In Proceedings of 2nd International Workshop on Grid and Cooperative Computing (GCC 2004), Shanghai, China, Lecture Notes in Computer Science 3032, Springer Verlag, December 2003, pp. 34-41.

Implementation of Grid-enabled Medical Simulation Applications Using Workflow Techniques

Junwei Cao, Jochen Fingberg, Guntram Berti, and Jens Georg Schmidt

C&C Research Laboratories, NEC Europe Ltd., Germany
cao@ccrl-nece.de

Abstract. GEMSS is a European project that aims at providing high performance medical simulation services in a distributed and grid computing environment. The GEMSS grid middleware is designed using web services technologies and standards and provides support for authorization, workflow, security, Quality of Service aspects. In this work, one of the GEMSS applications, maxillo-facial surgery simulation, is described, which includes a complete chain of tools necessary for the entire process from geometric model generation from scan data to computer simulation and visualization. The Triana workflow environment is utilized to implement the application for uniform access of local processes (e.g. image pre-processing and meshing), interactive tools (e.g. mesh manipulation) and grid-enabled remote services (e.g. HPC finite element simulation). It is concluded that workflow provides benefits to flexibility, reusability and scalability and is potential to become a mainstream grid application enabling technology.

1 Introduction

The grid builds on the accessibility of the Internet to allow effective use of geographically distributed resources [13, 1]. Grid computing technologies will provide the basis for the next generation of Internet-enabled HPC solutions. GEMSS (Grid Enabled Medical Simulation Services) [14] is a European project that aims to create a grid testbed for medical computing.

In GEMSS a grid middleware will be developed that can be used 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. The grid architecture is designed based on Web Services technologies and standards and supposed to meet various requirements from business, legal and social, security, performance and application aspects. Workflow specification, enactment and execution are essential supports of the GEMSS grid middleware for business process management, QoS negotiation and application enabling.

Maxillo-facial surgery simulation [12] is one of the medical service applications included in the GEMSS test-bed, which provides a virtual try-out space for the pre-operative planning of maxillo-facial surgery. The implementation of the maxillo-facial surgery simulation requires a chain of tools necessary for the entire process from geometric model generation from scan data to computer simulation and visualization.

The work presented in this paper takes maxillo-facial surgery simulation as an example application and provides an initial demonstration of using workflow techniques to enable the application for uniform access of local processes (e.g. image segmentation and mesh generation), interactive tools (e.g. mesh manipulation) and remote web services for large-scale simulation (e.g. HPC finite element simulation). An existing open source problem solving environment, Triana [24], is utilized as a workflow manager for the application. Triana is written in Java and provides abundant graphical user interfaces for both toolbox implementation and workflow construction. In this work, several Triana toolboxes and an example workflow model are developed and detailed information is provided on how to develop Java wrappers for various binaries and scripts, turn them into Triana toolboxes using the Triana unit, and set up web services for remote access of HPC resources.

2 GEMSS

The GEMSS middleware is built on existing web services technologies and standards and provides support for authorization, workflow, security, Quality of Service aspects. This section provides brief information on the GEMSS architecture and one of the six GEMSS applications, maxillo-facial surgery simulation.

2.1 GEMSS Architecture

The GEMSS architecture, shown in Fig. 1, uses a client/server topology employing a service-oriented architecture. Detailed information on individual modules is not provided below and can be found in [14].

The GEMSS client architecture is mainly a pluggable component framework, which aims to provide flexible support to various application scenarios by placing minimal demands upon the components themselves and providing sufficient means for them to interact. The component framework is used by applications as the entry point into the GEMSS grid middleware and hides the details of the grid as far as possible.

A GEMSS service can be implemented using existing web services technologies. Application services provide generic interfaces for starting application, uploading and downloading files. Resource manager and data storage provide interfaces for services to access actual HPC resources.

2.2 Maxillo-facial Surgery Simulation

One of the GEMSS target applications deals with computing pre-operative simulations for maxillo-facial surgeries. In clinical practice treatment of patients with in-born deformations of the mid-face is performed by cutting ill-formed bones and pulling them into the “right” position.

