Hybrid architecture for web simulators of pathological physiology

Tomáš Kulhánek[1,2], Marek Mateják[1], Jan Šilar[1], Pavol Privitzer[1], Martin Tribula[1], Filip Ježek[1,3], Jiří Kofránek[1]

March 4, 2013

[1]Institute of pathological physiology, 1.LFUK [2]CESNET z.s.p.o. [3]FEL ČVUT

1 Abstract

This work summarizes technology of web simulators with focus on the simulators running on the server and vizualization is performed on the client. This hybrid architecture of web simulator allows to perform long term simulation and demanding application on the server or remote capacity on computational grid and follow and control the simulation on mobile devices with limited computation capabilities. The result is a growing set of models and simulation application and a large scale physiological model Hummod published in web and web based simulation environment to enable basic simulation and manipulation with the available models.

2 Introduction

There are several methodologies and technologies how to model a complex reality in biological domain. Modelica is an object oriented language to express modeled reality in acausal way. Acausal modeling is a modeling style based on equations instead of assignment statements, which gives more flexibility to state which variables will be inputs and which will be computed as outputs without significant change of model itself[1]. Hester at al. published Hummod - a large scale physiological model, constructed from empirical data obtained from peer-reviewed physiological literature is described in

XML language which can be later modified by user to observe and test new approaches [2]. However, such description in XML language may be too complex for understanding and with a model introduces the proprietary modeling environment for the domain of human physiology. Our effort is to express the human physiology in standardized modeling language - Modelica. And it is now being used to update the model parts as well as to implement complex simulation application[3].

The Modelica language is implemented by several tools provided by different vendors. The open source OPENMODELICA `https://www.openmodelica.org/` developed by the Open Modelica Consortium, the DYMOLA provided by Dassault Systemes `http://www.dymola.com` and others. Current modeling tools implementing the Modelica language offer efficient way to develop a complex model with object-oriented techniques such as inheritance and encapsulation. These tools also offer an environment, where a simulation selected model and visualize and browse simulation results.

Current effort in education of human physiology is to publish the known models of human physiology as a web based simulation without need to install and maintain specialized tools by the user. The simulation capabilities of current modeling tools are limited to achieve this goal. Fortunately these tools offer several possibilities to export models and integrate them with different software technologies. Methods described bellow were mainly tested with the DYMOLA which implements Modelica language and allows some of the integration patterns to be implemented. OPENMODELICA was tested too as it offers open source tools for Modelica language.

3 web based simulators

The common characteristic of web-based simulation application is that they use web browser as a support for graphical interfaces connecting the user with simulation [4]. The downloading of a simulation package from a server to the client's computer and its execution as an application wholly independent of the web browser and internet is considered as native and not web-based simulation.

Byrne at al. [5] categorize the web based simulation into several categories, where we focus on the following three categories describing the application architecture:

(1) local simulation and visualization is where both the simulation engine and visualisation components are downloaded by the client to the user's local computer and the simulation and visualization is performed without the server interaction within the client web browser (Internet Explorer, Firefox,

Google Chrome, etc.). The enabling technologies for rich internet applications are e.g. Adobe FLASH, MS Silverlight, Java applets, HTML+Javascript, etc.

(2) remote simulation and visualization, where the simulation and animation rendering is performed on the server and the client only shows the rendered results.

(3) hybrid simulation and visualization architecture runs the simulation on the server. When the user connects to the server the client's visualisation engine is typically downloaded to the web browser and dedicated data connection may be established back to the server, which transfers simulation data for the visualisation engine.

The current effort the authors of this paper is focused on (1) and (3) approach. The (1) local simulation and visualization architecture is being developed and maintained within existing Atlas of physiology and pathophysiology [6]. The enabling technologies Adobe Flash and newly MS Silverlight are used to perform simulation and visualization within the client web browser. All computations are performed on clients, thus high client load is not threatening the server. This approach offer fast response time and is convenient for less complex models and for mass application deployment.

The (3) hybrid architecture is newly researched with the focus to execute long time simulation and computation on the server side and browse simulation results on the client. Furthermore the visualization could be performed on mobile devices (e.g. tablets) with limited performance.

