

GEM Portal Example

Preliminaries

To make full use of this portal, email the IP address of your browser machine and a request for a valid username/password to marpierc@indiana.edu with copies (cc:) to Jay.W.Parker@jpl.nasa.gov and Peggy.Li@jpl.nasa.gov. Some additional portal fields have been added since the screen shots below; these are not considered confusing.

Overview

The GEM portal and testbed consists of the following pieces:

1. A portlet-based portal environment that allows users to customize their displays and service interfaces.
2. Support for the GEM codes Disloc, Simplex, and GeoFEST.
3. XML based Web services that allow the user to
 - Transfer files from desktop to backend and between backend.
 - Submit jobs
 - Monitor job progress on backend resources
 - Archive and resubmit old jobs
 - Visualize GeoFEST output
 - Exchange fault and dislocation information between Disloc and Simplex.
 - Services for managing portal applications.

The general architecture is shown in Figure 1. For the testbed, we use the following resources:

- Complexity, a Sun Sunfire 880 server, acts as the main web server, runs the portal, and manages connections to remote services through client stubs.
- The fault database is hosted at USC.
- Services for job submission, file management, and job monitoring are hosted on a range of Linux and Solaris servers (danube, noahsark, solar, and grids).
- Complexity also hosts several local services for session management.

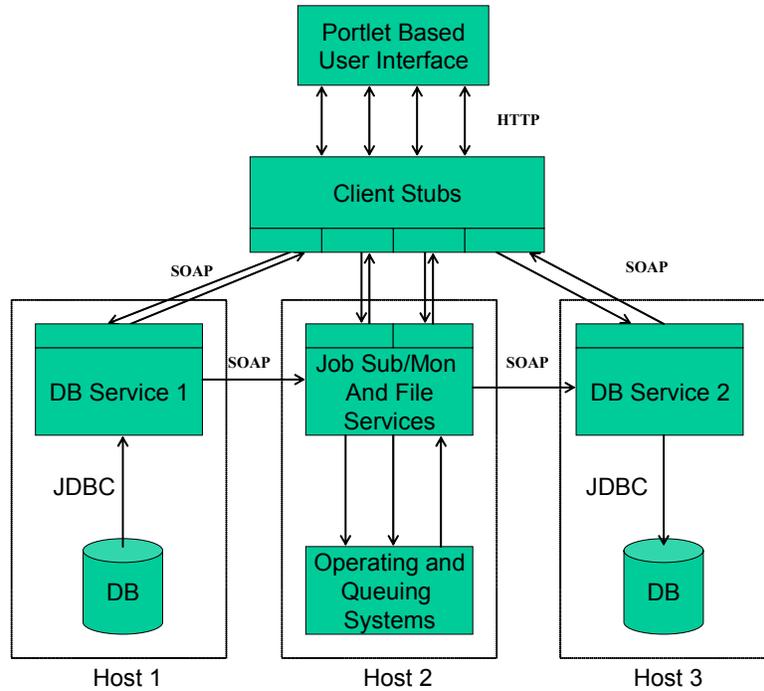


Figure 1 GEM Architecture

Running GeoFEST with Visualization

To simplify the demonstration of the portal capabilities, we have integrated several steps needed to set up, run, and visualize GeoFEST into a single set of linked steps. The portal should be accessed with Internet Explorer, Netscape 6 or 7, or Mozilla. *Please do not use Netscape 4.x, which has a well-known HTML table bug.*

Logging In

1. Go to <http://complexity.ucs.indiana.edu:8282/jetspeed>. You should see the screen below, Figure 2.
2. Log in with a valid username/password (see “preliminaries” section, this document). You should see the screen in Figure 3.

GEM Portal Example: Milestone I

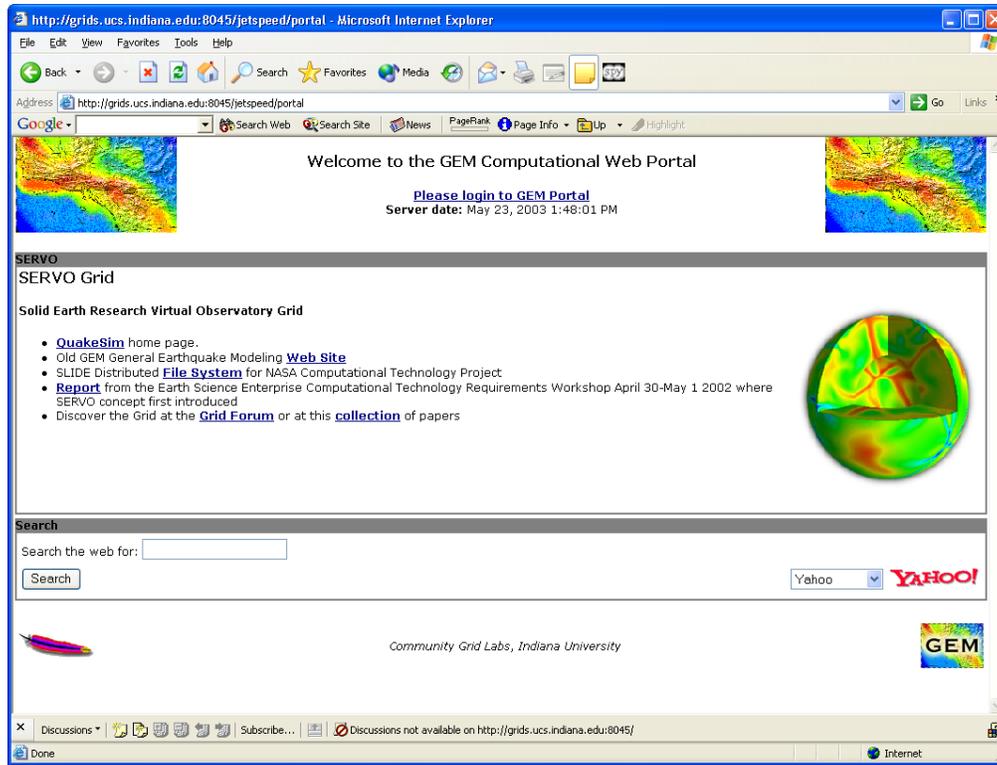


Figure 2 GEM Portal Login

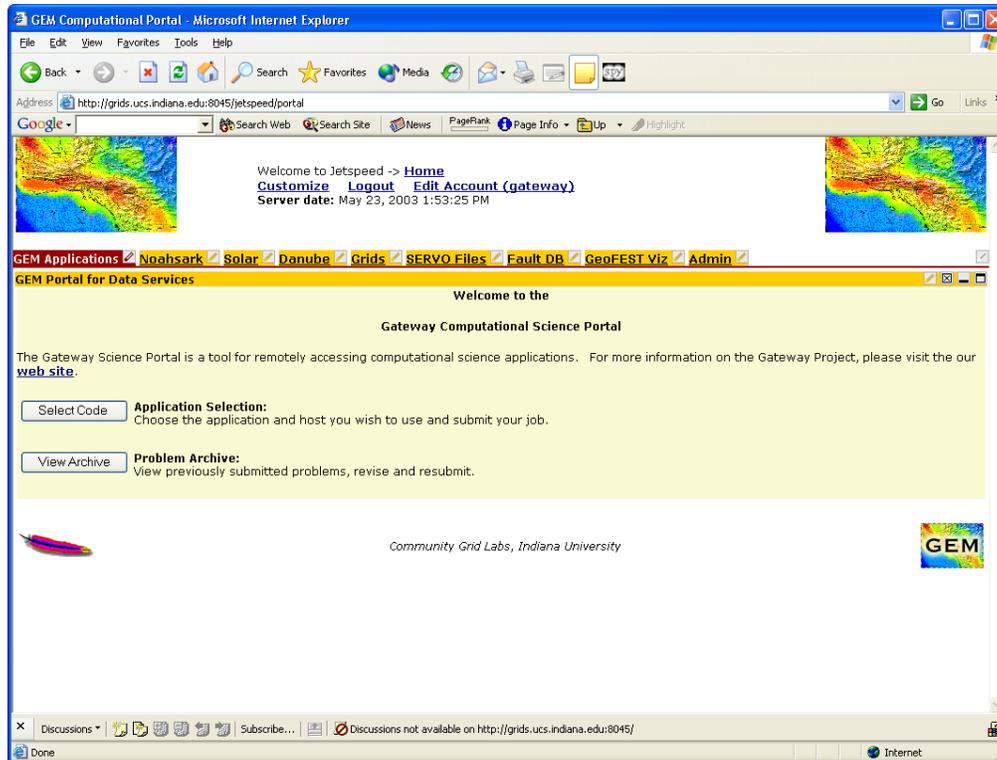


Figure 3 First page after login

Select and Setup GeoFEST

Choose "Select Code". You should see the the screen in Figure 4. Choose "GeoFEST_Plus_Viz".

