Publishing House
AKAPIT

COMPUTER METHODS IN MATERIALS SCIENCE
Informatyka w Technologii Materiałów

Vol. 10, 2010, No. 3

WEB SERVICE FOR FINITE ELEMENT MESH GENERATOR

NORBERT SCZYGIOL, JULIUSZ MIKODA*, ANNA WAWSZCZAK

Institute of Computer and Information Sciences, Czestochowa University of Technology
ul. Dąbrowskiego 73, 42-200 Częstochowa
**Corresponding juliusz.mikoda@icis.pcz.pl*

Abstract

In this paper, we present a system that could be used for finite element mesh generation for two- and three-dimensional domains and an interface that enables use of this system with the aid of a web site. The authoring mesh generator was used for creating finite element meshes. It uses the Advancing Front Technique (for 2D meshes) and the Delaunay tetrahedralization in connection with the Advancing Front Technique (for 3D meshes) used for inner nodes generation. The architecture of presented system was built on the basis of the architecture design pattern Model-View-Controller (MVC). The mesh generator is available with the aid of the interface that is presented in this paper or any other interface created by the user. The communication between View and Controller, based on Web Services architecture, enables use of whichever device and operating system, that provides basic network services. Thanks to complete separation of the View layer, it is possible to use the generator described in this paper inside another engineering application.

Key words: mesh generator, Delaunay meshes, FEM, MVC, WebServices, Advancing Front Technique

1. INTRODUCTION

Over recent years the Internet has significantly changed its character. Web sites, that offer not only texts, graphics or multimedia but also different types of services, constitute bigger and bigger part of it. Such services could be successfully used by developers that design engineering software, as well as by users of such applications. Very often the calculations that are made during simulation process could be divided into a few step. Each of them has a significant impact on the reliability of the obtained results. In the case of the Finite Element Method, these steps are: mesh generating, matrix building, system equation solving.

In this paper, we describe a system providing a service that enables accomplishing the first of these steps. This service is built on the basis of the Web Services technology. Finite element mesh generation could be done, among others, with the aid of the web site. However, it is just an example of a user interface that could make use of the described service. Therefore such services are accessible through SOAP (W3C Group, 2000), they could be successfully invoked by external engineering packages without relaying on the web interface.

2. TETRAHEDRAL AND SURFACE ELEMENTS MESH GENERATOR

In the system described in this article, the authoring two-dimensional (triangles) and three-dimensional (tetrahedrons) elements mesh generator is used. Mesh creating method that was used in this generator, is based on the Advancing Front Technique. Additionally, it uses Delaunay meshes.

The basic presumptions of the mesh generation method that was used in this work, were described by Mikoda and Sczygiol (2005). The algorithm that is described there, was extended with addition of a module that enables controlling correctness of meshes that are created. This module creates Delaunay triangulation – an initial mesh, that encircles whole domain that is to be filled up. All triangles that belong to this mesh are transformed in order to fulfill Delaunay criteria. The next step that has to be made, depends on creating inner elements. Their shapes are determined in accordance with the Advancing Front Technique criterion. The initial mesh is extended by insertion of triangles and nodes. This mesh is used during the phase of nodes insertion to check the position of new inserted node relatively to opposite front. The method of generation of the tetrahedral elements meshes was described by Sczygiol and Mikoda (2006). There is one great weakness of this method, that the positions of nodes and tetrahedrons inserted to the mesh must be precisely checked. The determination of a mutual position of the opposite fronts, that is required to get through the mesh generation process, is very time-consuming. It has a significant effect on the increase of the amount of time that is needed to preform this task. The same as in the case of 2D mesh generator, the algorithm of 3D mesh generation that is used in this work is extended by the module that enables controlling of correctness of generated meshes. At the beginning, an initial mesh is created. It encircles whole domain that is intended for being filled. The initial mesh takes shape of a hexahedron. All nodes situated on the surface of the domain that is to be filled up, are inserted into this mesh. The elements of the initial mesh could be divided into two groups. The elements that are situated outside the domain that is to be filled up and these that are placed inside the domain. At the interface of these two groups of elements arises a front that is used to generate the inner elements of the mesh. The mesh generated in this way enables control location of inserted nodes, whose coordinates are determined according to the Advancing Front Technique criterion (Sczygiol & Mikoda, 2006). An inserted node lands up into the mesh of elements inside the domain that is being filled up. The location of the opposite front could be determined by finding the element of mesh, in which the new inserted node is placed. Using this two methods, the Advancing Front Technique and the Delanay meshes, allows to combine full control over the size of created elements (the Advancing Front Technique) and assurance that the mesh fills up whole domain (the Delanay mesh). A use of initial mesh has a significant influence on the decrease of the number of operations needed to check location of inserted point. Furthermore, it decreases a risk of creating incorrect mesh.