3.1 model simulator integration

Modeling tools allows 2 approaches how to integrate model and it's solver into application and make a simulationa application. The first is to export model into an independent application able to be performed via command line and input parameters and files. This type of integration is usually referred as File-Transfer[7]. It's simple type of integration for building complex and enterprise application, the place and format of data must be known for both parts of the system. It's not possible to control simulation in real-time.

The second approach allows to export model into a library that follows either proprietary interface or some standard API. Dymola exports models into C sources which is then wrapped by the solver and compiled into DLL using it's own proprietary, but well documented API. Open source tool OpenModelica is being enhanced by the ability to export model directly into .NET libraries and which should be then controlled by .NET application, e.g. some of the simulation application published in Atlas of physiology and patophysiology [6] ....

The vendors of modeling tools agreed and developed a standard Functional

Mockup Interface(FMI) - a tool independent standard for the exchange of dynamic models and for co-simulation[8]. Main goal of FMI is to support exchange simulation models between different suppliers. Functional Mockup Unit (FMU) exported from the modeling language can be independently integrated into external application and accessed via FMI application interface.

Exported FMU prepared for co-simulation contains platform specific binaries. E.g. The Dymola or OpenModelica tool generates first source code in C language and then it is compiled and packaged using platform specific compilers into FMU containing dynamically linked libraries for the specific platform, dynamic-link library (DLL) for Microsoft Windows platform or shared objects (SO) library for Linux platform.

To verify the capabilities of hybrid architecture we developed a web service, which combines architectural style REST[9] and controlling of multiple simulators. The architectural style -Representational State Transfer- (abbreviated as REST) recommends to represent functionality as a set of resources identified by theirs URL and limited set of operation which are create, read, update, delete (abbreviated usually as CRUD) [9]. This limited set of operation can be mapped to the web protocol - HTTP methods (PUT,GET,POST,DELETE). Then the functionality of a simulation can be offered via several resources identified by URL and this fixed set of operation. Such service is usually called RESTfull service or RESTfull webservice.[9]

The proposed RESTfull web service currently stores these data: result of simulation, metadata about simulated model, description of visualization screens in an own domain specific language. The web service also provides different views on stored data - data with only relevant quantities can be retrieved. The functionality is being enhanced. Data are exchanged via HTTP protocol and Javascript Object Notation (JSON) data format with both parts. Simulation engine is invoked only if such task is needed to obtain new data not presented in database. ASP.NET and MVC framework is used in pilot implementation to build the REST service on top of the IIS - web server. Because the data of simulation related to the mentioned model Hummod ocupies several GB in binary form server stores the data in an SQL database using Entity Framework.

4 Discussion

The hybrid architecture of web based simulation doesn't want to replace the current technologies oriented on local web based simulation and visualization. Rather it allows to develop enhanced portfolio of application including long term simulation, identification of physiological systems and vizualization on