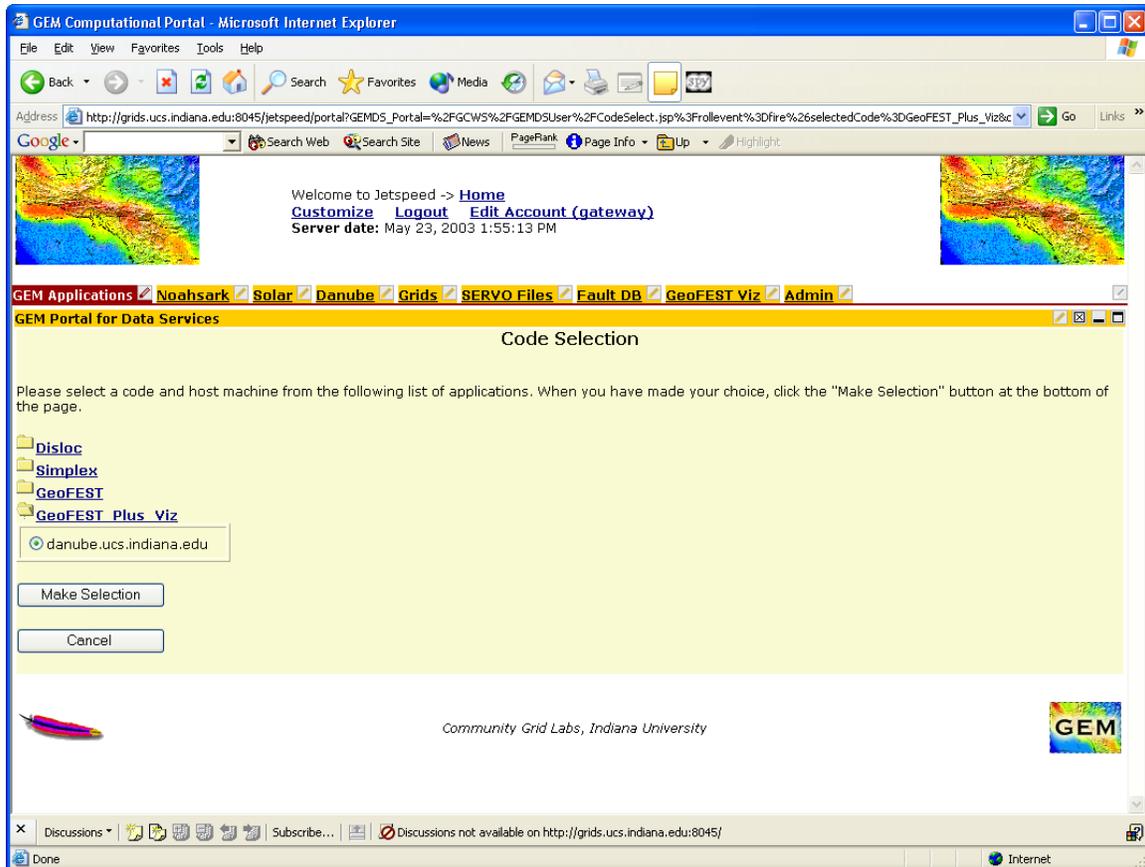


Figure 4

You may next either set up a new project or choose from archived GeoFEST projects (Figure 5). In either case, you will be able to create layer geometries and faults using the interface shown in Figure 6. The “Load Project” option allows you to reload a previously created session, while the “New Project” option creates a blank project.

Important Note: the current version of the system requires that layers be added from bottom to top. For example, the Northridge layers in the layer database should be entered in the following order: NorthridgeAreaMantle, NorthridgeAreaMidCrust, and NorthridgeAreaUpper.

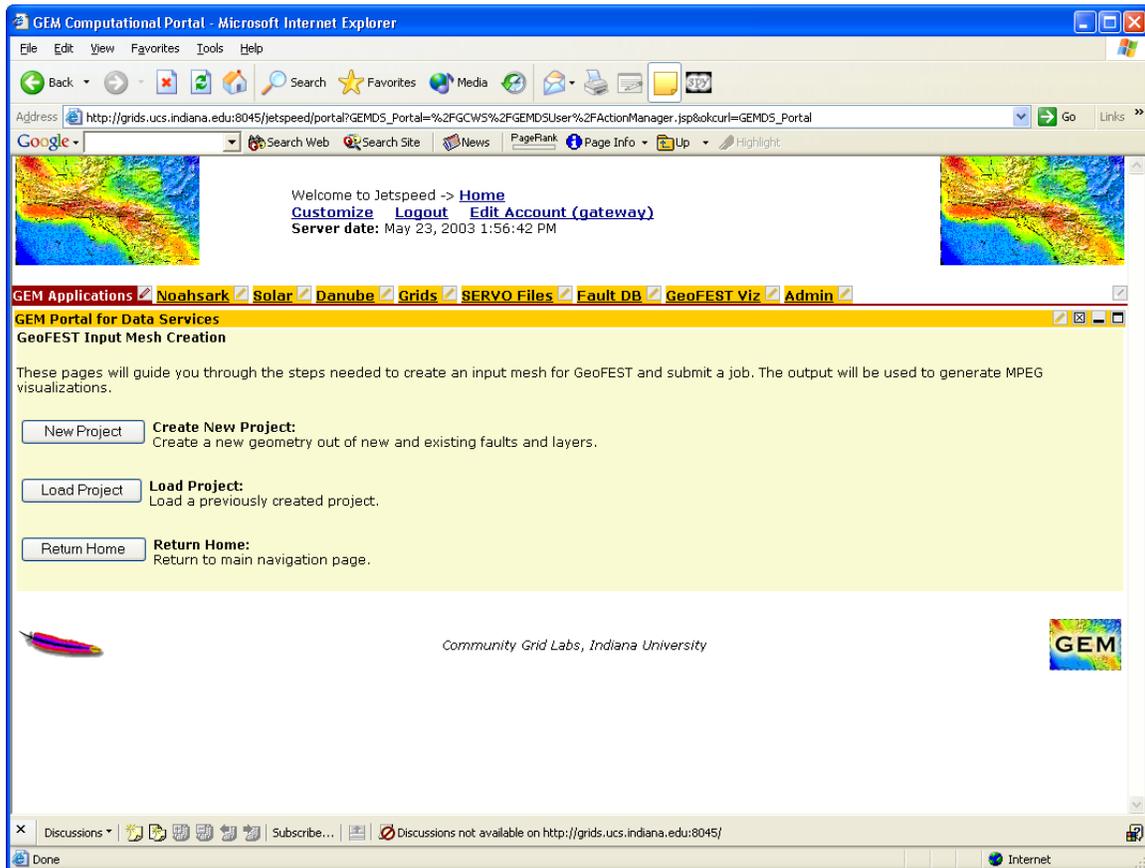


Figure 5 Create a new project or load a previous project.

Where prompted in Figure 6, enter a name for your project. You may also select to add layers and a fault to the system. Both layers and faults may be either entered in by hand or obtained from system's databases. Figure 7 shows a typical screenshot for adding a layer geometry. Fault input forms are similar.

