Porting a Hemodynamics Simulator for a Grid Computing Environment

Paulo G. P. Ziemer, Ramon G. Costa, Pablo J. Blanco,
Bruno Schulze, Raúl A. Feijóo

{ziemer, ramongc, pjblanco, schulze, feij}@lncc.br

[1]National Laboratory of Scientific Computing - LNCC
Quitandinha, 25651-075 Petris - RJ, Brazil

Abstract. *The modeling of human cardiovascular system may represent a breakthrough in current way how physicians and researchers understand the general working of cardiovascular system and also could help the development of new treatments to cardiovascular pathologies through the detailed information provided by computer simulation. However, computer simulation of highly detailed models usually demands great computing power and may take long computing time, depending on the size and precision of the model. This way, in order to decrease computing time, the model resolver (SolverGP) uses MPI to share load among different processors. This work presents the main steps of the porting process of SolverGP to a grid environment and also covers the basic characteristics of the middleware gLite, the software tools used in the porting process, such as Watchdog, SecureStorage and MPI-Start and the principal strategies used to enable the execution of SolverGP in the grid environment from EELA-2 project.*

1. Introduction

Diseases related with the human cardiovascular system, such as coronary heart disease, are the leading cause of death worldwide [Mackay et al. 2004]. The use of computer modeling can provide exclusive and detailed insight about the blood flow behavior to Medicals and Researchers, giving them new perspectives about treatments of pathologies and also improve the general understanding of the main physical phenomenons of human cardiovascular system. The computational hemodynamics field presents an increasing need for reproducing accurately physiological blood flow regimes encountered in the cardiovascular system, as well as to simulate coupled global/local phenomenons, with the purpose of retrieving as much information as possible from the numerical simulations [Blanco et al. 2009a].

This way, the HeMoLab project [HeMoLab] was created with the purpose of developing computational tools that could make easier the process of creation, edition and visualization of results of models of the Human Cardiovascular System (HCS). Then, it was developed the software named HeMoLab as an extension of Kitware's visualization software Paraview [Moreland 2009]. HeMoLab allows the creation and customizing of general simplified models (1D models), restricted detailed models (3D), and coupled models (1D-3D). It is also possible to simulate cardiovascular related pathologies, e.g., artery aging process, aneurysms and stenosis, and surgical procedures such as insertion of stent, creation of artery bypass and etc. The models created/customized by HeMoLab are saved in special format files and used after as input to the software that resolves the simulation (SolverGP). Visualization of produced simulation data is a crucial stage in order to allow a better understanding of main physicals phenomenons that occur within determined model. Thus, several schemas of scientific data visualization have been developed, such as blood particle tracing effects, artery flow 3D visualization with aid of CAVE (Cave Automatic Virtual Environment) and etc.

The experience with the computer simulation of 3D models have shown that the demanded processing time is of the order of weeks or even months in case of more complex models. This way, the use of high performance computing techniques such grid computing is important to the success of adoption of the tool by Medical community. Grid computing [Kesselman and Foster 1998] is a paradigm that proposes aggregating geographically distributed, heterogeneous computing, storage and network resources to provide unified, secure and pervasive access to their combined capabilities. We hope that using grid computing applied to the simulation of cardiovascular models may provide speedup for the execution of computer simulations (since a larger number of processors is available in some grid sites - if compared to the regular number of processors commonly used) and also increase the number of simulations made concurrently.

In order to execute an application in a grid environment, the software must be carefully analyzed by looking for requirements that if not well ”carefully observed” may cause the fail of the execution of the application in the remote site. Examples of restrictions include the use of specific libraries (which may be not ”installed” in remote computers), interaction of application with local Operating System (OS) (for instance, some OSs like Linux allows by default the use of a limited amount of dynamic memory).

The HeMoLab was selected among e-science applications from Latin America to participate of the Second EELA-2 Grid School [Grid-School 2009]. This event aims to instruct how applications that usually runs on centralized computing resources, such as homogeneous local clusters, may be adapted to be executed at decentralized heterogeneous distributed computing resources from EELA-2 project [EELA]. During the school, it was presented the basic and advanced characteristics of the gLite middleware [Meglio 2007] like searching of computing resources, jobs types and specifications and etc.

