Numerical Simulation for eHealth: Grid-enabled Medical Simulation Services

Siegfried Benkner[a], Gerhard Engelbrecht[a], Werner Backfrieder[b], Guntram Berti[c], Jochen Fingberg[c], Greg Kohring[c], Jens Georg Schmidt[c], Stuart E. Middleton[d], David Jones[e], John Fenner[e]

[a]Institute for Software Science, University of Vienna, Vienna, Austria

[b]Department of Biomedical Engineering and Physics, University of Vienna, Austria

[c]C&C Research Laboratories, NEC Europe Ltd., St. Augustin, Germany

[d]IT Innovation Centre, University of Southampton, UK

[e]Medical Physics and Engineering, The University of Sheffild, UK

The European GEMSS Project[1] is concerned with the creation of medical Grid service prototypes and their evaluation in a secure service-oriented infrastructure for distributed on-demand supercomputing - the GEMSS test-bed. The medical prototype applications include maxillo-facial surgery simulation, neuro-surgery support, radio-surgery planning, inhaled drug-delivery simulation, cardiovascular simulation and tomographic image reconstruction. GEMSS will enable the wide-spread use of these computationally demanding tools originating from projects such as BloodSim, SimBio, COPHIT and RAPT as Grid services. The numerical High-Performance Computing core includes parallel Finite Element software and Computational Fluid Dynamics simulation as well as parallel optimization methods and parallel Monte Carlo simulation. Targeted end-user groups include bio-mechanics laboratories, hospital surgery/radiology units, diagnostic and therapeutic departments, designers of medical devices in industry as well as small enterprises involved in consultancy on bio-medical simulations. GEMSS addresses security, privacy and legal issues related to the Grid provision of medical simulation and image processing services and the development of suitable business models for sustainable use. The first prototype of the GEMSS middleware is expected to be released in February 2004.

1. Introduction

Computationally demanding methods such as parallel Finite Element Modelling, parallel Computational Fluid Dynamics and parallel Monte Carlo simulation are at the core of many advanced bio-medical simulation applications. Often, however, such applications have a very limited clinical impact, because there is no convenient or practical way for the typical medical end user to access the necessary software and hardware resources. Grid technology has the potential to provide medical practitioners and researchers with access to advanced simulation and image processing services for improved pre-operative planning and near real-time surgical support by providing on-demand transparent access to remote HPC hardware and services over the Internet. The Grid will also allow computational resources to be brought to the medical technology industry, already using bio-numerics for virtual prototyping, but requiring larger-scale compute resources due to the growth in the complexity of design problems made tractable by numerical methodology advances.

The European GEMSS Project [7] is concerned with the creation of medical Grid service prototypes and their evaluation in a secure service-oriented infrastructure for distributed on-demand supercomputing. The medical prototype applications include maxillo-facial surgery simulation, neurosurgery support, radio-surgery planning, inhaled drug-delivery simulation, cardio-vascular simulation and advanced image reconstruction. GEMSS will enable the wide-spread use of these computationally demanding tools originating from projects such as BloodSim [4], SimBio [8], COPHIT [5] and

[1] GEMSS, EC project number IST-2001-37153, is a 30 month project which commenced in September, 2002. Project web site: `http://www.gemss.de`

RAPT [6] as Grid services.

The GEMSS Grid infrastructure and middleware is being built on top of existing Grid and Web technologies, maintaining compliance with standards thereby ensuring future extensibility and interoperability. Furthermore, GEMSS aims to anticipate privacy, security and other legal concerns by examining and incorporating into its Grid services the latest laws and EU regulations related to providing medical services over the Internet.

The rest of this paper is organized as follows: Section 2 discusses the provision of medical simulation services via the Grid. Section 3 presents an overview of the GEMSS Grid infrastructure currently under development. Section 4 discusses the HPC simulations considered in GEMSS and outlines how these applications are being Grid-enabled. Section 5 discusses related work followed by conclusions in Section 6.

2. Providing Medical Simulation Services via the Grid

GEMSS is concerned with creating an environment in which computationally demanding tools relevant to the health sector can be made easily accessible to a wide spectrum of users.

2.1. Benefits of Grid-enabled Simulation Services

In a Grid scenario a medical end-user (client) would simply use a browser or a client software to access a service via a provider's portal or server. In this way doctors, clinicians or medical researchers can be provided with advanced tools at their workplace through easy-to-use interfaces without requiring their institutions to invest in expensive HPC hardware and related IT or engineering specialist know-how. End users will only pay a negotiated price per use. Furthermore, a reliable, pervasive and interoperable Grid infrastructure can accommodate new services and updates to existing ones immediately as they become available.

2.2. GEMSS High-Level Grid Architecture

The GEMSS architecture is based on a client/server topology employing a service-oriented architecture as shown in Figure 1. It relies on Web Service technology that will allow integration, via OGSA [13], with various hosting platforms such as UNICORE [21] and GLOBUS [12].

