How to Port Applications on Grid: A Short Review

Stefano Cozzini *

CNR-INFM DEMOCRITOS National Simulation Center, Trieste, Italy

Lecture given at the
Joint EU-IndiaGrid/CompChem Grid Tutorial on
Chemical and Material Science Applications
Trieste, 15-18 September 2008

LNS0924002

*cozzini@democritos.it

Abstract

In this paper we will address the problem of how to port scientific applications on Grid infrastructures. This is indeed a complex task: to Grid-enable applications, users have to cope with many different aspects: some are related to the infrastructure (the complexity of Grid architectures and wide spectrum of Grid middleware tools and services) while some others are strictly connected to the application itself (like, for instance, its architecture and the algorithms used). The goal of this paper is to guide interested scientific users through the important stages of implementing applications on Grid infrastructures. Our presentation is general but emphasis will be given toward EGEE/gLite infrastructures. We discuss in detail our succesful methodology adopted in the context of the EU-IndiaGrid project. A discussion about the lessons learned so far, the important challenges and their potential solutions are given in the Conclusion.

Contents

1 Introduction

The task to enable an application to benefit from Grid computational environment is not a straightforward procedure for almost all kinds of Grid infrastructures. Despite the many efforts done in the last years still enabling a simple application can represent a formidable mission requiring a high level of expertise and involving a considerable amount of work. There are infact important barriers to overcome due to the nature of the Grid itself. The most limiting factor that stands out it is probably the complexity associated with Grids both from a programmatic point of view as well as from technology, management and deployment. Scientific users face several layers of complexity when porting applications to a Grid computing environment composed by distributed networked systems. As an example, let us consider the complexity at hardware layer. This is given by the heterogeneity of the resources on the infrastructure: we can range from a loosely coupled system composed by multicore cpus up to high end HPC clusters with tightly coupled processors. To exploit these resources in an efficient way applications have to be adjusted and sometimes heavily modified.

Moreover, it has to take into account the specific requirements posed by the many components of the Grid services architecture. There are four general classes of Grid services: security, resource management, information and data management services; each of them have to be considered in the porting procedure.

It is impossible to review and present, in a single article, all the issues related to application porting and development on Grid infrastructures. For a more general review on this topic references [2] [3] are recommended. We restrict our discussion here on a few general concepts with the aim to highlight some lessons learned so far and to give some general guidelines. Our analysis relies on the experiences accumulated thus far interacting with a certain number of scientific communities, which were given the opportunity of using Grid infrastructures based on gLite/EGEE middleware.

The remainder of this paper is organized as follows: in the next section we will introduce some definitions of the most used term in this context. We then examine the major issues that can impact on the porting procedure in section three. In section four we discuss in detail the Grid-enabling procedure focusing on our EU-IndiaGrid experience. In section five we briefly review some approaches and associated tools that can be used to facilitate the port activities. We conclude our review presenting our lesson learned on the issues

still present in the Grid enabling procedures.

2 Some definitions

This section attempts to list definitions for terms frequently encountered throughout this paper. It is important, in our view, to define precisely a few terms often used in this context with some approximation. Our definitions are working definitions which are probably not universally applicable nor rigorous. However we feel that, while simple and basic, they are enough to facilitate our discussion and to avoid some misunderstanding and imprecisions.

We first define the following terms:

- **Grid infrastructure:** *A distributed infrastructure of computation and storage resources, which can be used by users in a transparent way (i.e. without need to know about the location of the resources etc.)*
- **Grid-enabling procedure:** *The procedure that allows a scientific (or generic) user to solve her scientific computational problem by means of a Grid infrastructure*
- **application:** *A collection of work items to solve a certain problem or to achieve desired results using a Grid infrastructure. (...) In other words, a Grid application may consist of a number of jobs that together fulfill the whole task.*

The definition of application is taken from reference [3]; keeping in mind such definition we can make the following observations:

- An application is not only a software application but rather a complex computational problem to be solved in a Grid environment.
- A successful story of "Grid-enabled applications" refer not just to the porting of some scientific software on the Grid infrastructure but instead to a real exploitation of such software by a scientific community.
- Each application is unique in that it tries to solve a specific computational problem clearly stated by a scientific user or a scientific community. In this context we can use as a synonym for application the term **computer experiment**.

- The same scientific software package can be used in many different computer experiments and, in principle, each of them should require its own "Grid-enabling" procedure.