3. ARCHITECTURAL DESIGN PATTERN MODEL-VIEW-CONTROLLER

The architecture of the system presented in this paper is designed on the basis of the architectural design pattern Model-View-Controller (MVC) described by Trygve (1979). This architecture is presented in figure 1.

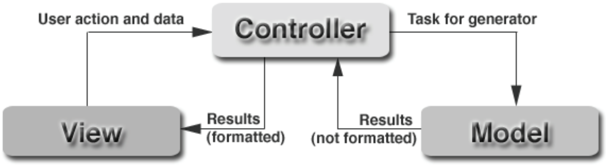


Fig. 1. *Design Pattern Model-View-Controller.*

The main target of using MVC pattern is to separate View, that is part of code specific for the user interface, from Model, that can be treated as a business layer. In this case the finite element mesh generator constitutes the Model module. The Controller module is placed between View and Model. It is responsible for interpretation and preparation of data that are submitted by Model. Additionally, Controller is responsible for providing event handler for events that are generated by the user interface and submitting information about these events to Model. The application of such solution enables using different user interfaces without necessity of making any changes in other parts of the system.

4. THE MODULE RESPONSIBLE FOR REMOTE MANAGEMENT OF THE GENERATOR

In described system, the role of the Controller module is played by the module responsible for remote management of the generator. It is responsible for bidirectional data exchange between View and Model. This module makes two main tasks:

- processes events generated by the View and on its basis controls work of the generator,
- gets data from the generator, formats them and sends the formatted data to the View.

The system described in this paper enables use of them by a lot of users simultaneously. To realize this goal, there is a queue of submitted tasks created in Controller. At the moment when the task is submitted, Controller checks correctness of the shape description that is sent by View. The shape description is then written into the database and the task is given an unique ID. The task created in such way is put into the queue and waits for the possibility of being processed by the generator. Currently, the number of generator instances running simultaneously is limited to four, due to the number of processor cores that are available.

5. COMMUNICATION BETWEEN THE VIEW AND THE CONTROLLER

Communication between View and Controller is realized with the aid of a Web Services technology described by W3C organization (2004). The module that plays role of Controller provides a service of finite elements mesh generation. This service is described by the WSDL document (Web Services Description Language) described by W3C Group (2001). The module gets the WSDL document from a given localization and then calls the service provided by the Controller. The service request is prepared on the basis of the information included in this document. The Simple Object Access Protocol (SOAP), described by W3C group (2000), is used to remote call of this service.

Figure 2 shows the model of communication between View and Controller.

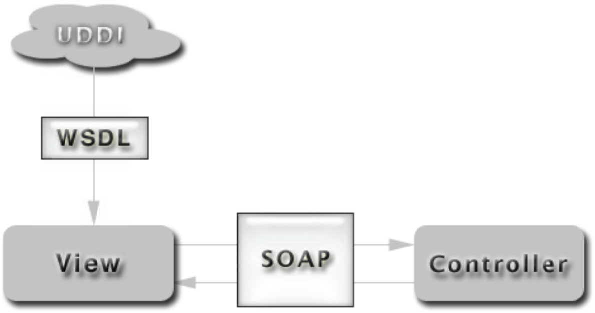


Fig. 2. Communication between View and Controller.

