parameters such that each of them will be responsible for its own job, i.e. solving the structure and the fluid sub-problems, respectively.

These configuration files will be transferred to corresponding grid nodes, and act as running input parameters when starting programmings on three grid nodes.

4. Numerical Results On the Austrian Grid

4.1. Testing Environment

We use three different nodes for testing the algorithm:

1: the Server (master) node for synchronizing the whole process, *alex.jku.austriangrid.at* in Linz, which is a cluster with a total of 384 Xeon (nehalem) cores connected via gigabit,

2: the Client (slave) node for running the structure solver, *altix1.uibk.ac.at* in Innsbruck, which is a cluster of four 16-way SGI Altix 350 systems interconnected by an Infiniband fabric,
3: the Client (slave) node for running the fluid solver, *lilli.edvz.uni-linz.ac.at* in Linz, which is a shared-memory single node with 256 CPUs and 1 TB RAM, connected via gigabit.

See details in Table 1 and [2].

Table 1. Grid Nodes Information

Site	Grid Node	Processors Types (Number of processors)	RAM
ALTIX-UIBK	altix1.uibk.ac.at	Intel Itanium-2 (16)	1.5 GB
JKU	lilli.edvz.uni-linz.ac.at	Intel Itanium-2 (256)	1.0 TB
JKU	alex.jku.austriangrid.at	Xeon (nehalem) Cores (768)	1.5 TB

In addition, the user machine *agrid-01* in Linz is a desktop with two AMD Opteron processors and 4 GB RAM, which is responsible for transferring data and binary files from the user to the grid nodes.

However, as mentioned before, for this moment, neither of the structure and the fluid solvers is parallelized, so we only utilize one of processors from each grid node. In order to obtain good scalability and high performance, we will implement the structure and fluid solvers in parallel for the future plan, e.g. see the parallel technique in [6].

4.2. Material Parameters

We simulate a pressure wave in a cylinder of length 5 cm and radius 5 mm at rest. The thickness of the structure is 0.5 mm. The structure is considered linear and clamped at both the inlet and outlet. The fluid viscosity is set to $\mu = 0.035$, the Lamé constants to $\mu^l = 1.15 \times 10^6$ and $\lambda^l = 1.73 \times 10^6$, the density to $\rho^f = 1.0$ and $\rho^s = 1.2$. The structure and the fluid domains are initially at rest and a pressure of 1.332×10^4 is set on the inlet for a time period of 3 ms. For details concerning material parameters, we fefer to [5].

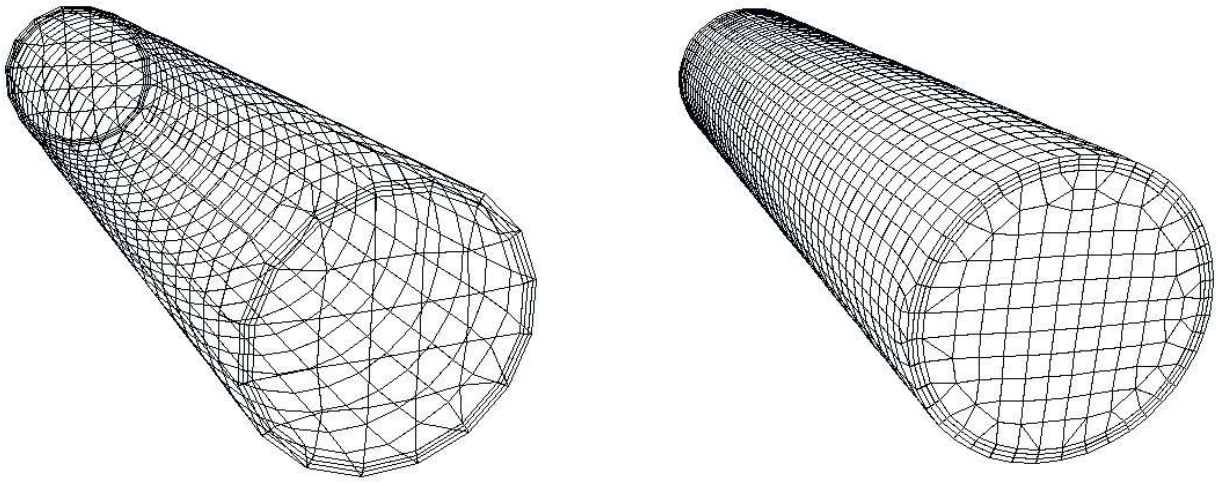
4.3. Meshes

All meshes in our test examples were provided by Dipl.-Ing. Ferdinand Kickiger, CAE Software Solutions Wolfkersbhelstr. 23, A-3730 Eggenburg, Austria (See webpage: www.meshing.org). We used two meshes for simulations (see surface meshes in Fig.8 as an illustration). The coarse mesh contains about $4,000$ vertices and the fine mesh $20,000$ vertices. The overall number of unknowns for these two testing cases are about $28,000$ and $140,000$, respectively.

