GILDA Grid Demonstrator - Overview

The GILDA Grid Demonstrator is a customized version of the full GENIUS web portal, jointly developed by INFN and NICE, from where everybody can submit a pre-defined set of applications to the GILDA Testbed. It's a powerful dissemination tool available 24x7 to everybody.
This guide is intended for users of the *GILDA Grid Demonstrator*. Within these pages, users will find an introduction to all the functions provided by the *GILDA Grid Demonstrator* and descriptions of how to use them. Users will also find an ensemble of examples to illustrate these functions. Finally, in the Appendix A, users will find details of all the jobs that they can submit including examples of with the associated output.

For the GENIUS portal, each user can directly access the *GILDA Grid Demonstrator* through the "secure" URL https://grid-demo.ct.infn.it. The GILDA Grid Demonstrator' home page is shown in figure 1.

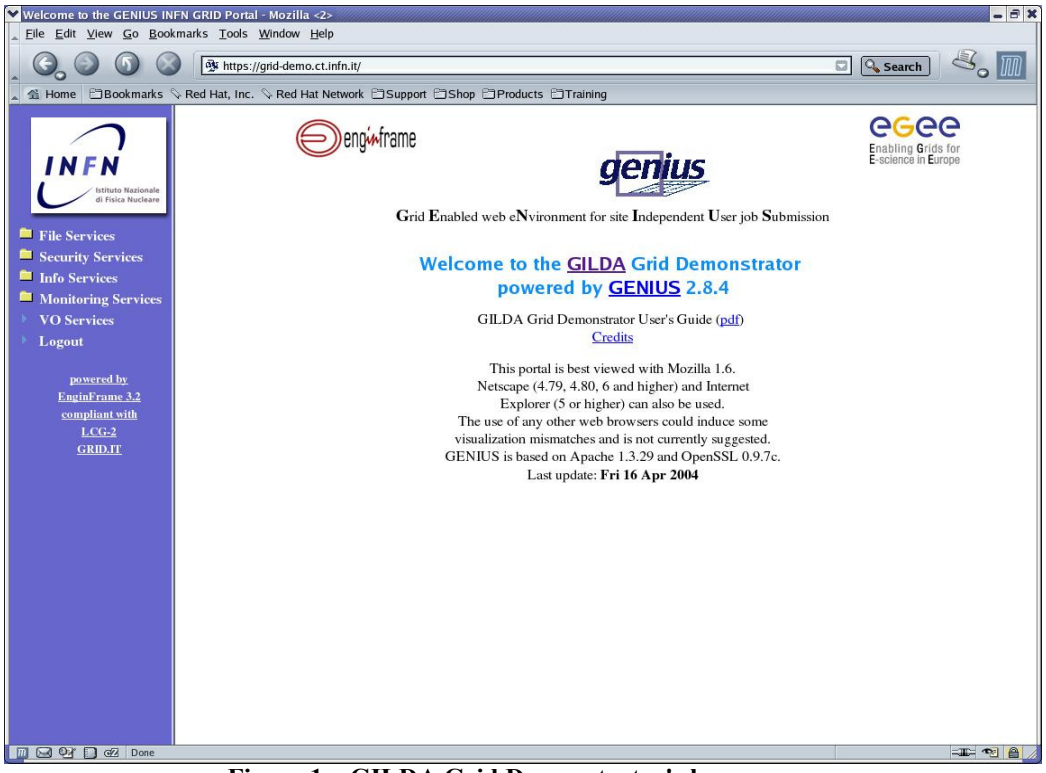


Figure 1 – GILDA Grid Demonstrator's home page.

As shown, in figure 1, portal layout comprises two frames. In the left frame services are grouped, and users can browse them, choosing the desired one; its details are then shown in the central frame, from which users can manage a service.

1 GILDA Grid Demonstrator – Functions

The goal of this section is to describe in detail the guidelines for accessing the functions provided by the *GILDA Grid Demonstrator*. On choosing "VO Services" a user sees the screen shown in figure 2, with a pre-existing username assigned. Before using *GILDA Grid Demonstrator,* users have to authenticate themselves to access grid functions (see fig.2). This authentication will be valid for one day.

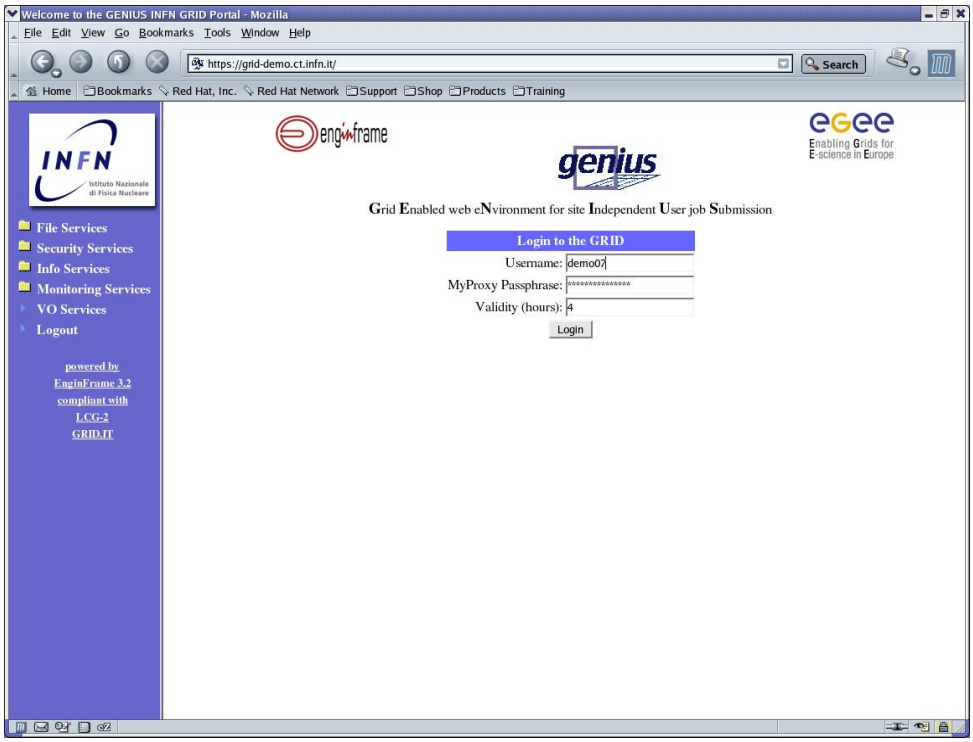


Figure 2 – GRID authentication.

Once users have authenticated themselves, by using the already entered passphrase and username, they can access all *GILDA Grid Demonstrator* services, from the left frame (see fig.2). In particular users can choose one of the following services:

- File Services
- Security Services
- Info Services
- Monitoring Services
- VO Services
- Logout

2 File Services

These services are intended to let the user interact with their files stored on the remote UI machine.

2.1 View a File

With this service, a user can view, but not modify, one of his/her files stored in the UI machine. In order to choose the file, the user has two possibilities:

- Provide the full path of the file in the dialog box
- Click on the Select button, to list all his/her file (including subdirectories) on the UI machine, and select the desired one.

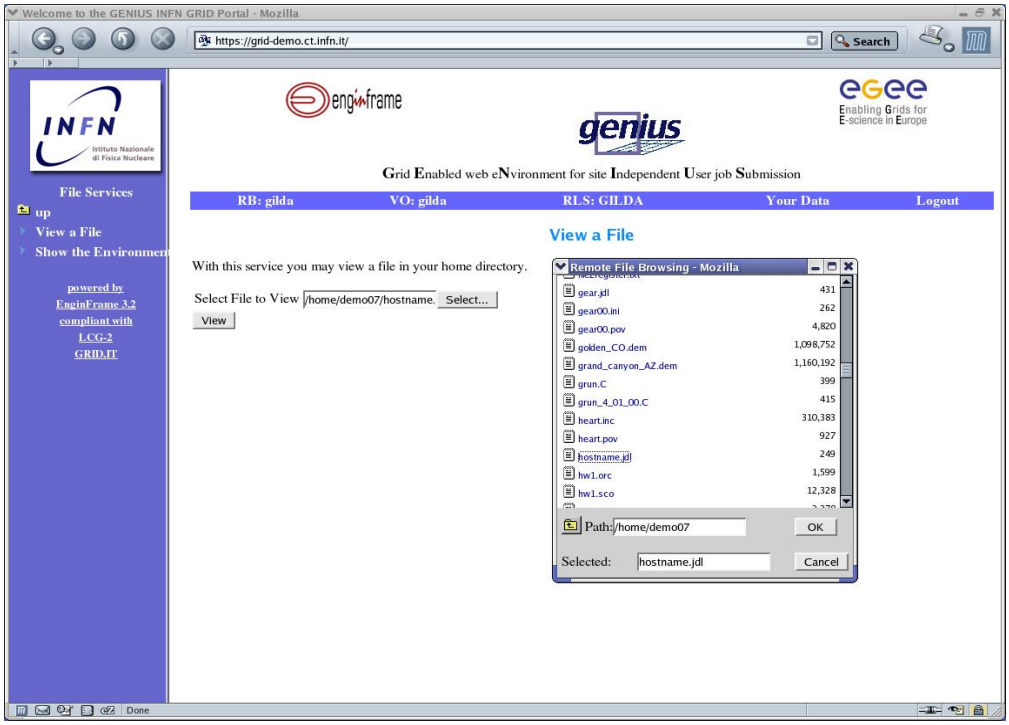


Figure 2.1 – View File.

After selecting the file, the user can click on the *View* button, and the file contents will be shown on the screen.

2.2 Show Environment

With this service users can inspect their environment variables on the UI machine.

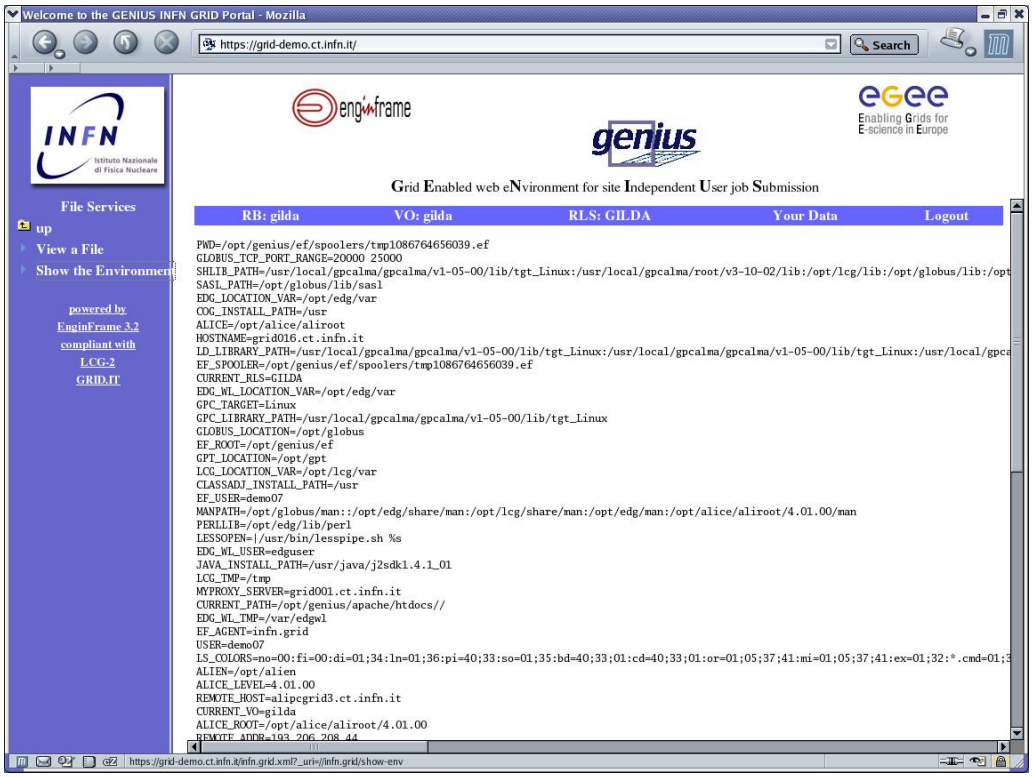


Figure 2.2 – Show the Environment.

3 Security Services

These services are meant to let users manage the tokens needed to run jobs on the grid. To use this service, a user needs to be logged on the Grid.

3.1 Info on Proxy

With this service, users can get information about their grid proxy and, the temporary token needed to run jobs on the grid. The subject of the certificate, the issuer, the type, strength and the remaining period of validity are shown.

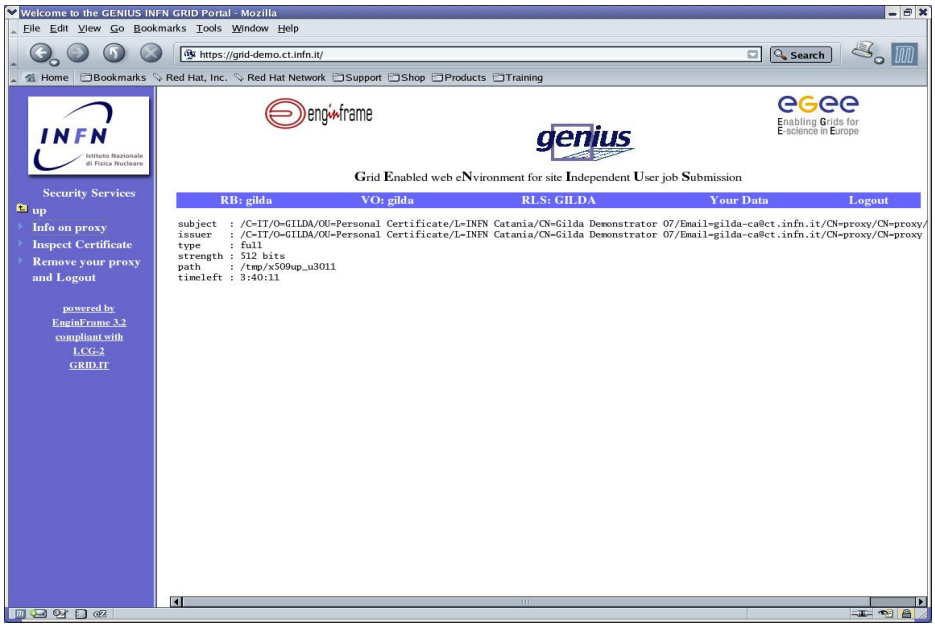


Figure 3.1 – Info on Proxy.

3.2 Inspect Certificate

With this service, users can get inspect and verify their certificates. The subject of the certificate, the issuer, the period of validity, and the verify status are shown.

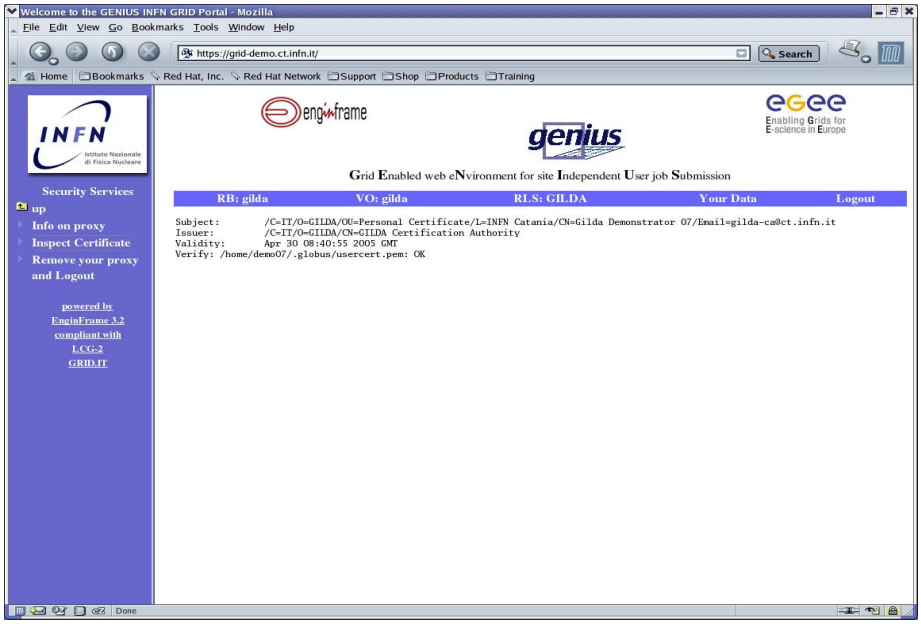


Figure 3.2 – Inspect certificate.

3.3 Remove your proxy and Logout

When users request this service, they are automatically logged out of the *GILDA Grid Demonstrator*, while their temporary proxy and myProxy are destroyed. However, it can always be renewed next time that user try to use *GILDA Grid Demonstrator*.

4 Info Services

These services are meant to let the user browse the testbed which he/she can use, depending on the VO to wich they belong.

4.1 Check testbed

This service allows user to retrieve general information about the sites accessible by his/her VO; for any CE of the site, is shown the number of CPUs, number of jobs running and waiting, the application that can be run, and the time limit of the queue. For any SE of the site the amount of free space is shown.

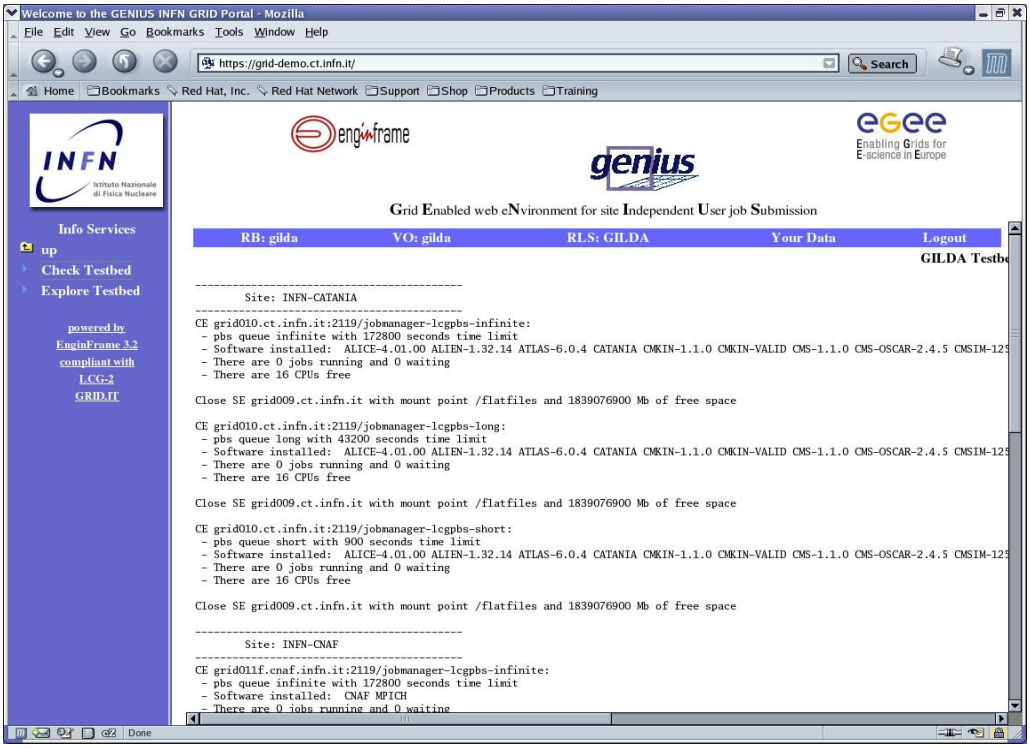


Figure 4.1 – Check Testbed.

