e-Spine: FEM Simulation System for Spine and Implant Based on RESTful Architecture in HPC Environment

Jaehyoung Lim, Junglok Yu, Dukyun Nam, Kumwon Cho

Korea Institute of Science and Technology Information
245 Daehak-ro, Yuseong-gu, Daejeon, 305-806, Korea
{david.j.lim, junglok.yu, dynam, ckw}@kisti.re.kr

Abstract. In medical area, the importance of computer simulation is growing faster. Spinal medical area is also one of practical application domains to prevent side effects and to reduce cost and time in many aspects. However, finite element method (FEM) software programs widely used have four major problems – cost-inefficiency, complex user interface, low flexibility, and lack of data management. The programs are expensive and the licenses limit users to use only allowed computers. Also they are too complex and heavy for novice and casual users such as medical doctors to learn and use. The preprocessing and postprocessing programs are fully hard-coded and tightly coupled with corresponding simulation tool such as the coupling of Abaqus and Abaqus CAE. Users must manage lots of simulation data for themselves. To solve the problems, we propose a Representational State Transfer (REST) architecture for FEM simulation. The proposed approach brings extensibility and flexibility to the system. Detaching clients including pre- and post-processors from the simulation tool and introducing the middleware interfacing clients through REST API make both clients and the middleware extensible and flexible. Client programs and the middleware can implement additional functionalities easily while they keep compatibility to the REST API. Proposed approach allows users to conduct simulations through the client remotely. By the proposed approach, users can retrieve and reuse simulation data.

Keywords. FEM, simulation, RESTful, spine, HPC

1 Introduction

Computer simulation is getting more important recently. Simulation using computer takes the important role to reduce time and cost in experiment and verification, preventing expectable side effects. Likewise, the importance of simulation is huge in spinal medical area. Especially simulation on spinal implant before conducting surgery helps predicting the result and any side effects, previewing the joint and fixture of patient's spine bone and the implant and checking effects of the movement and the loads. Computer simulation is also major concern for medical engineers and researchers to develop the spinal implants that target specific patients in specific condition, or to invent implants that show better cost-effect or better performance.

Despite the importance of computer simulation in spine medical area, software programs widely used for finite element method (FEM) simulation have four major problems – cost-inefficiency, complex user interface, low flexibility, and lack of data management. Those software programs are expensive. Their licenses limit users to use only allowed computers. Also the programs are too complex and heavy for novice and casual users such as medical doctors to learn and use for their purpose. The pre-processing and post-processing programs are fully hard-coded and tightly coupled with corresponding simulation tool such as the coupling of Abaqus and Abaqus CAE. Users must manage lots of simulation data for themselves. In addition, high performance computing (HPC) resources are very expensive and require extremely high level of security. For the security purpose, we must provide clients with the custom-made security function based on certificate to meet the security level that the institute requires.

To solve the problems, the client program (pre- and post-processing program) and the simulation tool software must be separated, introducing an abstract interfacing layer in-between to provide both sides with independence while keeping the interface – it is a loosely-coupled system architecture rather than the tightly-coupled.

There are requirements to design the loosely-coupled system architecture for our spine-implant FEM simulation system. First, to get the maximum level of extensibility and flexibility, the solution is required to support various clients on multi-platform such as Windows, Linux, and so on. Second, thin-client approach can allow users to spend less cost and resource to manage simulation details such as synchronization over different client software programs in multiple work places.

Therefore, we propose a RESTful architecture using Web services. Representational State Transfer (REST) is a style of information system architecture where clients and the servers communicate through the messages that represent states of certain resources [5]. An architecture is called as being 'RESTful' if it conforms the REST constrains – client-server, stateless, cacheable, layered system, uniform interface, and code on demand [6]. Currently most RESTful systems use Web services. The advantages of Web service provide functionalities to build loosely-coupled system architecture and meet the requirements listed above.