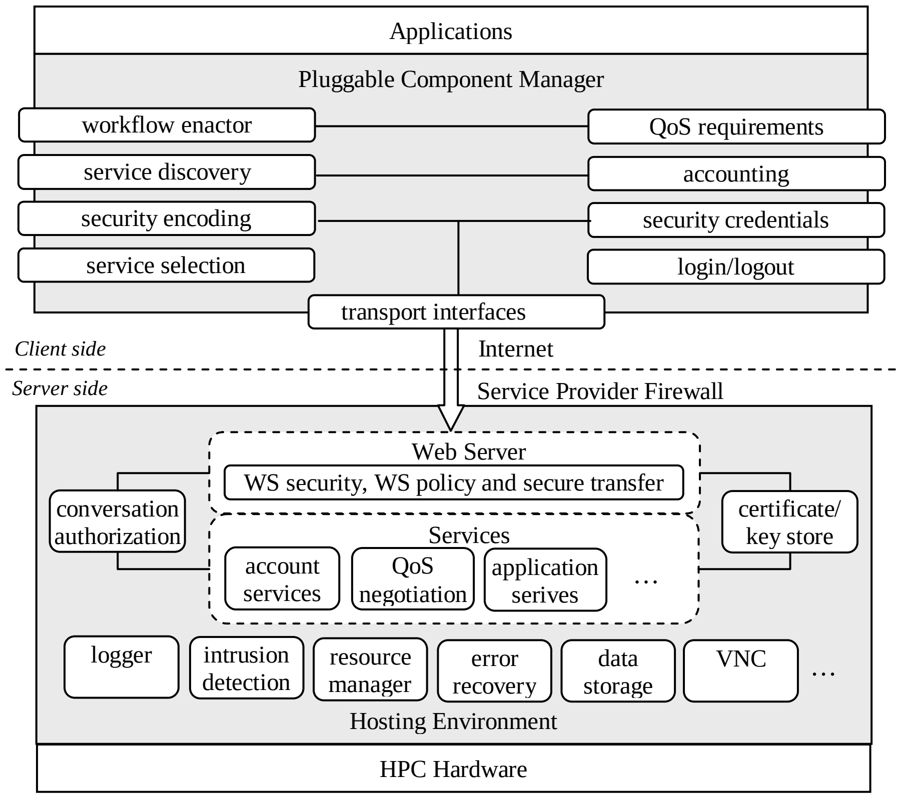

Fig. 1. The GEMSS architecture is composed with the client side architecture and service provider architecture, which plays as middleware between grid applications and resources.

A halo frame is mounted to the patient's skull (see Fig. 2a). Distractors, mounted to the halo fixed at certain points in the midface, exert a force on a midface region, gently pulling it to a pre-defined position in order to achieve a medically and cosmetically pleasing result. The simulation of the process includes a complete chain of tools [12], which are partly described below.

- *Image segmentation.* As shown in Figs. 2b, 2c and 2d, bone and soft tissue parts can be identified from the original CT image. This can be performed at the client as an image pre-processing step.
- *Mesh generation.* The process allows the generation of meshes suitable for Finite Element simulation (see an example in Fig. 2e) and could be computation intensive if the mesh size is very large. In our initial implementation, this step is also considered as a client side process.
- *Mesh manipulation.* This process provides the meshing model with initial and boundary conditions for simulation using a user friendly tool. An example of virtual bone cutting is shown in Fig. 2f. This is an interactive process and can be only carried out at the client side.

- *Finite Element analysis.* The Finite Element simulation capability is provided by a fully parallel program linked with a load balancing library. Example visualizations of simulation results are illustrated in Figs. 2g and 2h. In this case the simulation result shows that a better bone position can be achieved after the surgery. This is the part that required grid enabling to access remote HPC resources.