In this context, this paper describes the main steps of the porting process of SolverGP to gLite environment and also covers the main characteristics of gLite and the tools used during porting process.

2. The HeMoLab project

2.1. Modeling the Human Cardiovascular System

The simulation of the HCS is performed through the use of the following types of models: unidimensional (1D) and three-dimensional (3D). The 1D model represents the cardiovascular system in a simplified fashion, where main arteries of human body are modeled as a set of straight lines (segments) and points (terminals). Segments are used to represent the individual arteries, while terminals are used to model the connection points (between arteries that have different properties) or are used to represent the effects of capillary artery sets. Each segment is sub-divided in elements, called nodes, where the values of blood pressure, velocity and artery area are calculated during the simulation time. The human heart is also modeled through an element that imposes a regular pressure variation to the whole arterial tree. Each artery has mechanical properties, such as elastin (that indicates how elastic the artery is) and also geometric parameters like artery radius in some given point, wall thickness and etc.

3D models are used to provide a greater level of detail of the blood flow behavior inside a specific artery structure. The geometry of a 3D model may be obtained through the analysis and processing of medical images, such as computer tomography, magnetic resonance, intravascular ultrasound and etc. The use of patient medical image presents the advantage of creation of custom model this way allowing a more realistic hemodynamics study. The artery to be modeled is identified on the image with the use of image segmentation algorithms, such as region growing, topological derivative [Larrabide et al. 2008] and etc. As output of the algorithm, a new image is produced, where the artery area is easily identifiable and then allowing the use of a isosurface algorithm that finally generates the triangle mesh. This mesh usually has not good quality (i.e. triangles not corrected aligned with others - needle shape- or the number of triangles in some surface area is not refined enough to correctly model the artery wall behavior). In order to solve the stated problems, mesh quality improvement algorithms are used in mesh refinement, mesh smoothing, removal of needle triangles and etc. Figure 1, shows an example of a surface model containing cerebral arteries, where one of them presents an aneurysm.

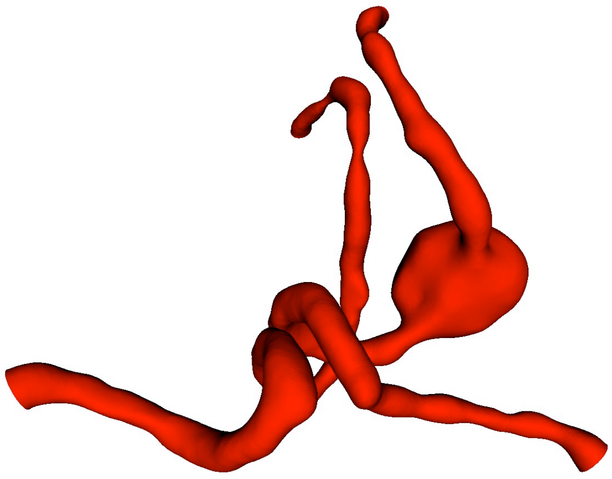

Figure 1. Model Geometry of two cerebral arteries containing an aneurysm.

Once the surface mesh has good quality, it is now possible to obtain the volume mesh using Delaunay triangulation process. The spatial components of velocity and displacement of blood flow and also the value of blood pressure are calculated for each created point inside the surface. The recently created internal structure is named of Volume mesh. After the surface and volume meshes have been created, the mechanical parameters of the model are defined. These include, artery wall type, that can be defined as rigid or deformable allowing the simulation of artery wall movement imposed by the pressure variation (set by the heart). A set of surface triangles can have custom properties, for instance, the modification of value of elastin allows the simulation of aging process in human arteries. At this time, the model is ready to enter the computer simulation stage, where the results of blood dynamics during simulation time are calculated.

3D models can be used in a standalone way or integrated with 1D models. 1D-3D integrated models are called of Coupled models [Blanco et al. 2009b], where the 3D elements from inflow covers are connected with Terminals or Nodes from 1D model. This way, the simulation result data from the 1D model points, where the 3D model is currently connected with, are used as boundary condition to the 3D model. If a 3D model is used as standalone, then the boundary conditions must be setup in advanced. These must be a constant or variant value of blood pressure or a blood velocity value.