The proposed approach brings extensibility and flexibility to the system. Detaching pre-/post-processor from the solver and introducing the middleware interfacing clients through RESTful API make both clients and the middleware extensible and flexible. Client programs and the middleware can implement additional functionalities easily only when the implementations are compatible to the RESTful API. Proposed approach allows users to conduct simulations through the client remotely. To retrieve and reuse simulation data, the system can store and manage various simulation data such as metadata, parameters, and files.

Chapter 2 presents overview of some related systems and researches in computer simulation for science. In chapter 3, We discuss how the proposed simulation system, e-Spine, can support for medical and engineering purpose of spine and implant and how it can solve the problems raised in chapter 1, applying RESTful architecture. The implementation of e-Spine system is described briefly in chapter 4. Lastly chapter 5 concludes the paper.

2 Related Work

Abaqus [1] is a software application for finite element analysis (FEA) and computer-aided engineering (CAE). In detail, as supporting a post-processing step, it is used for the modeling and analysis of mechanical components and assemblies. After running the simulation, it can visualize the FEA result as a post-processing step. Abaqus is popular with not only academic and research institutions, but also the automotive, aerospace, and industrial products industries. However, the license fee of this product can be burden to users and the functionalities of this product may be over-qualified because it provides the wide material modeling capability.

HUBzero [2] is the platform to support nanoHUB.org, an online community for the Network for Computational Nanotechnology (NCN). It allows users to develop simulation and modeling tools online. The HUBzero platform provides a typical X11 Window System environment through the embedded VNC viewer in the web browser. The Rappture library makes it possible to generate the graphical user interface of the resulting tools. The web-based simulation environment atop HUBzero has a limitation to provide rich pre-processing and post-processing graphical interfaces due to the web-based visualization technology.

We have recently launched 'EDucation-research Integration through Simulation On the Net' (EDISON) project [3] under the financial support of the Ministry of Education, Science and Technology of Republic of Korea, where the main goal of the project is to construct the web-based cyber infrastructure to support the education and research via computational simulations. Like HUBzero case, it is hard to provide rich graphical user interface in the EDISON environment which intends to be developed with the web technology.

3 e-Spine System Architecture

We propose an FEM simulation system for spine and implant based on RESTful architecture, e-Spine. Proposed system – e-Spine has three layers – client, middleware, and backend layers. REST interface lies between client and middleware layers. Clients can have various implementations only when they keep the communication methods specified by REST interface. Middleware also can modify function modules without any change on clients. The independence between the client and middleware layers brings extensibility and flexibility to the system. Figure 1 shows the architecture of e-Spine system.

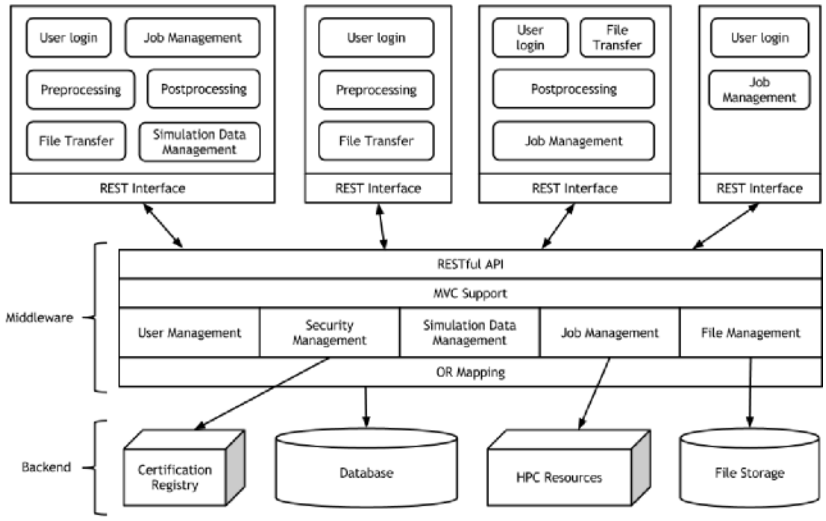

Figure 1. e-Spine System Architecture

User can do login to the middleware and the backend server through the client. Using pre-processing functions, the user prepares input data in files. There are some specific functions for spine-implant simulation such as automatic merger of spine-implant FEM models. The input files are automatically saved in file storage of backend under middleware's management. The remote-yet-connected storage provides user with convenience that there is no need to manage synchronization of many files from different clients in multiple locations. The middleware keeps the only copy of the file in the file storage in backend. User will always open the same file without noticing.