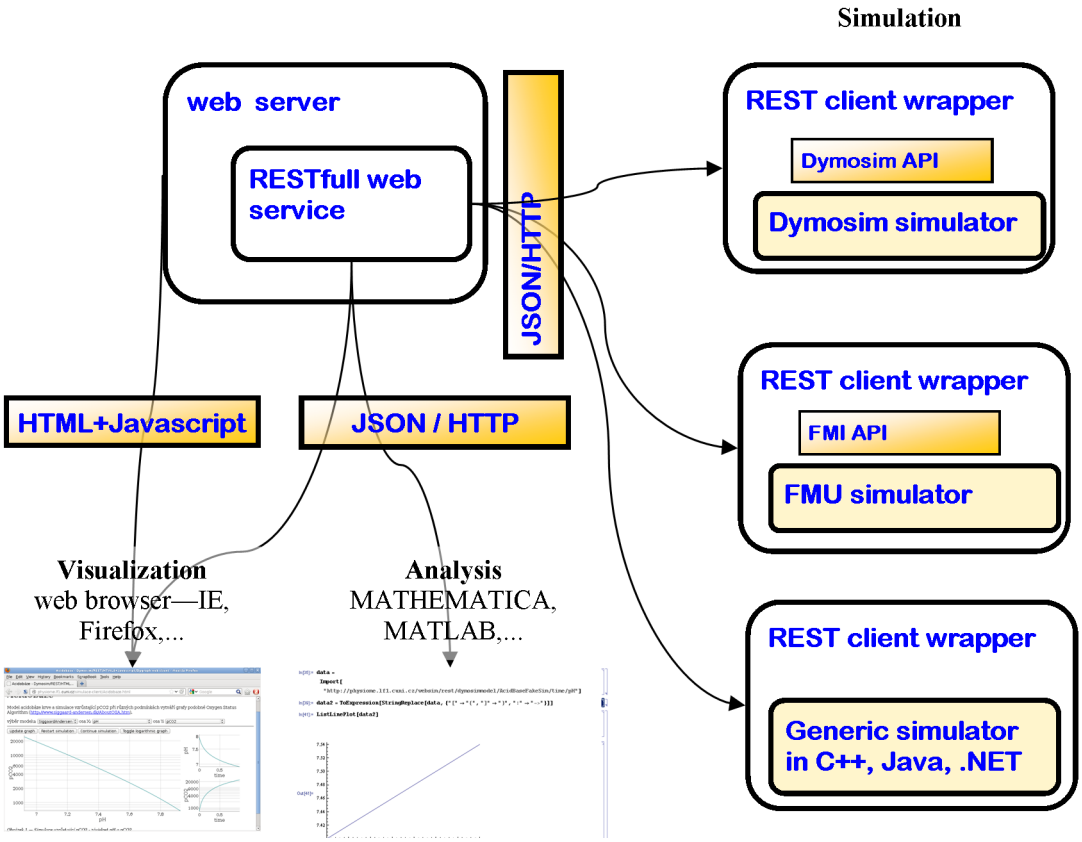

Figure 1: Hybrid architecture using RESTfull web service and simulation on server and visualization/analysis provided on client

mobile platforms with limited computational capabilities.

Tha atlas of pathological physiology can be enhanced by repository of data which are not present in current implementation, including source codes of models, it's description, definition of vizualization and simulation which can be performed remotely. The loose coupling between web service and simulation exported from modeling tools can be used to delegate simulation to the grid infrastructure available for scientific usage in a grid initiatives, e.g. european grid initiative `www.egi.eu`. However this new functionality needs to implement robust solution for authorization which will allow only certain users to perform computation demanding simulation etc.

5 Acknowledgment

This work was funded by the fund of development association CESNET.

References

[1] P. Fritzson and Brunus P. Modelica – a general object-oriented language for continuous and discrete-event system modeling and simulation. *Simulation Symposium, 2002. Proceedings. 35th Annual*, pages 365–380.

[2] Robert Hester, Alison Brown, Leland Husband, Radu Iliescu, William Andrew Pruett, Richard L Summers, and Thomas Coleman. Hummod: A modeling environment for the simulation of integrative human physiology. *Frontiers in Physiology*, 2(12), 2011.

[3] J. Kofranek, M. Matejak, and P. Privitzer. Hummod-large scale physiological models in modelica. *Proceedings 8th Modelica Conference, Dresden, Germany, March 20-22, 2011, 713-724, Linkoping Electronic Conference.*

[4] Bencomo S.D. Control learning:present and future. *Annual Reviews in Control*, 28:155–136, 2004.

[5] J. Byrne, C. Heavey, and P. J. Byrne. A review of web-based simulation and supporting tools. *Simulation Modelling Practice and Theory*, 18(3):253–276, 2010.

[6] J. Kofránek, S. Matoušek, J. Rusz, P. Stodulka, P. Privitzer, M. Mateják, and M. Tribula. The atlas of physiology and pathophysiology: Web-based multimedia enabled interactive simulations. *Computer methods and programs in biomedicine*, 104:143, 2011.

[7] Gregor Hohpe and Bobby Woolf. *Enterprise integration patterns: Designing, building, and deploying messaging solutions*. Addison-Wesley Professional, 2004.

[8] Torsten Blochwitz, M Otter, M Arnold, C Bausch, C Clauß, H Elmqvist, A Junghanns, J Mauss, M Monteiro, T Neidhold, et al. The functional mockup interface for tool independent exchange of simulation models. In *Modelica'2011 Conference, March*, pages 20–22, 2011.

[9] Roy Thomas Fielding. Chapter 5: Representational state transfer (rest). *Architectural Styles and the Design of Network-based Software Architectures, Dissertation*, 2000.