4.4. Iteration Numbers

For all simulations, we use a time step size of $\delta t = 1$ ms and run the simulation until time $t = 20$ ms. The iteration numbers are listed in the following:

1: for each time step, we need 2-3 Newton iterations for a relative error reduction by a factor of 10^{-5} for solving the nonlinear interface equation (1),
2: for each nonlinear iteration, 6-8 GMRES iterations are required to reach a relative error reduction by a factor of 10^{-5},
3: for each GMRES iteration, we apply the structure and fluid solvers once,



(a) coarse mesh (b) fine mesh

Figure 8. Fine and coarse meshes for simulations.

4: for solving structre sub-problem, 10-12 preconditioned conjugate gradient iterations with AMG (algebraic multigrid method) preconditioning are need to get the error reduction by a factor of 10^{-8},
5: for solving fluid problem, 5-8 AMG iterations are needed to obtain the error reduction by a factor of 10^{-8},
6: almost the same numbers of iterations for the coarse and the fine mesh.

Note that for a detailed description concerning the AMG solvers for both sub-problems, we refer to [28].

4.5. Computational Complexity

As we see in Algorithm 2, its main cost is reduced to solving the structure and the fluid sub-problems on client grid nodes, i.e, the solving time using AMG solvers for each sub-problems. Since the gird nodes are placed in different locations, we will take the communicational cost between the server and the client node into account. However, as we discussed, these two sub-problems are distributed independently, and run in parallel. Therefore, they have overlap in the running time such that it can save the running time of the whole process.

If we look at closely the iteration numbers in subsection 4.4., the following cost will be of importance: the cost for each time step, for each Newton iteration, for solving the structure and the fluid sub-problems once, and for communication between the client and the server nodes once. The cost for one time step is obtained by multiplying the cost for each Newton step with the steps, around 2-3. The whole simulation cost is obtains by multiplying the cost for each time step with the time steps of 25. We will present the cost for the simulation on two meshes in Fig. 8, see Table 2 and Table 3. All the cost is measured in second (s).

We use some abbreviations in Table 2 and Table 3: 'Newton Step'-one Newton iteration which in-

volves applying GMRES iterations several times, 'S-Solver'-solver for the structure sub-problem, 'F-Solver'-solver for the fluid sub-problem, 'S-Comm'-communication between the server node and the client node for the structure sub-problem which involves the cost for the structure solver and one Globus_IO reading/writing operation, 'S-Comm'-communication between the server node and the client node for the fluid sub-problem which involves the cost for the fluid solver and one Globus_IO reading/writing operation.

Table 2. Computational cost on the coarse mesh

	Newton Step	S-Solver	F-Solver	S-Comm	F-Comm
Cost (s)	1268.26s	10.27s	11.14s	38.97s	11.15s

Table 3. Computational cost on the fine mesh

	Newton Step	S-Solver	F-Solver	S-Comm	F-Comm
Cost (s)	4411.78s	38.70s	71.19s	347.6s	71.19s

The computational cost of one solver and communication operation is reflected in 'S-Comm' and 'F-Comm' for the structure and the fluid sub-problems, respectively. For the structure part, due to the distance and aconet networking connection between Linz and Innsbruck [1], the communicational cost is rather expensive and cannot be avoided, but can be acceptable, about 70% and 88% for coarse and fine meshes, respectively. The networking connection for the two nodes in Linz is fast such that the solver cost will take the main portion of the whole cost, about 100% for both fine and coarse meshes. On the other hand, the overlapping cost for 'S-Comm' and 'F-Comm' is about 11s on the coarse mesh, and about 70s on the fine mesh. Pay attention that, for one Newton iteration, we need to apply the 'S-Solver' and the 'F-Solver' several times, and some other routines, e.g. the mesh handling, matrices assembling and preconditioning techniques, which we do not discussed in this paper. This explains the large cost for each Newton iteration. However, due to the networking latency between Linz and Innsbruck, we expect no better performance at this stage except that we are using grid nodes connected via fast networking, e.g. Infiniband fabric or gigabit connection.