Once client send a request for simulation job with input data to the middleware, the middleware stores simulation data in database. There are two types of simulation data – simulation input data such as input files and input parameters, and meta data of the simulation jobs. The middleware submits the simulation job to the HPC resource after the storing data. The middleware monitors the simulation jobs that are running on HPC resource. When a simulation job finishes, the middleware will move the result files from HPC resource to backend file storage and write other related data in database.

Client can check whether a simulation is finished or not and find the result files. The client program downloads the result file and visualizes it in various ways of post-processing.

e-Spine middleware and backend play critical roles in processing user and job management. The middleware consists of user management and job management modules, which interact with a security module, certification registry, and job execution engine, to handle user authentication and job lifecycle management.

We adopt Grid Security Infrastructure (GSI) based security model, i.e., MyProxy, which open source software for managing X.509 Public Key Infrastructure (PKI) security credentials (certificates and private keys) developed at the National Center for Supercomputing Applications (NCSA). MyProxy combines an online credential repository with an online certificate authority to allow users to securely obtain credentials when and where needed. For users' login/logout requests, our user management module uses MyProxy APIs to authenticate and/or authorize the users, and finally stores X.509 proxy certificates (that usually have limited lifetime), on a temporal storage for the future use.

Job execution engine handles HPC job submission and monitoring via RESTful service, to provide simple and lightweight middleware for job management. Basically, Job execution engine provides flexible services for HPC job submission and monitoring due to using a workflow module. Job execution engine provides RESTful endpoints to submit and monitor jobs. Therefore the engine can process any simulation job submission from clients even of different domains from FEM simulation.

Our job management module supports basic job lifecycle (i.e., job submission, cancel, etc.) management and job monitoring. Upon the arrival of a job submission request, the job management module builds a job description XML document which includes HPC resource URI, job script (in our case, LoadLeveler command script), and the user's proxy certificate, etc. and calls job execution engine through RESTful endpoints with the job description as HTTP request body. Note that our module can renew the user's expired proxy certificate without password based on the certificate which has been temporally saved via the authentication procedure above. To manage up-to-date jobs' status efficiently, we employ some worker threads that periodically update job status for corresponding job_id onto the local database.