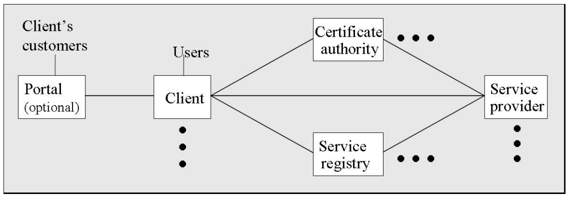

Figure 1. GEMSS High-Level Architecture

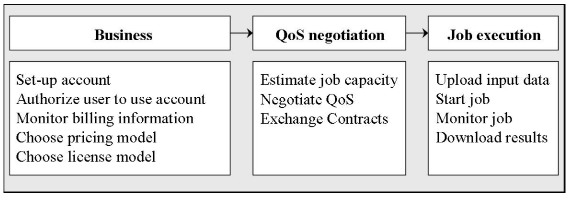

Figure 2. Three step job execution process

At its simplest, the Grid client is an internet enabled personal computer loaded with client software that permits communication with a service provider through the GEMSS middleware. The client side applications handle the creation of the service input data and visualization of the service output data.

Service providers expose numerical simulation applications running on HPC hardware as generic application services accessible over the network. Generic application services provide support for quality of service negotiation, data staging, job execution, job monitoring, and error recovery.

One or more service registries will be employed holding a list of service providers and the services they support. The certificate authority, which is usually managed by a third party, provides certificate authentication after appropriate identity checks. In order to provide a custom interface for running Grid jobs, an optional client portal may be utilized. However, even if the GEMSS environment is accessed via a portal, the client will still be billed for jobs run through a portal, making it the responsibility of the client to pass these costs on to its customers.

The GEMSS design supports a three step process to job execution (see Figure 2). In the initial business step, accounts are opened, payment details are fixed and a pricing model can be chosen. Next, a job's quality of service and price, if not subject to a fixed price model, is negotiated and agreed. Once a contract is in place, the job itself can be submitted and executed.

2.3. Prerequisites for Grid-enabling Medical Simulation Applications

In order to Grid-enable a simulation application under the GEMSS environment it must be possible to partition the application between client and server. Data must be separated from the application software which implies that I/O operations access files only by relative path names. In order to comply with the GEMSS QoS model, for each application a machine specific performance model must be provided, which allows the determinination of performance estimates (e.g. run time, memory requirements, etc.) on the basis of meta data charaterising a service request. Support for error recovery, if required, must be ensured through an appropriate checkpoint/restart facility within the application. Moreover, the application must be compliant with the GEMSS security model and must support the integral business model as well as the Quality of Service model as described in the following sections.

3. The GEMSS Infrastructure

The GEMSS middleware exposes medical simulation applications installed on various Grid hosts as services which support a common set of methods for data staging, remote job management, error recovery, and QoS support. GEMSS services are defined via WSDL and securely accessed using SOAP messages. HTTPS will be initially used to securely transmit SOAP messages, and the WS-Security standard will be applied where possible. For large file transfers SFTP will be investigated, as well as SOAP attachments.

3.1. Client and Service Provider Infrastructure

Figure 3 depicts the main architectural components of the GEMSS infrastructure for the client and the service provider.

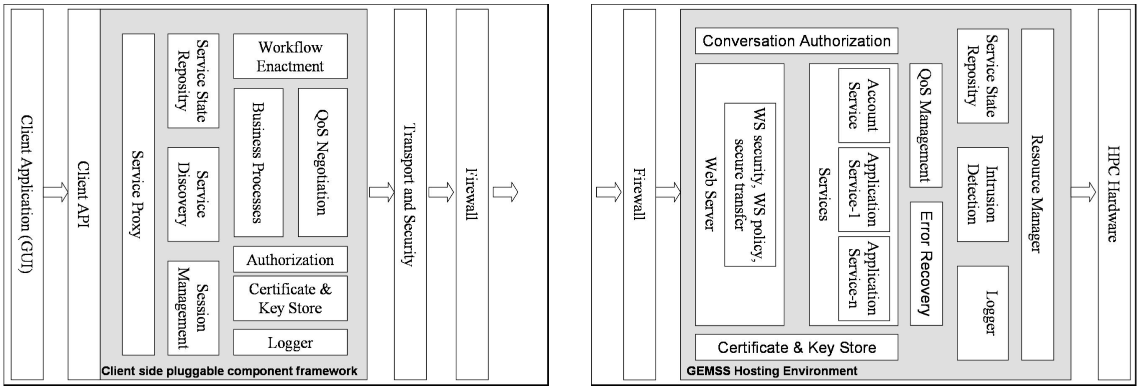

Figure 3. GEMSS Client (left hand side) and Service Provider Infrastructure (right hand side)

On the service provider side, medical simulation applications are exposed as Web Services and hosted using a secure web server and a service container (usually Apache and Tomcat Axis). Services are executed under the direction and orchestration of the client, subject to availability of resources and authority to use them. The quality of service management component handles reservation with the resource manager (job scheduler) and provides input to the quality of service negotiation process so that sensible bids can be made in response to client job requests. The error recovery component handles checkpointing and re-starting of services if required. The logger component manages a database for logging auditable information and performs most of the low level event logging for intrusion detection. The service state repository component manages a conversational state database that contains information about any client-service conversation allowing it to be resumed at a later time if the user logs off. The provision of applications is based on the concept of generic application services as described in detail in the next section.