4.6. Visualization

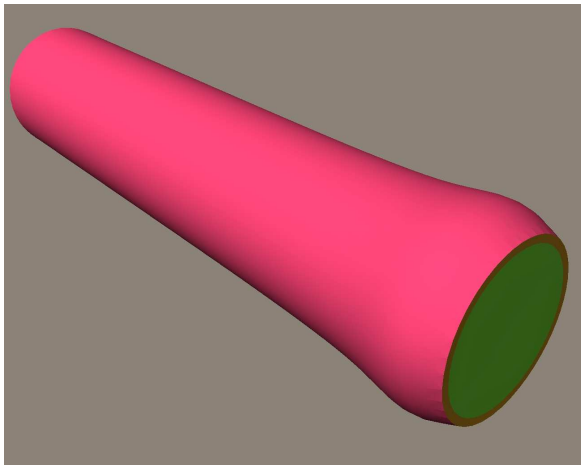
For visualization purposes the deformation is amplified by a factor of 12. It shows the pressure wave propagation on the fine mesh at different time levels in Fig.9.

Acknowledgments

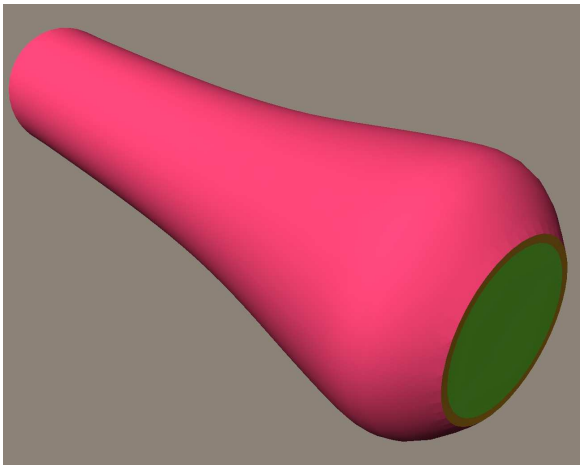
The authors gratefully acknowledge the Austrian Grid project funded by the BMWF (Federal ministry of Science and Research). Special thanks go to Ferdinand Kickinger who supplies us with meshes. Thank Frauk Kujundžić and Markus Baumgartner from JKU for their kind help on running the code on the Austrian Grid environment.

References

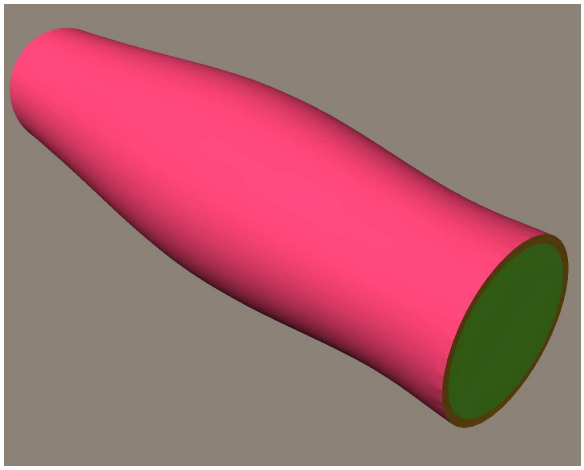
[1] Austrian academic computer network. http://www.aco.net/.