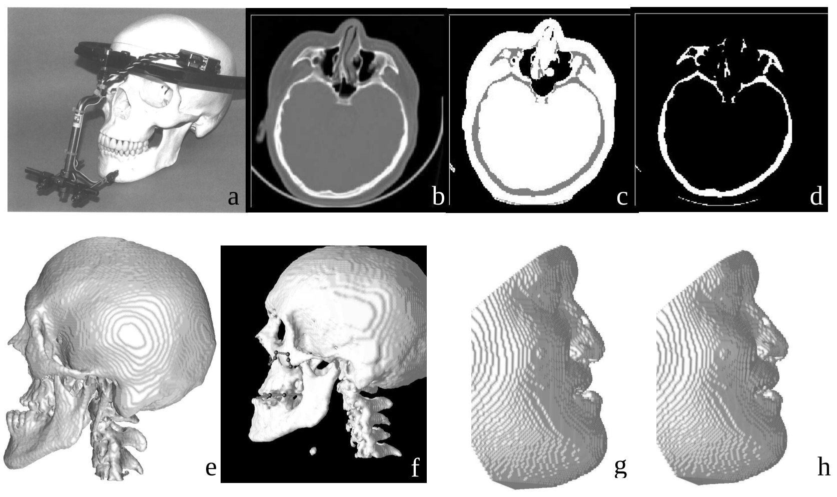

Fig. 2. Maxillo-facial surgery simulation. (a) The halo device mounted on the head; (b) CT images; (c) Identification of substructures (bone, soft tissue and background); (c) Selection of bones from others; (e) Mesh of a skull; (f) Virtual bone cutting; (g) Visualization of the geometric face change I (before the surgery); (h) Visualization of the geometric face change II (after the surgery).

3 Application Workflow

Workflow techniques are utilized in the GEMSS project for multiple purposes. In this work, only application workflow is implemented using an existing workflow tool.

3.1 Triana Environment

The Triana software environment is intended to be flexible and can be used in many different scenarios and at many different levels. One of the Triana applications is described in [23], where Triana is equipped with P2P computing mechanisms for galaxy visualization. In another European grid project GridLab [17], Triana is integrated with grid environments via grid application toolkit (GAT) interfaces

In this work, Triana is used as a management environment for the simulation application workflow. It provides an effective wizard to build high level toolboxes in a semi-visual way so that workflow activities are uniformly encapsulated and handled by the Triana workflow. The Triana unit is the base class on which that all of toolboxes are built. The unit class provides abundant methods for each toolbox to handle inputs, outputs, parameters and even corresponding graphical input interfaces in a standard way. Fig. 3 provides a screenshot of the Triana implementation of maxillo-facial surgery simulation. Toolbox implementation and workflow construction are described below in separate sections.

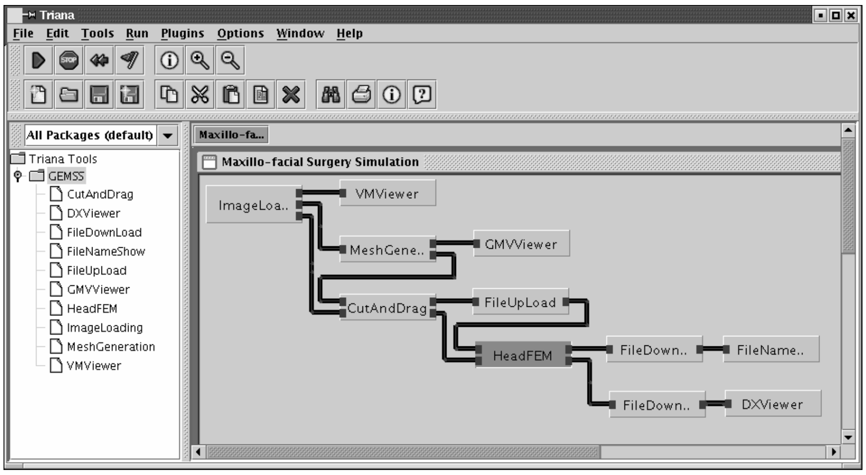

Fig. 3. Triana implementation of maxillo-facial surgery simulation, with a complete list of the GEMSS toolboxes in the left column and the example workflow in the main window.

3.2 Toolbox Implementation

There are three types of toolboxes used for building the maxillo-facial surgery simulation workflow: input, processing and output. The only input toolbox shown in Fig. 3 is the image loading toolbox. Main processing toolboxes include those for mesh generation, bone cutting and dragging, and Finite Element analysis. Output toolboxes are implemented for result visualization.

- *ImageLoading*. This is the entry point of the simulation workflow. The user is required to input the image file name to start the workflow execution.
- *MeshGeneration*. Since image processing and meshing described in Section 2.2 are both client-side local processes, a shell script is developed to describe the whole chain of the process. The toolbox is implemented via a Java wrapper to the script execution. The mesh generation toolbox takes input from the image loading and outputs a result mesh file for mesh manipulation.
- *Cut&Drag*. The activity allows the user to define initial and boundary conditions for simulation via virtual bone cutting and dragging operations. The toolbox implements an iterative process so that the user can repeat the activity until a