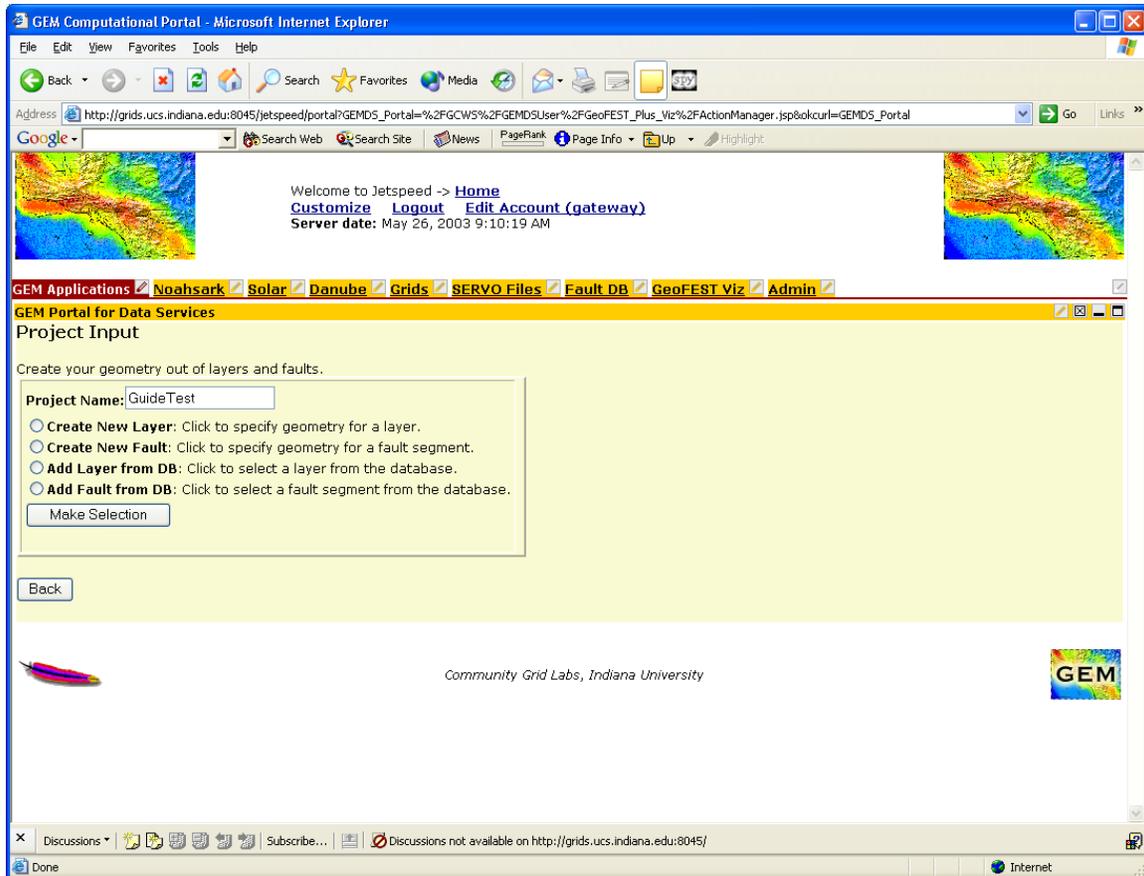


Figure 6 Give the project a name and select to add either layers or faults.

GEM Portal Example: Milestone I

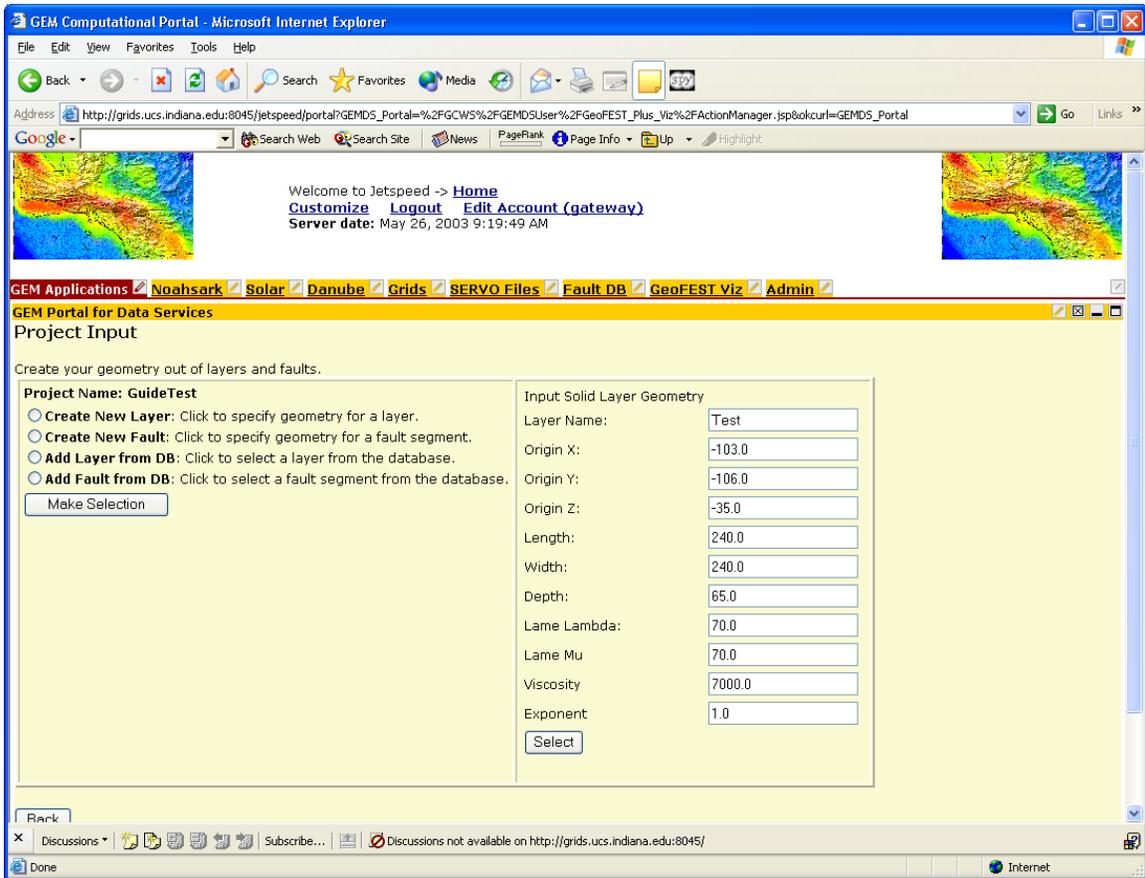


Figure 7 Add a layer geometry by filling in forms.

Database values may be selected by either the Layer or Fault name, as shown in Figure 8.