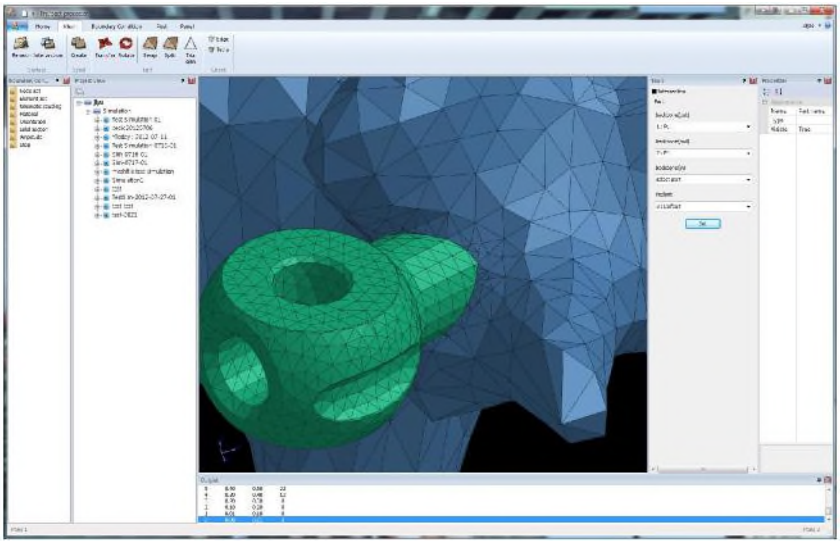

Figure 2. Screen of e-Spine Client

4 Implementation

Current version of e-Spine, in most cases, utilizes Abaqus that is a structural dynamics and heat-fluid dynamics simulation tool. However, any simulation tool can be installed in HPC resource and it can be served through middleware. The current implementation of e-Spine client is of Win32 platform. The RESTful interfacing modules of client is based on .NET framework. Pre-processing modules in the client use CM2 library in order to process mesh (finite elements: FE). The current implementation uses Abaqus post-processing API. The current version of e-Spine client is implemented in Win32 platform because the CM2 library is Win32. Figure 2 shows the screen of e-Spine client. Automatic merger of prmade spine and implant FEM models uses the library heavily as shown in Figure 3.

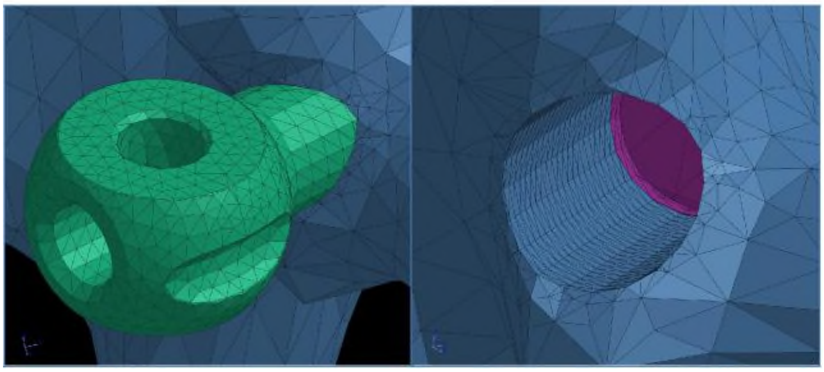

Figure 3. Automatic Merger of Spine and Implant using e-Spine Client

The e-Spine middleware was implemented in Java platform. The RESTful API layer was developed using Spring framework. We used MyBatis for OR mapping and data exchanging between database and other modules. The database in backend is MySQL. HPC resources are the super computer and PLSI resources that are owned and operated by Korea Institute of Science and Technology Information (KISTI). PLSI stands for the Korea national project – Partnership and Leadership for the Nationwide Supercomputer Infrastructure. The middleware and HPC resources communicate through 10Gbps high-speed network. Therefore the high-speed network guarantees that the system can stably support transferring huge-sized files among HPC resources, file storage, and middleware.

5 Conclusion

We described how e-Spine system maximizes flexibility and extensibility through RESTful architecture. Proposed system architecture enables e-Spine system to utilize custom-made security technology, to store and manage simulation data, to implement different types of clients, and to extend or modify middleware without any affects to clients. Moreover, new simulation tools can be installed in HPC resource without any change of existing client.

Using the extensibility and flexibility of e-Spine system, we plan to build the model database for standard and patient spine models in middleware and install additional functions for clients to retrieve and reuse for simulation. Though current application domain is a narrow one – spine-implant FEM simulation, it is possible to apply the system to general FEM. Also any simulation tools in domains different from FEM (now we have Abaqus) can be installed, so that various clients for different domains can conduct simulations at once. We can apply the simulation tools totally different from FEM. For example, simulation tools for nano-physics and chemistry can be applied.

Reference

1. Abaqus Unified FEA, http://www.3ds.com/products/simulia/portfolio/abaqus/overview/, Retrieved: August 23, 2012.
2. M. McLennan and R. Kennell, "HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering," Computing in Science and Engineering, 12(2): 48-53, March-April 2010.
3. EDISON project, http://www.joomla.edison.re.kr/, Retrieved: August 23, 2012.
4. R. Fielding, "Architectural Styles and the Design of Network-based Software Architectures", Ph.D Dissertation, University of California, Irvine, 2000
5. L. Richardson, S. Rudy, "RESTful Web Service", O'Reilly, ISBN 978-0-596-52926-0, 2007