We now define the actors of the Grid-enabling procedure introducing the following definitions:

- **Grid users or user communities:** *People interested in using a Grid infrastructure. They usually have an application and/or computational problem to solve. They are not interested in technical details about the infrastructure and the concepts behind.*

- **Grid providers:** *People who setup and manage the infrastructure. Generally computer scientists and technical people with little or no experience in science. They have a deep knowledge about the infrastructure.*

- **Grid developers and/or programmers:** *People that write and/or use scientific codes and want to enable them on the Grid infrastructure. They are familiar with either the Grid infrastructure and its technicalities or the scientific package they are dealing with.*

A few more observations follows from the above definitions:

- The first two classes (users and providers) rarely overlap and they interact on the basis of some agreement. Grid providers "sell" the Grid infrastructure. Users try to "buy" it in order to use it for scientific research. The way the "sell/buy" relationship is handled depends upon the Grid infrastructure. In EGEE the end-user just joins one Virtual Organization (VO) and can then use the available resources according to the rules of the virtual organization itself. In some VO scientists have to contribute their own computational resources to get involved while in others they can just use the infrastructure without any kind of fee, but also without any service level agreement: i.e. just using them on best effort basis.

- There could be a communication problem between users and Grid providers, due to different scientific background, different motivation and different goals. This should not be underestimated.

- The third class of people involved in the process is a little bit more fuzzy: sometimes Grid developers or application programmers are belonging to scientific communities, other times have to be accounted for as Grid providers.

We finally give our definition of **Grid environment:** *a Grid infrastructure complemented by people (users and providers and developers) and with some applications enabled or to be enabled on the top of it.*

We stress here the fact that we can define a Grid environment where the three major components are all present: the infrastructure itself, the people involved and the computational problem to be solved.

3 What is impacting in porting applications to Grid?

There are several aspects that can play an important role in the Grid-enabling procedure of a scientific application. Among the most important ones we need to examine are the infrastructure itself, the data management and application architecture issues. These three major features could determine the success of porting and the resulting performance of the Grid application. Users should therefore be made aware of the importance of the role played by each of them. In the following subsections we just summarize the extensive discussions presented in references [2] and [3] about the role of the three features considered above.

3.1 Applications and the Grid infrastructure

The Grid infrastructure forms the core foundation for successful Grid applications. A Grid infrastructure is a complex combination of a number of services and resources; these should be clearly stated and identified for the specific problem to be solved on it. As already pointed out in the introduction the four major classes of services offered by a Grid infrastructure are security, resource management, information and data management. Each of these kinds of services can affect the application architecture, its design, deployment, and performance. Therefore, the user has to go through the process of matching the application (structure and requirements) with these components of the Grid infrastructure, as described in detail in reference [3]. This is a rather complex task that can be correctly addressed only if

users and application developers have good knowledge about the Grid services available on the infrastructure they are working on. The situation is even more complex when a user community plans to use more than one Grid infrastructure: in this case the matching procedure should be repeated for each infrastructure, due to the fact that quite often Grid services are different on different infrastructures.

There are, therefore, many efforts in the Grid community to develop tools and standards which simplify the above procedure by enabling the application to make easy use of the Grid middleware services. The Open Grid Forum, an international committee that coordinates standardization of Grid middleware and architectures proposed recently a Simple API for Grid Applications (SAGA) [4].

The goal of SAGA is two-fold:

1. Provide a simple API that can be used with much less effort compared to the interfaces of existing Grid middleware.

2. Provide a standardized, portable, common interface for the various Grid middleware systems.

SAGA should simplify the development of Grid-enabled applications, even for scientists without any background in computer science or Grid computing. Using SAGA will also guarantee a sufficiently high-level of abstraction so as to be able to be independent from the diverse and dynamic Grid environments. SAGA is on the road to becoming a community standard and many important Grid infrastructures are offering now a SGA API: EGEE/gLite middleware is actively working toward it but still does not implement fully the SAGA standard. For this reason we do not yet have direct experience with the SAGA approach. We also note that using an API is not always an easy task in case of legacy code and/or well structured packages: this could also limit the adoption of the standard.

3.2 Data management

Data have a strong influence on many aspects of the design and deployment of a scientific application. A careful analysis of all data involved in the application can determine whether an application deserves to be ported or not on some specific kind of Grid infrastructure. There are many issues with respect to data handling which might affect the process of Grid-enabling

an application; they can range from the simplest ones, like data types and size of input and output to the more complex problems about security and privacy of sensible data in case of specific applications.

For a more in-depth general discussion of data management related tasks and issues we refer to reference [3]. With respect to EGEE/gLite Grid infrastructure, we note that such infrastructure provides several important advanced tools for data management of very large datasets. These services are developed to fulfill the strong requirements coming from its largest scientific community: the High Energy Physicists involved in Large Hadron Collider (LHC) experiments. These services are now complemented by other advanced services like, for instance, data encryption and privacy required by some other user communities [5].

3.3 Application architecture issues

We examine here the role that the application program itself plays in the Grid-enabling procedure. In this context two different approaches to deploy an application on a Grid are possible. The first approach is when developers try to enable an existing application code or set of codes on a Grid infrastructure. We define such an approach *a Grid application porting scenario*. The second one is when developers start from scratch and design and develop a new application program with the Grid (or with a certain type of Grid infrastructure) in mind. We refer to this approach as *a Grid application developing scenario*. In both scenarios, the effort of deploying an application can be huge and requires a careful analysis beforehand on many different aspects. A partial list of such aspects comprises:

- the user requirements for running the application on a Grid (e.g. cost, time);
- application type (e.g. compute or data intensive);
- application architecture and algorithms;
- application components and how they interact (e.g. loosely or tightly coupled, or workflows);
- what is the best way to map the application onto a Grid;
- which is the best suited Grid architecture to run the application in an optimally performing way.

The two scenarios pose however different challenges in the implementation phase; in the first case (application porting) we generally deal with an application that often, in the past, has been developed and optimized with a specific computer architecture in mind. As a consequence we are facing a monolithic application not always flexible enough to be easily executed on a different infrastructure. In some other cases the application with its inherent numerical algorithms is optimally mapped onto a specific architecture. This again reduces the chance to exploit it easily on different and more complex infrastructures. In the second case (application developing) difficulties are of other kinds. We cite here just two of them: the lack of common standard to interact with Grid services (as already pointed out in the previous section) and the adoption of the correct Grid enabling strategy to be implemented to guarantee at the same time the exploitation of the underlying infrastructure and the portability among different infrastructures.

Our own experience is mainly related with the first scenario: we generally face the problem of porting already established scientific software that we adopted to run (in some cases successfully, in some others no) onto the Grid. Difficulties of this process depend also on which kind of code you are dealing with. Based on our experience we classify three classes of codes:

- **Home made codes:** In this case the Grid enabling procedure should be easy because users are supposed to know everything about the application and its requirements. For some applications belonging to this class we were also able to rather develop a Grid wrapping application instead of porting the code itself in a sort of mixing of the two scenarios mentioned above.

- **Some well-known packages used with slight modification/ adaptations:** An increasing number of scientific communities nowadays rely upon third-party software (including both open source community science projects and commercial applications). Porting such kind of software is complex because of the limited knowledge of it by the actual users: it is, however, relatively easy to get information if the package is open source or at least widely adopted by the scientific community of reference. The most important aspect for this class of codes is that users often lack the specialized expertise to deal with the complex Grid environments.

- **Black box application (a.k.a. legacy application):** This is the most complex case because information about the code are usually not available. The Grid-enabling procedure can be very time-consuming and therefore a careful analysis about pro and cons should be performed to understand if the porting deserves to be done. For this class of codes, as we will discuss later, a Web browser-based portal could be the correct approach.

Finally we note that in many cases today, users are looking to identify specific applications that they could adapt quickly to a Grid environment to gain immediate benefits and to gain experience and knowledge around Grid computing as well. Developing a complete Grid application from scratch is at the moment not so common in the scientific communities we interacted with.

4 The EU-IndiaGrid Grid-enabling procedure

A short guideline on how to enable applications on Grid infrastructure has been drawn up by the EU-IndiaGrid project, [6], based on the experience collected within different user communities. The document has the goal to offer a first support to users interested in using the EU-IndiaGrid infrastructure to its best. In such a document we propose that a successful procedure for Grid-enabling application should be performed in the following five major steps:

- **Initial Step**: Awareness of Grid computing opportunities/analysis of the computational requirements
- **Step 1**: Technical deployment on the infrastructure
- **Step 2**: Benchmarking procedures and assessment of the efficiency
- **Step 3**: Production runs and final evaluation
- **Final Step**: Dissemination of the results among peers.

At the end of each intermediate step an evaluation procedure takes place to understand if the action should move to the next step, should be stopped with the execution of the final step, or should go back and repeat the steps previously done. The flowchart is reported in figure 1. It is worth to note

that such procedure was elaborated keeping in mind the point of view of the users: this is why we insist that the procedure should have in any case (even if it is stopped after the initial step) a final dissemination phase. The results (even negative) can be of great importance for other users as initial input when starting the Grid-enabling procedure.

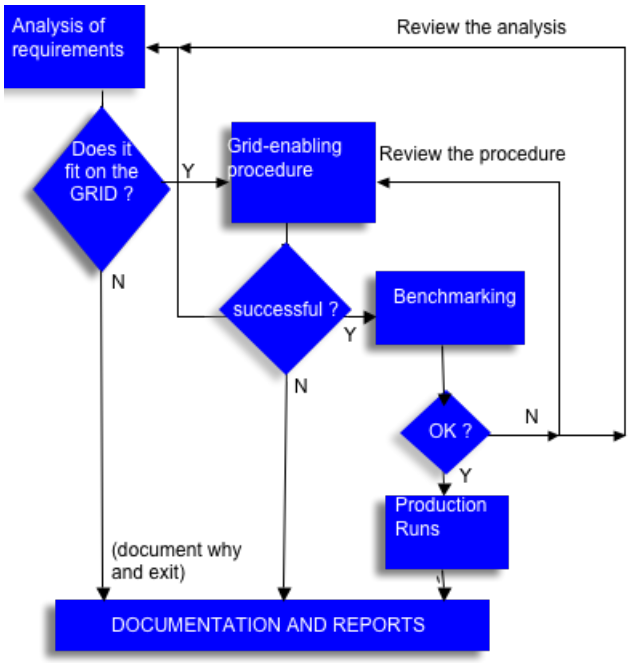

Figure 1: The porting procedure and its major steps.

We review in the next subsections each of the steps discussing some important aspects at the light of the ideas and concepts presented in the previous sections.

4.1 Initial step: Awareness of Grid computing opportunities/analysis of the computational requirement

The initial step is actually a preliminary one in the sense that it does not require any specific interaction with Grid infrastructure. The goal of this step is to make Grid newcomers understand if there is any advantage for their application in the adoption of Grid technology. This is a crucial point:

in many cases, based on our experience, we noticed that interested scientists have no precise understanding of which infrastructure provides which kinds of computational resources and functionality, and how this functionality relates to their particular needs. The user community, therefore, must be prepared to spend some time on mastering the technology and adapting it to their own application if this is feasible. There are, in fact, chances that the application they wanted to port does not fit at all the resources offered by the Grid infrastructure they are looking at. For example, some personal productivity applications are tightly coupled with a user's interface and do not consume a large amount of computing resources. Running them on a Grid may not provide significant benefits. Grid infrastructure opportunities and advantages should be clearly presented by Grid providers and carefully evaluated by user communities when they decide to port new applications on the Grid.

Once the users are aware of Grid potentialities, they should then analyze in detail the applications and their requirements to understand how the application could be ported and/or developed to obtain benefits from a Grid infrastructure. In this phase it could be extremely useful to fill in some predefined questionnaire which provides a guideline for all aspects to be taken into account for an application. Several EGEE related projects developed such tools (see for instance [7]). We do observe here that to be useful the questionnaire should be filled in strict collaboration with Grid providers. Inexperienced Grid users generally do not provide correct and/or sufficient detailed information.

4.2 Step 1: Technical deployment on the infrastructure

This is an operative deployment step and should be done in strict collaboration with application users/developers: the existent application will be enabled on the infrastructure with respect to the specific requirements and restrictions (e.g. OS, network capacity, computing/storage resources, etc.) identified in the previous step. In this phase the software codes part of the application will be adopted and modified in order to make them run on the Grid infrastructures.

So far we have performed a few different kinds of technical deployment:

1. Simple serial porting of an application and setup of a few simple tools (scripts etc.) to run the computational experiment (generally a parametric study)

2. Porting of parallel applications with MPI: This again is quite simple in principle but present MPI limitation on the infrastructure makes this strategy not very efficient.

3. Porting of complex codes/packages that requires important changes on the original way to run the application and the experiment associated.