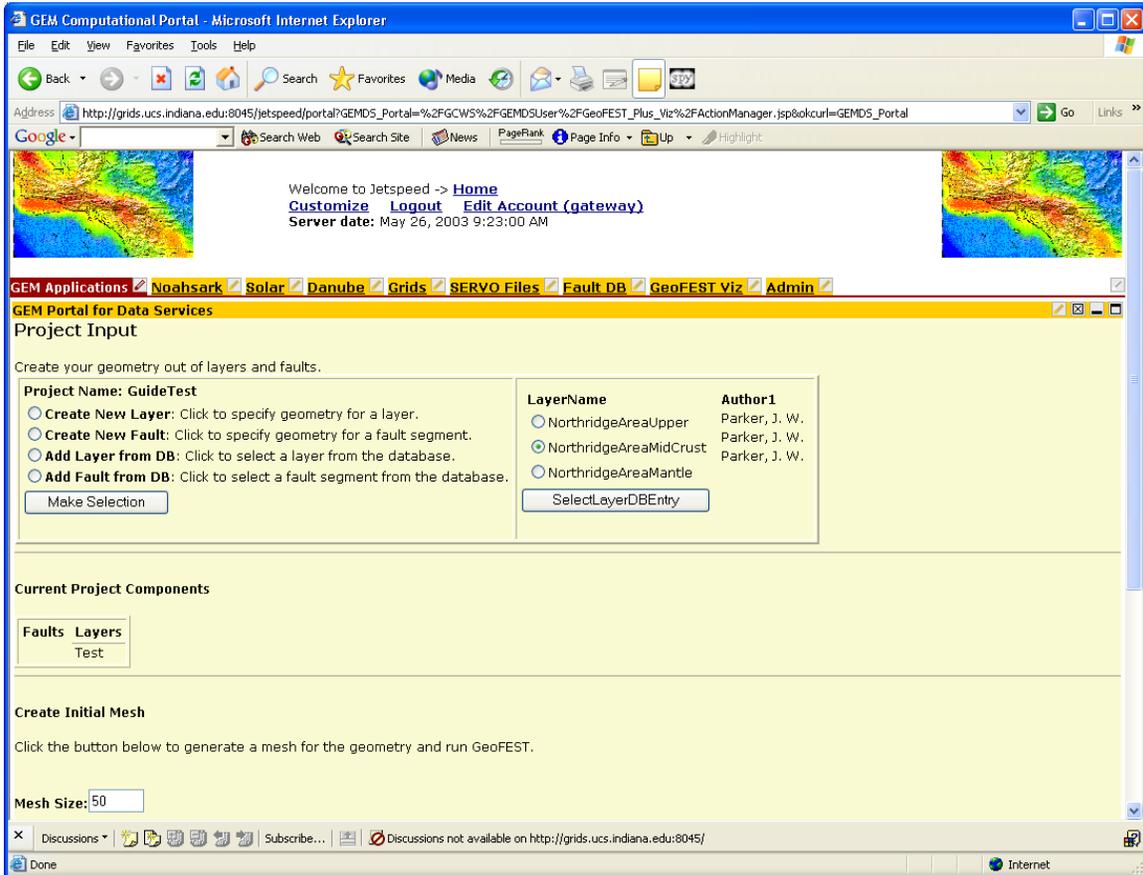


Figure 8 Layers may be selected from the database by name.

Create an Initial Mesh

After you have created your layer and fault geometry, you may generate and refine your finite element mesh. Click the “Generate Mesh” button (in the “Create Initial Mesh” region of Figure 8) and you will be directed to the screen shown in Figure 9.

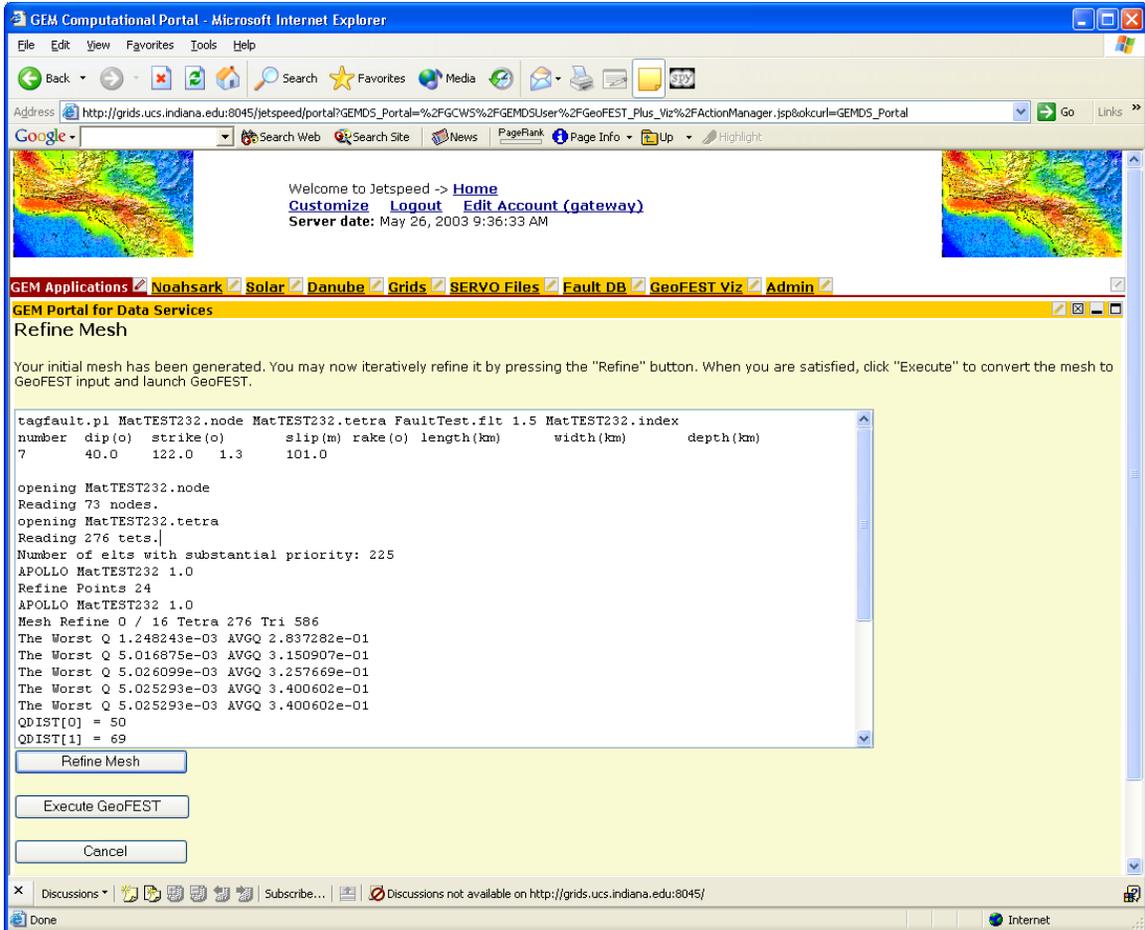


Figure 9

Successively refine your mesh by selecting the “Refine Mesh” button. The portal will block for increasingly longer times while the mesh generator is working (a few seconds at most). As a rule of thumb, the mesh should be refined until the “number of elements with substantial priority”, after steadily increasing with each refinement, begins to drop. For the sample Northridge case, this occurs when the total number of points is approximately 10,000 and the number of tetrahedrons is about 50,000.

Submit GeoFEST

After you have satisfactorily refined your mesh, you may convert this into a GeoFEST input file and run GeoFEST. You should provide values for the indicated fields or use the defaults. Enter your email address and you will be emailed a link to the movie when the job has completed.