2.2. Simulations Results

After the computer simulation is done, computer graphics techniques can be applied to the produced data and then allow and make easier the process of getting insight about data. In Figure 2, it is possible to see an example of such process on the simulation results of an artery that contains an aneurysm. Streamlines indicate the instantaneous trajectory of blood particles and the value of velocity magnitude. Warping is also used to indicate the value of velocity on the transverse sector of the aneurysm area.

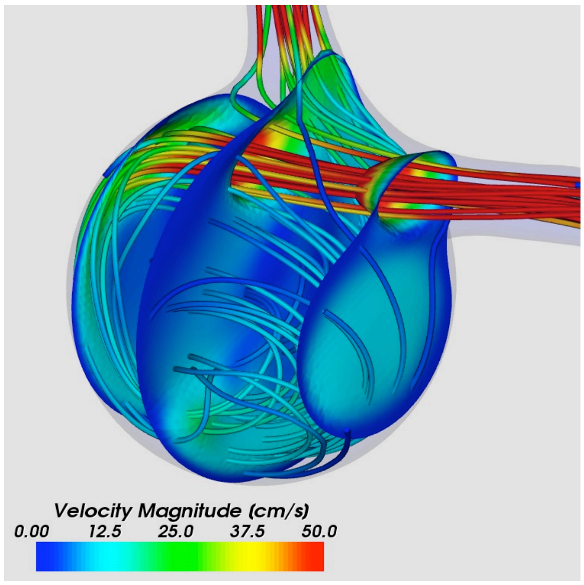

Figure 2. Example of an aneurysm model with simulation results

2.3. General Purpose Solver: SolverGP

The computer simulation of a model is performed through the resolution (using Finite Element Method - FEM) of a Navier Stokes equations system (that describe the motion of fluids, such as blood inside of an artery). Numerical methods are used to provide an approximated result of the analytic result of the equation system. So, these numerical methods were implemented using the programming language Fortran 90 in a system that was named of SolverGP (General Purpose Solver) and this uses the PETSc library [Balay et al. 2008] to make the distributed resolution through MPI (Message Passing Interface) [Forum 1994] of the equation system and also the Parmetis library [Karypis et al. 2003] which is responsible for mesh partitioning.

In order to start a computer simulation, SolverGP reads the set of files that describe the model parameters such as geometrical, mechanical and simulation parameters. Each simulation period (usually one period refers to one heart beat) is sub-divided in time steps (a typical value for it is 640) and for each time step, SolverGP updates the file that stores the simulation results at disk (Dataout.txt) with the values of velocity, displacement and pressure for each volume point. The computing of the simulation can be initialized with some previous calculated result or with a defined state for the model, using for that the file Inifile.txt. This file is updated for each time step and this procedure allows the resuming of a previous computing that has failed.

The SolverGP uses MPICH2 [Gropp et al. 2009] as the MPI standard implementation. MPICH2 programs require that mpd daemons be running in the machines that form the computing processing ring. In Code 1, are shown the needed commands to perform the execution of SolverGP in such environment. In the listed example, the mpdboot command initializes six mpd process in the list of machines specified by the file machine.txt. After that, the mpirun command triggers the process SolverGP and also specifies all the PETSc parameters, such as the maximum number of iterations for the current numerical method `(ksp_max_it)`, parameters related with the linear solver in use `(ksp_gmres_restart`, `ksp_rtol`, `ksp_atol)`, information about convergence `(ksp_monitor)` and finally the name of log file used to store status information `(run.parallel.log)`.