6. E-MESHGEN WEBSITE - AN EXAMPLE OF USER INTERFACE

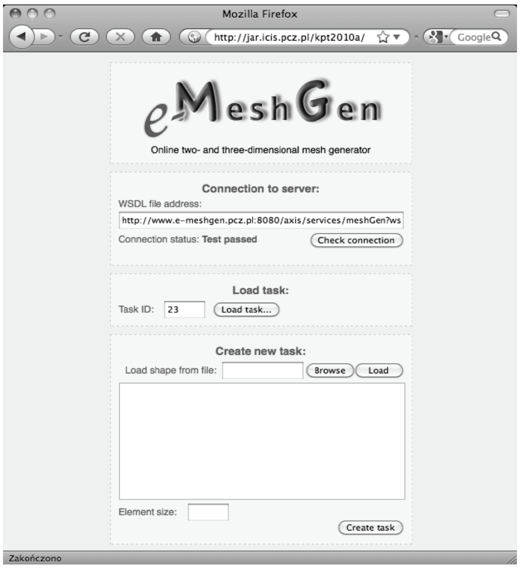
The Web Services technology enables use, as an user interface, any application that is running on device and operating system that provides basic network services. In the system described in this article, the website e-MeshGen perform a function of View. It provides the interface that enables:

- loading shape description and launching mesh generation process,
- supervision of the tasks launched earlier,
- displaying the shape of the domain, two variants of surface mesh and generated 3D mesh (as graphic and as text).

The description of the area's shape should be prepared in GID ASCII format and loaded as text file on the website. The resulting mesh could be obtained as text transcribed in format dedicated for NuscaS System. It can be downloaded as text or binary file. Generated mesh is also displayed as GIF image that shows full 3D mesh or surface mesh. These images are created on the server side using graphical context provided by Java module. This mesh could be rotated, moved and scaled. The mesh could be also obtained as a set of points dedicated for OpenGL visualization. During the process of mesh generation, the user that launched such process could get the state of it. The are 6 states of process:

- "Waiting" – waiting for processing,
- "Shape loading" – reading mesh description from file,
- "Shape testing" – testing correctness of shape description,
- "Surface mesh generating" – generating mesh on the surface of shape,
- "Nodes and tetrahedrons inserting" – generating mesh elements inside area that is to be divided,
- "DONE" – mesh generation process is finished.

Figure 3 shows the interface that enables creating and submitting the new task as well as loading the task submitted earlier, identified by given ID. Figure 4 shows interface that enables supervise of task state and display shape and meshes that have been generated.


Mozilla Firefox

http://jar.icis.pcz.pl/kpt2010a/

eMeshGen

Online two- and three-dimensional mesh generator

Connection to server:

WSDL file address: http://www.e-meshgen.pcz.pl:8080/axis/services/meshGen?ws

Connection status: Test passed

Load task:

Task ID: 23

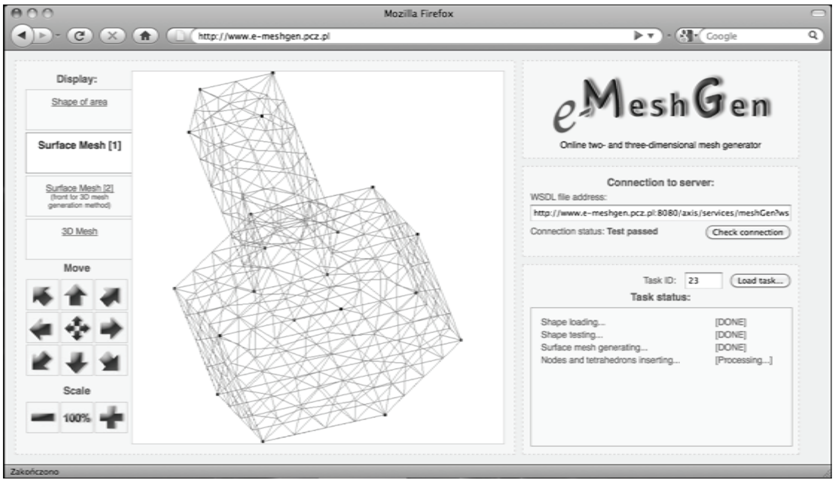
Create new task:

Load shape from file:

Element size:

Zakończono

Fig. 3. Portal e-MeshGen – creating and loading tasks.


Mozilla Firefox
http://www.e-meshgen.pcz.pl
Google
Display:
Shape of area
Surface Mesh [1]
Surface Mesh [2] (front for 3D mesh generation method)
3D Mesh
Move
Scale
100%
eMeshGen
Online two- and three-dimensional mesh generator
Check connection
Load task...
Zakończono

Connection to server:

WSDL file address: http://www.e-meshgen.pcz.pl:8080/axis/services/meshGen?ws

Connection status: Test passed

Task ID: 23

Task status:

Shape loading... [DONE]
Shape testing... [DONE]
Surface mesh generating... [DONE]
Nodes and tetrahedrons inserting... [Processing...]