GEM Portal Example: Milestone I

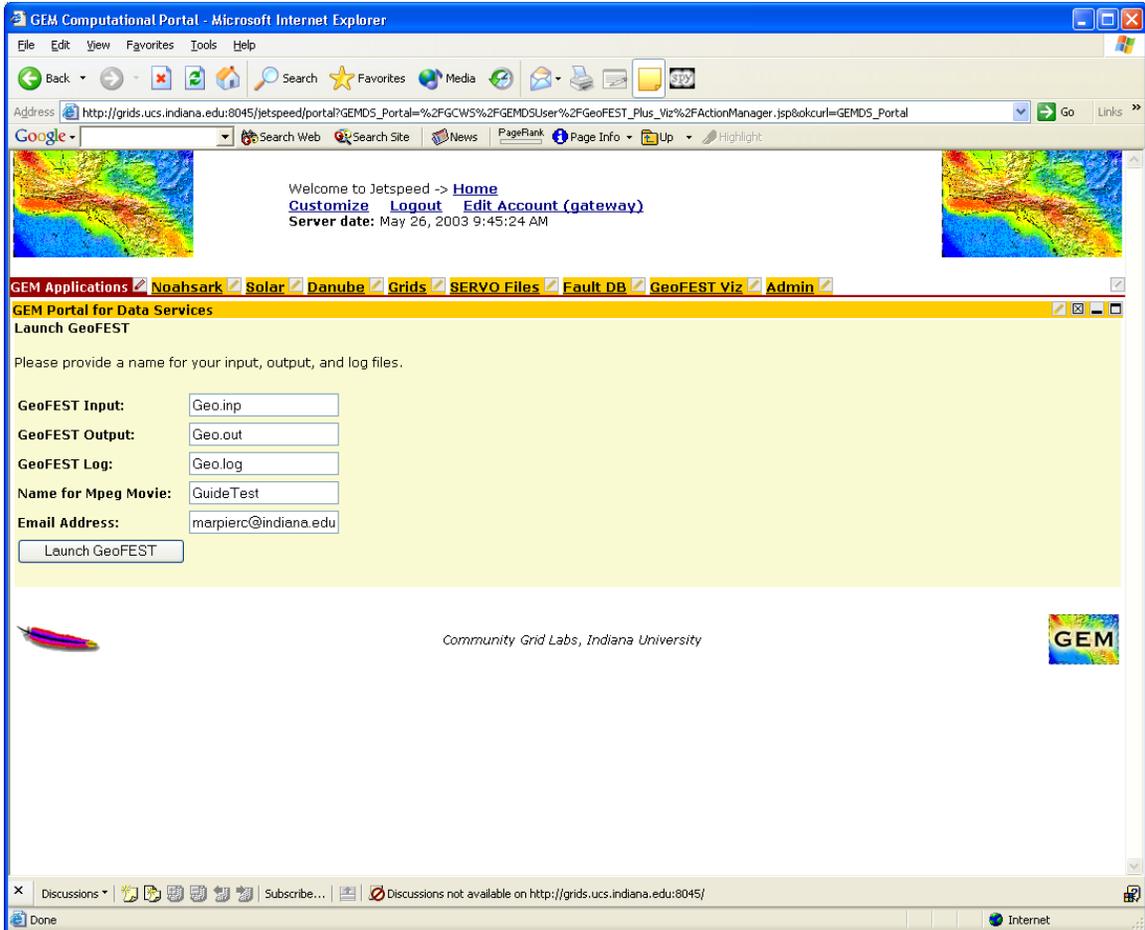


Figure 10 Prepare to launch GeoFEST.

You may verify that your job is running by selecting the “Danube” tab on the top navigation bar. You should see a screen similar to Figure 11.

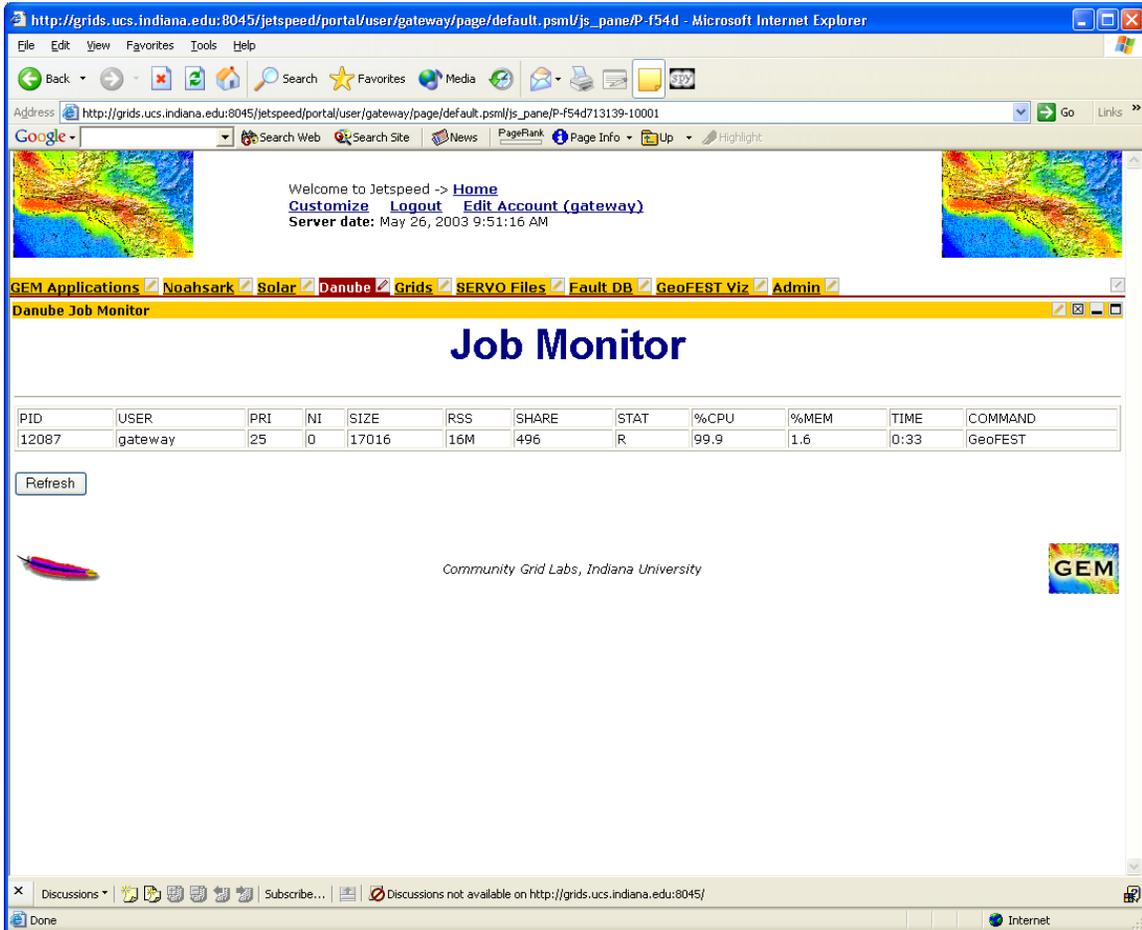


Figure 11 Verify that GeoFEST has been launched.

A GeoFEST run on Danube with 10,000 nodes/50,000 elements for the Northridge simulation takes about 45 minutes. Generating the movie of the output requires another 15-20 minutes.

Running GeoFEST, Disloc, and Simplex

Guide to be written. Interface may be used.

Troubleshooting Problems

- If you have problems or wish to report bugs, please contact Marlon Pierce: marpierc@indiana.edu or 812-856-1212. You may also contact Choonhan Youn, cyoun@ecs.syr.edu if the servers need to be restarted.
- The portal is just beginning user testing and bugs will be found, so if you find a problem that you can't recover from, close your browser and open a new one. This will remove all browser sessions.
- Remember that the portal currently only supports meshes created from 3 layer geometries with a single embedded fault.

Third Party Codes and GeoFEST Helper Applications

The portal is based primarily on the following technologies:

- Jetspeed is used to build the portal framework (for which we have added some extensions).
- JavaServer Pages are used to create the web pages.
- Remote invocations and communications between various machines and databases in the testbed use Web services running in the Apache Axis implementation of WSDL and SOAP.
- We use apache-ant (with various custom extensions) to coordinate related tasks (such as the steps needed to create the initial mesh or to generate the movie).

The mesh generation, refinement, input creation and movie creation are all done through calls to several external programs. We typically use apache-ant to invoke these non-Java applications.

- AKIRA is used to generate the mesh. We call Akira directly and also through a legacy control code, CROM.
- APOLLO is used for mesh refinement (through tagfault.pl).
- GeoFEST uses a number of helper Perl scripts (refonce.pl, tagfault.pl, and lee2geof43m) to create the mesh and apply boundary conditions and material properties.
- The mpegs are generated using RIVA.

The appendix describes an earlier graphical interface for AKIRA, CROM and APOLLO (the mesh generation technology used in this portal).

Very Near Future Work

- We need to make the portal more robust by incorporating error handling of the third party codes. Currently, crashes or problems with these codes cannot be handled by the portal user.
- We are in the process of adding simple web visualizations for the fault and layer geometries using VTK. We will follow this with visualizations of the mesh. These initially will be non-interactive, with interactivity (zooms, rotations, transparency adjustment) added as follow on work.

Final Report on guiVISCO: A Graphical User Interface for Mesh Generation for Viscosity Analyses

By Jin-fa Lee, Ohio State University

guiVISCO

On the Linux command line, make sure guiVISCO program is located in your command path, simply type in guiVISCO and a pull down menu as the one shown in Fig. 1 will pop up on your window. Currently, guiVISCO has only three major menu: File, Cmds, and View.

Briefly, the *File* menu is used to create projects, *Cmds* includes commands to create layers as well as faults, and also the command to create a uniform mesh once a project is defined (through File menu), and *View* provides very limited viewing capabilities of geometry as well as tetrahedral meshes.