satisfying result is obtained. It takes inputs from both the mesh file and the original image file and outputs an all-in-one model file ready for simulation.
- *HeadFEM.* The Finite Element simulation takes two inputs from the *Cut&Drag* toolbox and outputs two sets of files, one including detailed simulation data and the other including a ready input for visualization. In order to implement the toolbox, the HeadFEM service has to be set up for remote access of HPC resources. This is implemented using the Apache Tomcat and Axis. In the client side toolbox implementation, the JAX-RPC call provided by the Apache Axis library is used to access the HeadFEM service.
- *FileUpload and FileDownload.* These two toolboxes are necessary before and after the simulation activity and temporarily implemented using FTP. In the later GEMSS implementation, these will be supported by application services as well.
- *VMViewer, GMVViewer and DXViewer.* These output toolboxes are used in the workflow for result visualization. As shown in Fig. 3, the VM tool is used to display the original image file, the GMV tool for mesh visualization and the DX tool for the visualization of simulation results. The use of these tools during workflow execution is also illustrated in Fig. 4.
- *FileNameShow.* This is an output toolbox that can be used to pop up a window and display the input file name.

3.3 Workflow Construction

Workflow construction is straightforward in the Triana environment given all toolboxes ready. The developer can drag and link toolboxes to directly construct a workflow ready for execution. The workflow developed for maxillo-facial surgery simulation is shown in Fig. 3. The screenshot of an example workflow execution is illustrated in Fig. 4. In this case the input is a CT image file shown in Fig. 4b. Output toolboxes in the workflow invoke different tools to visualize results (see Figs. 4b-4e).

4 Related Work

Apart from the GEMSS project, there are some other projects that are using grid computing for medical or biological application. These include European MammoGrid project for breast cancer screening [21], UK e-Science Programme MyGrid [22] and Swiss BioOpera [5] projects for bioinformatics, Japan BioGrid project [3], Singapore BioMed grid [4], and so on.

Workflow techniques are also used in many other grid projects. Pioneering WebFlow project [2] uses workflow as high level mechanism for high performance distributed applications. Early grid projects Condor [8] and UNICORE [25] provide workflow capability to manage task dependencies. The Globus project [15] is also developing a grid service flow language (GSFL). Current other projects that involve workflow development include USA ASCI grid [6], UK MyGrid [22], European projects GRASP [16] and CrossGrid [9].

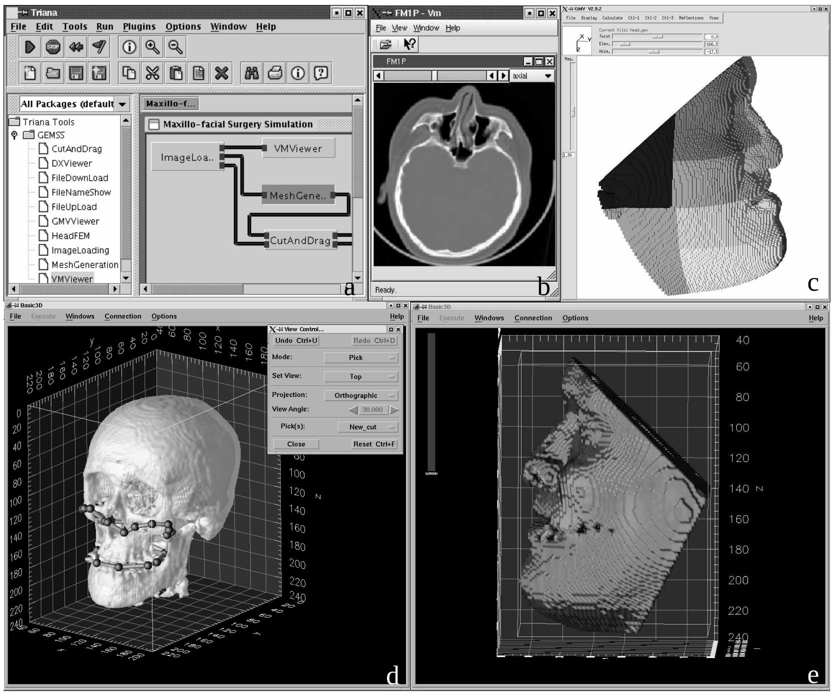

Fig. 4. An example workflow execution given a CT image file. (a) The Triana software environment; (b) The VM tool for browsing the input image file; (c) The GMV tool for browsing the mesh file; (d) The DX tool for interactive virtual bone cutting and dragging; (e) The DX tool for simulation result visualization. (Note: The data set and guidance for placing the bone cuts was kindly provided by Dr. T. Hierl, University Clinic Leipzig.)