The GEMSS client application programming interface hides most of the complexity of dealing with the remote services from the application developer by providing appropriate service proxies. Service proxies are in charge of discovering services and negotiating with grid services to run jobs. The session management component manages client sessions and maintains a security context, authenticating the current user and providing access criteria for the certificate and key stores. A service discovery component is available for looking up suitable services in a registry. The workflow enactment component is in charge of running business and negotiation workflows, as well as service orchestration.

The client typically runs a business workflow to open negotiations with a set of service providers for a particular application. The quality of service negotiation workflow is then run to request bids from all interested service providers who can run the clients jobs subject to QoS criteria required by the client; this results in a contract being agreed with a single service provider. The client then uploads the job input data to the service provider and starts the server side application by calling appropriate methods of the service.

The client infrastructure is based on a pluggable client side component framework which is described in detail in Section 3.3.

3.2. QoS-enabled Generic Application Services

A generic application service is a configurable software component which exposes a native application as a service to be accessed by multiple remote clients over the Internet. A generic application service provides common methods for data staging, remote job management, error recovery, and QoS support, which are to be supported by all GEMSS services. In order to customize the behavior of these methods for a specific GEMSS application, an XML application descriptor has to be provided. Besides general information about a medical simulation service, the application descriptor specifies the input/output files, the script for initiating job execution, and a set of performance-relevant application parameters required for QoS support.

GEMSS services support a model and process for agreeing, dynamically and on a case-by-case basis, various QoS properties, including service completion time, cost, availability and others. For this purpose, each GEMSS service provides methods for enabling clients to negotiate required QoS properties with a service provider before actually consuming a service. QoS enabled GEMSS services are capable of providing an estimated service completion time, based on meta data about a specific service request (e.g. image size, required accuracy, etc.) supplied by the client.

The QoS support infrastructure for GEMSS is based on QoS contracts (XML), request descriptors (XML), and performance models, and relies on a resource scheduler that supports advanced reservation. For each GEMSS application, the service provider or application developer has to specify a set of application specific performance parameters in the XML application descriptor. For example, in the case of an image reconstruction service, performance parameters typically include image size and required accuracy (i.e. number of iterations). In order to compute various QoS properties (e.g. service completion time) on the basis of the specified performance parameters, a machine-specific performance model has to be provided. Since in general, it will not be possible to build a simple analytical performance model for all GEMSS applications, we plan to build a data base relating typical problem parameters to resource needs like main memory, disk space and running time, which will initially be populated using data from test cases.

During QoS negotiation, the client has to supply a request descriptor containing concrete values for all performance-relevant parameters specified in the application descriptor. Moreover, the client has to pass to the service provider an initial QoS contract, specifying the required QoS properties and other conditions to use the service. On the service side, the request descriptor is fed into the performance model in order to determine whether or not the client's QoS requirements can be fulfilled.

A generic application service is realized as a Java component, which is transformed automatically into a Web Service with corresponding WSDL descriptions and customized for a specific GEMSS application using the XML application descriptor. In order to provide a native GEMSS application as a Web/Grid Service, the application has to be installed on some Grid host, and a job-script to start the application as well as an XML application descriptor have to be provided. Finally, the generic application service has to be deployed within an appropriate hosting environment e.g. Apache Tomcat/Axis. As a result, the native application will be embedded within a generic application service and will be accessible over the Internet. In the future, GEMSS services will be extended in order to be compliant with the Open Grid Services Architecture (OGSA).

3.3. Client Side Component Framework

On the client side, the GEMSS Project has chosen a flexible approach built around an SDK consisting of pluggable components. Using the SDK, developers can easily integrate Grid functionality into existing applications. The SDK is conservative, in that it does not expose any low level Grid concepts which applications should not need to deal with, while at the same time providing applications full control over higher level abstractions. Among the high level components identified to date are: Business Processes, Service Discovery, QoS Negotiation, and Workflow Enactment. The GEMSS SDK exposes the interfaces for these components to the applications, while keeping their underlying implementation in terms of Web Services hidden from view.

All the components making up the SDK are pluggable in the sense that they can be replaced by any implementation which supports the well defined public interfaces and properly replicates a component's documented behaviors. Support exists for simultaneously using multiple components from different providers, as well as the run-time exchange and update of the individual components. It is also planned to introduce an autonomic maintenance system in order to relieve users of the burden of downloading and installing updates to various components which are being developed by different institutions.

Through the SDK, applications can use the system in one of two ways; either execute each component separately, thereby exercising full control over the process of service discovery, QoS negotiation and the service invocation, or, use a convenience component to automatically find the desired service, perform the QoS negotiation and create a service proxy. Either way, at the end of this process the application has a service proxy through which it can communicate directly with the desired service. Although most applications will use the SDK to interact mainly with GEMSS services, support is provided for interacting with any standard Web Service. (Additional support for working with OGSA services will be incorporated later.)

Finally, the SDK also includes methods for application dependent session handling. Applications can designate those service proxies or other objects which are to constitute a "session" and then have them serialized for later re-instantiation. This provides a lightweight incarnation of a persistent environment useful for working with the types of long running services commonly found in GEMSS.

3.4. Business Models for a Medical Grid

With an eye to possible exploitation, GEMSS will maintain a flexible business model to allow commercial operation of Grid services. There is a concept of clients having an account with each service provider, allowing payment details to be provided and monthly bills generated for Grid use. Business models considered for GEMSS include airline reservation/business models and telephone business models. A negotiation model is also supported so a client can shop around, negotiating with all the service providers who provide the required service to get the best deal. Within each negotiation the quality of service terms associated with the required job can be discussed, as well as the price involved. The aim for GEMSS is to provide a viable and flexible approach to operating a commercial Grid.

3.5. Privacy and Security for Grid-based Medical Data

Since GEMSS is concerned with the processing of highly confidential and private information, such as images of patient heads and commercially sensitive fluid dynamic models, there are serious privacy issues to consider. Within GEMSS, EU law is being examined to identify privacy, contractual and ethical issues. To operate within EU law an appropriate level of security must be applied, with medical data anonymised where possible and not held for longer than is required to achieve the purpose of the Grid processing. Grid security must be periodically reviewed and patients made aware of the processing that will occur, and be able to review and correct the information held about them. It is expected that future Grid processing would be integrated into a hospital's existing data processing procedures.

The GEMSS Grid infrastructure will employ commercial off the shelf (COTS) technology, making full use of well-tested technology, regular security patches and best practice security procedures. Such security technologies will be evaluated with respect to the GEMSS Grid, in addition to the creation of a methodology for assessing the security needs at each GEMSS partner's site. The full GEMSS infrastructure is designed to provide a public key infrastructure, X.509 compliance, RSA encryption, service level authorization, logging and intrusion detection. The GEMSS Grid is designed to work with existing site firewalls, and does not require any insecure ports to be opened through them.

4. High Performance Medical Simulation Applications

Medical simulation applications considered within the GEMSS include maxillo-facial surgery planning, neuro-surgery support, medical image reconstruction, radiosurgery planning and fluid simulation of the airways and cardiovascular system. In the following we discuss these applications and outline their realization within the GEMSS environment as HPC services.

4.1. Maxillo-facial surgery simulation

In patients suffering from severe facial malformations like maxillary hypoplasia (see Fig. 5) and retrognathia, conventional therapeutic surgery often fails to guarantee long-term stability. The use of a rigid external distraction system for midfacial distraction osteogenesis (Fig. 4) is a new method to correct the underdevelopment of the midface, surpassing traditional orthognathic surgical approaches for these patients. The treatment consists of a midfacial osteotomy (bone cutting) followed by a halo-based distraction (pulling) step.

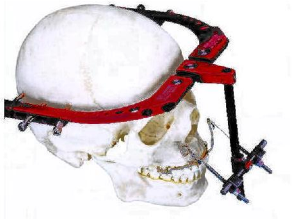

Figure 4. The halo device for distraction

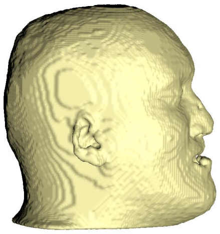

Figure 5. Patient before distraction

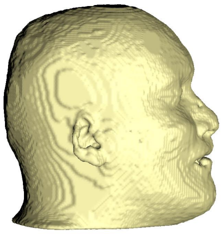

Figure 6. After distraction (geometric linear simulation)

Currently surgical planning is only based on CT (computed tomography) images, and the surgeon's experience is the only means of estimating the outcome of the treatment. The goal of this application is the modeling of this distraction process to allow predictions of its outcome and enable the medical user to try out several treatments in silico before selecting the most promising plan.

The tools required to perform such a simulation form a complex workflow graph. First, the CT image data may have to be filtered to reduce noise effects like streak artifacts due to teeth fillings, and must be segmented into different tissue classes. Next, the data set has to be visualized, and the surgeon has to specify cuts and distraction parameters. From these parameters a geometric model of the surgical operation is generated as a basis for the following simulation of the facial distraction process by an FEM (Finite Element Method) application. Finally, the results of this simulation (Fig.

6) can be visualized and used by an expert to evaluate the quality of the treatment plan. Earlier versions of the FEM package and the mesh generation tool are described in [9].

Most tasks of this toolchain are demanding in terms of memory and computation time. The FEM simulation task stands out as the most critical and challenging component in this simulation. A typical mesh (already clipped to the region of interest) involves about 0.5M elements. This number is likely to increase when the soft tissue is modeled in more detail. Thus, Grid computing is needed to obtain results with sufficient accuracy and reliability for clinical use on remote HPC facilities. Currently, a 64 node PC cluster located at NEC's C&C Research Laboratories in St. Augustin is used. The FEM code has been parallelized for distributed memory machines based on MPI, using DRAMA [10] as mesh partitioning library.

Proper modeling of the head turns out to be the most delicate problem, both in terms of correct reconstruction of patient geometry from the image data, and the appropriate modeling of material properties. Currently, soft tissues are modeled as a single type using either hyper-elastic or visco-elastic material laws. The next step involves to further distinguish skin, fat, and muscles. Different discretizations as (tri-)linear elements, quadratic elements, and elements with internal degrees of freedom using the enhanced assumed strain (EAS) approach have been compared. The choice of the discretization turns out to be very important. While linear elements are quite robust and need less time and memory than the other elements, the results are rather inaccurate. The quadratic elements yield very accurate results, but need more memory and solution time. The EAS elements seem to be the best choice. They need more time than the linear elements, but only the same amount of memory and lead to highly accurate results.

A comparison of algebraic multigrid (AMG) and Krylow solvers confirmed the well-known result that AMG solvers are optimal in a sense that the number of required floating point operations is proportional to the number of unknowns in the discretization.

4.2. Quasi real-time Neurosurgery support by non-linear image registration

Assistance for image-guided surgical planning by correction of the brain shift phenomenon is the focus of this application. The occurrence of surgically induced deformations invalidates positional information about functionally relevant areas acquired from functional MRI (fMRI) data. This problem is addressed by non-linear registration of pre-operative fMR images to intra-operative MRI, or to intra-operative 3D ultrasound data.

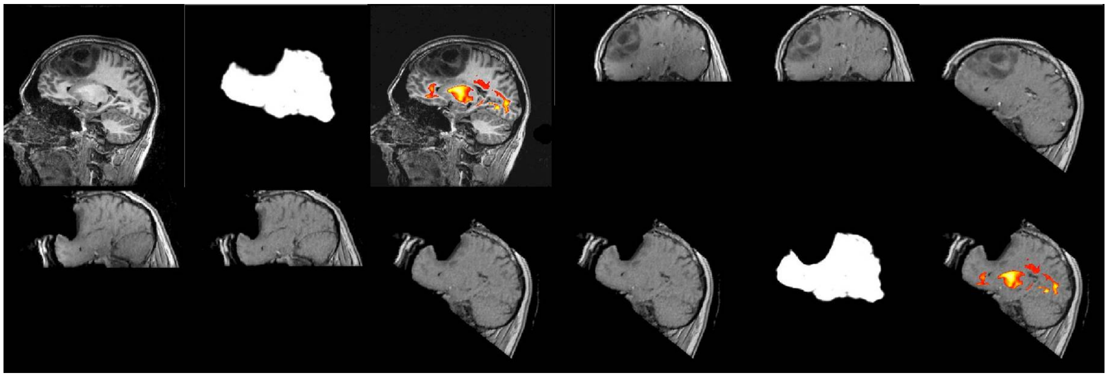

Figure 7. Single steps of the neurosurgery image processing chain

Whenever an intra-operative dataset is acquired, the following image processing chain must be executed (see Figure 7): (a) Transfer of anatomical images from the scanner and conversion into a machine-independent data format, (b) Correction of intensity non-uniformities in the scan data, (c) Linear registration of the (low-resolution) intra-operative scan with a (high-resolution) pre-operative scan, (d) Intensity adjustment between both scans, (e) Non-linear registration of both scans to yield a 3D deformation field, (f) Application of the deformation field to the pre-operative fMRI datasets, (g) Overlay of the deformed functional information onto the intra-operative data, (h) Conversion and transfer to the presentation device (e.g., monitor, surgical microscope).

The current processing time for the image processing chain is about 4 h (Intel Pentium III, single processor). However, minimal disruption of the progress of surgical intervention requires a maximum processing time of approx. 10 min. The time-consuming steps (b)-(e) can benefit from computation on the Grid and are readily parallelisable on shared or distributed memory high performance computing platforms.

However, the Grid application also involves the transmission of large image datasets (hundreds of Mbytes) between client and server and therefore internet bandwidth may be a problem in such time critical applications. The data volume is sufficiently large so that it is impractical to keep moving data sets between Grid server and client as each of the steps (a) - (h) is completed. Even though not all steps require the compute power of a multi-processor environment it is likely that all corresponding modules reside on the same server to minimise the data transfer overhead.

The data/application partitioning required by all GEMSS applications is clearly evident in this example and will be readily accommodated within GEMSS. A client GUI will use the GEMSS API calls to negotiate a Grid service, establish quality of service parameters and negotiate the cost of the job. Accessibility to sustained computing resources is very important in this application and timely response must be guaranteed during surgical intervention.

4.3. Near real-time Cranial Radiosurgery Simulation

This application (RAPT) is concerned with the provision of physics-based simulation for radiosurgery treatment planning. Accurate focusing of radiation to the treatment site requires a combination of stereotactic localisation and accurate modelling of the radiation dose distribution within the head. The best description of the radiation distribution can be obtained using complex, compute intense Monte-Carlo simulations. The need to treat patients as soon as possible after the stereotactic frame is fitted means that rapid computations of these distributions are needed. The use of an efficient parallel Monte-Carlo code (EGS) running on the Grid will enable high accuracy treatments to be planned and executed within existing time constraints. The accuracy of the treatment plans and the response time of the GRID system will be evaluated.

The Grid-enabled radiosurgery application uses RAPT as a front end to the EGS Monte Carlo engine to model ionising radiation transport through the head of the patient. It requires: (a) Definition of patient geometry (b) Specification and distribution of material types contained within the geometry (c) Position of beam isocentre (d) Beam properties - intensity profile, spectrum (e) Beam distribution - number of beams and their arrangement (f) Quality of simulation parameters - total number of photons, interaction types.

This data is currently specified by the contents of numerous text files that are converted and loaded into the EGS solver at startup. The simulation process simulates the dose given to the region of interest by modelling millions of photons, and following their paths, employing information from photon scattering data, to correctly give the photons their positions, angles of deflection, and energies or to absorb them in the tissue as they interact with the atoms of the target. The energy distribution within the geometry equates to the dose distribution of the model. Thus the output from the modelling process is a 3D patient mesh geometry with the accumulated dose and flux variance at each element of the solution mesh.