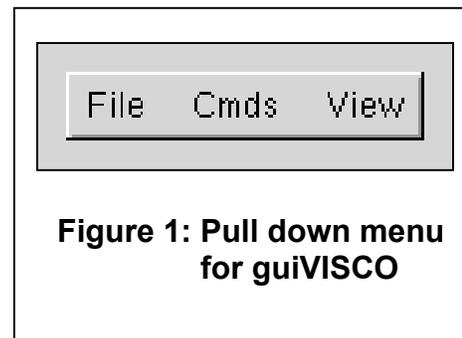


Figure 1: Pull down menu for guiVISCO

File menu

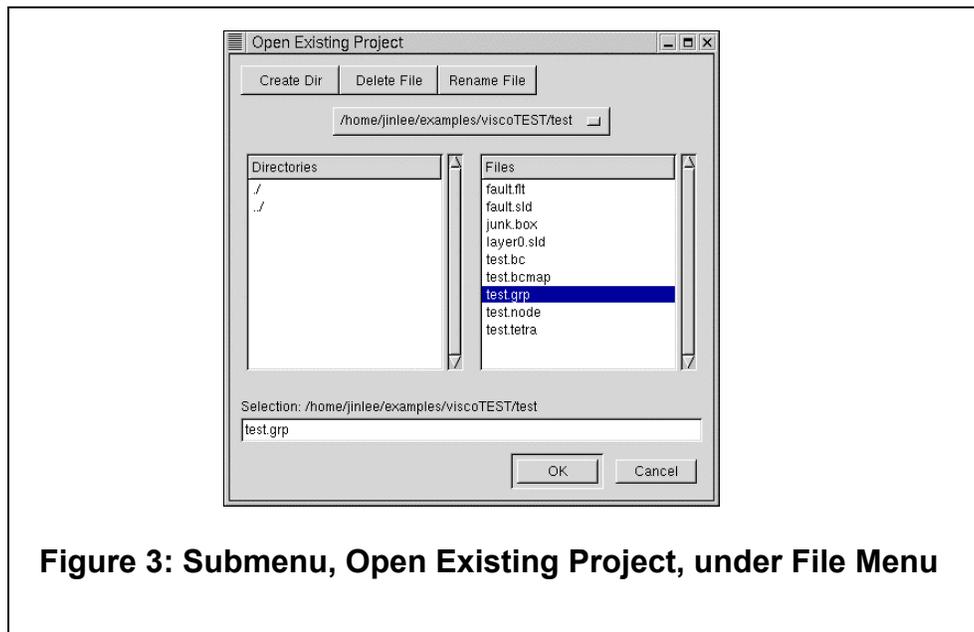
Under the File menu there are three submenus: Open Existing Project, New Project, and Exit, as shown in Figure 2. The Exit submenu is straightforward, it simply close the guiVISCO application.



Figure 2: Pull down menu under File menu.

1) Open Existing Project

The submenu, Open Existing Project, allows the user to continue working on a project that has been defined/constructed previously. Click on the Open Existing Project, a window similar to Figure 3 will pop up on the screen. It includes all the files in the current working directory, and the user will need to select the project that he/she wants to work on. Note that, the project file is always end with an extension of .grp (group file).



2) New Project

The New Project submenu is provided for the user to put together a project by including layer solids as well as faults. The submenu is shown in Figure 4, on the left pane, all the files with .sld (for layer solid) and .flt (for fault plane) extensions are included, the user can then select from this group of layer solids and faults to form a project. The selected layers and faults will then shown in the right pane, the user should make sure the project includes the right layers as well as the right faults before he/she proceeds. Additionally, the user is asked to specify the unit that is used in defining the coordinates as well as the project name. Once the user clicks OK, a file FileName.grp will then be generated and resides on the current working directory.

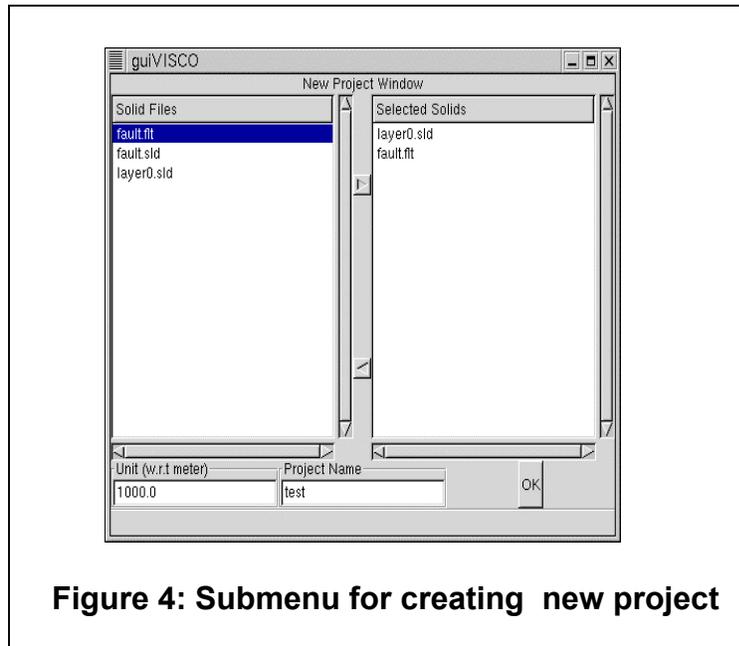


Figure 4: Submenu for creating new project

Cmnds menu

In Figure 5, we show the pull down of the Cmnds menu. As can be seen in Figure 5, there are three submenus in this pull down menu: Layer Creation, Fault Creation, and Initial Mesh.

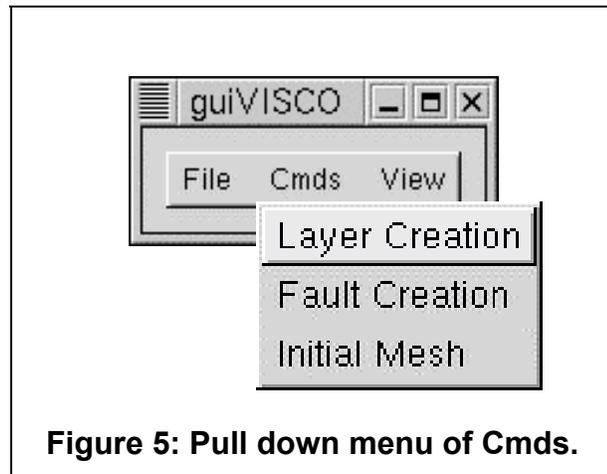


Figure 5: Pull down menu of Cmnds.

1) Layer Creation

The Layer Creation submenu is used to create layer solids. Click on the Layer Creation button, a pop-up window will show up as in Figure 6. The inset in the figure describes the meaning of each parameter in creating the layer solid. The length, width, and depth will determine the layer physical dimensions as indicated in Figure 6. The coordinate of the origin will then place the layer in the corresponding location. The layer name specification will allow the user to piece together different layers and faults to create a project in the File menu as mentioned in earlier section.