4.2 Explore a Testbed

This service allows a user to browse an LDAP server containing detailed information for any resource and service on the testbeds accessible for users, depending on the VO they are

logged into. These resources are hierarchically organized, with a tree structure; the root of the tree is the information index machine which VO is referring to.

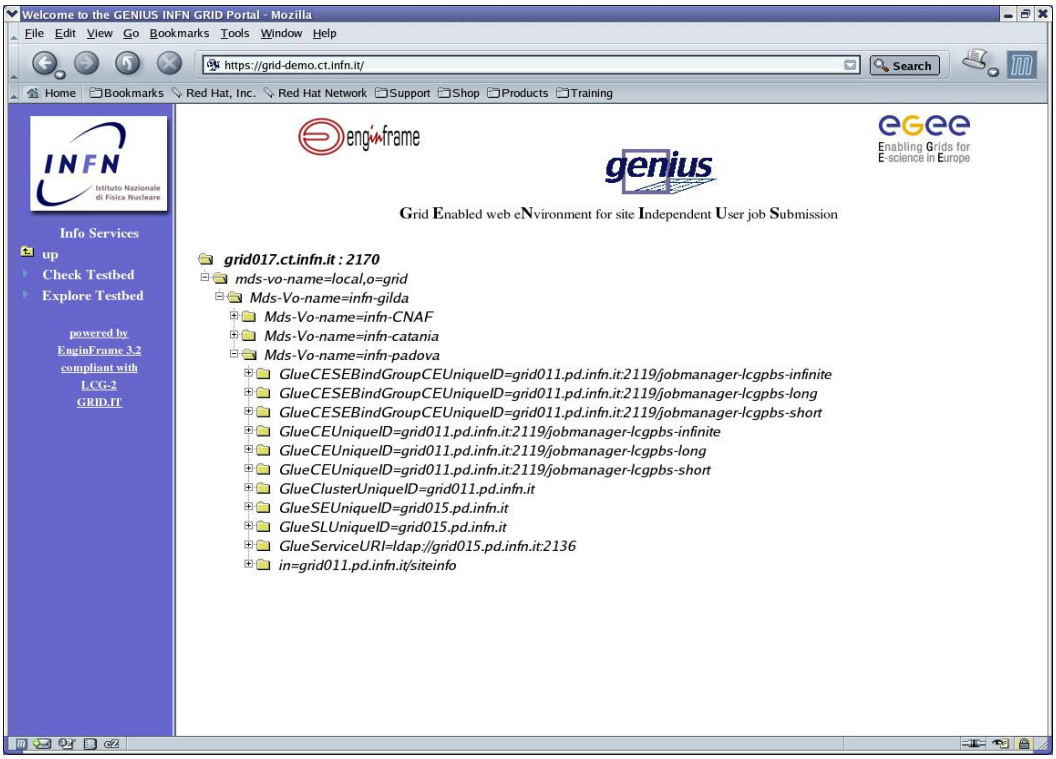


Figure 4.2 – Explore Testbed(1).

Each node of the tree has a label reflecting element name in GLUE Schema, and when this label is hit, a lot of useful information about the element is shown. If the node is not a leaf it could be expanded, showing a subtree which represents all elements "logically" composing the node.

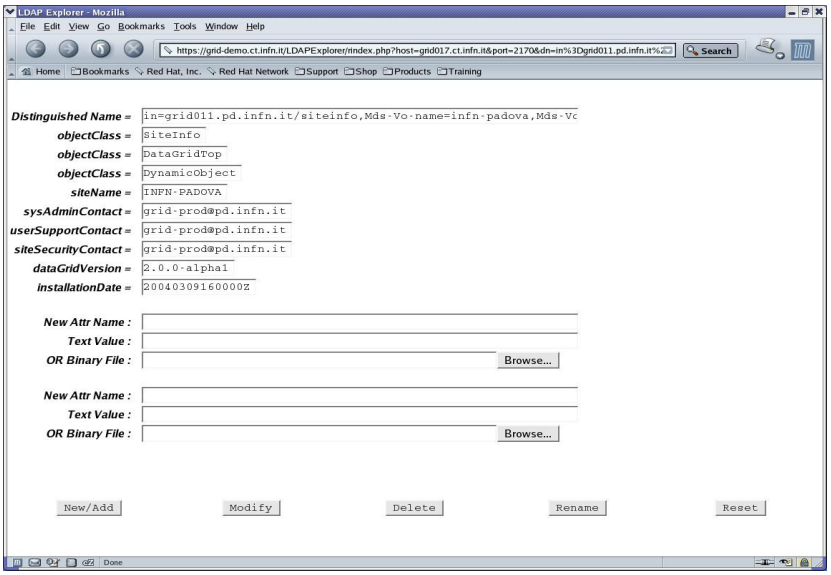


Figure 4.3 – Explore Testbed(2).

5 Monitoring Services

These services allow users to monitor and measure significant grid resources in order to analyse usage, behaviour and performance of the grid, and eventually to detect faults.

Figure 4.4 – GridIce(1).

Figure 4.5 – GridIce(2).

6 VO Services

VO Services give users possibilities of managing HadronTherapy Services, Video on Demand, Other Job Services and Data Services. In the next sub-sections we will describe each of this services in detail.

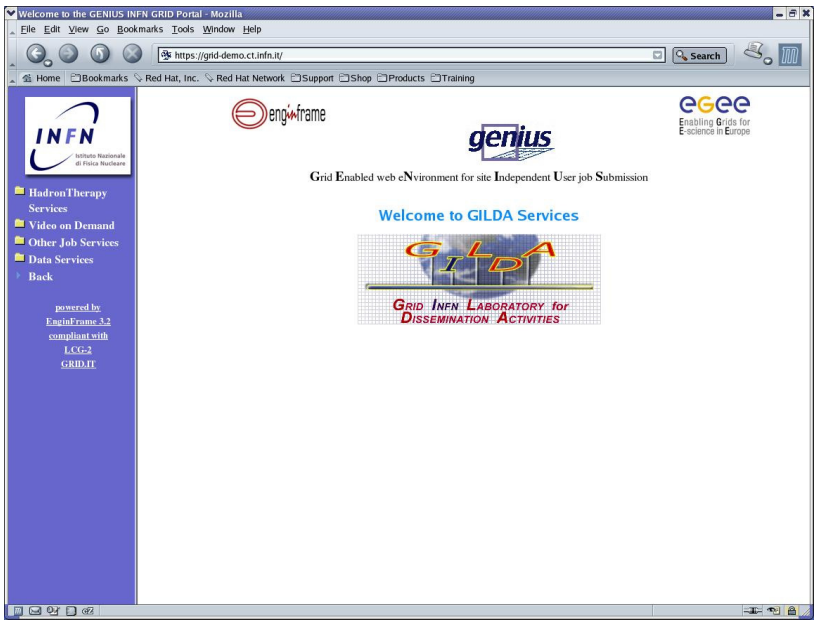


Figure 6.1 – VO Services.

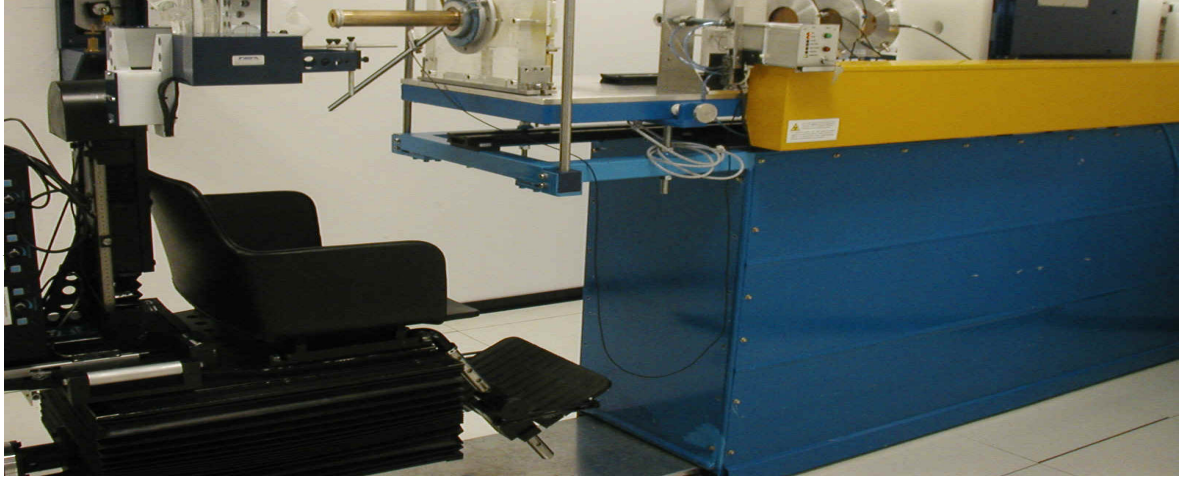


Fig. 6.2 -CATANA proton therapy center

Developing a personalized hadron therapy beam line requires a long and accurate experimental work, and for this reason the simulation tool has been created. GEANT4 simulation, on which is based *hadronTherapy*, are implemented via MonteCarlo method : this lead a great lack of time, in order to provide an accurate simulation, because a very large number of basic events have to been generated.
To reduce computational times, *hadronTheraphy* has been ported on the GILDA Dissemination Grid, taking advantages of distributed architecture.
Instead of launching a single job (on a single machine) with M events (say *O(f(M))* its execution time), we launch N jobs, each with M/N events. Because execution is on different Worker Nodes, we can consider it parallel, and so time is O*(f(M/N))*.

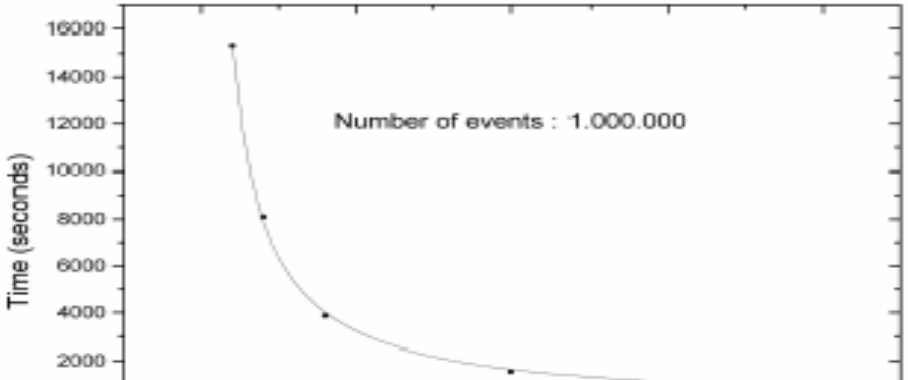


Fig. 6.3 - Cpu Time vs. Cpu Number

An accurate simulation (i.e. 1 million of protons) requires 4.16 hours on a single CPU (Pentium 4 @ 2.4 GHz), but distributing computation on many CPU decreases times exponentially : the same track number, elaborated with 20 machines, needs 15 minutes, and increasing CPU number up to 85, the track of 1 million of protons would need only 3 minutes…
With these execution times a routinary use of this Monte Carlo code is reasonable, if a simple interface is provided. With this issue, *hadronTherapy* has been integrated in GENIUS.

Through a form users can specify parameters for the production (i.e. group of jobs) that they want to execute. Once submitted the production, users can inspect its status from **hadronTherapy Job Queue,** where they will find an item named as the production. The page that appears is refreshed each 30 seconds, informing users on the production execution status. When it is completed, a link to the spooler containing a copy of the production output directory appears. So, following this link, user can inspect directly output from the browser : there is a directory for each job in the group, and a file containing the graphic of simulation, obtained with CERN tool Root. The directory in the spooler have a limited lifetime, while the original, in the users home, will stay undefinitely.

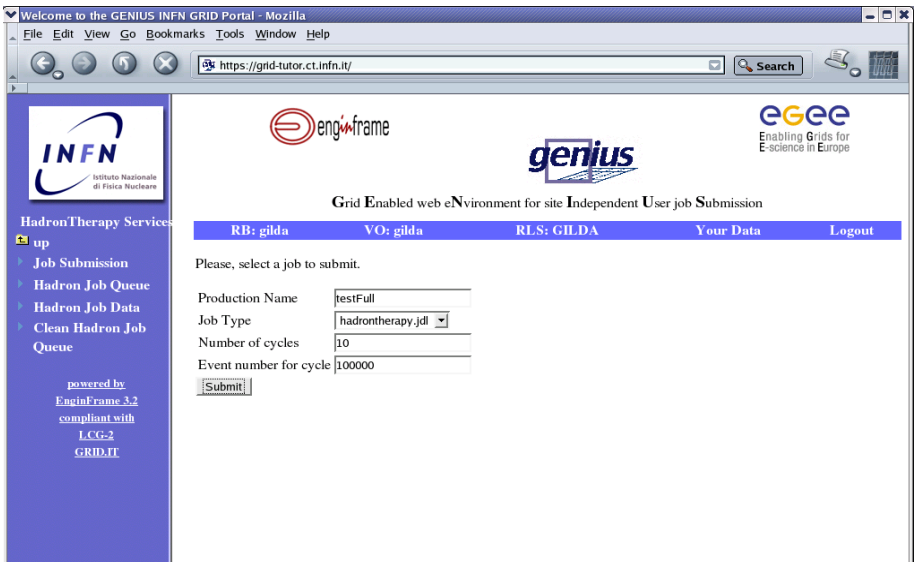
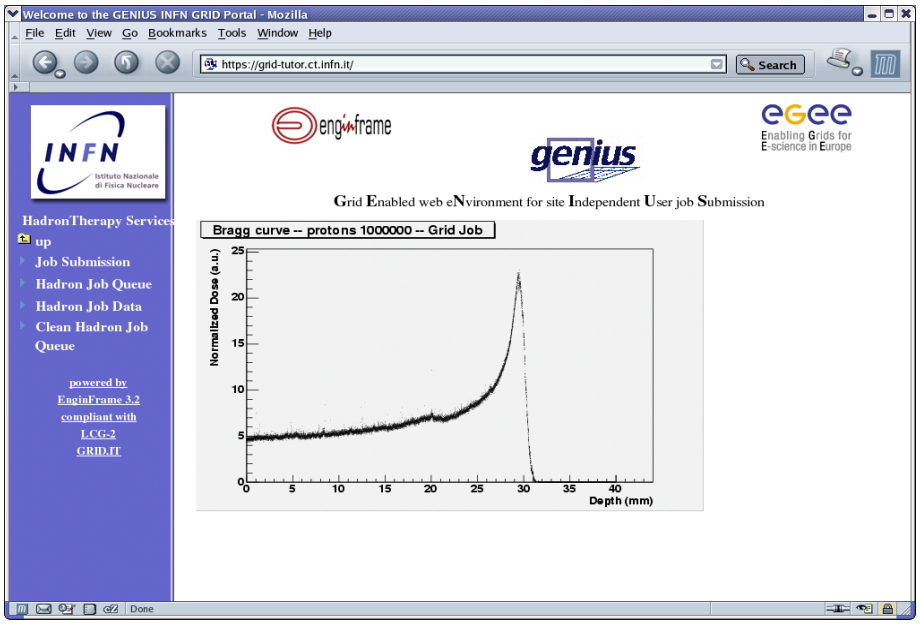


Fig. 6.4 - GENIUS Interface for specification of production parameter



6.2 Video on Demand

In this sub-section we will describe the steps to request a video streaming. To do this the user has to click on Request Video folder and select the video that he/she wants to see. The playlist of all the videos available is collected in a list as shown in the fig. 6.6.

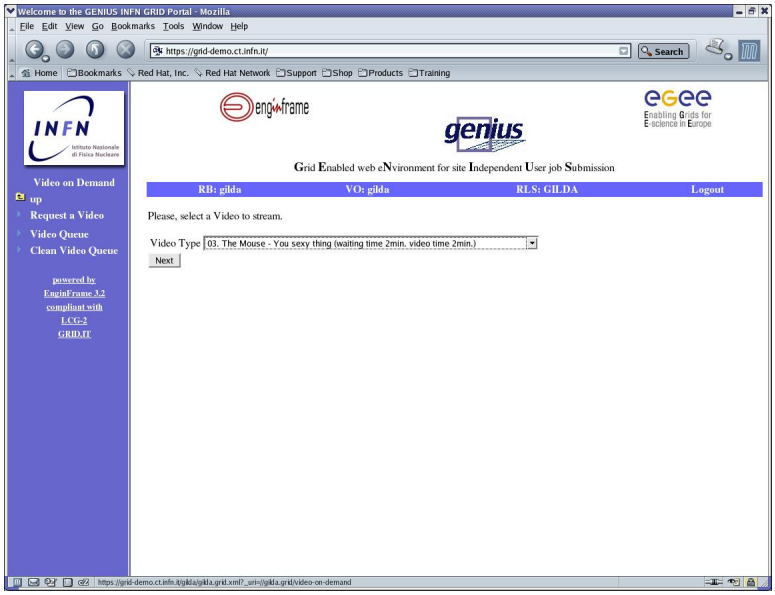
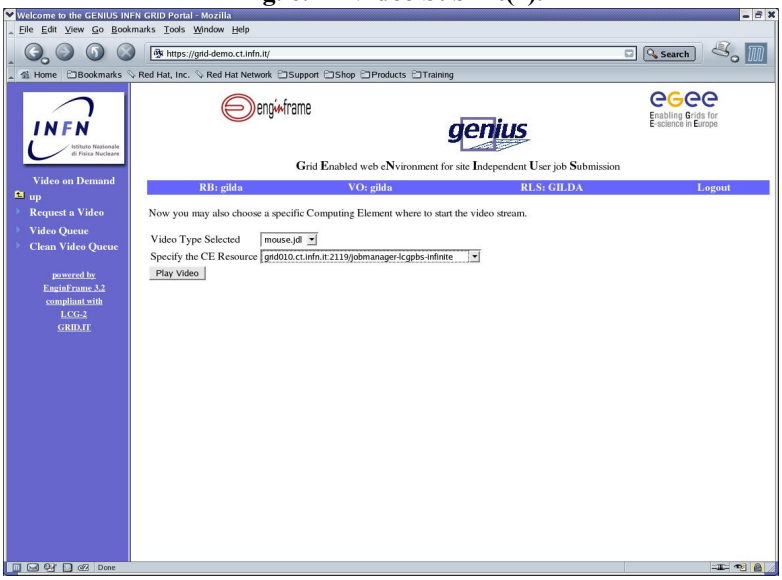


Fig. 6.6 – Video Submit.