There are also many grid-aware workflow environments and tools under development. IBM BPWS4J is a web services flow execution engine designed for BPEL4WS [10]. Symphony [20] focuses more on security issues. GridFlow [7] focuses more on workflow simulation and scheduling. In several most recent work, workflow techniques are used to address various interesting issues in grid environments, including failure handling [18], authorization [19], automatic workflow generation and grid mapping [11].

5 Conclusions

There are many discussions in grid computing community on programming models for grid application development. According to our experiences on using the Triana workflow for building GEMSS applications described in this work, advantages of using workflow techniques are summarized as follows:

- *Flexibility*. The application can be constructed at different levels easily and quickly using workflow mechanisms.

- *Reusability*. A toolbox once developed can be reused in multiple applications.
- *Scalability*. Workflow-based applications can be constructed in a hierarchical way, which provides possibilities for applications to scale up in a grid environment.

It is expected that workflow will become one of mainstream programming models for future grid application development.

References

1. F. Berman, A. J. G. Hey, and G. Fox, Grid Computing: Making The Global Infrastructure a Reality, John Wiley & Sons, 2003.
2. D. Bhatia, V. Burzevski, M. Camuseva, G. Fox, etc., "WebFlow – a Visual Programming Paradigm for Web/Java Based Coarse Grain Distributed Computing", Concurrency: Practice and Experience, Vol. 9, No. 6, pp. 555-577, 1997.
3. BioGrid. http://www.biogrid.jp/.
4. BioMed. http://bmg.bii.a-star.edu.sg/.
5. BioOpera. http://www.inf.ethz.ch/personal/bausch/bioopera/main.html.
6. H. P. Bivens, "Grid Workflow", GGF Working Document, 2001.
7. J. Cao, S. A. Jarvis, S. Saini, and G. R. Nudd, "GridFlow: Workflow Management for Grid Computing", in Proc. of 3rd IEEE Int. Symp. on Cluster Computing and the Grid, Tokyo, Japan, pp. 198-205, 2003.
8. Condor. http://www.cs.wisc.edu/condor/.
9. CrossGrid. http://www.crossgrid.org/.
10. F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, S. Weerawarana, "Business Process Execution Language for Web Services, Version 1.0", July 2002. http://www.ibm.com/developerworks/library/ws-bpel/.
11. E. Deelman, J. Blythe. Y. Gil, C. Kesselman, et. al., "Mapping Abstract Complex Workflows onto Grid Environments", J. Grid Computing, Vol. 1, No, 1, pp. 25-39, 2003.
12. J. Fingberg, G. Berti, U. Hartmann, and A. Basermann, "Head-Mechanical Simulations with SimBio", NEC Research & Development, Vol. 43, No. 4, 2002.
13. I. Foster, and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure, Morgan-Kaufmann, 1998.
14. GEMSS. http://www.ccrl-nece.de/gemss/.
15. Globus. http://www.globus.org/.
16. GRASP. http://eu-grasp.net/.
17. GridLab. http://www.gridlab.org/.
18. S. Hwang, and C. Kesselman, "Grid Workflow: a Flexible Failure Handling Framework for the Grid", in Proc. of 12th IEEE Int. Symp. on High Performance Distributed Computing, Seattle, USA, pp. 126-137, 2003.
19. S. Kim, J. Kim, S. Hong, and S. Kim, "Workflow-based Authorization Service in Grid", in Proc. of 4th IEEE Int. Workshop on Grid Computing, Phoenix, USA, 2003.
20. M. Lorch, and D. Kafura, "Symphony – A Java-based Composition and Manipulation Framework for Computational Grids", in Proc. of 2nd IEEE/ACM Int. Symp. on Cluster Computing and the Grid, Berlin, Germany, pp. 136-143, 2002.
21. MammoGrid. http://www.healthgrid.org/.
22. MyGrid. http://www.mygrid.info/.
23. I. Taylor, M. Shields, I. Wang, and R. Philp, "Distributed P2P Computing within Triana: A Galaxy Visualization Test Case", in Proc. of 17th IEEE Int. Parallel & Distributed Processing Symp., Nice, France, 2003.
24. Triana. http://www.triana.co.uk/; http://trianacode.org/.
25. UNICORE. http://www.unicore.de/.