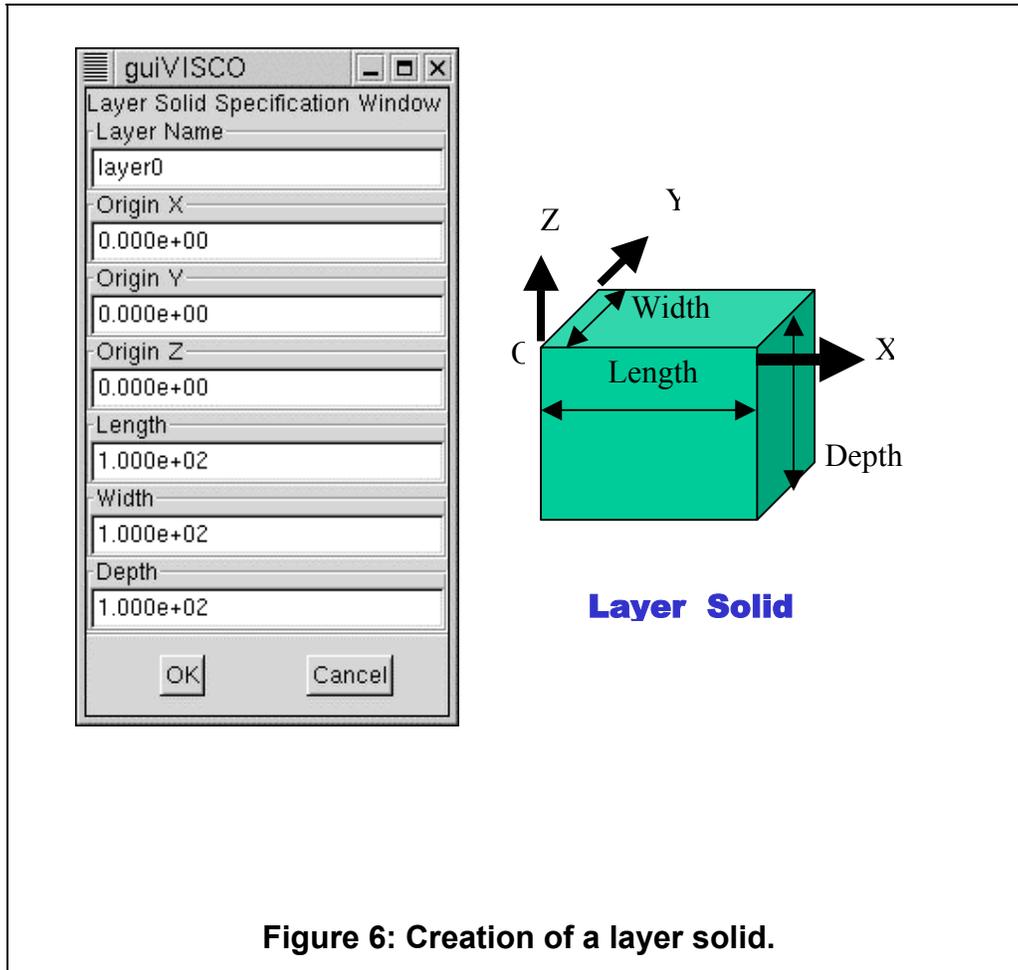


Figure 6: Creation of a layer solid.

2) Fault Creation

Similarly, the fault creation submenu is used to create faults. The pop-up window corresponds to Fault Creation is included in Figure 7.

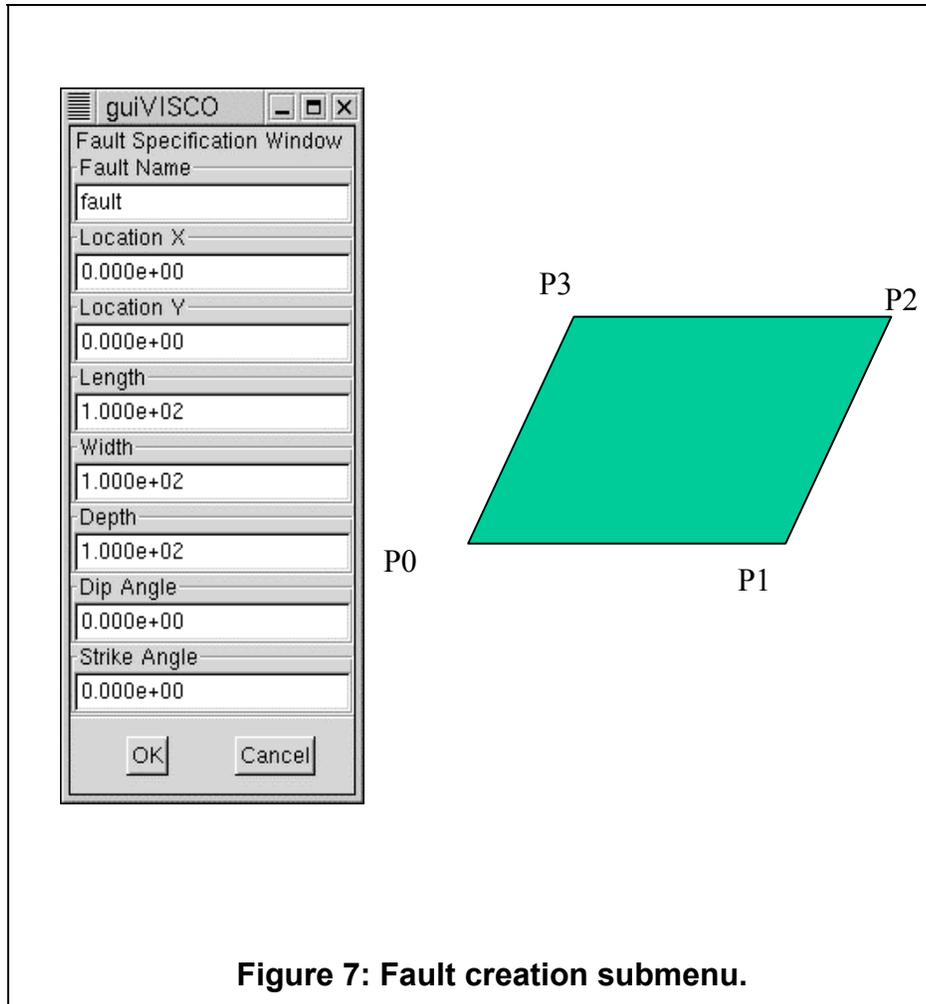


Figure 7: Fault creation submenu.

The fault creation command will generate a four-point plane; the coordinates of the four corner points are given by

$$\begin{aligned}
 P_0 & (X, Y, -D) \\
 P_1 & (X + L \sin \phi, Y + L \cos \phi, -D) \\
 P_2 & (X + L \sin \phi - W \cos \phi \cos \delta, Y + L \cos \phi + W \sin \phi \cos \delta, -D + W \sin \delta) \\
 P_3 & (X - W \cos \phi \cos \delta, Y + W \sin \phi \cos \delta, -D + W \sin \delta)
 \end{aligned} \tag{0.1}$$

where

- $X \longleftrightarrow$ Location X
- $Y \longleftrightarrow$ Location Y
- $D \longleftrightarrow$ Depth
- $L \longleftrightarrow$ Length (0.2)
- $W \longleftrightarrow$ Width
- $\delta \longleftrightarrow$ Dip Angle
- $\phi \longleftrightarrow$ Strike Angle

3) Initial Mesh

The last command in the Cmds menu is the Initial Mesh submenu. This command is used, after a project has been properly defined, to create a uniform mesh with specified largest edge length. As shown in Figure 8, there is only parameter that needs to be specified in this command, the desired mesh size in terms of the unit. Note that the unit is specified through the project creation submenu in File menu and is simply shown in the pop-up window to inform the user what unit he/she is working with in the current project and can not be changed here.

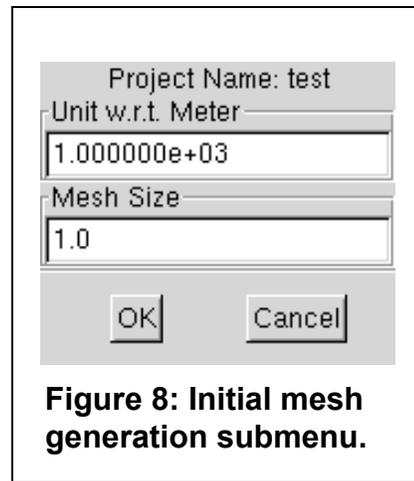


Figure 8: Initial mesh generation submenu.

View menu

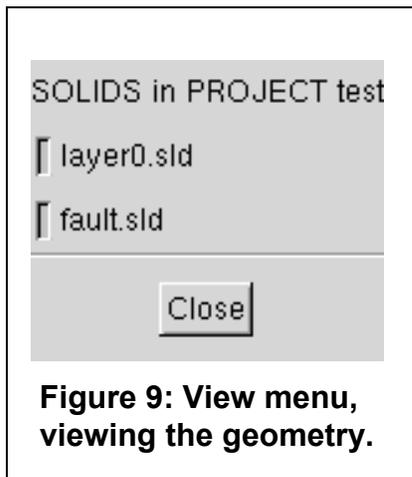


Figure 9: View menu, viewing the geometry.