Code 1. Commands used to initialize a SolverGP computer simulation

```
mpdboot -n 6 -f ~./machines.txt
mpirun -np 6 SolverGP.x -ksp_gmres_restart 100 -ksp_max_it 1000 \
-ksp_rtol 1.0e-5 -ksp_atol 1.0e-10 -ksp_monitor -log_summary
</dev/null> & run.parallel.log &
```

The computer simulation of 1D models does not demand high performance computing power. Those simplified models are commonly solved using a sequential SolverGP version on ordinary desktop computers and the solving time is about 10 to 30 minutes. However, more complex models, such as the 3D or coupled models, may demand a high computing power in order to be solved. For instance, the computer model of an aorta, the largest artery of the human body, has approximately 500,000 points and then originating a system with 3,500,000 equations. The computing of just one heart beat period in such model takes approximately 35 days (with 640 time steps) - using eight processors in a 2X quad-core Xeon 3GHz / 64GB RAM machine.

3. Technologies and concepts

During the HeMoLab porting process, several tools were used to support the creation and configuration of grid services, such as Secure-Store Client [Scardaci and Scuderi 2007] and Watchdog [Bruno et al. 2009].

Secure-Store [Scardaci and Scuderi 2007] is used to provide encryption of the files stored at the Storage Elements (SE - grid sites used only for storage). This tool provides command line interface and API for the programming langauges: Java, C++, and etc. Commands are integrated in the gLite User Interface to encrypt/upload and decrypt/download files using the user private key as cryptographic key. The encryption of files is an important action, since models may contain sensitive information that must be protected from unauthorized access.

The computing of SolverGP may fail (diverge) if simulation parameters are not correctly set. Once that happen the simulation process must be restarted with a new set of simulation parameters. This way, it is necessary that the user keep a certain level of control over how the simulation may develop. In a local cluster, this process is straightforward. However, to monitor files that are located in remote sites is a difficult task since files are not directly available to user. This way, the Watchdog [Bruno et al. 2009] tool was used to provide real time information about remote computing. This tool is a shell script which is included in the JDL (Job Description Language) main script and presents the following features: i) it starts in background before to run the long term job; ii) the watchdog runs as long as the main job; iii) the main script can control, stop and wait until the watchdog has finished; iv) easily and highly configurable and customizable; v) the watchdog does not compromise the CPU power of the Worker Nodes;

Before running a MPI job in a grid Computing environment, one must know some characteristics of remote site, such as, if the remote home disk area is shared between the assigned cluster nodes, the list of machines available (selected by local scheduler) to current session and etc. In order to hide the complexity of establishing the necessary infrastructure to the execution of a MPI job, the set of scripts MPI-Start [HLRS] were developed for the Interactive European Grid Project [I2G]. These scripts supports the initialization of MPI jobs from the following implementations: MPI, OpenMPI, LAM-MPI and MPICH2.

The use of scripts in MPI session configuration allows the user to focus principally in the application related tasks. The number of processing cores is defined through the JDL file and the name of the executable and any parameters are specified using environment variables in the script file MPI-start-wrapper.sh. The functions, pre_run_hook and pos_run_hook from the script file MPI-hooks.sh make possible to specify the previous and post job operations, such as, pre-compiling the MPI code remotely or making some filtering on the data produced after the application has finished its execution.

Beside the present tools, there are several tools to support the creation of a grid environment [JRA1]: Dirac, Transactional Grid Storage Access Framework (GSAF), Grid2Win User Interface, and etc.

4. Aspects related with Software Porting Process

4.1. The structure

The EELA-2 project (E-science Grid facility for Europe and Latin America) aims at building and operating a production-quality grid infrastructure, which provides computing and storage resources from selected partners across Europe and Latin America. Currently, the EELA-2 Consortium encompasses seventy eight member institutions from sixteen countries (five from Europe and eleven from Latin America). GLite [Burke et al. 2009] provides a framework for building grid applications tapping into the power of distributed computing and storage resources across the Internet.

At the time of this writing, twenty four resources centers were connected to the EELA-2 infrastructure and with more than 9600 job slots (CPUs) and around 10 PB of disk storage [gStat]. As SolverGP requires the use of MPICH2 library, then the number of Computer Elements (CE - grid computer node that actually runs the job) that can be used for the execution of jobs is reduced. The Table 1 shows CEs that are suitable for HeMoLab simulation jobs.