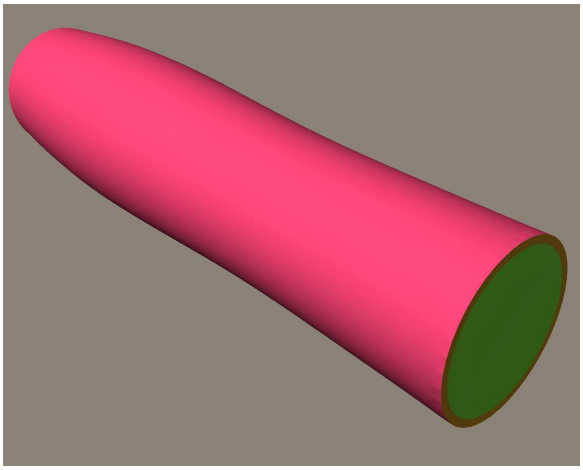
(a) $t = 1$ ms



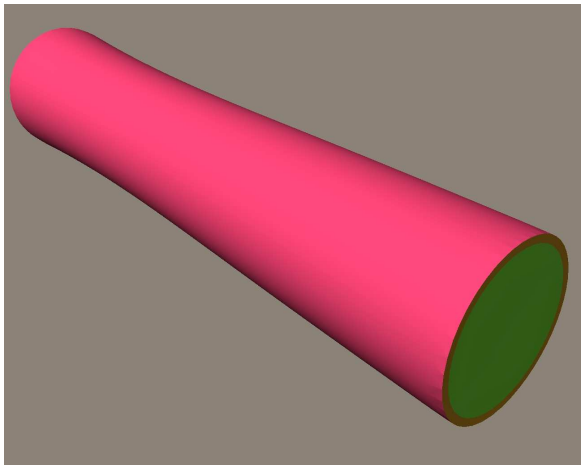
(b) $t = 5$ ms



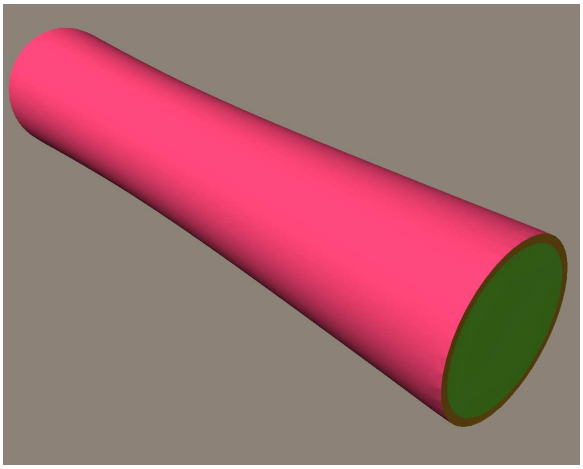
(c) $t = 10$ ms



(d) $t = 15$ ms



(e) $t = 20$ ms



(f) $t = 25$ ms

Figure 9. Simulation results at time $t = 1$ ms (upper left), $t = 5$ ms (upper right), $t = 10$ ms (middle left), $t = 15$ ms (middle right), $t = 20$ ms (lower left) and $t = 25$ ms (lower right).

[2] Austrian grid nodes information. http://agrid.uibk.ac.at/austriangrid/.

[3] Globus toolkit. http://www.globus.org/toolkit/.

[4] M. Baumgartner, C. Glasner, and J. Volkert. An Overview of the Austrian Grid Infrastructure. In J. Volkert, T. Fahringer, D. Kranzlmüller, and W. Schreiner, editors, *Proceedings of* 1^{st} *Austrian Grid Symposium*, pages 277–286, 2006.

[5] S. Deparis, M. Discacciati, G. Fourestey, and A. Quarteroni. Fluid-structure algorithms based on Steklov-poincaré operators. *Comput. Methods Appl. Mech. Engrg*, 195:5797–5812, 2006.

[6] C. Douglas, G. Haase, and U. Langer. *A Tutorial on Elliptic PDE Solvers and their Parallelization*, volume 16 of *SIAM Series on Software, Environments, and Tool*. Philadelphi, PA, 2003.

[7] C. Farhat, M. Lesoinne, and P. Le Tallec. Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: momentum and energy conservation, optimal discretization and application to aeroelasticity. *Comput. Methods. Appl. Mech. Eng.*, 157(1-2):95–114, 1998.

[8] C. Fellenstein and J. Joseph. *Grid Computing*. Prentice Hall Ptr, 2003.

[9] M. A. Fernández and M.Moubachir. A Newton method using exact Jacobians for solving fluid-structure coupling. *Comput. and Struct.*, 83(2-3):127–142, 2005.

[10] L. Ferreira, V. Berstis, and J. Armstrong. *Introduction to Grid Computing with Globus*. IBM Corp, 2003.

[11] C. Förster. *Robust methods for fluid-structure interaction with stabilised finite elements*. PhD thesis, University Stuttgart, 2007.

[12] I. Foster and C. Kesselman. *Grid 2 Blueprint for a New Computing*, volume 13 of *Elsevier Series in Grid Computing*. Morgan Kaufmann Pub, 2003.

[13] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *High Performance Computing Applications*, 15(3):200–222, 2001.

[14] U. Langer, T. Odaker, H. Yang, and W. Zulehner. Fluid-structure Interaction (FSI) Simulation Under the Grid Environment. Poster at the supercomputing conference '08, Austin, Texas, USA, November 15-21 2008.

[15] U. Langer and H. Yang. A parallel solver for the 3D incompressible Navier-Stokes equations on the Austrian Grid. SFB-Report 06-12, SFB "Numerical and Symbolic Scientific Computing" F013, Johannes Kepler University of Linz, 2006.

[16] U. Langer, H. Yang, and W. Zulehner. A grid-enabled solver for the fluid-structure interaction (fsi) problem. Technical Report 2009-07, Institute of Computational Mathematics, Johannes Kepler University of Linz, 2009.