To provide this simulation code as a Grid service, the RAPT application code has been separated into client and server parts. The parallel RAPT kernel has been transformed into a Web Service, and the client has been realized with Java and visualization tools written in MatLab.

A typical usage scenario comprises the following steps: (a) a text editor is used to create input files, (b) the Grid client is run to negotiate for reservation of the Grid computing resource, (c) an appropriate reservation is confirmed and the input file(s) are uploaded by the Grid client, (d) RAPT job runs as scheduled on the compute resource, (e) user receives notification of completed job, (f) user downloads the output file(s), (g) MatLab application visualization software is used to view the results.

As an example test case (see Figure 8), a homogeneous spherical water phantom of radius 8 cm

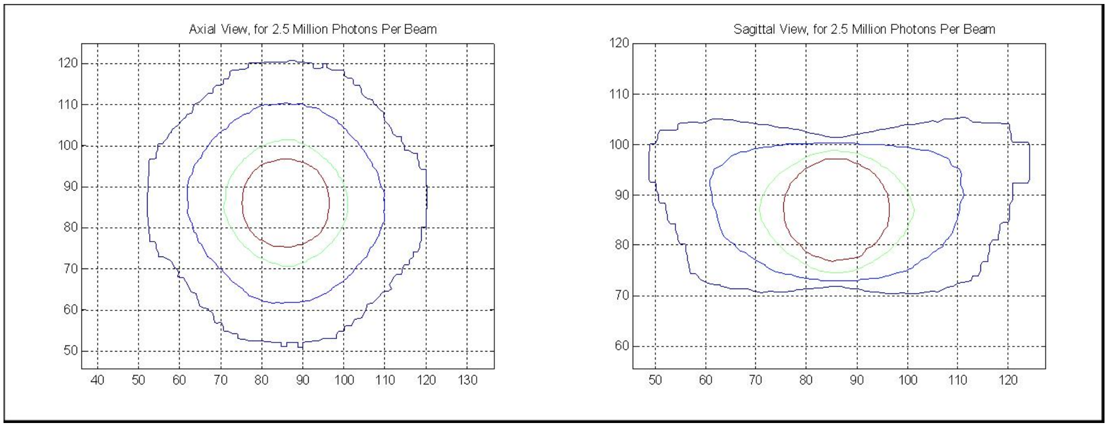

Figure 8. Monte Carlo computed dose distribution.

was modelled. The dose distribution resulting from irradiation by 201 beams (500 million photons) was calculated both in a local setup and a Grid-setup using 3 PCs. In the local set-up the overall run time amounted to approximately 22 hours on a 2.5GHz single processor PC with 1GB of RAM. In a preliminary prototype Grid set-up a run time of 18 hours was obtained using a 4-way Xeon PentiumIII, 500 MHz, 2GB RAM. The job input size amounted to 117 kBytes and the job output size to 4 MBytes.

4.4. Inhaled Drug Delivery Simulation

This component of GEMSS is a comprehensive simulation tool for the study of new treatments and drug delivery to the lungs. It has been originally developed in the COPHIT project and involves a respiratory simulation that integrates medical images, mesh generation, Computational Fluid Dynamics (CFD), compartment modelling of the lungs, and the simulation of inhalation devices. The fluid dynamics elements are very computationally demanding and enable the simulation to derive significant benefit from the Grid. The simulation results are used to inform the user of the requirements for targeted delivery of medication to the lung and systemic circulation.

There are several stages in the simulation process, including pre-processing steps that create the model geometry and define its properties. Pre-processing includes description of airways geometry from CT scans, mesh generation and specification of parameters that describe various aspects of the model (e.g. drug formulation, inhalation profile etc). Although these steps require a high level of user interaction, they are not very computationally demanding and are therefore allocated to the local client rather than the Grid server. Determination of the pattern of air flow and drug deposition throughout the model is achieved using the CFX computational fluid dynamics solver. This is the most computationally demanding part of the simulation process and therefore relies on the Grid. The modelling of physiologically relevant dynamic input flows requires a transient analysis which is particularly time consuming because it requires repetitive solving over many timesteps (possibly several hundred). However, the solution step does not need any user interaction and can be run in batch mode once the input files have been encrypted and dispatched to the Grid server. The CFX results data is encrypted and returned to the client where it can be viewed using the CFX post-processor (see Figure 9).

The Grid is essential for the CFD computation stage of a COPHIT simulation, and calculation may take several days. The input files sent to the Grid server are typically many MB in size, but benefit from compression. The size of the results files depends upon the solution parameters the user has requested. Since results are produced at each timestep, the amount of data can be potentially very large typically several hundreds of megabytes. In the face of excessive download time resulting from large files and limited internet bandwidth, the user could either request to receive a selected subset of the available data or some form of results processing could be performed on the Grid server to