Under the view menu, there are three sub-menu: view Project Geometry (solids), view Tetrahedral Mesh, and view Boundary Conditions (not implemented yet).

1) View Project Geometry

For example, click on the geometry sub-menu, in this case a project called test has been defined, a window will pop-up as the one shown in Figure 9. In this example, there are basically one layer and one fault. The geometry is shown in Figure 10.

Note that in the geometry, the fault is completely inside the layer, which is the normal operation. However, the current guiVISCO code also allows for the fault

to touch the layer boundaries as shown in Figure 11. However, it does NOT allow for a fault to cut through different layers. If it happens, the user will have to input it as multiple faults, as was done in Figure 11.

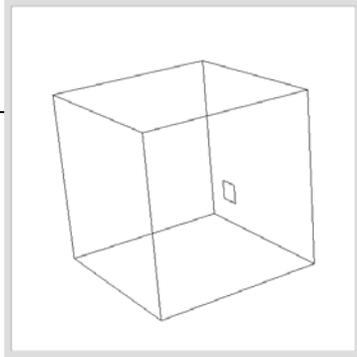


Figure 10: sample geometry, one layer and one fault. Note that the fault is completely inside the layer.

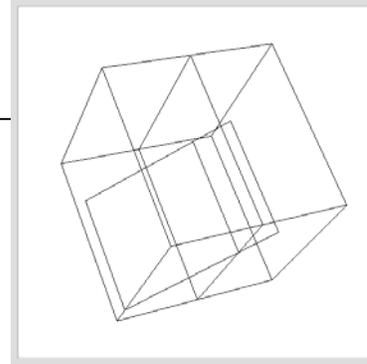


Figure 11: A sample geometry, which has two layers and two faults.

2) View Tetrahedral Mesh

The Tetrahedral Mesh under view menu is provided to view the generated tetrahedral mesh. As the input, there should be two files, FileName.tetra and FileName.node, where the FileName should be the same as the project name. A sample mesh-viewing example is included in Figure 12.

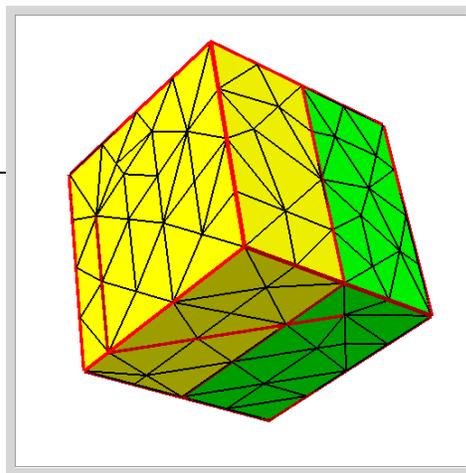


Figure 12: A sample tetrahedral mesh for two layers and two faults as in Figure 11.

AKIRA_v1 for Initial Mesh Generation

The input files that are needed in order to AKIRA_v1 are: FileName.grp, and the layer solids files as well as fault files that are included in the .grp file. For example, a project file say test.grp has the following contents:

```
1000.0 2
layer0.sld
fault0.flr
```

The first number, 1000.0, in test.grp means that the unit is 1 km, the second number indicates the number of solids included in this project, in this case 2. The first solid is described in full in layer0.sld file and the second solid (in this case a fault plane) is included in fault0.flr. Both the .sld and .flr have exactly the same file format. Therefore, we shall only discuss the file format of .sld file. For example, the layer0.sld could have the description as:

```
8
-1 -1 -1
-1 1 -1
1 1 -1
1 -1 -1
-1 -1 1
-1 1 1
1 1 1
1 -1 1
6
1 1
4 1 2 3 4
1 1
4 1 2 6 5
1 1
4 2 3 7 6
1 1
4 3 4 8 7
1 1
4 1 4 8 5
1 1
4 5 6 7 8
```

Now, an explanation of the file format is in order here. The first number in layer0.sld, 8, means that there are 8 vertex points forming this solid. Subsequently, there are 8 coordinates being given following the number 8, and they are sequentially numbered as (x1, y1, z1), (x2, y2, z2), ..., (x8, y8, z8). After the coordinates of the vertex points, an integer number, 6, specifies the number of facets which forms this solid. Subsequently, there would be 6 data sets

describing these 6 facets. The facet description is a very simple format, the first integer assign the boundary conditions (for viscosity application, it is dummy), the second integer indicates the number of loops in this facet, a number 1 means simply connected loop. The next integer, in the current case, 4, specifies the number of vertex points that form this facet. Finally, there would be equal number of indexes, which correspond to the vertex ids. For example, the first facet in layer0.sld is described as

```
1 1
4 1 2 3 4
```

It means the boundary condition of this facet is assigned $bc_id = 1$, and there is one loop for this facet, and this loop is formed by 4 points, and they are point 1, 2, 3, and 4.

Once the .grp file has been created and the corresponding solid files are in place, to run AKIRA_v1 simply type

```
AKIRA_v1 FileName 0
```

This will then generate a minimum mesh for the project FileName.grp. Two output files are created:

```
FileName.tetra and FileName.node.
```

meshRefine For Uniform Mesh Refinements

The program meshRefine is employed for two purposes:

- To uniform refine the mesh until the largest edge length is smaller than the given desired length; and,
- To refine the mesh until the maximum aspect ratio of the tetrahedral is smaller than the given tolerance.

Assuming an initial mesh, FileName.tetra and FileName.node, are given, you can invoke meshRefine by typing

```
meshRefine FileName 0 edge_length
```

The FileName is the project name, and the integer 0 tells meshRefine program that you want to perform uniform refinement with maximum edge length edge_length. To achieve the purpose of getting the maximum edge length smaller than the desired edge_length, this command needs to be repeated many times until no more new points are added in the process. Alternatively, control program CROM will automatically call meshRefine program many times until the

maximum edge length is smaller than `edge_length`. We will discuss CROM in more detail later. The `meshRefine` program can be used to improve aspect ratio of the tetrahedral mesh too. To do so, simply type in

```
meshRefine FileName 1 max_aspect_ratio
```

The integer 1 tells `meshRefine` that you want to perform aspect ratio refinement. The desired maximum aspect ratio is input as `max_aspect_ratio`. Again, this process in general needs to be called many times before the aspect ratio of the mesh is under control. The control program CROM can be used to automate the entire process.

APOLLO For Adaptive Mesh Refinements

APOLLO is the program used to adaptively refine the current mesh, `FileName.tetra` and `FileName.node`, according to error indication stored in `FileName.index`. The `FileName.index` stores N_{tetra} error index number, and the sequence of these error indicators need to in-sync with the sequence in `FileName.tetra`. The larger the error index for a particular tetrahedron, the more likely the mesh will be refined around that tetrahedron. With `FileName.tetra`, `FileName.node`, and `FileName.index` present, you can simply invoke APOLLO by

```
APOLLO FileName 0  
APOLLO FileName 1
```

To do one iteration of adaptive mesh refinement, you need to invoke APOLLO twice, the first one with option 0 and the second one with option 1. After that `FileName.tetra` and `FileName.node` will be overwritten with the new mesh. Again, this process can be automated by using CROM.

CROM: A Top Control Program to Bind Together AKIRA_v1, meshRefine, and APOLLO

CROM is nothing but a very simple control program to automate mesh generation process. CROM can be employed to uniformly refine and perform aspect ratio refine at the same time. However, before you invoke CROM, make sure `FileName.tetra` and `FileName.node` exist in the current working directory. To run CROM simply type

```
CROM FileName max_edge_length 1 max_aspect_ratio
```

This will accomplish the uniform mesh refinement until the maximum edge length is smaller than `max_edge_length` and conduct the aspect ratio refinement until the maximum aspect ratio of the tetrahedral mesh is smaller than `max_aspect_ratio`.

To invoke CROM for adaptive mesh refinement, again, make sure FileName.tetra, FileName.node and FileName.index exist. Furthermore, make certain the error indexes included in FileName.index is in correct order as in the FileName.tetra. To perform adaptive mesh refinement, do

CROM FileName dummy_number

A new mesh will be replace the old one in FileName.tetra and FileName.node.