Table 1. list of sites that can run jobs from HeMoLab

CE	Number of Cores
ce-eela.ciemat	216
ce01.eela.if.ufrj.br	216
ce01.unlp.edu.ar	10
eelaCE1.lsd.ufcg.edu.br	8
grid001.cecalc.ula.ve	48
grid012.ct.infn.it	200
gridgate.cs.tcd.ie	715

4.2. The Architecture

A service layer was created between the SolverGP and the services provided by EELA-2. The Figure 3 illustrates the proposed architecture:

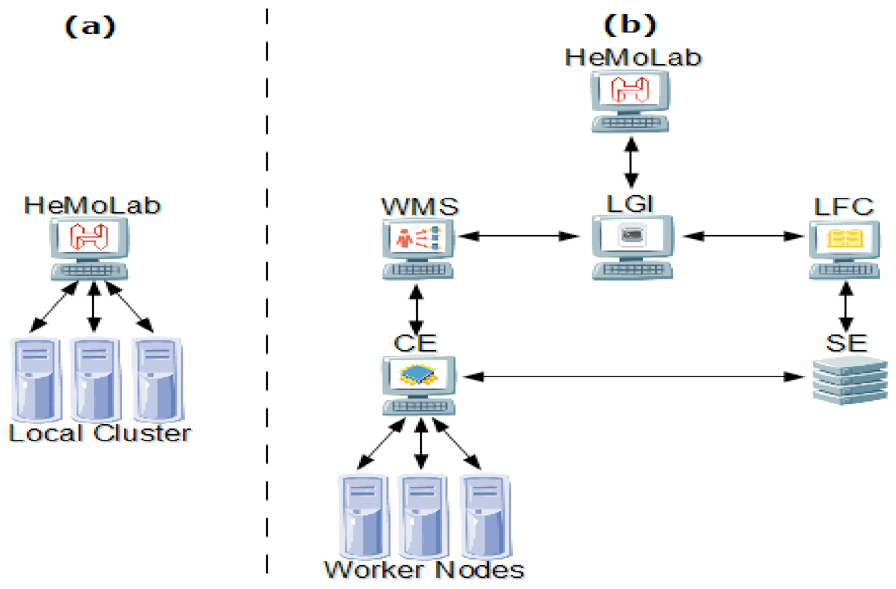

Figure 3. (a) The standard HeMoLab architecture; (b) The proposed architecture that integrates HeMoLab and the services of the EELA-2

The Local Grid Integration (LGI) is the component that retrieves input files and also selects the grid site that best matches with SolverGP execution requirements. At the selected CE, scripts are executed (prior the execution of simulator) in order to correctly configure the environment of the Worker Nodes. The LGI interacts with the Workload Management System (WMS) which has set of grid middleware components that are responsible for the distribution and management of tasks across grid resources.

The SE and LCG File catalog (LFC) are components that provides the storage services: storage of big files (files sent and received through JDL have a maximum size limitation) and availability (the use of replicas is allowed)/ localization of data files (catalog contains the logical to physical file mappings).

Summarizing, the LGI is responsible for:

1. creation of the interface between HeMoLab and CE - used to launch the remote script that executes the SolverGP;
2. encryption of input files and decryption of output data through the use of private key found at User's digital certificate;
3. triggering of jobs monitoring, which is required by HeMoLab in order to monitor the simulation process.
4. receiving the output files generated remotely.

4.3. The Workflow

The Figure 4 shows the workflow in the execution of a HeMoLab numerical simulation as well its interaction with the grid environment services.