[17] U. Langer, W. Zulehner, H. Yang, and M. Baumgartner. GStokes: A grid-enabled solver for the 3D Stokes/Navier-Stokes system on hybrid meshes. *Parallel and Distributed Computing, Sixth International Symposium on Parallel and Distributed Computing (ISPDC'07)*, 0:377–382, 2007.

[18] J. F. Gerbeau M. A. Fernández and C. Grandmont. A projection semi-implicit scheme for the coupling of an elastic structure with an incompressible fluid. *Int. J. Numer. Meth. Engng*, 69:794–821, 2007.

[19] D. P. Mok and W. A. Wall. Partitioned analysis schemes for the transient interaction of incompressible flows and nonlinear flexible structures. In *Trends in Computational Structural Mechanics (ed. by K. Schweizerhof, W. Wall)*, Barcelona, 2001. K. U. Bletzinger, CIMNE.

[20] F. Nobile. *Numerical approximation of fluid-structure interaction problems with application to haemodynamics*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2001.

[21] A. Quaini. *Algorithms for fluid-structure interaction problems arising in hemodynamics*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2009.

[22] B. Quatember and M. Mayr. GRID Software Solution for the Segmentation of the Co ronary Artery Tree in Biplane Angiograms. In *EUROCAST 2007*, pages 457–464, 2007.

[23] B. Quatember, M. Mayr, and W. Recheis. Patient-specific modelling and simulation of coronary haemodynamics. In *SpringSim 2008*, 573-580.

[24] B. Quatember and F. Veit. Simulation Model of Coronary Artery Flow Dynamics and Its Applicability in the Area of Coronary Surgery. In *EUROSIM 1995*, 945-950.

[25] H. Rosmanith and D. Kranzlmüller. glogin-A Multifunctional, Interactive Tunnel into the Grid. In *Proceedings of Grid 2004, 5th IEEE/ACM Intl.Workshop on Grid Computing*, pages 266–272, Pittsburgh, PA, USA, 2004. IEEE Computer Society.

[26] Y. Saad and M. H. Schultz. Gmres : A generalized minimal residual algorithm for solving nonsymmetric linear system. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.

[27] W. A. Wall. *Fluid-Structure Interaction with Stabilized Finite Elements*. PhD thesis, Institute of Structural Mechanics, University of Stuttgart, 1999.

[28] H. Yang. *Numerical Simulations of Fluid-structure Interaction Problems on Hybrid Meshes with Algebraic Multigrid Methods*. PhD thesis, Johannes Kepler University, 2009.

[29] H. Yang, W. Zulehner, U. Langer, and M. Baumgartner. A robust PDE solver for the 3D Stokes/Navier-Stokes systems on the grid environment. In *GRID '07: Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*, pages 145–152, Washington, DC, USA, 2007. IEEE Computer Society.

[30] T. E. Zezduyar, M. Behr, S. Mittal, and J. Liou. A new stategy for finite element computational involving moving boundaries and interfaces–the deforming- spatial-domain/space-time procedure. ii. *Computation of free-surface flows, two-liquid flows, and flows with drifting cylingers*, 94(3):353–371, 1992.

Latest Reports in this series

2009

2009-01	Clemens Pechstein and Robert Scheichl *Scaling Up through Domain Decomposition*	January 2009
2009-02	Clemens Hofreither, Ulrich Langer and Satyendra Tomar *Boundary Elements Simulation of Linear Water Waves in a Model Basin*	February 2009
2009-03	Huidong Yang and Walter Zulehner *Numerical Simulation of Fluid-Structure Interaction Problems on Hybrid Meshes with AMG*	April 2009
2009-04	Clemens Pechstein and Robert Scheichl *Analysis of FETI Methods for Multiscale PDEs - Part II: Interface Variation*	April 2009
2009-05	Alexsandr M. Matsokin and Sergey V. Nepomnyaschikh *Domain Decomposition Preconditioning for Well Models for Reservoir Problems*	June 2009
2009-06	Helmut Gfrerer *Generalized Penalty Methods for a Class of Convex Optimization Problems with Pointwise Inequality Constraints*	July 2009
2009-07	Ulrich Langer, Huidong Yang and Walter Zulehner *A Grid-enabled Solver for the Fluid-Structure Interaction (FSI) Problem*	August 2009

From 1998 to 2008 reports were published by SFB013. Please see
`http://www.sfb013.uni-linz.ac.at/index.php?id=reports`
From 2004 on reports were also published by RICAM. Please see
`http://www.ricam.oeaw.ac.at/publications/list/`
For a complete list of NuMa reports see
`http://www.numa.uni-linz.ac.at/Publications/List/`