After selected the video the user can choose the Computing Elements where the video will be executed.

Fig. 6.7 – Video Submit(2).



When the button Play Video is hit, if everything is OK, the video request will be submitted to the Grid and the user is notified about the submission status and gets the job identifier. If an error occurs the user is notified and an email containing the error log is automatically sent by *GILDA Grid Demonstrator* to the relevant site managers.

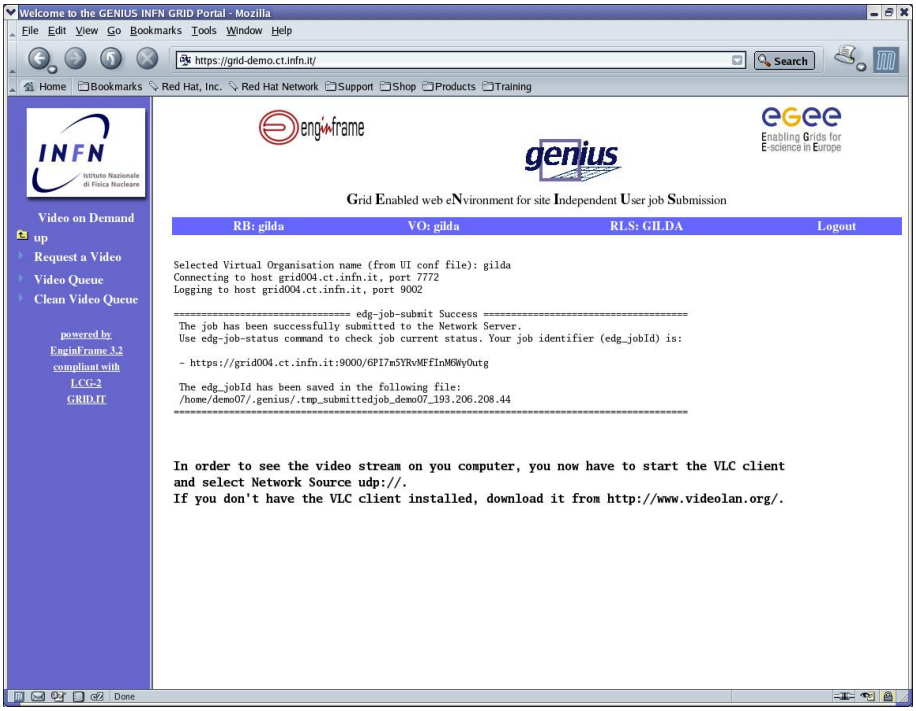


Fig. 6.8 – Video Submit(3).

At this point in order to see the video stream on own computer the user has to open the *VLC Client* and configure the Network Stream Panel (CTRL+N) and select UDP/RTP with port 1234 as indicated in the fig. 6.9. More information about the VideoLAN streaming solution and last release of VLC Client can be easily downloaded form http://www.videolan.org/

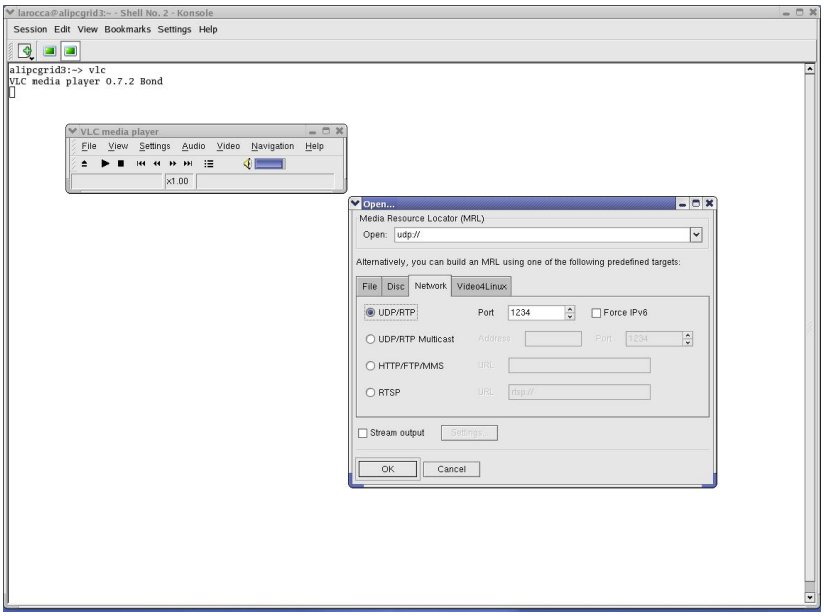


Fig. 6.9 – VLC Configure.

Now the user is ready to receive the video streaming selected in the previous sections.

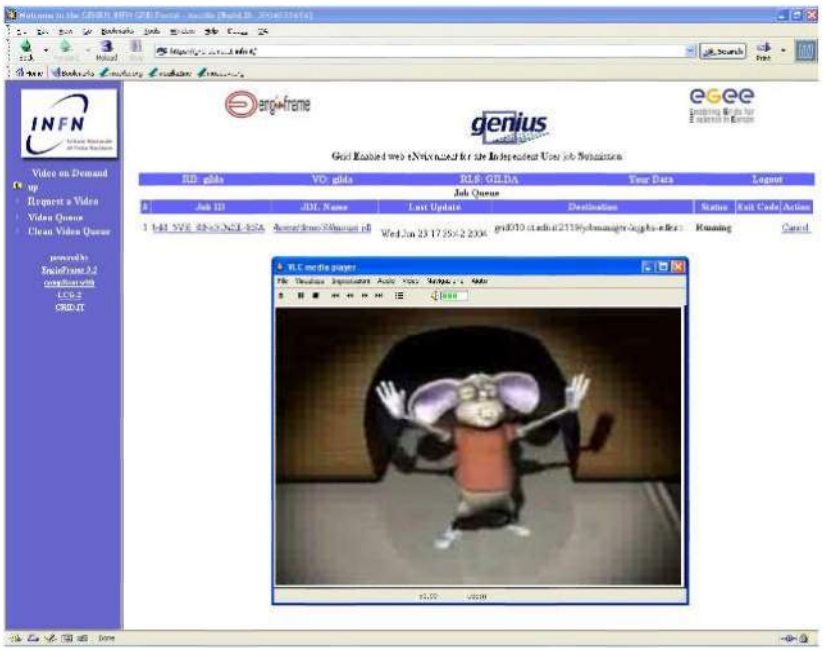


Fig. 6.10 – The Mouse.mpg video streaming.

With the Video Queue service users can query the status of the videos that he/she has submitted to the grid, while with Clean Video Queue service user can cleans his/her queue.

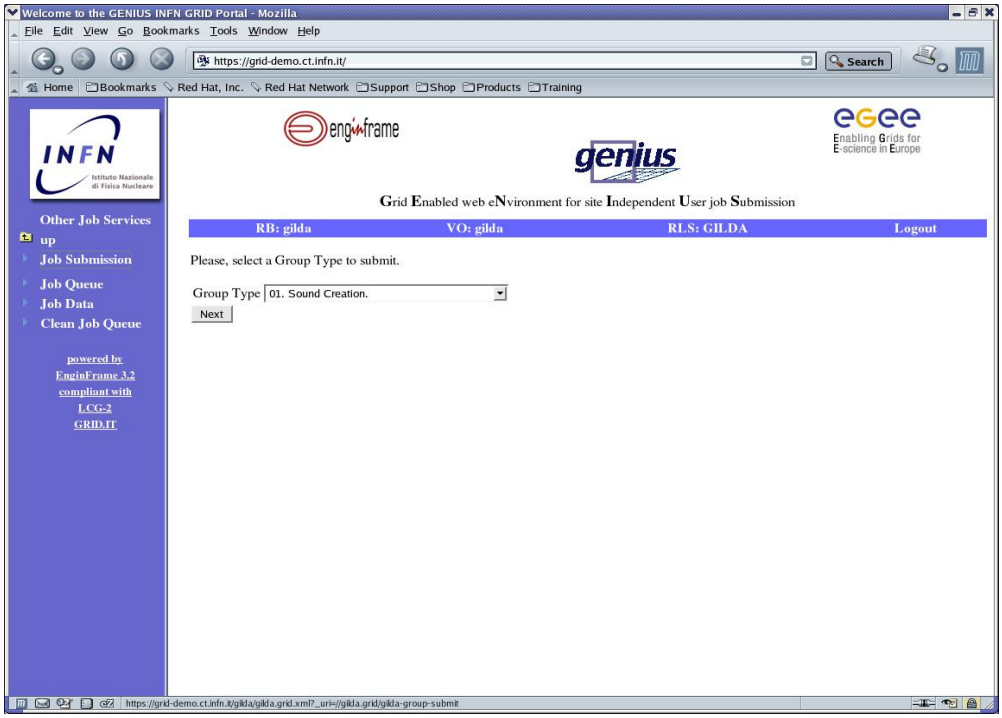
6.3 Other Job Services

Job Services provide commands to submit jobs, to cancels jobs, check their status, and retrieve any output obtained during their execution.

6.3.1 Job Submission

In this sub-section we will submit a job. To do this click on Job Submission folder and you will see something like this:

Fig. 6.11 – Job Submission(1).



Job submission is done in three steps.

- *Step 1*

  The user has to choose the Group type of the job he/she wants to submit. All the Group type available are collected in a list as shown in the fig. 6.11. Selecting a Group type file and clicking on the button "Next" will complete the first step.

- *Step 2*

  The user has to choose the JDL file which describes the job he/she wants to submit. All the JDL files available are collected in a list as shown in the fig. 6.12. Selecting a JDL file and clicking on the button "Next" will complete the second step.

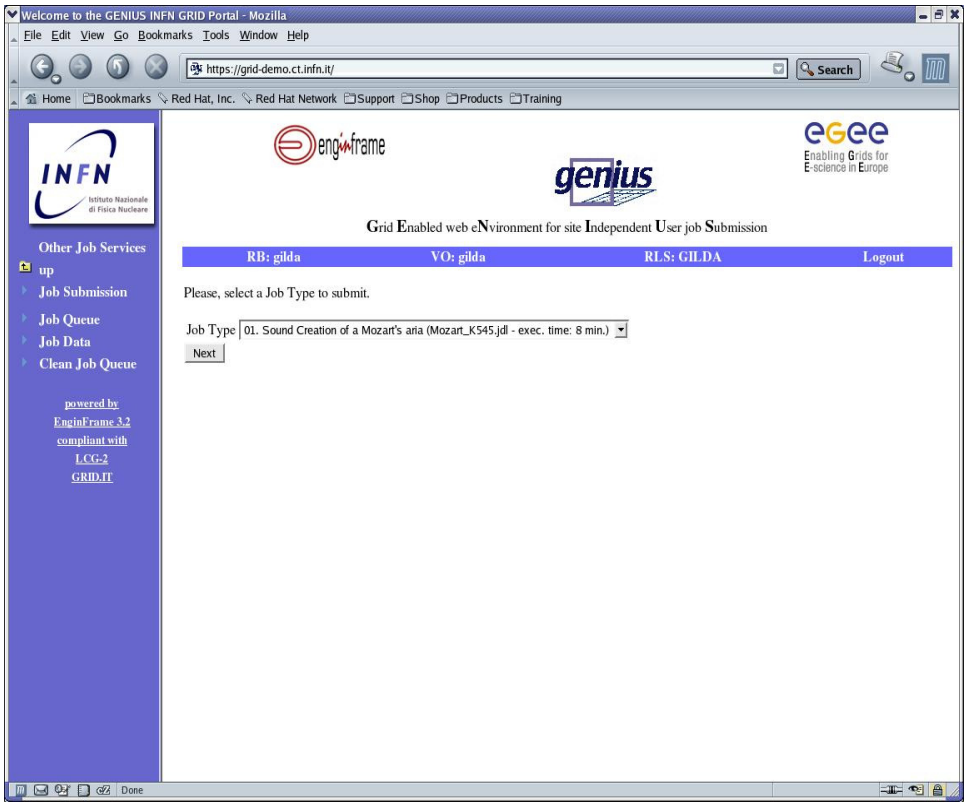


Fig. 6.12 – Job Submission(2).

- *Step 3*

  The JDL file which describes the job selected in the previous step, is shown on the screen (it cannot be changed) and a menu gives the ability to choose the Computing Elements where the job will be executed.

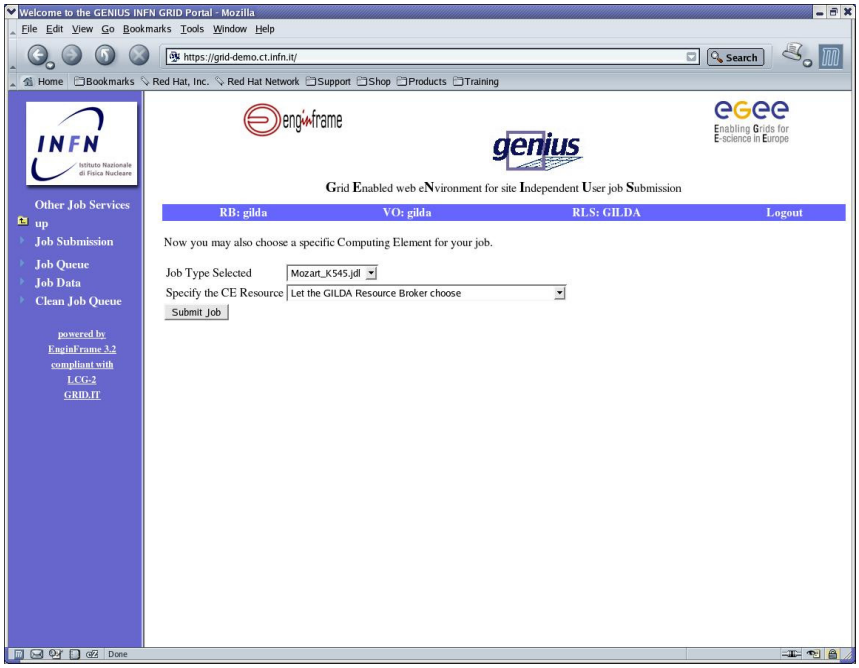


Fig. 6.13 – Job Submission(3).

When the button Submit job is hit, if everything is OK, the job is submitted to the Grid and the user is notified about the submission status and gets the job identifier. If an error occurs the user is notified and an email containing the error log is automatically sent by *GILDA Grid Demonstrator* to the relevant site managers.

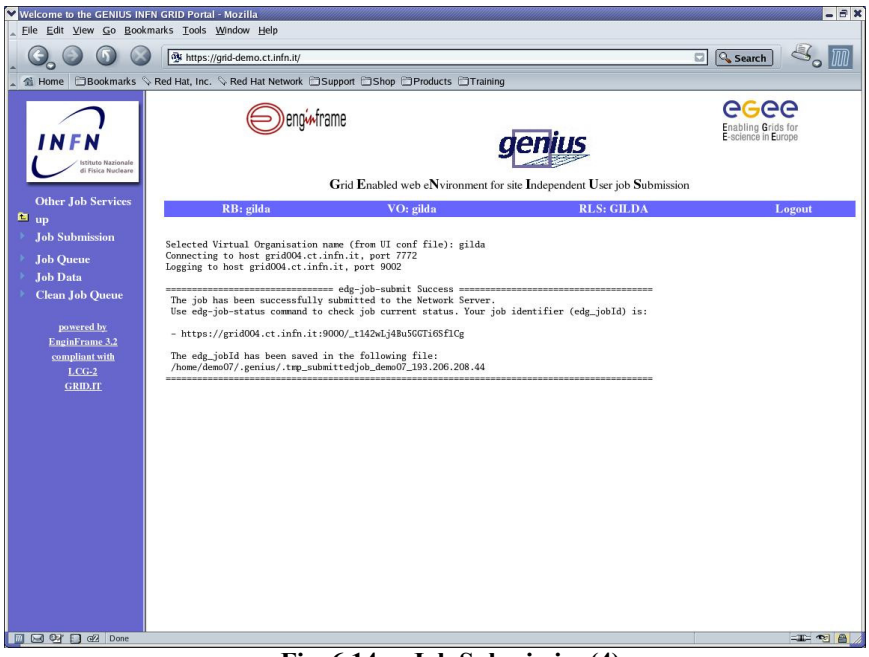


Fig. 6.14 – Job Submission(4).

6.3.2 Job Queue

If the job has been successfully submitted to the Network Server, a job identifier *(jobId)* is returned, which can be used to determinate the status of the job and eventually retrieve its output. With the Job Queue service users can query the status of the job they have submitted to the grid. So when the link Job Queue is clicked, users get the list of all the job they have submitted. Each row of this list contains the following information:

- job identifier : clicking on it we retrieve every step that the job has trnasited since is life-cycle get started
- JDL filename : clicking on it user can read jdl file content
- the time of job status last update (Ready, Running, Scheduled...)
- the Computing Element where it is actually running on : note that this CE may change during job life-cycle
- the present status of the job
- actions possible on the job: as long as the job has not reached the status “Ready” clicking on “Cancel” will abort it.

Job queue shows user jobs in order, starting from the most recent one. The frame is refreshed every five minutes. However, a user can always refresh the frame manually, by right-clicking with the mouse on the frame and selecting “Reload frame” in the browser pop-up menu.
As said above, clicking on the Job ID retrieves job logging information. This information is stored permanently by the Logging and Bookkeeping service and can also be retrieved after the job has terminated its life-cycle.

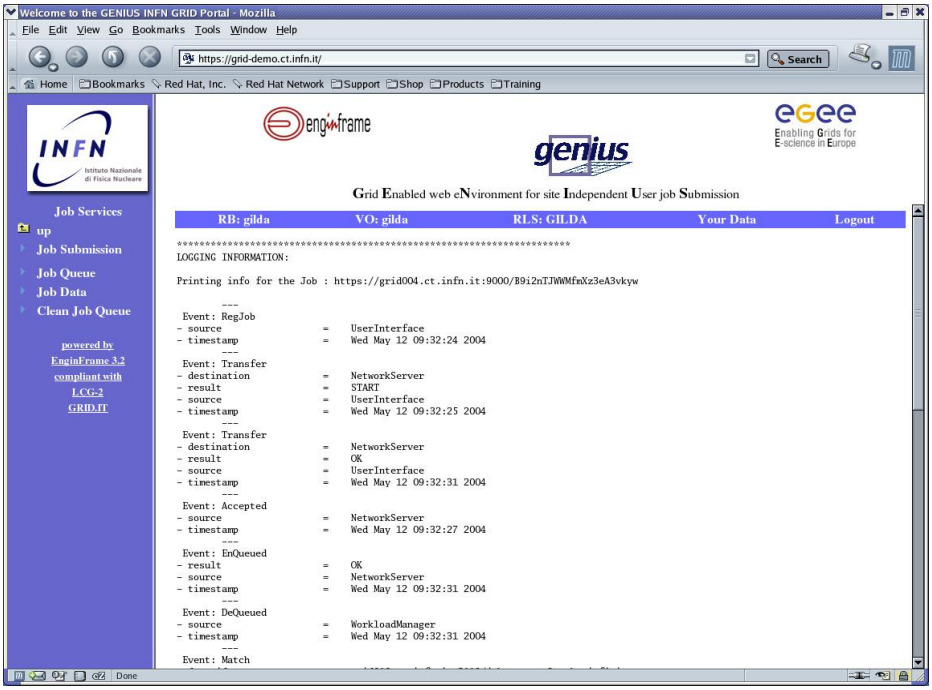


Fig. 6.15 – Logging Information for the job.

In the same way, by clicking on the JDL filename, user can see the JDL file of the job

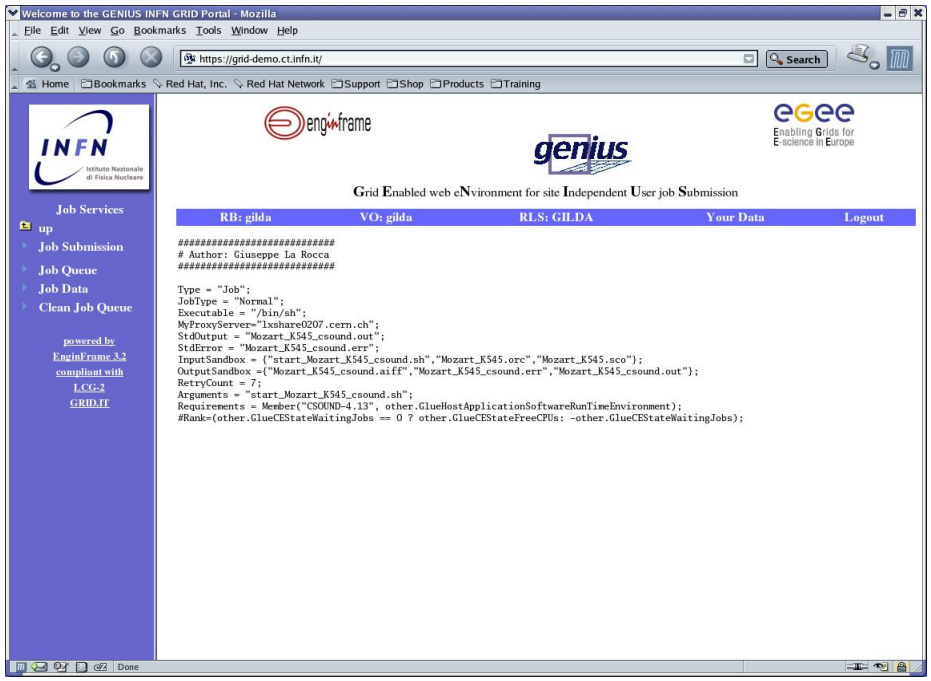


Fig. 6.16 – JDL file of the submitted job.

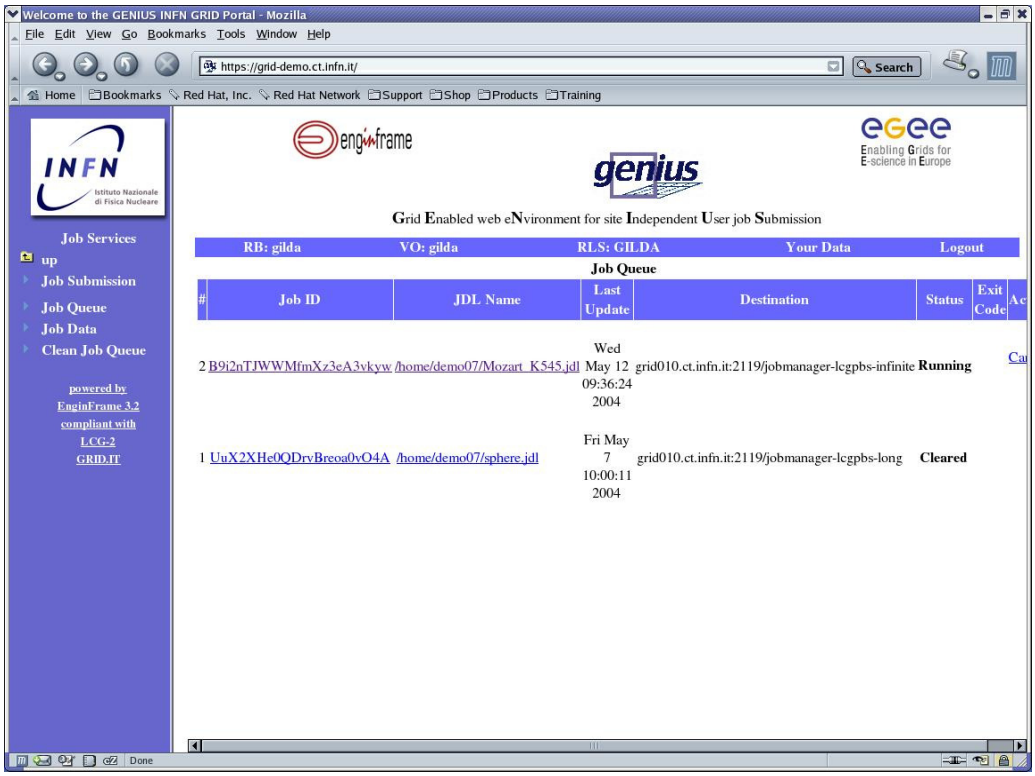


Fig. 6.17 – Job Running.

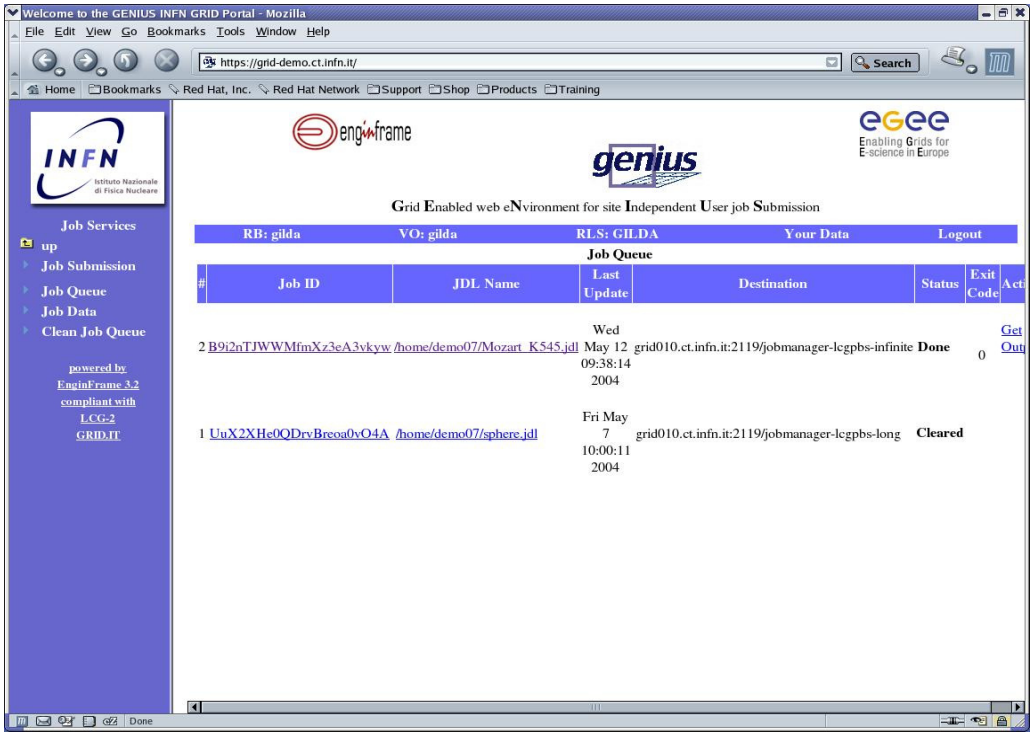


Fig. 6.18 – Job Done.

When the action button Get Output is hit, the output files are copied from the temporary

spooler system of *GILDA Grid Demonstrator* to the user's home directory on the remote UI machine, in the sub-directory *<user_name>_<jobID>* (as example, if the job identifier is *abc1215031* the output files are stored in the UI at */home/<username>/<username>_abc1215031/* .
Files can be inspected through the web browser (by clicking on the file name) or copied onto the user's local machine (by clicking on *"Save Link Target As..."* from the pop-up menu that appears right-clicking on the file name).

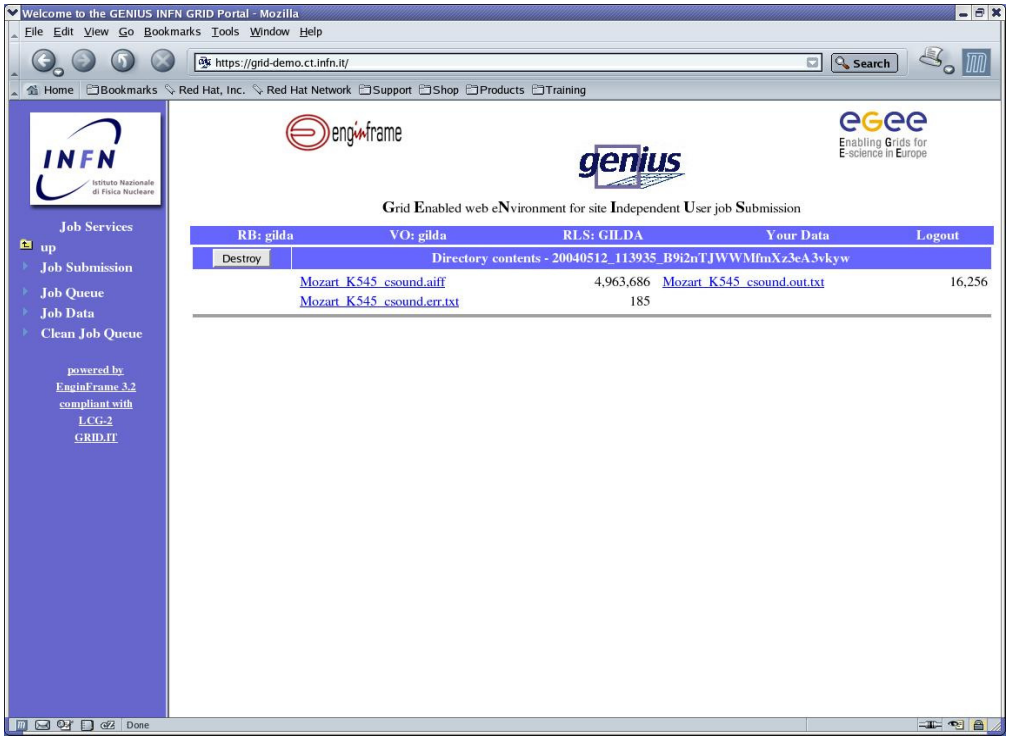


Fig. 6.19 - Job Output.

6.3.3 Job Data

With this service the user can view his/her personal spooler area. This is a virtual disk where the output file of all the jobs are stored, and they can be inspected through the web browser (clicking on the file names with the left button of the mouse) and/or copied on the user's local machine (clicking on the file names with the right button of the mouse). For obvious reason of disk space, job output file stays in the spooler area only for one day (24 hours). After this time, they are removed. This service shows a list of jobs output available in the spooler area.

Each row in the list contains the following information:

- the job identifier; clicking on it user accesses job spooler area

- data and time of job creation
- the number of output file (items) of the particular job
- the action possible on the job spooler area; if the action button Destroy is selected then the job spooler area relative will be deleted; this will not affect the corresponding job directory on the user's home directory in the UI.

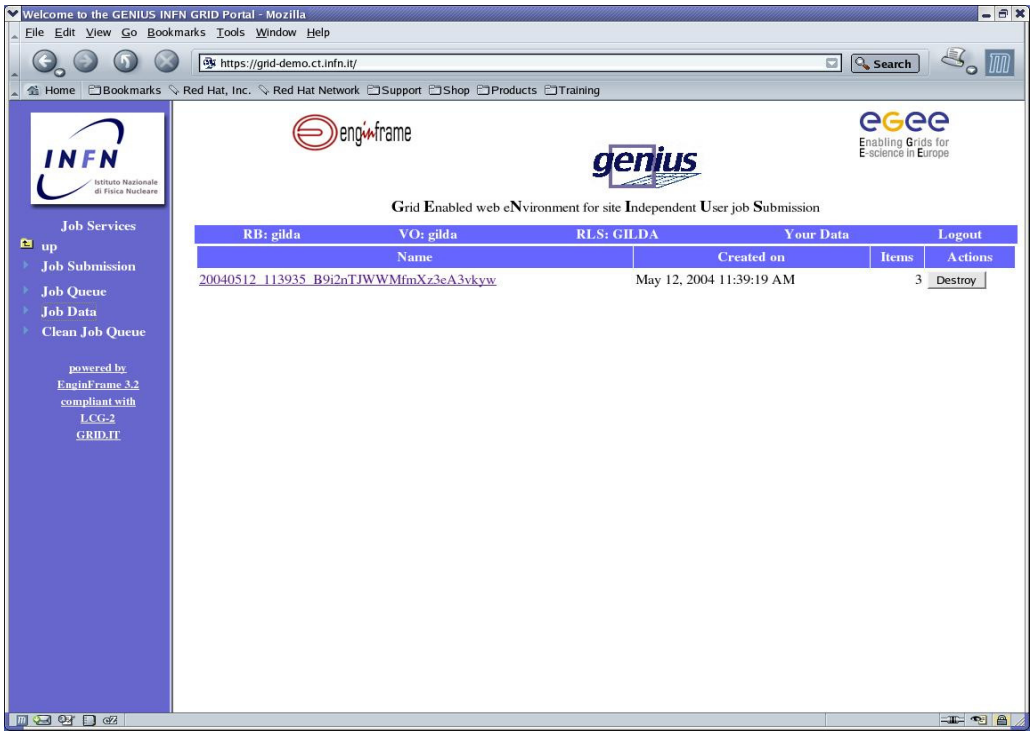


Fig. 6.20 - Job Data

6.3.4 Clean Job Queue

With this service users can clean their job queue list on the GILDA Resource Broker.

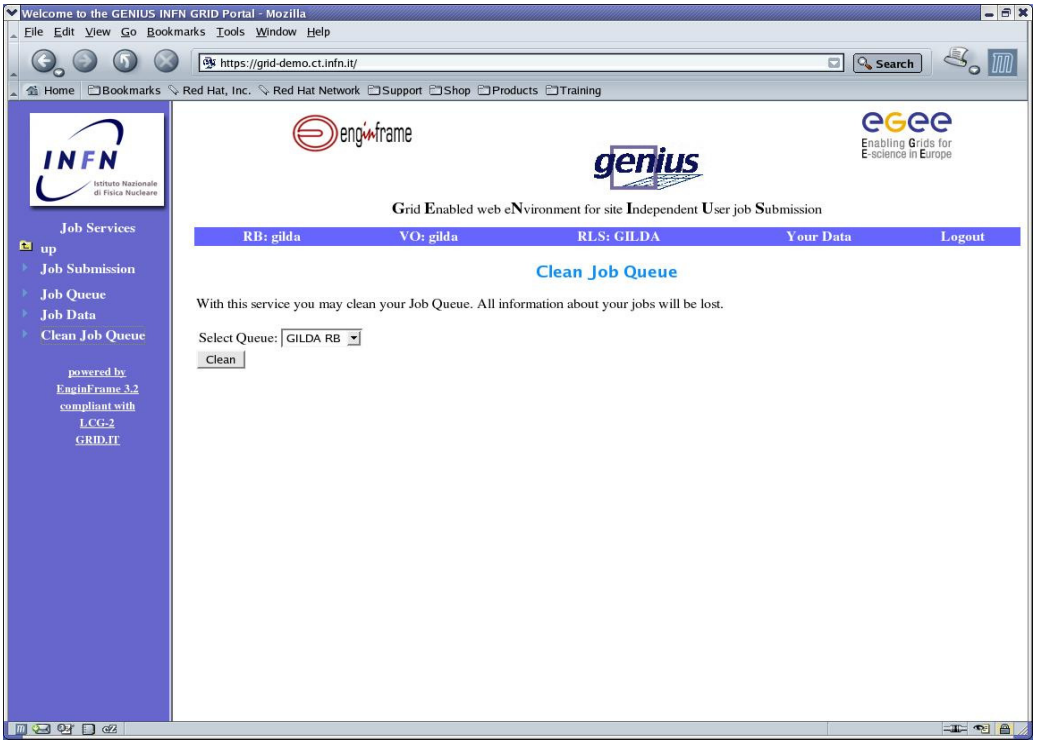


Fig. 6.21 – Clean Job Queue.

6.4 Data Services

A large part of users' tasks on the grid consist of access to data and management of the files containing data. Users would like to access the Grid service from all kinds of appliances (such as desktop, laptop, PDA, cell phone...) and from everywhere in the world, as easily as can be achieved through the World Wide Web. They would like to manage their data in a transparent way, that is as if these were local, independently of the physical storage elements where data are saved. Through *GILDA Grid Demonstrator* it is possible to satisfy these requirements in relatively simple but efficient way.
The section Data Service provides functions for data management, supporting the creation, replication and deletion of data files in the Grid and their registration in a Replica Catalog.

6.4.1 Navigate RLS

This service provides to user a support for choosing how to manage its data. In this case user can choose to do one of the following tasks:

- Browse & View
- Replicate a File
- Download a File

6.4.1.1 Browse & View

Selecting this item, and pressing the Navigate button, the user will obtain the complete list of the files published from any members of the VO which he/she belongs to (see fig. 6.23). Clicking on one of this file name, he/she can knows one or more SURL of this file in Storage Element (see fig. 6.24).

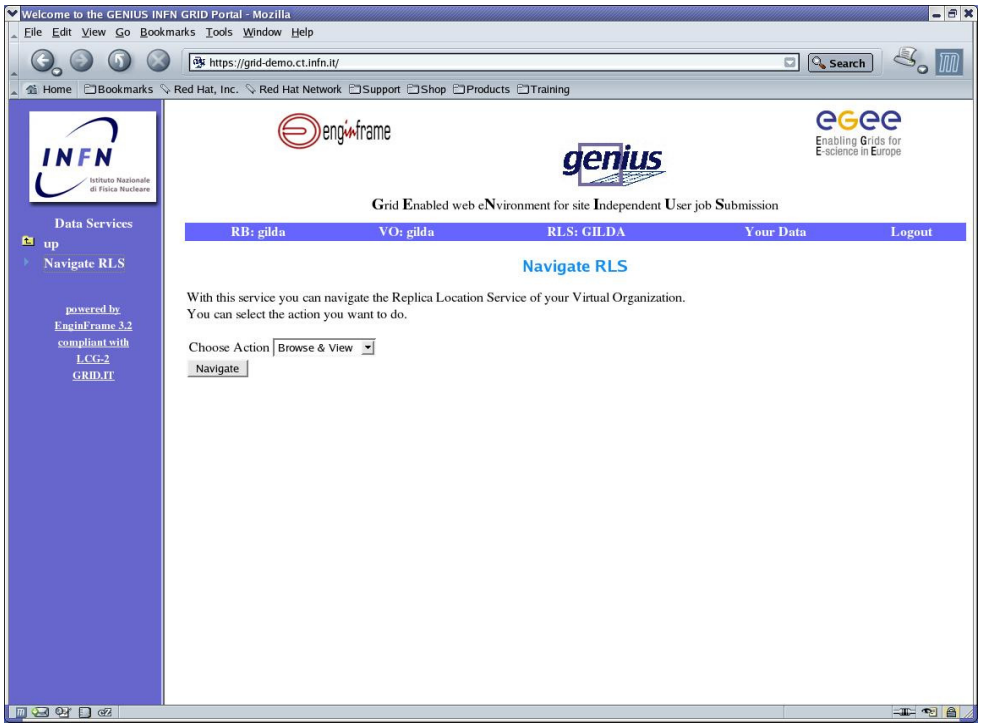


Fig. 6.22 – Navigate RLS - Browse and view(1).

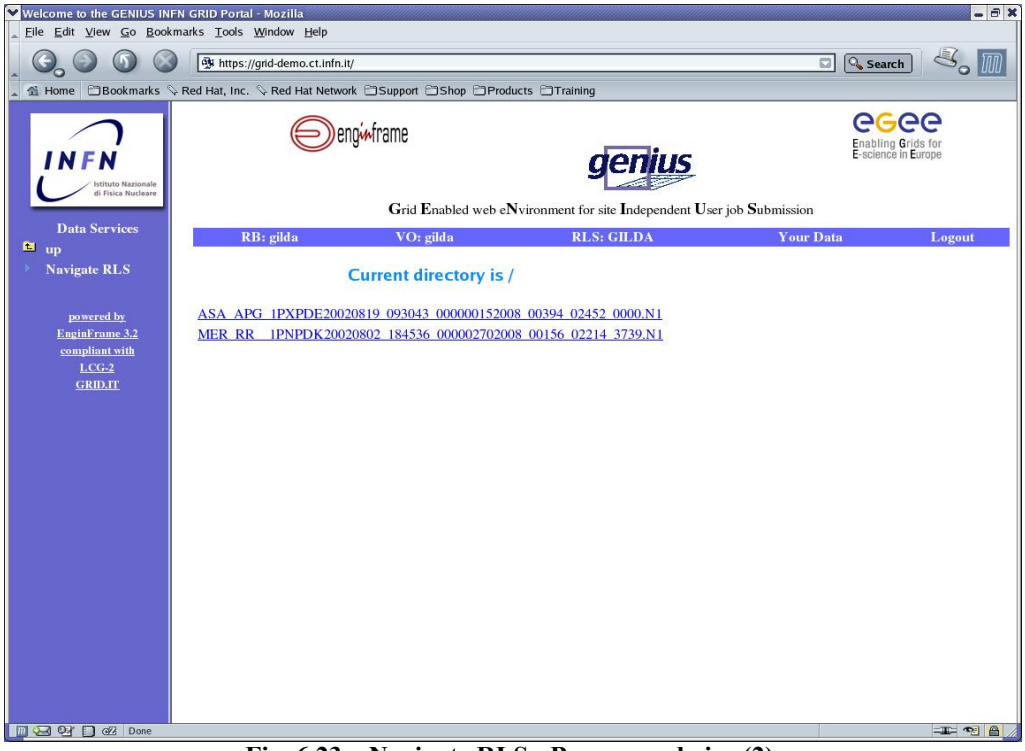


Fig. 6.23 – Navigate RLS - Browse and view(2).

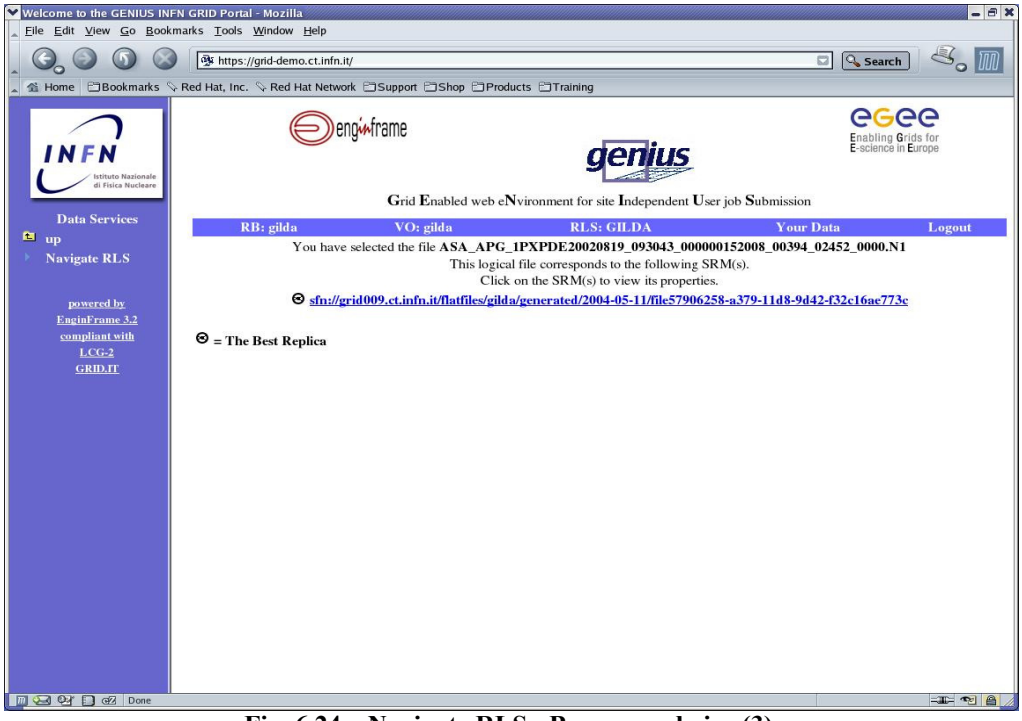


Fig. 6.24 – Navigate RLS - Browse and view(3).

Clicking on the SRM(s) is possible to view file contents. At this time, only ASCII file is

supported, so user cannot see it here any other kind of file. It's worth nothing that is indicated Best Replica's SURL, i.e. the SURL of the closest Storage Element in terms of total transfer time.

6.4.1.2 Replicate a File

This service allows a user to copy (replicate) a file already stored in a SE onto another Storage Element. Select Replicate a File in the menu item of the starting page in Navigate RLS and click on the navigate button. Clicking on Navigate button, a list of all files published by VO's members is returned to the user (see fig. 6.22).
To replicate a file a user has to click on its name in the list; a new page is shown, reporting the file(s) SURL('s). As indicated, to replicate the file, click on one of its SURL, and another page is opened, indicating the Storage Element where actually is the file, the destination SE and the file's LFN.

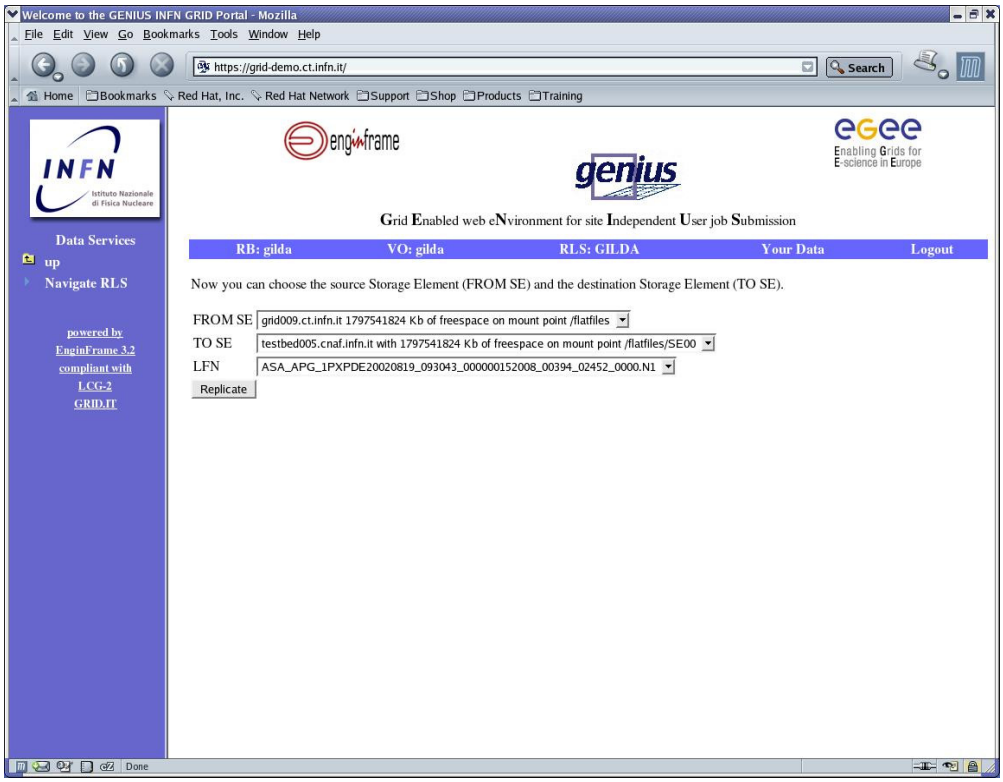


Figure 6.25 – Replicate a file.

By clicking on Replicate, the file indicated in the **LFN** field, which is stored in the SE specified in the **FROM SE** field, will be copied to the SE indicated in the **TO SE** field.
A page reporting the result of this operation is the shown.

6.4.1.3 Download a File

This service can be used to copy a Grid file to a non-grid storage resource; this may be useful for users to have a local copy of their files. The procedure for selecting a file is

identical to the ones described in precedent section, and as usual an output page is shown to user with the result of operation.

7 Logout GILDA Grid Demonstrator

With this service users can logout from *GILDA Demonstrator.* The browser remains open.

APPENDICE A – Jobs Application

As explained earlier, in this appendix the user will find, in detail, the output of all the possible jobs that he/she can submit to the grid. Some *GILDA Grid Demonstrator*'s jobs need three different applications installed on the grid (CSOUND, DEMTOOLs, POVRAY) to be run.

A.1 Hostname JOB.
A simple job which prints the worker node where it has been submitted.

A.2 CSOUND JOBs.
In this first section we will explain the characteristics of the jobs belonging to the first group of applications. *Csound* is a programming language/environment for music composition, sound design, and audio processing. Creating an output soundfile consists of writing and compiling two text-files: the orchestra (*.orc*) file which contains the instruments, and the score (*.sco*) file which plays the instruments.
GILDA Grid Demonstrator allows users to submit three jobs of this type: ***csound.jdl***, ***Mozart_K545.jdl*** and ***csound_hw1.jdl***. Each of these jobs produce an audio file (*.aiff*) which can be heard with the *play* application.

A.3 DEMTOOLs JOBs.
DEMTOOLs is a useful satellite image rendering program. Input is an ASCII files in the *.DEM* format (*Digital Elevation Model).* Common in geographical information systems and sometime acquired by high resolution remote sensing satellites, a DEM comprises a regular array, each cell of which holds the elevation of data in a corresponding rectangle of the earth's surface). DEMTOOLS is used to produce graphical virtual reality images, in *.wrl* file format, which can be browsed and rotated after output retrieval. There are three jobs of this type, as described below.

A.3.1 demtools.jdl
With this first job we create views of Mont Saint Helens and the Grand Canyon. The DEMTOOLs program produces virtual images in *.wrl* file format (*mount_sainte_helens_WA.wrl* and *grand_canyon_AZ.wrl)*. At the same time the DEMTOOLs program produces other output files using the DEM to PPM conversion program.
After the job's execution, it's worth to note that a visualization tool called *"lookat"* is required to visualize the graphical output file. If this tool is not installed on the UI machine used by the user, he/she will not be able to look at the produced output files. By clicking the *L* button user can switch between a black/white or colours vision of the output file. Otherwise by clicking on the *W* button user can get a grid vision of the output. For some details about the rpm's installation see Appendice B.

The screenshots of the DEMTOOLs 3D Map viewer is shown in the figures 6.26-6.27-

6.28.

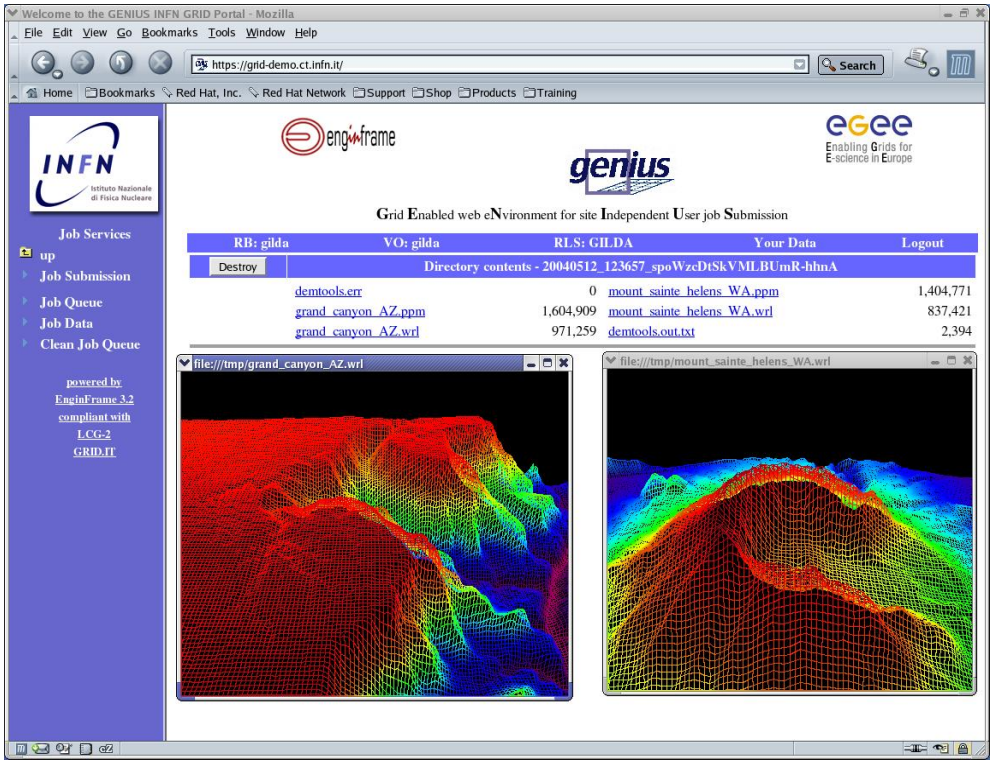


Fig. 6.26 – grand_canyon_AZ.wrl and mount_sainte_helens_WA.wrl.

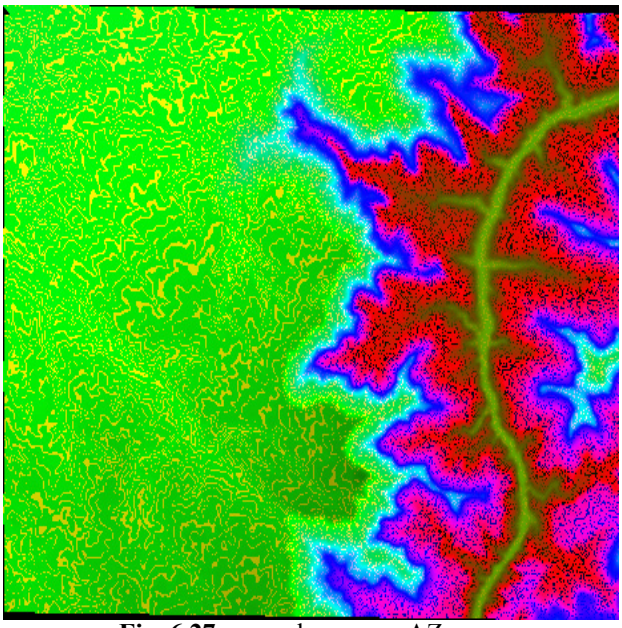


Fig. 6.27 – grand_canyon_AZ.ppm.

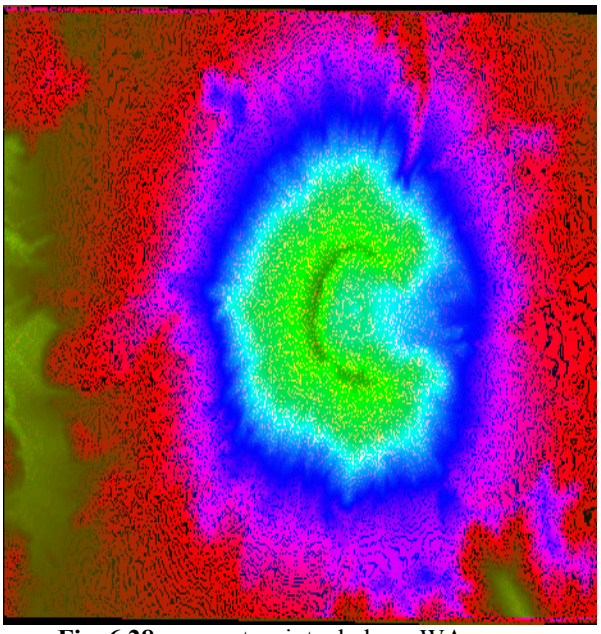


Fig. 6.28 – mount_sainte_helens_WA.ppm.

A.3.2 demtools2.jdl

In this second job we process the views of Etna and the Stromboli volcanos, and the DEMTOOLs program produces virtual images, in *.wrl* and *.ppm* file format. The virtual image of the Etna and Stromboli volcanos can be admired in the following figures.

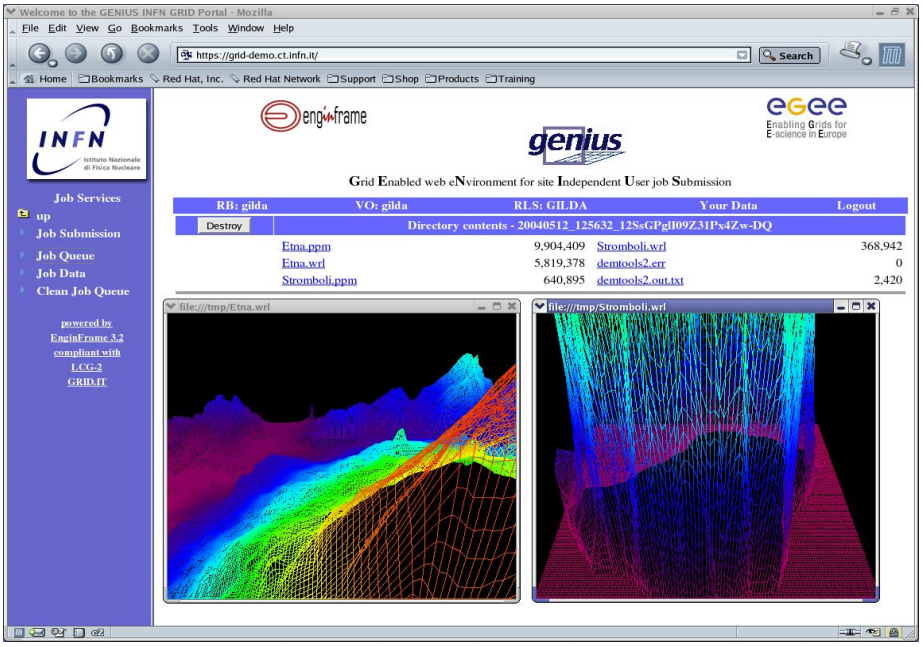


Fig. 6.29 – Etna.wrl and Stromboli.wrl.

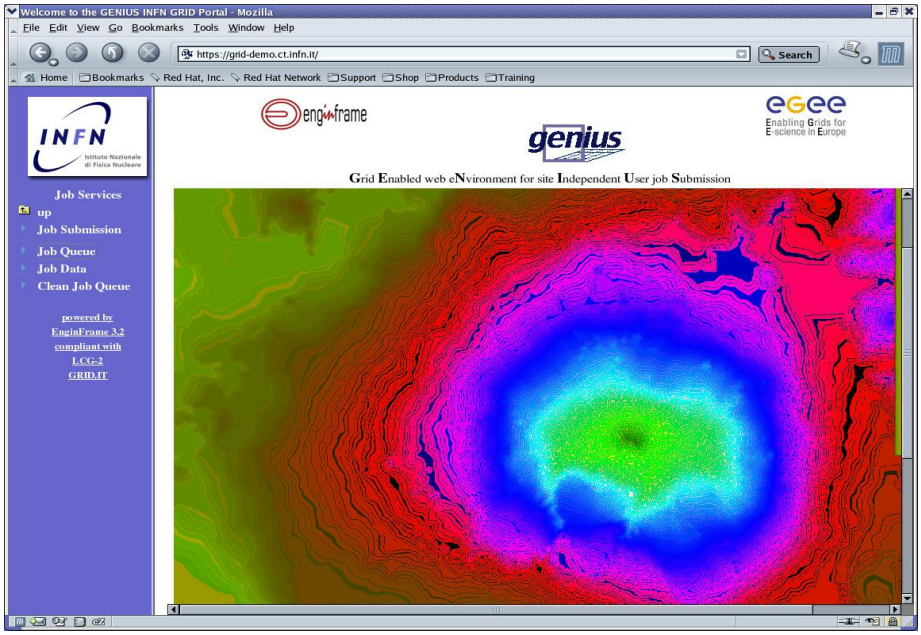


Fig. 6.30 – Etna.ppm.

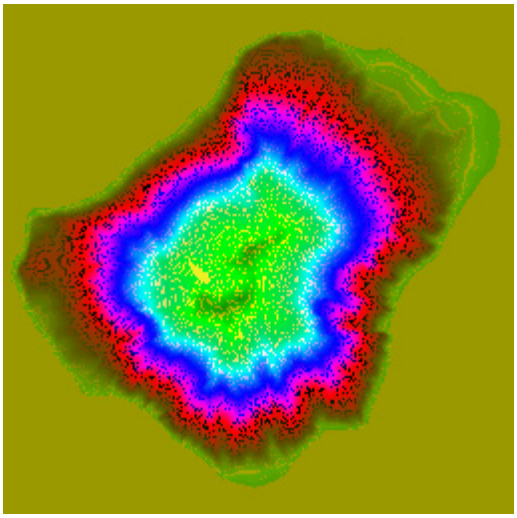


Fig. 6.31 – Stromboli.ppm.

A.3.2 demtools3.jdl

As in the previous examples we get virtual images of two American mountains.

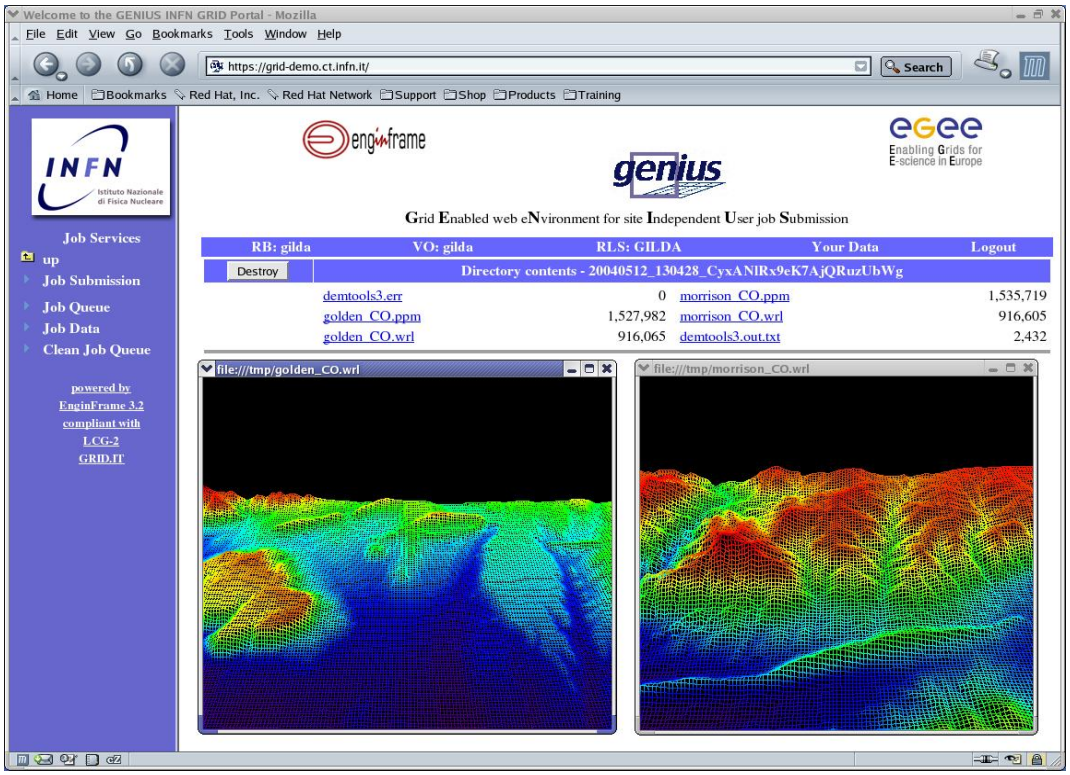


Fig. 6.32 – morrison_CO.wrl and Golden_CO.wrl.

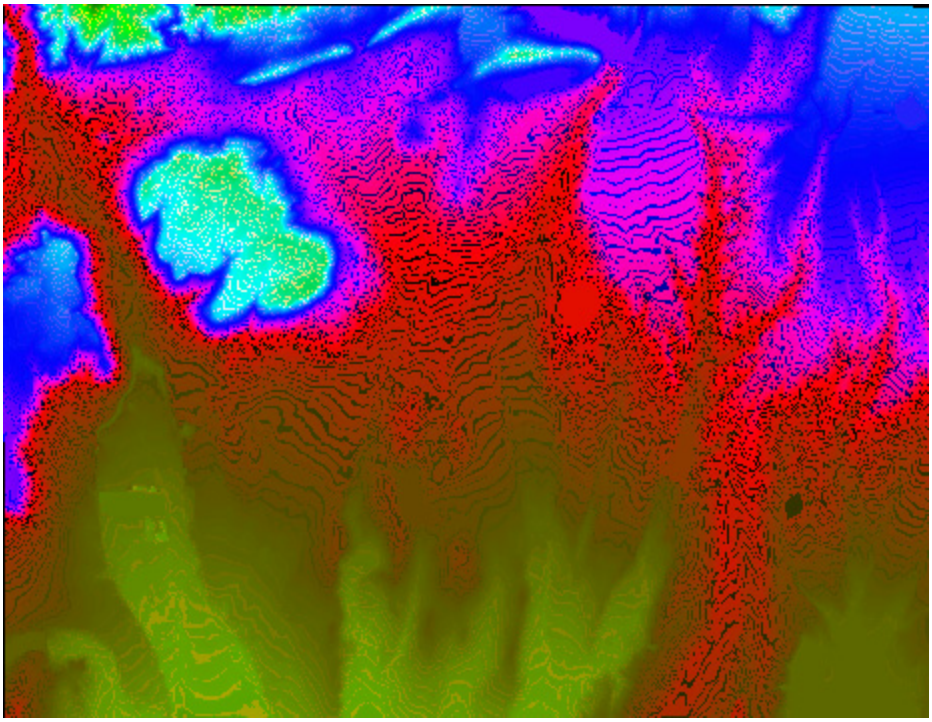


Fig. 6.33 – golden_CO.ppm.

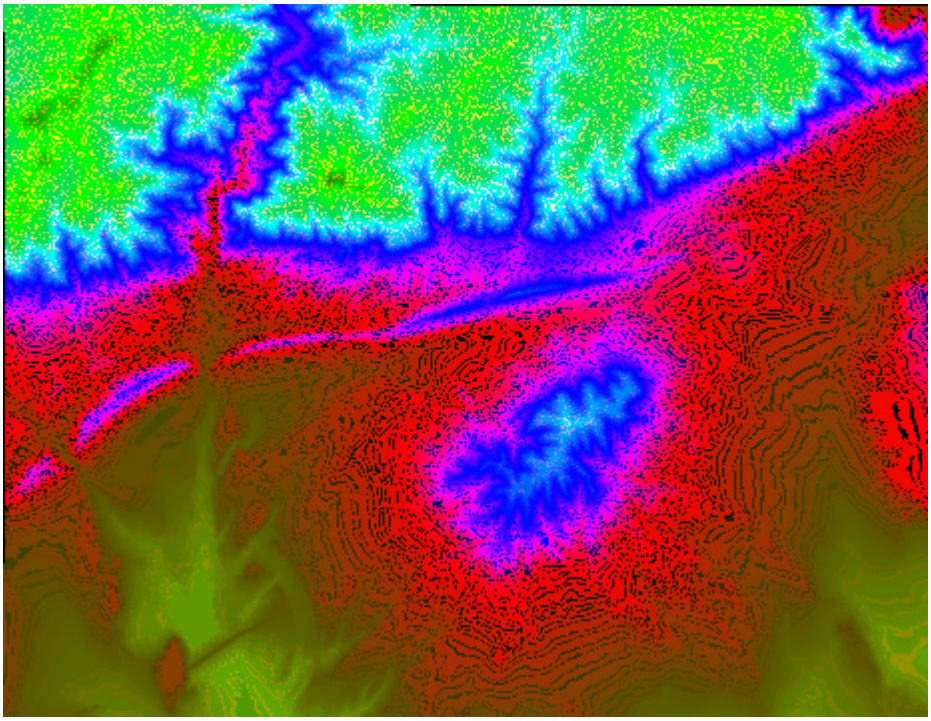


Fig. 6.34 – morrison_CO.ppm.

A.4 POVRAY JOBs.

The *Persistance of Vision Raytracer*, or just *POVray*, is a powerful raytracing package, which is freely available and runs on all major platforms. Being a raytracer, it allows the creation of complex three dimensional scenes with stunning realism and incredible artistic beauty, achieving effects that a realtime 3D rendering system cannot or only with difficulty achieve. Hereafter we will briefly describe the variety of jobs at the user's disposal.

A.4.1 povray_cubo.jdl

The povray_cubo file produce a 3-D rendering of a cubic lattice of spherical objects and cylindrical segments. Its screenshot is shown in the figure 6.35.

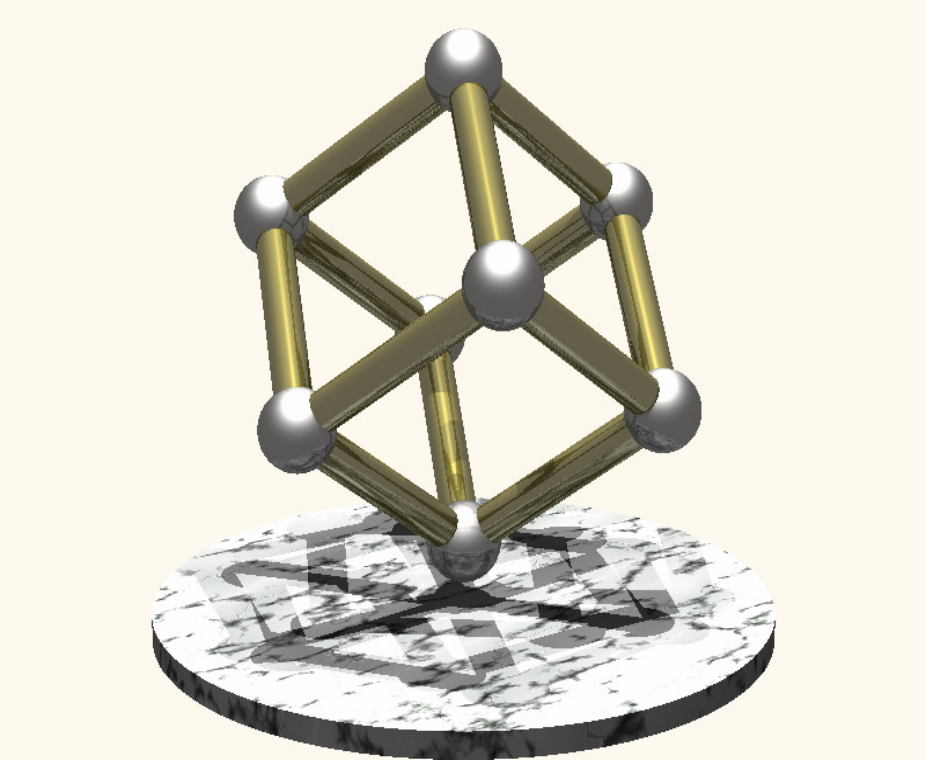


Fig. 6.35 – povray_cubo.jdl.

A.4.2 povray_defcity.jdl

The povray_defcity file produce a 3D rendering of a city as shown in the figure 6.36.

Fig. 6.36 – povray_defcity.jdl.

A.4.3 povray_dna.jdl

The povray_dna file produce a 3D rendering of human DNA as shown in the figure 6.37.

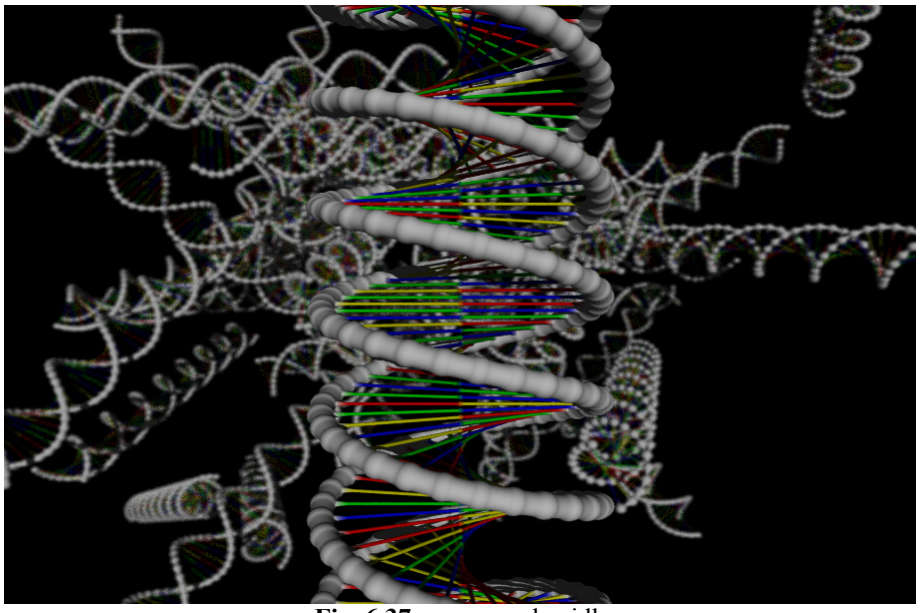


Fig. 6.37 – povray_dna.jdl.

A.4.4 povray_heart.jdl

The povray_heart file produce a 3D rendering of human heart as shown in the figure 6.38.

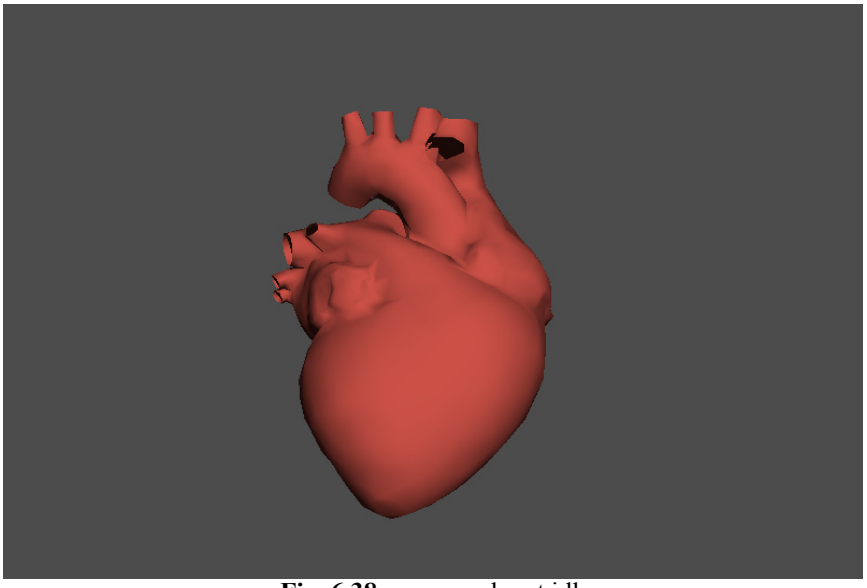


Fig. 6.38 – povray_heart.jdl.

A.4.5 povray_loop74.jdl

The povray_loop74 file produce a 3-D rendering of a concatenation of two ring objects joined with a sphere. Its screenshot is shown in the figure 6.39.

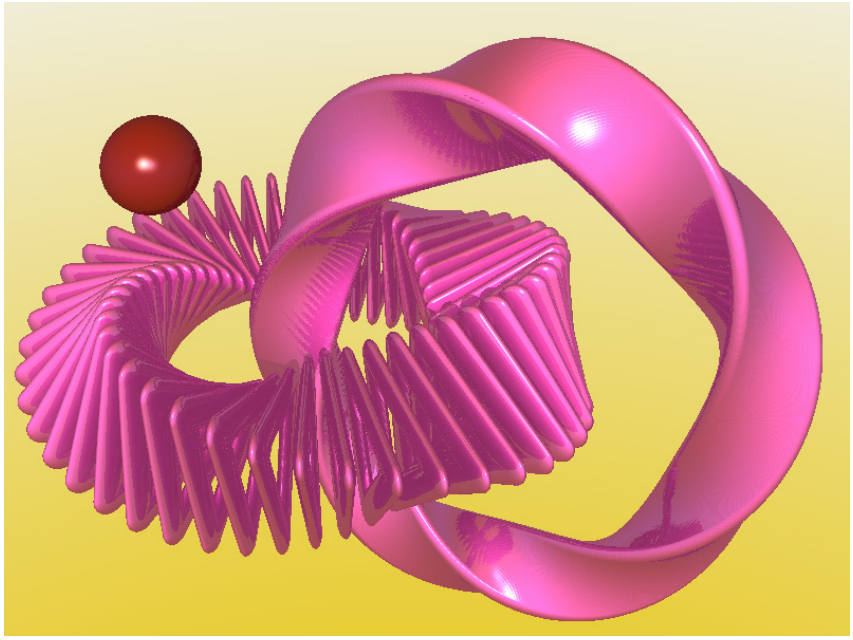


Fig. 6.39 – povray_loop74.jdl.

A.4.6 povray_loop7f_1.jdl

The povray_loop7f_1 file produce a 3D rendering of a circular object. Its screenshot is shown in the figure 6.40.

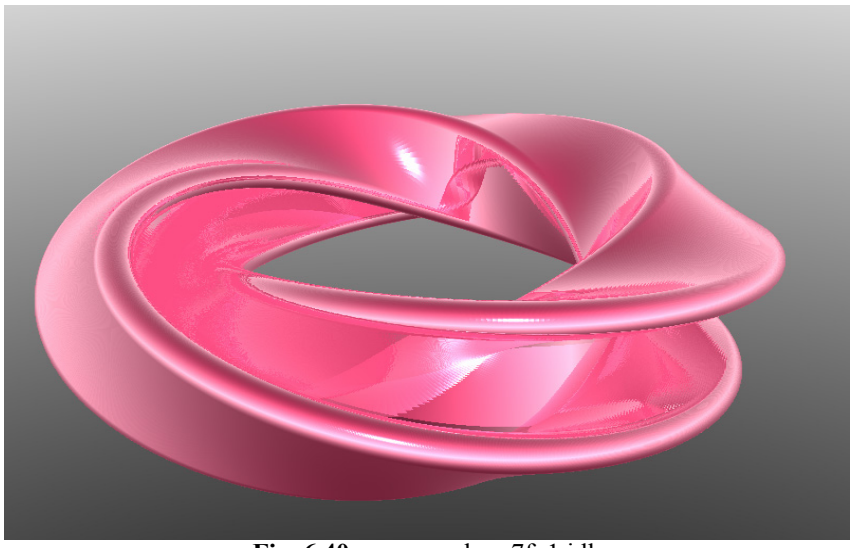


Fig. 6.40 – povray_loop7f_1.jdl.

A.4.7 povray_loop7g1.jdl

The povray_loop7g1 file produce a 3D rendering of a ring of two objects. Its screenshot is shown in the figure 6.41.

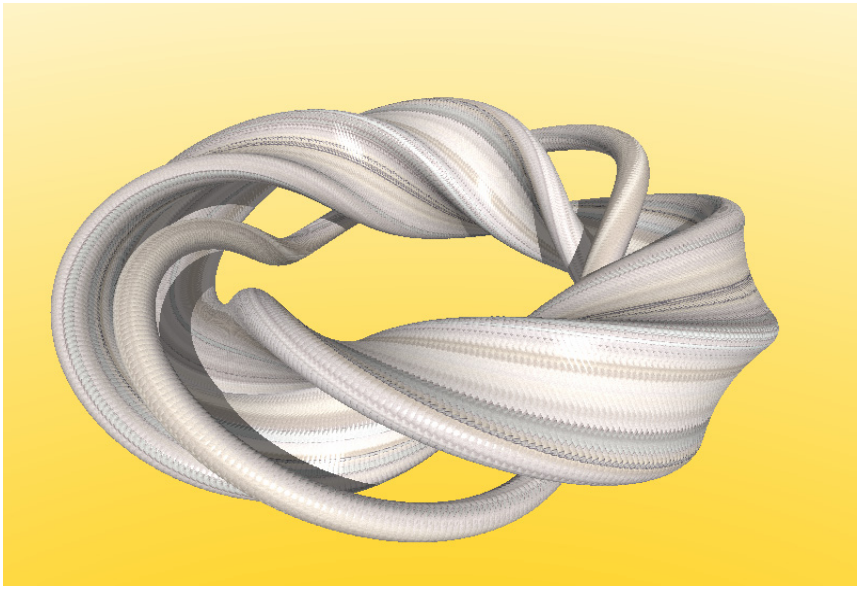


Fig. 6.41 – povray_loop7g1.jdl.

A.4.8 povray_loop7g3.jdl

The povray_loop7g3 file produce a 3D rendering of a pressed ring of two objects. Its screenshot is shown in the figure 6.42.

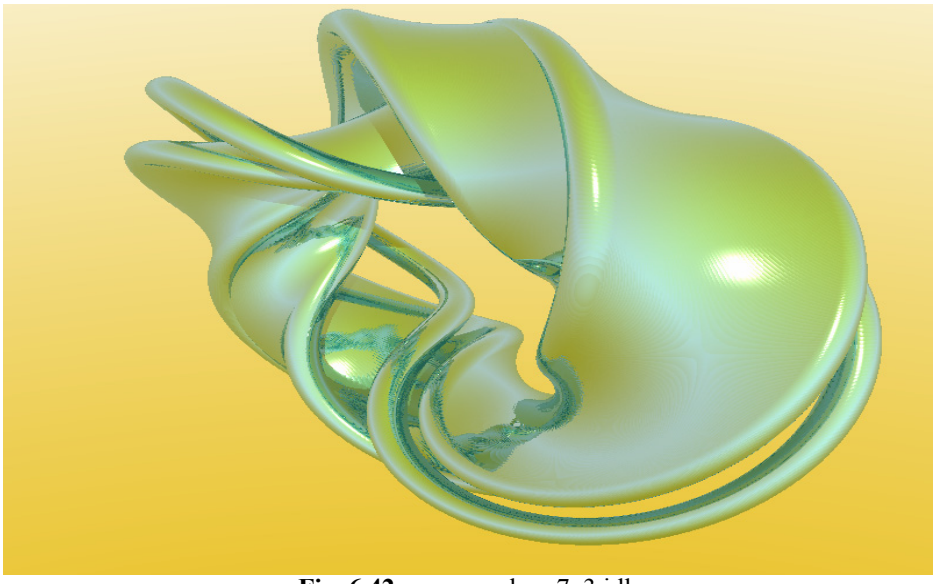


Fig. 6.42 – povray_loop7g3.jdl.

A.4.9 povray_loop7h1.jdl

The povray_loop7h1 file produce a 3D rendering of two helicoidal object as shown in the figure 6.43.

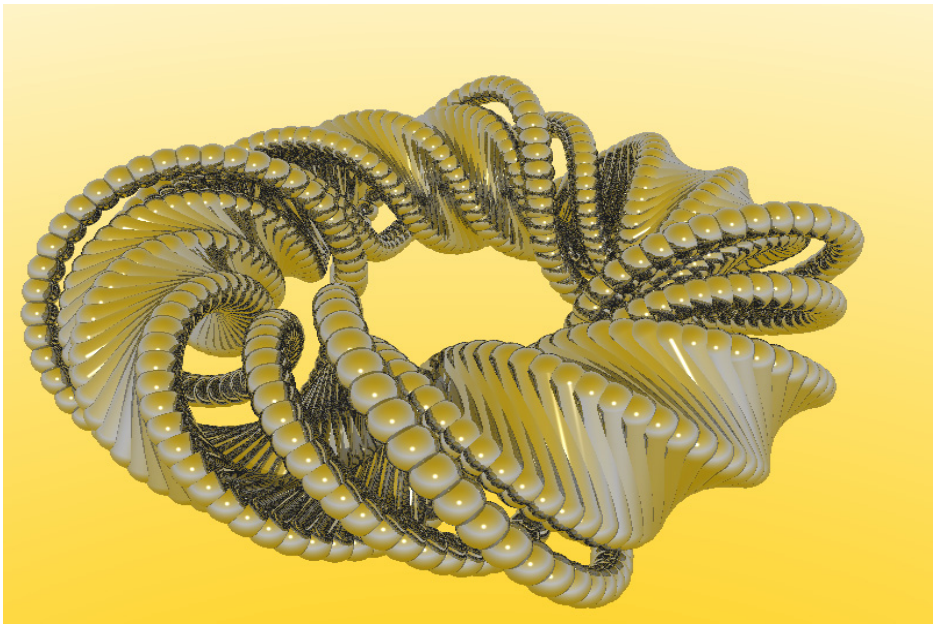


Fig. 6.43 – povray_loop7h1.jdl.

A.4.10 povray_loop7k.jdl

The povray_loop7k file produce a 3D rendering of a ring spherical object reflected in a mirror. Its screenshot is shown in the figure 6.44.

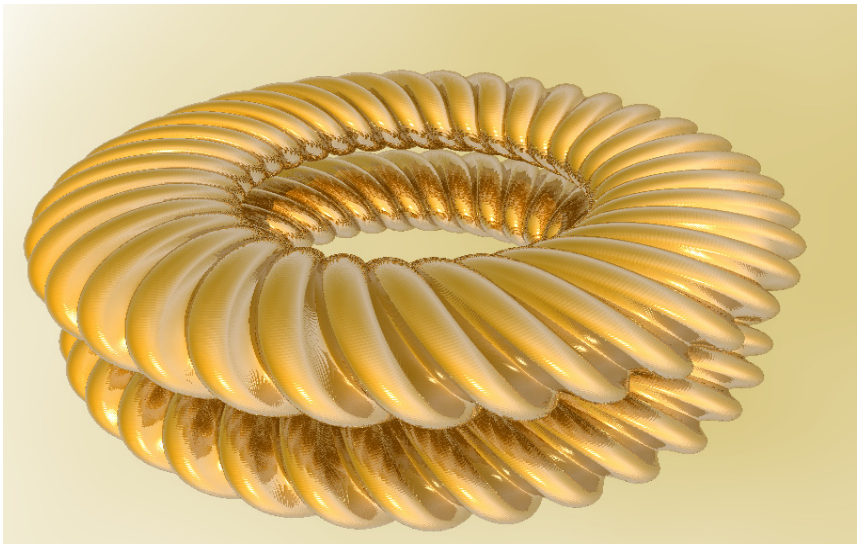


Fig. 6.44 – povray_loop7k.jdl.

A.4.11 povray_pblb3e.jdl

The povray_pblb3e file produce a 3D rendering spherical objects with tetragonal function. Its screenshot is shown in the figure 6.45.

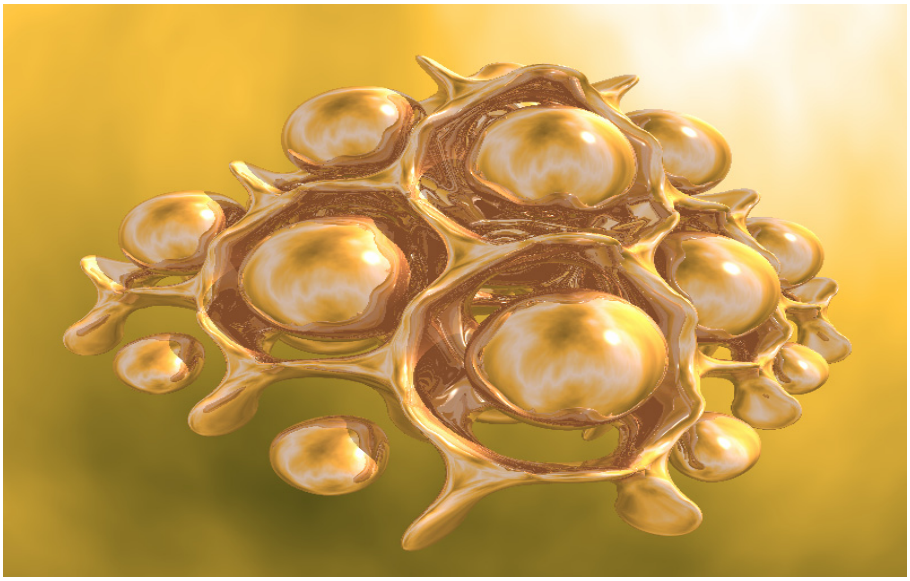


Fig. 6.45 – povray_pblb3e.jdl.

A.4.12 povray_hyper.jdl

The povray_hyper file produce a 3D rendering of a hyperbole with rhombus meshs as shown in the figure 6.46.

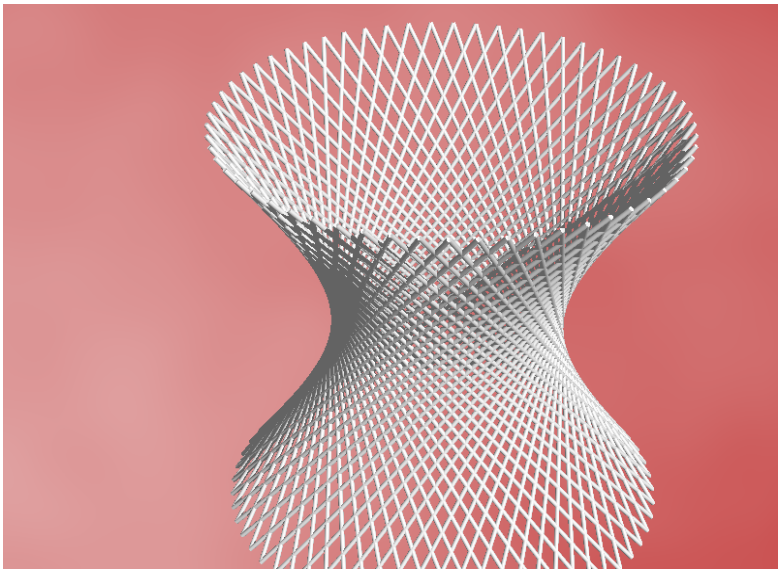


Fig. 6.46 – povray_hyper.jdl.

A.4.13 povray_pyranet.jdl

The povray_pyranet file produce a 3D rendering of a pyramidal reticulum of spherical objects and cylindrical segments as shown in the figure 6.47.

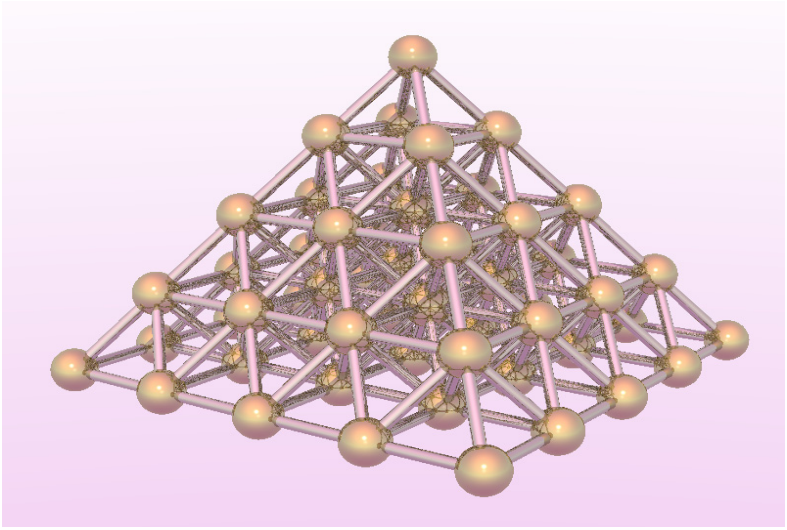


Fig. 6.47 – povray_pyranet.jdl.

A.4.14 povray_sinus0.jdl

The povray_sinus0 file produce a 3D rendering of a spherical objects as shown in the figure 6.48.

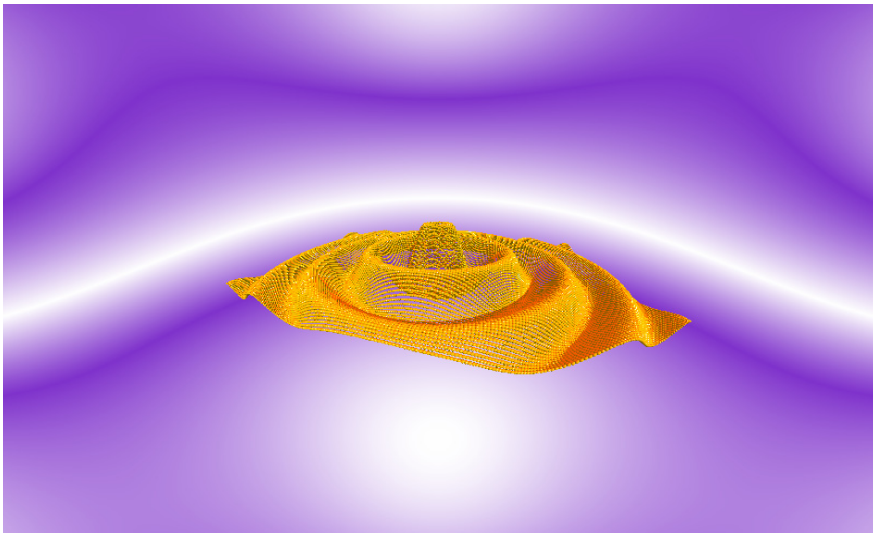


Fig. 6.48 – povray_sinus0.jdl.

A.4.15 povray_valve.jdl

The povray_valve file produce a 3D rendering of a mechanic valve as shown in the figure 6.49.

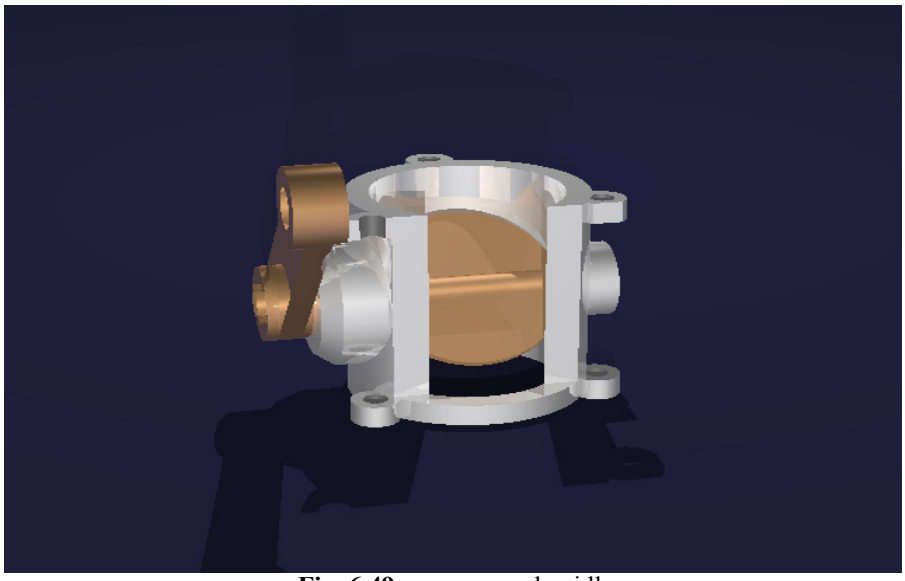


Fig. 6.49 – povray_valve.jdl.

A.4.16 povray_con_rod1.jdl

The con_rod1.jdl file produce a cyclic animation of a connecting rods using the sine and cosine function as shown in the figure 6.50.

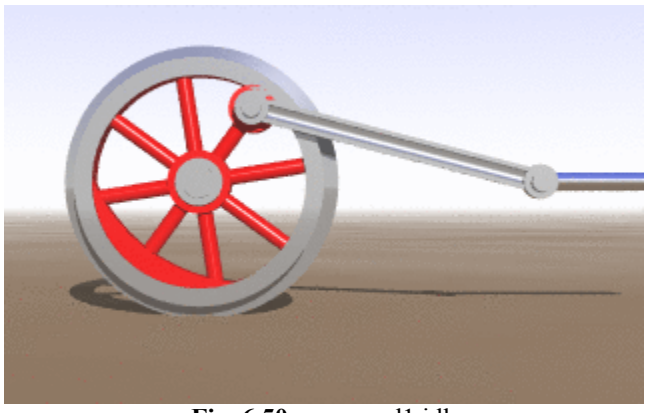


Fig. 6.50 – con_rod1.jdl.

A.4.17 gear.jdl

The gear.jdl file produce a cyclic animation of two dental wheels as shown in the figure 6.51.

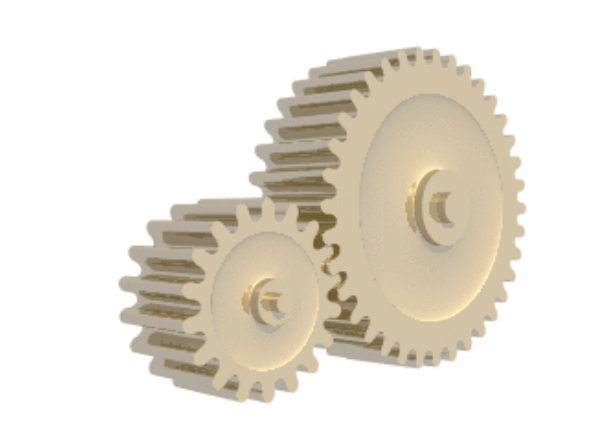


Fig. 6.51 – gear.jdl.

A.4.18 sphere.jdl

The sphere.jdl file produce a cyclic animation of a sphere as shown in the figure 6.52.

Fig. 6.52 – sphere.jdl.

A.4.19 spiral.jdl

The spiral.jdl file produce a cyclic animation of a spiral pendulum using the sine function as shown in the figure 6.53.

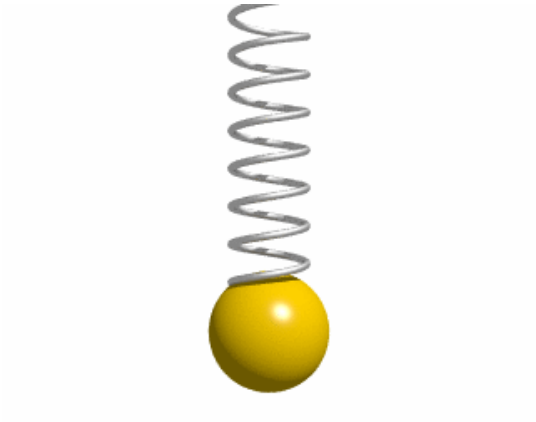


Fig. 6.53 – spiral.jdl.

A.4.20 spline.jdl

The spline.jdl file produce an animation with Spline curves as shown in the figure 6.54.

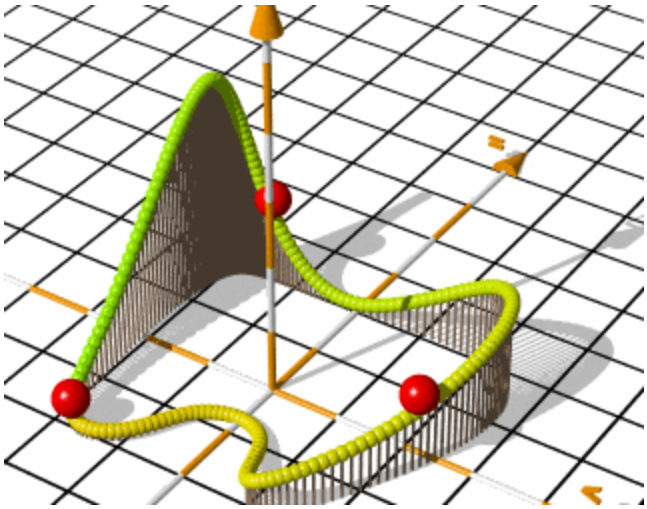


Fig. 6.54 – spline.jdl.

A.4.21 ball.jdl

The ball.jdl file produce a cyclic animation of a bouncing ball as shown in the figure 6.55.

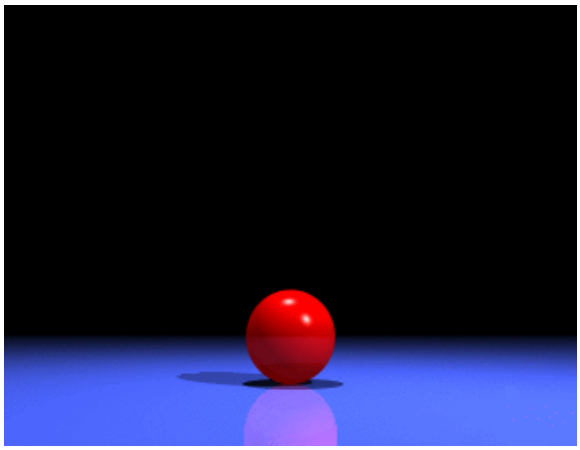


Fig. 6.55 – ball.jdl.

A.5 MERIS JOB.

MERIS (MEdium Resolution Imaging Spectrometer Instrument) is one of the 10 instruments flying on board of ENVISAT Earth Observation satellite.

MERIS is a 68.5-deg field-of-view pushbroom imaging spectrometer that measures the solar radiation reflected by the Earth, at a ground spatial resolution of 300m, in 15 spectral bands, programmable in width and position, in the visible and near infra-red. MERIS allows global coverage of the Earth in 3 days. The primary mission of MERIS is the

measurement of sea colour in the oceans and in coastal areas. Knowledge of the sea colour can be converted into a measurement of chlorophyll pigment concentration, suspended sediment concentration and of aerosol loads over the marine domain.

Four broad domains of applications of ocean-colour data can be identified:
- the ocean carbon cycle
- the thermal regime of the upper ocean
- the management of fisheries
- the management of coastal zones

MERIS is also capable of retrieving cloud top height, water vapour total column, and aerosol load over land. These measurements constitute MERIS secondary mission.

A.5.1 pds2jpg-MERIS-DEMO.jdl

The output of the submitted processing is an image over the Pacific Ocean built after the composition of spectral bands 2, 5 and 8 of a MERIS Reduced Resolution Level 1 product. *This is a product by courtesy of European Space Agency*.

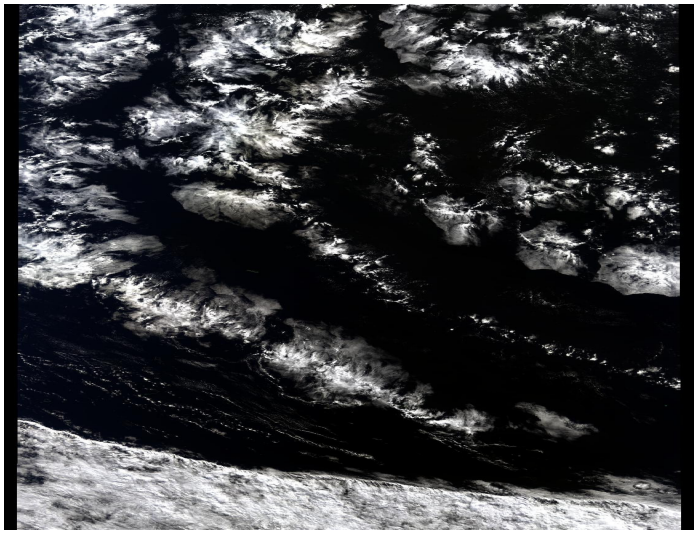


Fig. 6.56 – pds2jpg-MERIS-DEMO.jdl.

A.5.2 pds2jpg-MERIS-ETNA.jdl

The output of the submitted processing is an image of the Sicily and Eolie' isles. *This is a product by courtesy of European Space Agency*.

Fig. 6.57 – pds2jpg-MERIS-ETNA.jdl.

A.6 ASAR JOBs.

ASAR (Advanced Synthetic Aperture Radar) is one of the 10 instruments flying on board of ENVISAT Earth Observation satellite.

Operating at C-band, ASAR ensures continuity with the image mode (SAR) and the wave mode of the ERS-1/2 AMI. It features enhanced capability in terms of coverage, range of incidence angles, polarisation, and modes of operation. This enhanced capability is provided by significant differences in the instrument design: a full active array antenna equipped with distributed transmit/receive modules which provides distinct transmit and receive beams, a digital waveform generation for pulse "chirp" generation, a block adaptive quantisation scheme, and a ScanSAR mode of operation by beam scanning in elevation.

The Advanced SAR is built up on the experience gained with the ERS-1/2 active microwave instrument (AMI) to continue and extend Earth observation with SAR. Compared to ERS AMI, ASAR is a significantly advanced instrument employing a number of new technological developments which allow extended performance. The replacement of the centralized high-power amplifier combined with the passive waveguide slot array antenna of the AMI by an active phased array antenna system using distributed elements is the most challenging development. The resulting improvements in image and wave mode beam elevation steerage allow the selection of different swaths, providing a swath coverage of over 400-km wide using ScanSAR techniques. In alternating polarisation mode, transmit and receive polarisation can be selected allowing scenes to be imaged simultaneously in two polarisations."

A.6.1 pds2jpg-ASAR-DEMO.jdl

The output of the submitted processing are two images over Central Europe corresponding to polarisation combination sub-modes HV and HH of an ASAR Alternating Polarisation mode ellipsoid geocoded image product. *This are two products by courtesy of European Space Agency*.

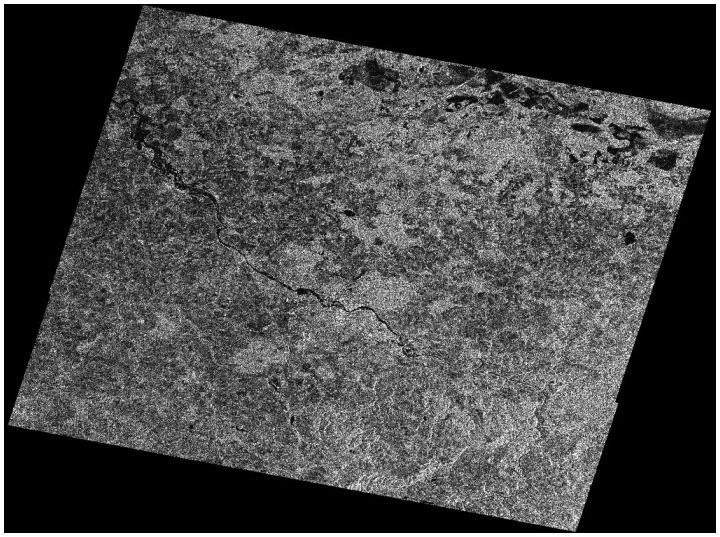
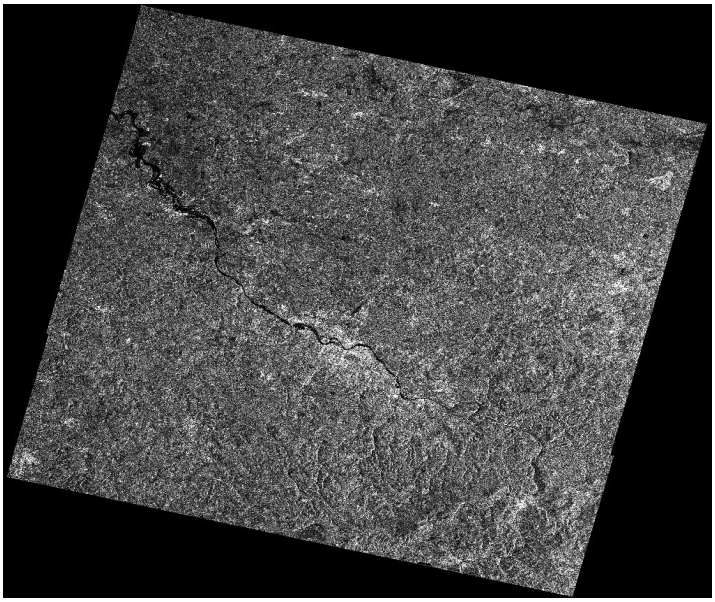


Fig. 6.58 – pds2jpg-ASAR-DEMO.jdl(1).

Fig. 6.59 – pds2jpg-ASAR-DEMO.jdl(2).



A.6.2 pds2jpg-ASAR-Prestige.jdl

The output of the submitted processing is an image of the Prestige tanker after the misfortune of the 17/11/2002. *This is a product by courtesy of European Space Agency*.

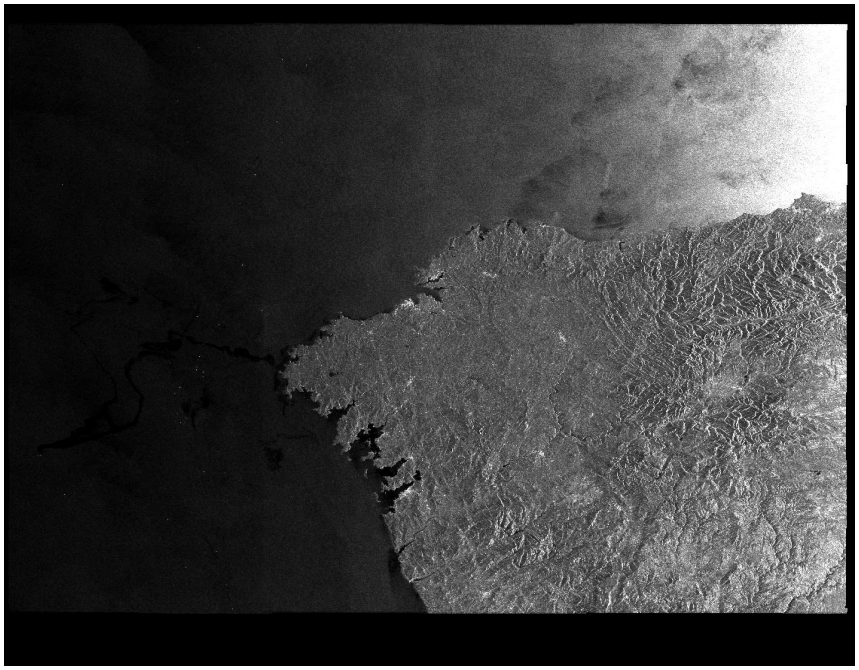


Fig. 6.60 – pds2jpg-ASAR-Prestige.jdl.

APPENDICE B – Installing plugins

1. VRML plugin

A good and exciting example of a Grid application is 3-D rendering. Demtools is a program used in the Grid to perform the rendering of satellite DEM (Digital Elevation Model). Starting from Remote Sensing Satellite data files (*.dem files), we will use Demtools to generate Virtual Reality images (.wrl image files).

If you want to see the output of a .wrl file you need to install a vrml-file viewer. An open VRML file viewer is **lookat**. The described procedure of installation is for this program on

a Linux machine with the web browser Mozilla; however there are not many differences for a different OS, browser or viewer.

1) Go to -> http://cic.nist.gov/vrml/vbdetect.html .You will find here also links for other vrml plugins, also for different OS.

2) In the *VRML Plugin* section, click on **OpenVRML-Lookat** for Linux, and click on *download* for **openvrml** version 0.14.3.

3) The packages to download are:
lookat-0.14.3-1.i386.rpm
openvrml-0.14.3-1.i386.rpm
openvrml-gl-0.14.3-1.i386.rpm

4)Go to http://rpmfind.net, and digit **glut** on the Search window

5)Download the **glut** RPM appropriate for your system above version 3.7
(e.g., for RedHat-9, choose glut-3.7-12.i386.rpm)

6) Become root on your machine, go to the directory where you put the downloaded RPM's, and install them with :

rpm -ivh lookat-0.14.3-1.i386.rpm openvrml-0.14.3-1.i386.rpm openvrml-gl-0.14.3-1.i386.rpm <glut_for_your_system>.rpm

7) Open Mozilla and go to **Edit -> Preferences**

8) Choose **Helper Application** and then click **New Type**

9)In the field *MIME Type* insert *model/vrml* and fill the field *Extension* with *wrl*; select the option *Open it with* in the section *When a file of this type is encountered* inserting */usr/bin/lookat*; at the end confirm with OK.

2. AIFF plugin
The CSOUND program outputs files in .aiff format. On Linux they can be played with the **play** utility; you can check if **play** is installed on your machine with (be root first)

rpm -q -a | grep play

If **play** is installed, follow the above instruction for .wrl (starting from step 7) to add it as plugin for .aiff files for your web browser. If **play** is not installed, try to download it or find another player which can acceppt .aiff files, and then set is as plugin for you web browser.

3. VLC Media Player
VLC (initially VideoLAN Client) is a highly portable multimedia player for various audio and video formats (MPEG-1, MPEG-2, MPEG-4, DivX, mp3, ...) as well as DVDs, VCDs,

and various streaming protocols. It can also be used as a server to stream in unicast or multicast in IPv4 or IPv6 on a high-bandwidth network. VLC media player is a cross-platform media player and server. It works under Linux, Windows, Mac OS X, BeOS, BSD, Solaris, Familiar Linux and QNX.

- For Fedora Core 2 download the RPM packages tarballs listed below

vlc-binary.tar.gz (VLC Binaries packages)
redhat9-updates.tar.gz (Updates required to run VLC on Red Hat 9)
fedora1-updates.tar.gz (Updates required to run VLC on Fedora Core)

Uncompress them in the same directory (as root)
tar zxf <package name>
rpm -U vlc/* --force

Insert a new row like this:
-A <Chain Name> - m state –state NEW -m udp -p udp –dport 1234 -j ACCEPT
in the file */etc/sysconfig/iptables* to receive the udp stream from the 1234 port.

- For Windows platform download the self-extracting package: **vlc-0.7.2-win32.exe**

Configure your VLC client opening the *Network Stream Panel* (CTRL+N) and select UDP/RTP with port 1234.

More information about the VideoLAN streaming solution can be found in the streaming section http://www.videolan.org/.