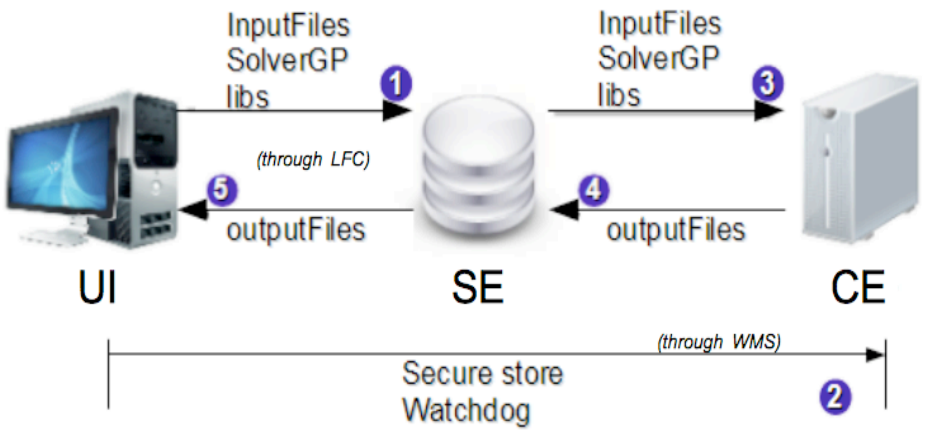

Figure 4. The workflow for a job submission

The User Interface (UI) is the submission machine (grid entry point) and is considered part of the WMS. The LGI is located at UI and contains the JDL file that describes the file names and also details the type of job to be executed. According to the enumeration in Figure 4, follows a brief workflow description:

1. Before submitting a job to remote processing, three files must be stored in SE: i) inputFiles - files generated by HeMoLab that contain model and simulation information required for the numerical solver; ii) SolverGP - the executable file of the SolverGP (responsible for the numerical simulation); iii) libs - extra libraries required by the execution of SolverGP (only used if SolverGP is linked using shared libraries). The input files sent to the SE are encrypted locally in the UI, using the Secure Storage-client and the encryption key is the user's private key.

2. A JDL file defines the files to be sent to CE through WMS. The parameters inputSandBox and outputSandBox lists the files to be sent and received by UI, respectively. Three files are sent to CE through the WMS: i) solver.sh bash script responsible for the downloading of files stored in the SE and also launchs the execution of the SolverGP; ii) watchdog - files used to enable the online monitoring of the execution steps in the numerical simulation; iii) secure store - libraries required to decrypt the input files.

3. the remote script is executed, the files are moved from SE to CE, the numerical simulation is launched and finally the output data is generated. At the moment that the output data is being generated, it is possible to monitor the convergence of the simulation process, through the use of a Watchdog session and this way monitor problems that may be occurred in the numerical simulation, or check the output files for immediate analysis.

4. The output files are sent to the SE and the log files are returned to the UI through WMS.

5. Output files are now moved from SE to the local UI.

5. Conclusions and Future Work

Diseases related to the human cardiovascular system, such as coronary heart disease, are the leading cause of death worldwide. This way, the modeling of the behavior of such system may provide us a better understanding about how patologies develop and this information may be used, for instance, in the development of new treating procedures. The modeling process can make use of simplified general models (1D), restricted detailed models (3D), and coupled models (1D-3D). Those kind of models allow to perform study of cardiovascular related pathologies, e.g., artery aging process, aneurysms and stenosis, and the simulation of surgical procedures such as insertion of stent, creation of artery bypass and etc. However, the numerical simulation of 3D or Coupled models have shown that the demanded processing time is of the order of weeks or even months in a more complex models. This time factor definitely is a limiting issue to spreading the use of HeMoLab tool. This way, the use of grid computing is very important since that the expected improvements of simulation process may help the adoption of the tool by Medical community.

This paper presented the main steps performed to allow the execution of SolverGP in EELA-2 computer grid and also the tools used in this process, such as watchdog - that is used to extract ”on the fly” information from remote simulation- and SecureStorage - which allows the encryption of the sent data to remote grid sites. During grid school, only small models (that are solved quickly) were tested in order to allow the quick reply of the submitted jobs. As future work, it is expected the execution of more computing intense simulations needing for that the increase of the number of processors. Another very important work to do is the specification of the optimum number of processors used for the resolution of a given model, i.e., to make a deeper study of the speedup curve of SolverGP. The expected output from this study is to know in advanced the number of processors that faster solves a model with a specific size. It is also expected to use the gLite programming API [gLite 3.1 API Usage Examples] in order to allow the submission and monitoring of jobs directly from the HeMoLab application (client side). This brings the use of grid technology to users that are not aware that the simulation is actually being performed in a remote computing grid.