Fig. 4. Portal e-MeshGen – task management.

Currently, this web site could be accessed only from the local network of the Technical University of Czestochowa. In the future, the service that provide functionality of mesh generation will constitute part of wsMES system that will be accessible from the Internet for all engineers registered in this system.

As another example of user interface could be considered an application running on mobile phone that enables supervision of time-consuming process of large and complicated three-dimensional meshes generation. For instance, such application could get intermediate results of generation process or just notify user of the completion of this process.

Additionally, complete isolation of the View layer enables use of described generator inside another engineering application. It is possible to call web service that generate mesh directly from source code of another application. This application could download generated mesh and use it for calculation realized in this application. This operation could be perform transparently for the user.

7. SUMMARY

The system described in this paper shows the possibilities that are given by new internet technologies and network services. This possibilities could be successfully used during development of engineering applications. The Model-View-Controller architecture gives a possibility of extending and modifying described system in a very simple way. It applies to the interface dedicated for the end-user as well as to the module responsible for mesh generation. The system extended by additional mesh generators, make it possible to use this application by larger number of users at the same time. Bringing this modification into the system is completely invisible for the end-user. It is unimportant for the end-user where his data are processed. For instance, modified Controller module could use a lot of distributed machines and divide tasks between them depending on their load. Furthermore, realization of respective levels of engineering simulations as independent modules that are provided as services, will significantly simplify and speed up development of the authoring engineering applications. Applying ready-made mesh generator will help saving time and allows to concentrate on work relevant to other levels of simulation. There is also a lot of possibilities of extending described system, for example adding new modules responsible for realization of other parts of engineering simulations. Integration of such modules enables development of complete engineering tools that provide functionality compared to the functionality of commercial desktop applications. These modules could be developed independently by different teams and running on different machines. For the end-user, such system will constitute an entirety and any modifications brought in to respective modules will remain imperceptible for him.

REFERENCES

Mikoda, J., Sczygiol, N., 2005, Nowy frontalny generator siatek elementów skończonych, *Materiały XII konferencji Informatyka w Technologii Metali - KomPlasTech'2005*, 183-190 (in Polish).

Sczygiol, N., Mikoda, J., 2006, Niestrukturalny generator przestrzennych siatek elementów skończonych oparty na metodzie postępującego frontu, *Materiały XIII konferencji KomPlasTech 2006*, 61-68 (in Polish).

Trygve, R., 1979, *Models-View-Controllers*, Xerox PARC technical note.

W3C Group, 2004, *Web Services Architecture*, W3C Working Group Note.

W3C Group, 2001, *Web Services Description Language*, W3C Note.

W3C Group, 2000, *Simple Object Access Protocol*, W3C Note.

USŁUGA SIECIOWA UMOŻLIWIAJĄCA GENEROWANIE SIATEK ELEMENTÓW SKOŃCZONYCH

Streszczenie

W niniejszej pracy zaprezentowany został system umożliwiający tworzenie siatek elementów skończonych dla obszarów

2D i 3D oraz interfejs umożliwiający wykorzystanie tego systemu za pośrednictwem strony internetowej. Do tworzenia siatek elementów skończonych zastosowany został autorski generator wykorzystujący metodę postępującego frontu (2D) i tetrahedryzację Delaunaya (3D) w połączeniu z metodą postępującego frontu, stosowaną do generowania elementów wewnętrznych. Architektura prezentowanego systemu zbudowana została w oparciu o wzorzec projektowy Model-Widok-Kontroler (MVC). Generowanie siatek dla zadanego obszaru możliwe jest za pośrednictwem zaprezentowanego w pracy interfejsu webowego lub innego interfejsu stworzonego przez użytkownika. Oparcie komunikacji pomiędzy widokiem a kontrolerem na technologi WebService umożliwia wykorzystanie dowolnych urządzeń i systemów operacyjnych udostępniających podstawowe usługi sieciowe. Całkowita izolacja warstwy widoku pozwala również na użycie opisywanego generatora wewnątrz własnych aplikacji symulacyjnych.

Received:
Received in a revised form:
Accepted: