DELIVERABLE D4.9

APPENDIX 1: FINAL DEMO REQUIREMENTS ON THE TESTBED

WORKPACKAGE 4

Document Filename:	**CG4-D4.9-v1.2-CSIC22-Appendix-Demo.doc**
Work package:	**WP4 International Testbed Organisation**
Partner(s):	**CYFRONET, ICM, INS, UvA, II SAS, FZK, PSNC, UCY, TCD, CSIC, UAB, USC, DEMO, AUTh, LIP**
Lead Partner:	**CSIC**
Config ID:	**CG4-D4.9-v1.2-CSIC22-WP4-Appendix-Demo**
Document classification:	**PUBLIC**

Abstract: This appendix describes the requirements from the final demos on the testbed.

Delivery Slip

	Name	Partner	Date	Signature
From	Integration team	CSIC	14-Dec-2004	
Verified by	Santiago Gonzalez	CSIC	21-Dec-2004	
	Robert Pajak	CYFRONET	9-Feb-2005	
Approved by	Jesus Marco	CSIC	21-Dec-2004	

Document Log

Version	Date	Summary of changes	Author
0.1	1-Dec-04	Initial draft and template	Jesús Marco
0.2	14-Dec-04	Filled by Iteam	Integration Team
1.0	21-Dec-04	First release version	Integration Team
1.1	4-Feb-04	All sections updated with real demos details	Jesus with input from Branislav, Roland, Fran, David
1.2	8-Feb-04	Further updates	Jesus with input from Alfredo, David
	9-Feb-05	Verified by the Quality Engineer	Robert Pajak

CONTENTS

1. INTRODUCTION

OVERVIEW

The CrossGrid project is approaching its exploitation phase, and an important point is the preparation of demos running in the testbed showing the successful integration of applications, middleware and tools.

Along the integration work, demos have been presented along the different meetings to show the status in a direct way. In particular at the integration meeting in Dagstuhl, an explicit list of demos was included in the integration document. After the integration meeting in October 2004 in Lisbon the final project months are devoted to prepare the final review and demo in February in Amsterdam.

Notice that this report is an appendix to D4.9, Final Demo and Report.

DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

CrossGrid/X#	The EU CrossGrid Project IST-2001-32243
DataGrid/EDG	The EU DataGrid Project IST-2000-25182
GRID	Grid framework for sharing of distributed resources.
MPI	Message Passing Interface.
MPICH	A portable implementation of MPI (CH stands for Chamaleon)
WP	Work Package
VO	Virtual Organization
CE	Computing Element (EDG).
SE	Storage Element (EDG).
API	Application Programming Interface
RPM	Red Hat Package Manager
LCFG	Local Configuration Tool

REFERENCES

- Integration Team Web Page, Forum, and Repository at GridPortal
  http://gridportal.fzk.de/websites/crossgrid/iteam/
  and from there:
  - Developers guide
    http://savannah.fzk.de/websites/crossgrid/iteam/devguide/
  - Code Review recommendations
    http://gridportal.fzk.de/websites/crossgrid/iteam/CG4-D4.7-Code-Review.doc
- Testbed installation and support
  http://www.lip.pt/computing/projects/crossgrid/crossgrid-services/installing.htm
  and from there:
  - Installation Guide
    http://cgi.di.uoa.gr/~xgrid/cgfiles/LCFGng_v2.0.pdf
- Test and validation web pages
  http://www.lip.pt/computing/projects/crossgrid/task4/middleware-validation.htm
- Testbed tutorial and examples
  http://grid.ifca.unican.es/crossgrid/wp4/Tutorial/CG4-Tutorial.pdf
- CrossGrid HelpDesk
  http://cg1.ific.uv.es/hlpdesk/

2. FINAL PROJECT DEMO AND CURRENT TESTBED POSSIBILITIES

Compared to previous demos, the framework is now extended towards the preparation of final components and future exploitation, and should have into account technical issues but also packaging and ease of use and deployment considerations.