Among the main contributions, we can highlight the possibility of simulating multiples jobs in a parallel way, one in each remote cluster. The number of clusters that supports MPICH2 applications is expected to increase in the near future. It is being analyzed the possibility of dividing a single simulation among several clusters, allowing this way the explicit parallelization of simulation sectors that are data independent.

As main result of this initial work, is expected that the use of the grid enabled application may spread the use of HeMoLab tool among Brazilian medical research community, since a larger number of simulations are expected to be executed at same time and also to obtain smaller times from the execution of the simulations. In order to verify the possible improvements, further work and tests shall be performed.

References

Balay, S., Buschelman, K., and et. al. (2008). *PETSc - Portable, Extensible Toolkit for Scientific Computation Users Manual.* Argonne National Laboratory, US. `http://www.mcs.anl.gov/petsc/petsc-as/`.

Blanco, P. J., Pivello, M. R., Urquiza, S. A., and Feijóo, R. A. (2009a). Building coupled 3d–1d–0d models in computational hemodynamics. In *1st International Conference on Mathematical and Computational Biomedical Engineering - CMBE2009*, Swansea, UK.

Blanco, P. J., Pivello, M. R., Urquiza, S. A., and Feijóo, R. A. (2009b). On the potentialities of 3d-1d coupled models in hemodynamics simulations. *Journal of Biomechanics*, 42(7):919–930.

Bruno, R., Barbera, R., and Ingra, E. (2009). Watchdog: A job monitoring solution inside the eela-2 infrastructure. In *Proceedings of the Second EELA-2 Conference*, Choroni, VE.

Burke, S., Campana, S., and et. al. (2009). *gLite User Guide.* `http://glite.web.cern.ch/glite/`.

EELA. E-science grid facility for europe and latin america. `http://www.eu-eela.eu/`.

Forum, M. P. (1994). Mpi: A message-passing interface standard. Technical report, Knoxville, TN, USA.

gLite 3.1 API Usage Examples. Mpi-start. `https://grid.ct.infn.it/twiki/bin/view/GILDA/APIUsage`.

Grid-School (2009). Second eela-2 grid school. `http://indico.eu-eela.eu/conferenceDisplay.py?confId=200`.

Gropp, W., Lusk, E., and et. al. (2009). *MPICH2 - Message-Passing Interface (MPI) Implementation Users Guide*. Argonne National Laboratory, US. `http://www.mcs.anl.gov/research/projects/mpich2/`.

gStat. gstat monitoring page - grid information system. `from:http://goc.grid.sinica.edu.tw/gstat/eela/`.

HeMoLab. Hemodynamics modeling laboratory. `http://hemolab.lncc.br`.

HLRS. Mpi-start. `http://www.hlrs.de/organization/av/amt/research/mpi-start`.

I2G. Interactive european grid project. `http://www.i2g.eu/`.

JRA1. Eela jra1: Development of services for applications and infrastructure. `http://glite.web.cern.ch/glite/documentation/`.

Karypis, G., Schloegel, K., and Kumar, V. (2003). *ParMETIS - Parallel Graph Partitioning and Fill-reducing Matrix Ordering*. University of Minnesota, Department of Computer Science and Engineering. `http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview`.

Kesselman, C. and Foster, I. (1998). *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers.

Larrabide, I., Feijóo, R. A., Novotny, A. A., and Taroco, E. A. (2008). Topological derivative: A tool for image processing. *Comput. Struct.*, 86(13-14):1386–1403.

Mackay, J., Mensah, G. A., Organization, W. H., for Disease Control, C., and Prevention (2004). The atlas of heart disease and stroke - deaths from coronary heart disease. *World Health Organization, Geneva*, page 112.

Meglio, A. D. (2007). The egee glite middleware. *Second Meeting of the Spanish Thematic Network on Middleware for Grids, Barcelona, Spain (2007).*

Moreland, K. (2009). *The ParaView Tutorial - A Parallel Visualization Application*. Sandia National Laboratories. `http://www.paraview.org/`.

Scardaci, D. and Scuderi, G. (2007). A secure storage service for the glite middleware. *International Symposium on Information Assurance and Security*.