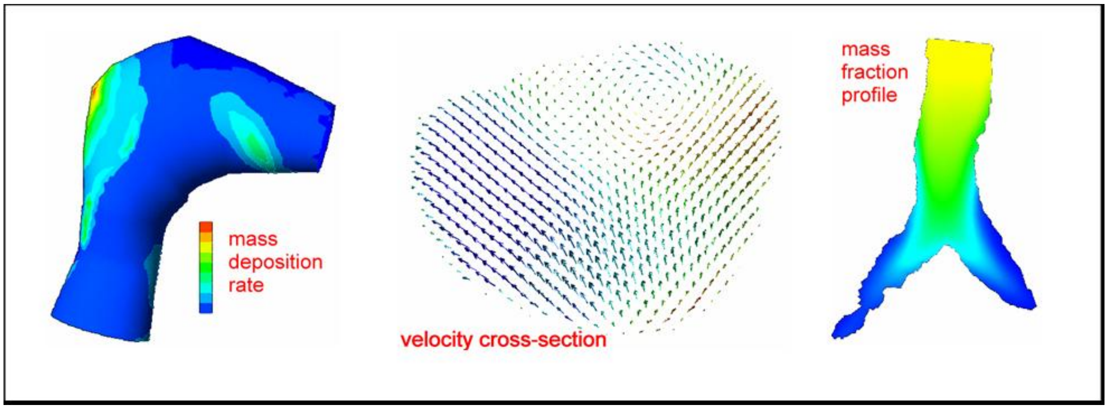

Figure 9. Results data of a COPHIT simulation viewed on the client using CFX-Post.

reduce the quantity of data. In a typical test case the input file amounted to 3.5MB (compressed). Solution was performed at 100 timesteps, requiring 14.5 hours on a Dell P4 Xeon with a 2.4 GHz processor. For each timestep of the results required for analysis, 19MB of data (compressed) must be downloaded.

4.5. Compartmental modelling approach for the Cardiovascular System

Simulation of the cardiovascular system is a valuable tool in the development of prostheses. Its role in surgical planning offers the opportunity to answer 'what if' questions, but there is a need for improved description of boundary conditions - a scientific issue that is being addressed within GEMSS.

As for the COPHIT application described previously, a tetrahedral model mesh is generated from a segmented CT or MRI scan dataset. Inlet and outlet boundary conditions and material properties are also specified. In this case the model is of a segment of vessel rather than airways, but the principle is the same. This means that the pre-processing tools developed for COPHIT can be adapted to the cardiovascular application. These steps require a high level of user interaction, but they are not very computationally demanding, and they are therefore allocated to the local client rather than the Grid server. They result in the production of a 'DEF' file, which is the input to the CFD solver (CFX). Compartment models, implemented using FORTRAN code, are coupled to the outlets of the CFD model, and these represent the peripheral circulation. The properties and initial state of the compartments are specified in various text files. Together with the 'DEF' file, these specify the complete coupled 3D model-compartment fluid dynamics problem. CFX is then used to determine the pattern of blood flow - a computationally demanding step which benefits from Grid resources. A typical transient analysis over the course of a cardiac cycle requires of the order of 100 or more timesteps, and may take several days to solve. The CFD and compartment software is able run in parallel and will thus benefit from multiprocessor Grid resources.

4.6. Fully 3D Reconstruction in SPECT

Single Photon Emission Computed Tomography (SPECT) is a popular diagnostic facility for tumour staging and examination of metabolisms. It provides valuable functional information complementary to MRI and x-ray CT showing anatomical details with high resolution. Accurate design of the system matrix based on empirical calibration data, allows inherent correction of spatially invariant line of response functions, and first order scatter correction.

An improved variant of the well-known ML-EM algorithm for emission tomography [20] has been parallelized in such a way that it is portable across distributed-memory architectures, shared-memory architectures, and SMP clusters. On a single SMP node equipped with 4 processors, we observed good speed-ups better than a factor of 3. On SMP clusters, a hybrid parallelization strategy based on a combination of MPI and OpenMP outperforms a pure process based parallelization strategy, relying

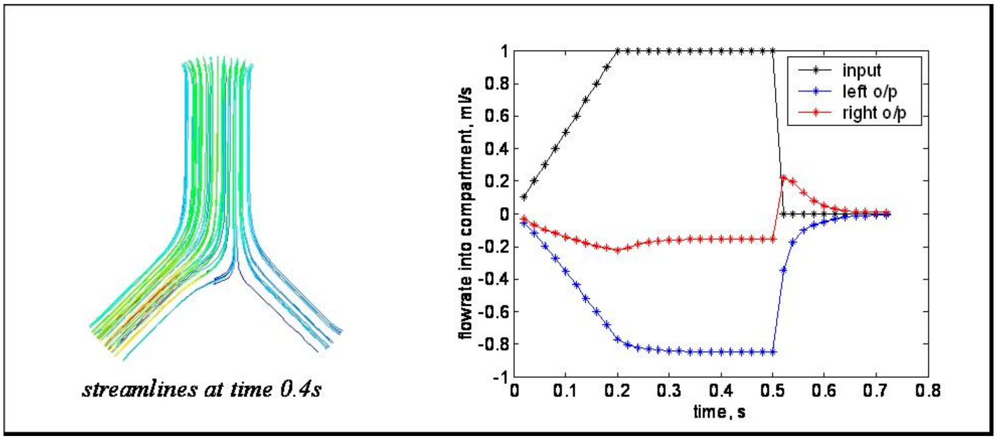

Figure 10. Test case for cardiovascular simulation: flow through a simple bifurcation model bounded by dynamic boundary condition constraints.

on MPI only. Due to the high computation-to-communication ratio the code exhibits a good scaling behaviour also on machines where an MPI-only parallelization strategy is used.