KEY COMPONENTS AND RELATION WITH DEMOS OF APPLICATIONS IN A GRID FRAMEWORK IN THE CROSSGRID TESTBED

The CrossGrid project includes four different applications that should run in a grid framework that provides to users with an interactive setup, with access to powerful resources, and control of the whole process. Regarding exploitation, it is important to remark the relation of applications with the Key Components.

Following the Integration document, it can be said that CrossGrid has chosen a clear model for the "gridification" of applications:

a) ***USE OF MPI:*** Application Developers do not need to learn complicated techniques to access distributed computing resources: use of MPI allows an almost transparent migration from a multiprocessor local environment (either cluster or SMP) to either a remote cluster or even to a distributed multi-sites framework (via MPICH-G2). The MPI use is possible thanks to several software components. The MPI-adapted **Resource Broker** allows the allocation and common use of a set of distributed computing nodes. It includes also a post-processing module to predict the availability of these resources.

   Two more tools are oriented to MPI use: the MARMOT package, and the Grid benchmarking and PPC components. The user also handles in a quite "transparent" way the access to distributed data resources, using the EDG replica-manager script tools. Examples of both MPI and EDG-RM use are given in the WP4 tutorial exercise description.

b) ***INTERACTIVE AND FRIENDLY USE:*** this is possible thanks to several components: the Interactive JSS service allows a direct flow of data up to the user; the Migrating Desktop and the Portal are two different front-ends to another key component: the Roaming Access Server (RAS) enables the user to access resources as in a local environment (similarly to the use of a web-mail vs. using a direct mail client). The RAS system provides also a clear and well defined environment to the user: a /home directory, a /group and /scratch directories, all as "virtual directories". The connection to the JSS is transparent through the translation of XML forms definitions (either from the Portal or from the MD) to define JDL scripts and actually launch, run and control jobs. All details are almost transparent to the final user. Graphics reach the final user via the Portal or the MD (or even via X11 if running from a UI).

c) ***CONTROL:*** the whole process is achieved thanks to several components monitoring, registering and analyzing the information on the infrastructure (JIMS), the network flow (SANTA-G) and the application itself (OCM-G). The user receives the information about the grid environment from these tools, and the admin managers can control the use of the resources using the same information (published through a relational database accessible as a web service, R-GMA)

The demo should show clearly how one or more of these key components are relevant.

PACKAGING KEY COMPONENTS FOR EXPLOITATION AND DEMOS

This item is being developed in the exploitation part.

3. GENERAL PLANS FOR DEMOS

The demos will be based on full applications, but should show the use of the key components.

The status by Lisbon and IST'04 time was:

- **Biomedical application**
  - MD access to SE; plugin for MD, job submission to testbed and monitoring using OCM-G/GPM
  - Visualization with GVK
  - Benchmarking
  - MARMOT on MPI code
- **Meteo and Air Pollution application**
  - Running in parallel COAMPS in different clusters using MPICH-P4
  - Running in parallel AIR POLLUTION model using MPICH-P4
  - MD integration and graphics
- **Flooding application**
  - MD plugin
  - Metadata Catalog
- **HEP towards use of Interactivity**
  - ***Histogram example***
  - ***ANN training: looking to the impact of initial seeds***

4. PROPOSAL OF TEMPLATE ON DEMOS AND TESTBED REQUIREMENTS

The "Demo" will in principle include both an "interactive" presentation showing some "display" and a parallel presentation (with slides).

This document gives only some advice about the demos that are under responsibility of the different WP and tasks in the project and tries to get the requests that they will pose on the testbed.

TEMPLATE

- Setup requested (example: a laptop for the presentation, connected to a projector, another laptop for the demo, with fast ethernet or wireless connection, connected to a second projector, with XGA resolution, etc.)
- Describe briefly (2-3 lines) your demo. Indicate if it is interactive and MPI based.
- Indicate where are the executable, sources, input data, and what output is expected, and give an example of JDL file describing a test run
- Estimate how many CPUs, at how many sites, and how much Storage and Data Transfer will be needed or would be ideal. In particular indicate speedup issues.
- Indicate the use of CrossGrid middleware and tools in your demo:
    - Release that will be used if specific
    - How the use of middleware will be indicated in the demo and how important it is
    - Why some middleware that could be relevant is not used or was not found adequate
    - What tools are involved, and for what explicit communities
- Draft a temporal scheme of the demo like the following
    - -30' Initiate transfer of files, or pre-computation, etc
    - 0' Introductory slides (1-4)
    - 5' Submission of job to the testbed from MD in laptop
    - 7' Continue showing slides (5-8)
    - 9' Monitor the job is running
    - 10' Continue slides 9-10
    - 12' Fetch the output, display it in MD

    …
- Indicate clearly the public/community addressed in the DEMO/Presentation (developers, final users, testbed administrators)
- Provide the numbers that will sustain "benefit of Grid computing":
    - Using 4 sites with 8 CPU each, we were able to provide an answer to 4 different configurations in 3 minutes, to be compared with the 100 minutes that it took previously in the desktop

PRELIMINARY LIST OF POINTS TO BE CONSIDERED IN THE PRESENTATION OF THE DEMO (AS IT COULD APPEAR IN A REVIEWER QUESTIONNAIRE):

- Is the presentation clear and well structured? Does it have a proper introduction and conclusion? Is the order of material being presented sensible?
- Is there a basic message well transmitted?
- Does the presentation make good use of visual aids? Are the slides clear, easy to red, used when appropriate in parallel to the demo?
- Does the presentation cover specific techniques used in the project?

- Do the slides talk about some specific lessons learned?
- Is there some creativity/originality in how the material is presented?
- The demo runs using the documented components and samples without obvious errors?
- The demo runs using the reviewer own reasonable samples and parameters without obvious errors?
- The commands and tasks asked to be performed seem to work reasonably well?
- The demo covers the complete system of just a part while others are not clearly working?
- Is the “product” easy to use, intuitive, and aesthetically pleasing?
- Is the role of the testbed in the demo clearly shown, and also a “grid-model” (including use of resources in a cooperative way, VO community, etc)?
- Are the benefits of grid computing described explicitly?
- Are key project ideas (MPI, interactivity) clearly shown?

5. DETAILED LIST OF REQUESTS FROM EACH DEMO AND SUMMARY ON TESTBED REQUIREMENT

The descriptions below gives only some advice about the demos that are under responsibility of the different WP and tasks in the project and tries to get the requests that they will pose on the testbed.

BIOMED DEMO

(Received from Alfredo Tirado)

The "Demo" includes 2 parts. The first 30 minute medical part includes a couple of short videos showing relevant medical procedures, a .ppt slide (workflow snapshot), and our normal whole workflow. It will use 2 projectors showing respectively the 2D CrossGrid and 3D rendering/image manipulation parts. The second part includes 10 minute segments showing our use of OCMG/GPM, GridBench, and MARMOT (if time allows).
BIOMEDICAL

- **Setup requested:**
  - a laptop for the 2D part of the presentation and the CrossGrid tool presentation, connected to a projector and the local network; the PSS 3D rendering hardware, with fast connection to the local network, connected to a second projector.
- **Demo description:**
  - The demonstration will show the medical application accessing medical data via the MD, data manipulation via our PSS machine, job submission for simulation and visualization via the whole CrossGrid infrastructure (as MPI job), and final rendering locally in PSS. The demonstration will then show access to CrossGrid tools (e.g., GVK, GPM, GridBench) via MD and some detailed analysis.
- **Indicate where are the executable, sources, input data, and what output is expected, and give an example of JDL file describing a test run**
  - Executables and data are stored at the Storage element at LUMC, LIP, and other sites. Output will be binary datasets and visualization.

    ```
    Executable = "wrapper.sh";
    Arguments = "10. bifurcation-stenosis --ocmg-appname Blood --ocmg-mainsm
    8d34a02a:4e24 --ocmg-debug";
    StdOutput = "bstream.out";
    StdError = "bstream.err";
    InputSandbox = {"wrapper.sh", "B_Stream_ocmg", "../data/bifurcation-stenosis.bs",
    "../data/bifurcation-stenosis.conf"};
    OutputSandbox = {"bstream.out",  "bstream.err", "velocity.tar.gz"};
    JobType = "MpiCh";
    NodeNumber = 4;
    ```
- **Estimate how many CPUs, at how many sites, and how much Storage and Data Transfer will be needed or would be ideal. In particular indicate speedup issues.**
  - One MPI job will use 4-8 CPUs, MPI class is MPICH-P4, data transfer will be 10s-100s of MBs. Due to high communication overhead of our MPI applications speedup is very small when running on more than 8 nodes.
- **Indicate the use of CrossGrid middleware and tools in your demo:**
  - **Release that will be used if specific:**
    Production testbed
  - **How the use of middleware will be indicated in the demo and how important it is**
    Middleware will be hidden, but behind-the-scenes action will be shown on the slides.
  - Why some middleware that could be relevant is not used or was not found adequate

- **What tools are involved, and for what explicit communities**
  MD, GVK, OCMG/GPM, GridBench, MARMOT. Other tools (L&B, RM, RB) are hidden behind.

- **Draft a temporal scheme of the demo like the following**
  - 0' Introductory videos and slide
  - 5' Access to MD, Virtual Directory browsing, download medical data to PSS
  - 10' Image manipulation, selection of ROI, LB mesh creation, upload to SE
  - 15' Simulation job submission and monitoring
  - 20' GVK job submission and monitoring
  - 25' Download of GVK results to PSS and rendering
  - 35' End of medical part of presentation, presentation of GPM use by Biomedical team, via MD.
  - 45' End of GPM demo, presentation of GridBench use by Biomedical team, via MD.
  - 55' End of GridBench demo, presentation of MARMOT use by Biomedical team, via MD.
  - 60' End of Demo

- **Indicate clearly the public/community addressed in the DEMO/Presentation (developers, final users, testbed administrators)**
  Final users
- **Provide the numbers that will sustain "benefit of Grid computing":**
  - During the biomedical demo there will be a need for running several simulations and visualizations with different parameters at the same time. An institution having one supercomputer would have to wait for one simulation to finish before running the other. Using distributed grid resources is ideal for covering such peak demands.

G-PM sub-demo

(Received from Roland Wismuller)

- Setup requested:
  - For presentation of slides: Laptop with Windows (PowerPoint), connected to beamer
  - For Demo: Laptop with LINUX (!) and X-Windows, with external mouse, Fast Ethernet or Wlan (ideally: incoming connections possible on non privileged ports), connected to a second projector, with XGA resolution. Java runtime (for MD).
- Brief description: The demo shows the performance monitoring of the WP1.1 blood flow solver. Two instances of the solver will be submitted (to different sites), and their performance will be compared with G-PM. The demo is MPI based and interactive (interactive monitoring).
- Indicate where are the executable, sources, input data, and what output is expected, and give an example of JDL file describing a test run
  - Executable and input files of the application are on SE. OCM-G is installed locally on each site.
  - For this demo, no output files are needed (monitoring data are sent on-line), although the application creates one.
  - JDL-File:

```
Executable = "B_Stream_ocmg.opt";
Arguments = "500 opt --ocmg-appname blood --ocmg-mainsm
8d34a02a:4e21";
StdOutput = "blood.out";
StdError = "blood.err";
InputSandbox = {"B_Stream_ocmg.opt", "opt.bs",
"opt.conf", "opt.decompose"};
```

```
OutputSandbox = {"blood.out",  "blood.err",
"velocity.tar.gz"};
JobType = "MpiCh";
NodeNumber = 4;
VirtualOrganisation = "cg";
Requirements    = other.GlueCEUniqueID ==
"cluster.ui.sav.sk:2119/jobmanager-pbs-workq";
```

- Estimate how many CPUs, at how many sites, and how much Storage and Data Transfer will be needed or would be ideal. In particular indicate speedup issues.
  - In total: 8 or 12 CPUs on 2 sites
  - Storage / Data transfer: about 2 MBytes, plus temporary storage on the WNs
- Indicate the use of CrossGrid middleware and tools in your demo:
  - Release that will be used if specific
    - CG 2.3.5 or higher (Need OCM-G 1.9.6)
  - How the use of middleware will be indicated in the demo and how important it is
    - Demo will use MD (+ RAS, RB, …) and OCM-G plugin to MD (In an urgent need, the demo could also be shown without it)
    - OCM-G, absolutely necessary, but invisible in the demo. It will be indicated in the presentation during the demo.
  - Why some middleware that could be relevant is not used or was not found adequate
  - What tools are involved, and for what explicit communities
    - G-PM tool. Communities: (expert) user of application, application developer
- Draft a temporal scheme of the demo
  - 0' Introductory slides (1-2)
  - 2' Submission of jobs to the testbed from MD in laptop, startup of OCM-G, G-PM
  - 3' Continue showing slides (3-4)
  - 5'-10' displaying and discussing measurement results
- Indicate clearly the public/community addressed in the DEMO/Presentation (developers, final users, testbed administrators)
  - expert users of applications, application developers
- Provide the numbers that will sustain "benefit of Grid computing":
  - Not applicable
  - Regarding the benefit of G-PM: By using G-PM and OCM/G to analyse the performance of the blood flow application, we were able to identify two bottlenecks: (1) a load imbalance problem, and (2) writing the output in ASCII format. By optimizing the data distribution and using a binary output format, the execution time could be reduced by 14% (from 118s to 102s).

FLOODING APPLICATION

(Received from Branislav Simo)

- **Setup requested**
  a laptop for the presentation, connected to a projector, another laptop for the demo, with fast ethernet or wireless connection, connected to a second projector, with XGA resolution
- **Demo description:**
  Demonstration will show application portal used for submitting a workflow of jobs to the grid (both MPI and sequential) that produce binary datasets that are visualized by appropriate tools. Visualized pictures are then viewed in the portal. Chosen resulting datasets can then be registered to replica manager and metadata catalog, and later on can be looked up.

- **Indicate where are the executable, sources, input data, and what output is expected, and give an example of JDL file describing a test run**
  Executables and data are stored at the Storage element at IISAS. Output will be binary datasets and visualization in form of pictures.
- **Estimate how many CPUs, at how many sites, and how much Storage and Data Transfer will be needed or would be ideal. In particular indicate speedup issues.**
  One MPI job will use 4-8 CPUs, MPI class is MPICH-P4, data transfer will be 10s-100s of MBs. Due to high communication overhead of our MPI applications speedup is very small when running on more than 8 nodes.
- **Indicate the use of CrossGrid middleware and tools in your demo:**
  - Release that will be used if specific:
    production testbed
  - How the use of middleware will be indicated in the demo and how important it is
    Middleware will be hidden, but behind-the-scenes action will be shown on the slides.
  - Why some middleware that could be relevant is not used or was not found adequate ?
  - What tools are involved, and for what explicit communities
    Application portal, job submission service, workflow service, metadata service. Other tools (replica manager, resource broker) are hidden behind.
- **Draft a temporal scheme of the demo like the following**
  - 0’ Introductory slides (1-4)
  - 5’ Submission of the workflow to the testbed from portal in the laptop
  - 8’ Monitor that the workflow is running
  - 10’ Show the results of an already finished workflow
  - 13’ Slides describing the metadata registration and searching (1-3)
  - 15’ Register chosen dataset to metadata catalog and then try to look it up
  - 18’ end of presentation (?)

  …
- **Indicate clearly the public/community addressed in the DEMO/Presentation (developers, final users, testbed administrators)**
  final users
- **Provide the numbers that will sustain “benefit of Grid computing”:**
  - During the flood threat or model calibration there will be need for running several simulations with different parameters at the same time. An institution having one supercomputer would have to wait for one simulation to finish before running the other. Using distributed grid resources is ideal for covering such peak demands.

METEO APPLICATION

(Received from Fran Rivera/Carlos Mouriño/Juliusz Gajewski)

- **Setup requested:**
  At least 2 laptops, one for the presentation, connected to a projector, another laptop for the demo, with fast ethernet or wireless connection, connected to a second projector, with XGA resolution

- **Demo description:**
  First the meteo application starts the execution, it is slower than the air pollution, so maybe it should start in the morning. Juliusz Gajewski makes a presentation. Then the air pollution application starts with the input from the meteo. Carlos Mouriño and/or Francisco F. Rivera

make a presentation. Next PPC is used with a presentation by Fran. After that, the results of the simulation are shown.

- **Indicate where are the executable, sources, input data, and what output is expected, and give an example of JDL file describing a test run**

  The source codes are in the CVS repository at FZK. For the Air pollution, the executable and input data are in the Virtual Directory and in some of the Storage Elements of the Testbed. For the PPC the executable is directly accessed from the MD or in the user interfaces of the testbed in /opt/cg/bin/PPCstandalone.sh. The input data are in some of the Storage Elements of the Testbed. For the Air pollution application, the output expected, apart the graphical output into the MD, are the files with the concentration values in all the points of the simulation mesh every time step. For the PPC the output is an interactive graphical Interface. A sample JDL for Air Pollution submission is:

  Executable ="STEMII-VD-viento5.exe";

  JobType ="MPICH";

  NodeNumber =2;

  Arguments ="20041021 0 1";

  StdOutput ="std.out";

  StdError ="std.err";

  OutputSandbox ={"std.out","std.err","ASTEP"};

  InputSandbox ={"STEMII-VD-viento5.exe"};

  Requirements =other.GlueCEStateStatus == "Production";

  Rank =other.GlueCEStateEstimatedResponseTime;

  VirtualOrganisation="cg";

  PPC is a tool that runs locally.

- **Estimate how many CPUs, at how many sites, and how much Storage and Data Transfer will be needed or would be ideal. In particular indicate speedup issues.**
  For the Air Pollution, the ideal number of CPU's to be used is 8. The data to be transferred is about 1.7MB every hour of simulation and about 0.5 Mb every minute of simulation for interactivity issues. PPC is a tool that runs locally.

- **Indicate the use of CrossGrid middleware and tools in your demo:**
  - **Release that will be used if specific:**
    production testbed
  - **How the use of middleware will be indicated in the demo and how important it is**
    In this demo the MD is used by all the applications. PPC uses JIMS Monitoring. Other tools used are not shown in the demo.
  - Why some middleware that could be relevant is not used or was not found adequate ?
  - What tools are involved, and for what explicit communities
    Migrating Desktop, job submission service, PPC. Other tools (replica manager, resource broker) are hidden behind.
- **Draft a temporal scheme of the demo :**
  - -30’ Initiate transfer of files, pre-computation,etc. Submission of Meteo application.

- 0’ Introductory slides by Juliusz about meteo application.
- 12’ Submission of air pollution application by Carlos to the testbed from MD in laptop.
- 15’ Showing slides by Carlos and Francisco about the air pollution application.
- 25’ Use of PPC from MD by Francisco
- 28’ Continue slides about PPC by Francisco
- 38’ Fetch the output, display it in MD for the Meteo and air pollution
- 50’ Fetch the output, display it in MD for the PPC.

- **Indicate clearly the public/community addressed in the DEMO/Presentation (developers, final users, testbed administrators)**
  Mainly for developers and final users.
- **Provide the numbers that will sustain “benefit of Grid computing”:**
  About the Air pollution application:

The use of several processors improves the performance of the simulation, but that processors must be of the same site, because there are a big amount of data to be communicated every iteration. Using several sites we recommend executing several simulations with different parameters, one in each site. See next tables:

nº processors	Total time	Communication time	Speedup
1	2200	-	-
2	1470	420	1.49
3	1106	340	1.99
4	934	294	2.35

Table 1. Execution time and communication time for execute 60 iterations of our program in nodes that belong to the same site.

nº processors	Total time	Communication time
1	2200	19
2	5107	428
3	20212	11372
4	37874	20745

Table 2. Execution time and communication time for execute 60 iterations of our program in nodes that belong to different sites.

HEP / INTERACTIVITY APPLICATION

(Received from David Rodriguez)

- **Setup requested:**
  a laptop for the presentation, connected to a projector, another laptop for the demo, with fast ethernet, connected to a second projector.
- **Demo description:**
  It will show interactive job submission using Migrating Desktop. The demo will be divided into two sections: the example histogramming interactive application (we have both sequential

and MPI versions), and the parallel ANN training application. We will first show the histogramming example, we will submit a job to the testbed using the MD and then show how to execute the interactive plugin in the MD. We will ask for several histograms to be generated, changing in real time the parameters (mean, width). The job will likely run using the glide-in mechanism on top of a batch job. For the ANN application, we will also show how to submit the job using the MD, then we will load the plugin and see how the training advances. We will eventually stop the training, to show interactive clean stop of the application. Furthermore, re-start of random weights will show the "importance" of re-training and first point, while technically indicates the possibility of input interaction.

- **Indicate where are the executable, sources, input data, and what output is expected, and give an example of JDL file describing a test run**
  The executable for the histogramming applications is at IFCA's SE. The executable for the ANN application and the necessary input files (dataset) are also stored at IFCA's SE. The actual data files used by the ANN application are at several storage elements in the testbed, and are acquired by the application using the Replica Manager. The output files are generated at a user chosen location, we usually use again IFCA's SE.
- **Estimate how many CPUs, at how many sites, and how much Storage and Data Transfer will be needed or would be ideal. In particular indicate speedup issues.**
  The histogramming example has no much requirements; it can be run in a single CPU (sequential version), or just in two CPUs in the parallel one. It does not require storage and all the data movement is limited to the interactive input/output streams. In the ANN case, we would like to use at least 4 CPUs, but the application parameters can be adjusted to cope with less. When the application is submitted the executable and three input files shall be sent to the CE. During the execution, the application gets the actual data files using the Replica Manager. The complete dataset is about 40 Megabytes, but it each slave would only get its slice of the data. Furthermore, during the execution the interactive input/output streams are used, although basically for loggin, and it should not be a great amount of data; meanwhile the output files containing the graphical information would be periodically refreshed and load in the MD plugin. These files are very small, just a few Kilobytes.
- **Indicate the use of CrossGrid middleware and tools in your demo:**
  - **Release that will be used if specific:**
    Production testbed.
  - How the use of middleware will be indicated in the demo and how important it is
    We will use the MD for submitting the jobs, we will talk about the interactivity mechanisms included both in the MD and JSS to make possible running our applications. We will also refer to the RB and the glide-in mechanism for priorization of interactive jobs.
  - Why some middleware that could be relevant is not used or was not found adequate
  - What tools are involved, and for what explicit communities
    Job Submission Service, Roaming Access Server, Migrating Desktop, Replica Manager.
- **Draft a temporal scheme of the demo like the following**

Complete scheme:

0-Introduction (Jesus) 3'
3- Histogramming running with steering parameters (David) 5'
4-Complete HEP application running (David+Jesus) 15'
1- JSS flow slide intro (Stefano? Marco?) 5' (1-2 slides)
2- Glide-in (Miquel?) 5' (1-2 slides)

With more details on parts 3 and 4:

- 0' Introductory slides + launching the MD.
- 5' Submit interactive histogramming job. Launch the plugin.
- 8' Ask for histograms using the plugin.
- 10' Stop the job.
- 11' Submit the ANN application. Launch the plugin.
- 14' Show the training error and the signal vs. background graphics.
- 17' Stop the job. Show the output file.
- 18' Conclusions slides.

- **Indicate clearly the public/community addressed in the DEMO/Presentation (developers, final users, testbed administrators)**
  Final users will be physicist using neural networks for analysis of HEP data. We will also address developers willing to adapt interactive applications to a grid environment.
- **Provide the numbers that will sustain "benefit of Grid computing":**
  ANN training time could be dramatically reduced using grid computing resources. The availability of more computing resources at several locations would let a user test different architectures and sets of input variables in parallel, and have all the results in less time that it was needed before for a single run. We have published several papers on the scalability of the parallel ANN training application. Using 64 CPUs we have reduced the user wait time from 8 hours to 8 minutes.

  The interactivity features included in the CrossGrid middleware lets the user control the execution flow of a program running on the grid. For the developer: the application does not need to use any special library to profit from this. On the client side developing a MD plugin for an interactive application is very similar to a normal one, it has the same functionality and adds the corresponding to the interactive input and output streams.