The ML-EM reconstruction has been transformed into a Web Service which can be deployed within Tomcat/Axis. The image reconstruction application relies on a Java-based client GUI which handles the acquisition of SPECT images from a scanner, the configuration and remote execution of reconstruction tasks, and the display of reconstructed images. All data transfers between clients and servers are currently based on SOAP. For typical SPECT resolutions, the size of the image files to be transferred between client and service is a few MBs and thus does not represent a problem, even when transferred within SOAP messages.

5. Related Work

There are a number of Grid projects that deal with bio-medical applications. The EU BioGrid [1] aims at the development of a knowledge grid infrastructure for the biotechnology industry. The main objective of the OpenMolGRID project [18] is to develop a Grid-based environment for solving molecular design/engineering tasks relevant to chemistry, pharmacy and bioinformatics. The EU MammoGrid [15] project is building a Grid-based federated database for breast cancer screening. The UK e-Science myGrid [16] project is developing a Grid environment for data intensive in silico experiments in biology. Most of these projects focus on data management aspects of the Grid similar to the EU DataGrid project.

In contrast, the GEMSS project focuses on the computational aspect of the Grid, with the aim of providing HPC hardware resources together with medical simulation services across wide area networks in order to overcome time, space or accessibility and ease of use limitations of conventional systems.

Other projects in the bio-medical field which also focus more on the computational aspect of the Grid include the Swiss BiOpera project [2], the Japanese BioGrid project [17], and the Singapore BioMed Grid [3].

6. Conclusion

Within the GEMSS Project medical Grid service prototypes are being developed and evaluated in a secure service-oriented infrastructure for distributed on demand/supercomputing. Grid technology has the potential for enormously enhancing the computing infrastructure and providing enhanced computational services to new user communities. The simulation services included within GEMSS is about to demonstrate potential impact on specific medical areas, aiming to improve: non-invasive diagnosis and pre-operative planning, operative procedures, therapeutic protocols, design, analysis and testing of biomedical devices, such as heart valves, stents, and inhalers.

Modelling of individuals is an ongoing research topic and targets the complete simulation of the human body — there is a great deal still to be understood and developed. Grid computing hopefully can bring those developments to the medical user community.

REFERENCES

[1] The BioGrid Project. http://www.bio-grid.net/index.jsp

[2] BiOpera - Process Support for BioInformatics. ETH Zürich, Department of Computer Science. http://www.inf.ethz.ch/personal/bausch/bioopera/main.html

[3] BiomedGrid Consortium. http://binfo.ym.edu.tw/grid/index.html

[4] The Bloodsim Project. `http://www.software.aeat.com/cfx/European_Projects/bloodsim/bloodsim.htm`, 1998. EU Esprit Project 28350, 1998-2001.

[5] The Cophit Project. `http://www.software.aeat.com/cfx/European_Projects/cophit/index.html`, 2000. EU IST Project IST-1999-14004.

[6] A. Gill, M. Surridge, G. Scielzo, R. Felici, M. Modesti, and G. Sardu, RAPT: A Parallel Radiotherapy Treatment Planning Code, in H. Liddel, A. Colbrook, B. Hertzberger, and P. Sloot, Eds. Proceedings High Performance Computing and Networking 1067, pages 183-193, 1996.

[7] The GEMSS project: Grid-enabled medical simulation services. `http://www.gemss.de`, 2002. EU IST project IST-2001-37153, 2002–2005.

[8] The Simbio Project. `http://www.simbio.de`, 2000. EU IST project IST-1999-10378, 2000–2003.

[9] A. Basermann, G. Berti, J. Fingberg, and U. Hartmann. Head-mechanical simulations with SimBio. *NEC Research and Development*, 43(4), October 2002.

[10] The DRAMA project (Dynamic Load Balancing for Parallel Mesh-based Applications) `http://www.ccrl-nece.de/DRAMA`, 1999. EU ESPRIT project 24953, 1997-1999

[11] The Datagrid Project. `http://eu-datagrid.web.cern.ch/eu-datagrid/`. EU Project IST-2000-28185, 2001-2003.

[12] I. Foster, C. Kesselman, S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations, International J. Supercomputer Applications, 15(3), 2001.

[13] I. Foster, C. Kesselman, J. Nick, S. Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002.

[14] The GRIA Project. `http://www.gria.org/`. EU IST Project IST-2001-33240.

[15] MAMMOGRID - European federated MAMMOgram database implemented on a GRID structure, contact person: AMENDOLIA, Salvator Roberto, Email: salvator.amendolia@cern.ch.

[16] The myGrid Project. http://mygrid.man.ac.uk/

[17] The Japanese BioGrid Project. http://www.biogrid.jp/

[18] OpenMolGRID- Open Computing GRID for Molecular Science and Engineering. Open Computing GRID for Molecular Science and Engineering (OpenMolGRID)

[19] M. Tittgemeyer, G. Wollny, and F. Kruggel. Visualising deformation fields computed by non-linear image registration. *Computing and Visualization in Science*, 5(1):45–51, 2002.

[20] L. A. Shepp and Y. Vardi, Maximum likelihood reconstruction for emission tomography. IEEE Trans. Med. Imaging, vol. 1, no. 2, pp. 113-122, 1982.

[21] The UNICORE Forum. http://www.unicore.org/