4. Client/Server mechanisms to run embarrassingly or loosely coupled applications/experiments.

This is not an exhaustive list and other approaches/strategies can be classified and used. For each of the four class of codes mentioned above we made use of specific tools available to speed up this phase. In some cases we developed "ad hoc" tools on the fly as well. Some of them were then easily adopted in subsequent porting experiences. A general discussion about general strategies and associated tools that can be used in this step is presented in section five.

4.3 Step 2: Benchmarking procedures and assessment of the efficiency

This step is one of the most important ones and, as far as we noticed many times, is not attended with due care. In this phase one should evaluate the convenience of executing the computer experiment associated with the enabled applications on the Grid infrastructure with respect to other computational facilities available. It should also asses the overall efficiency of the Grid-enabling procedure in terms of absolute performance of the code (i.e. Is the code, on average, running faster on the CPUs available on the Grid?), in terms of latency of the infrastructure (how much does it take, on average, to get the work running on the Grid/how much does it take to transfer data, etc. etc.). All these questions should be answered by establishing a care benchmarking procedure that should be set-up by applications users/developers with the help of Grid providers.

At the end of this phase it should be evident if the procedure should go further and start the production phase (step 3) or just stop. A negative result obtained in this phase allows a fine tuning of the whole procedure: it could in fact give important indications about a wrong approach in phase two/or some wrong assumptions in the initial step. It could also spot out some weaknesses of the Grid services (i.e. lacking of clear and standard interfaces) that should be taken into account by Grid providers.

4.4 Step 3: Production runs and final evaluation

Once the previous phases are successfully completed the computer experiments can finally be performed at production level in the Grid environment. This step should be managed almost independently by application developers/users with limited support by Grid providers. Grid provider support could be confined at the monitoring phase just to check the actual behaviour of experiments getting executed on the Grid and the responsiveness of the infrastructure. These monitoring procedures again offer a feedback to both the infrastructure and the application.

4.5 Final Step: Dissemination of the results among peers

The final step is intended to gather and organize information collected in any single step of the procedure with the aim of providing organized and documented experiences, which can be shared among fellows and colleagues by application developers/users. Even in case of failure of the porting procedure this documentation and dissemination phase should be completed in order to give information about the weak points encountered at both application and infrastructure level. The documentation effort should be done by both Grid providers and users involved in the procedure keeping in mind that the final audience is formed mainly by application users.

5 Strategies and tools to deploy application on the Grid

In this section we focus on some strategies used for the technical deployment on the Grid infrastructures. There are several classifications of such strategies and approaches in recent literature (see for instance [2] [8] [9] and reference therein) dedicated to application porting. In the following subsections we briefly present some of the most important strategies commenting and evaluating them based on our own direct experience.

We first point out however that a strategy should be chosen with the goal to increase the usability of the Grid infrastructure by the scientific users. In general, usability of a system is greatly increased if the latter is provided by an interface (that could be either high level or not) that supports the users natural way of working. It is therefore important to understand clearly which is the preferred work environment of the users in order to correctly choose

among different approaches. The key concept, in this case, is the integration of Grid enabled application into the prevalent working environment of the scientific users in a minimally disruptive way.

5.1 Command line approach and "ad hoc" tools

This is one of the most widespread approaches and is generally the preferred one by users coming from HPC environment. Here the application is ported and simply enabled to run on the Grid by means of simple scripts that can wrap-around the original software and offer the needed control functionalities of the jobs submitted to the Grid. Scientific users coming from HPC environment where job submission is generally done through command line interface and, with the help of simple script, will find moderate difficulties in using the command line. There is, of course, some overhead associated with the learning process of a new computational infrastructure but if the preliminary analysis (i.e. the initial step of the Grid enabling procedure) has been performed correctly this should be relatively modest.

This approach implies however that users have to create their own scripts and tools. They should, therefore, be aware of the potentials and drawbacks of scripting languages (perl python bash etc.). One of the potentials offered by this approach is the high level of flexibility: the power of Unix command line interface is available here and can be easily exploited also in the Grid environment. This approach was widely used in our activities, ranging from simple scripts to managing job submission up to the development of some command line tools to solve some specific issues on EGEE/gLite Grid infrastructures. We cite here, as an example, the reserve_smp_nodes [10] tool that allows a user to reserve multicore nodes for smp and/or parallel computations, a feature not yet implemented by gLite middleware.

5.2 Grid workflow environments

A different approach can be considered when the application program consists of a set of components or modules which interact with each other. In such cases data management is essential in that it often connects the different modules. The combined use of computational tasks and data management tasks in conjunction with numerous control functionalities leads to directed acyclic graphs (DAGs) that represent the overall scientific workflow. Most Grid middleware systems support complex scientific workflows by means of a middleware layer, the so-called workflow environment/system. Such layer

maps the application modules onto the different resources in the Grid. Many workflow environments have been made available in recent years for Grids: the most interesting ones are Triana [11] and Taverna [12]. They generally provide a Graphical User Interface for the definition of DAGs since defining complex scientific workflows with XML configuration files and command line interfaces is rather cumbersome. We have little direct experience with such tools, mainly due to the fact that the scientific communities we interact with do not need such kind of tools.

5.3 Portals

A Grid portal can be regarded as a Web-based portal able to expose Grid services through a browser to allow users transparent access to resources available on the infrastructure. The main goal of a Grid portal is to hide the details and complexity of the underlying Grid infrastructure from the user. This should improve usability and utilization of the Grid by users not familiar with command line interface. Usability should be even more enhanced by greatly simplifying the use of Grid-enabled applications through a user-friendly interface.

In addition Grid-enabled portals can make Grid resources available to users wherever they have access to a web browser running on the Internet without the need to download or install specialized software or worry about setting up networks, firewalls, and port policies.

Hence, Grid-enabled portals have been proven to be effective mechanisms for exposing computing resources and distributed systems to general user communities without forcing them to deal with the complexities of the underlying infrastructure.

Grid portals have become popular within some scientific research communities. One of the most interesting and successful examples in the area of life science is the MyGrid portal [13]. It provides access to bioinformatics tools running on a back-end Grid infrastructure. Scientific portals are usually being developed inside specific research projects, to offer specific applications and services satisfying particular research application areas. We defined such portals as vertical portals. In order to rapidly build such vertical and specialized Grid portals several more generic toolkits and frameworks have been developed. Here we cite for instance, Enginframe wich we recently used in some porting experiments and the P-grade [14] portal we used in order to develope our own portal for the EGrid research project [15].

5.4 Interoperability

This last approach should be considered when the complexity of applications requires a huge demand of computational resources of different quality and kind. Such diversity of resources is not always available on the same Grid infrastructure. The approach is then to consider already Grid-enabled applications on some kind of infrastructure and make them work on different ones in a transparent way for the final user. We call this approach the application level interoperability. The challenge is to develop, in the deployment phase, a methodology to allow the application to transparently access different kinds of resources and services. Nowadays, cases where such an approach is needed are getting more and more frequent; we discuss in the following two cases within the EU-IndiaGrid project where the application level interoperability is needed. In the first case, interoperability is needed because the size of the computational experiment to be performed is so large that users need to access as many as possible computational resources; therefore it is very useful to be able to recruit them across different Grid infrastructures. Applications that require such a large number of resources are generally quite insensitive about the quality and/or the kind of such resources: the amount of tasks assigned can be calibrated on the basis of the quality of resources and the overall load can be kept well balanced among the different resources. This is indeed the case of the Bemuse application aimed at tacking the protein folding problem and described in detail in a specific contribution in this volume.

The other case is given by computational problems which require to be solved on different Grids just because different Grids can provide different kind of resources. In this case, on different computational resources different computational experiments are performed using the same application. Here interoperability will not guarantee a larger experiment but permits different types of experiments. The particular example application is related to quantum atomistic simulation using the different scales of systems available in EGEE and DEISA infrastructure. The idea is to run the smallest simulations on EGEE smp resources while the largest ones can be executed on DEISA large HPC infrastructure using the same scientific package.

The two case studies reported above show clearly the need to use different Grid environments; this implies that Grid services should be the same on different Grid infrastructures, or better, transparency should be provided for both low level and high level services. The challenge here is to provide gate-

ways and tools which allow to create and manage workflows where resources of different nature belonging to different infrastructure can be intermixed transparently.

6 Conclusion: Lessons learned

We conclude our review on the subject presenting a checklist of important aspects which should be taken into account in the process to Grid enabling an application. For each item of the list, built upon our experience, we report about the lessons learned and analyzed/underline important open issues.

- **Item 1: Motivation**

  Any successful porting activity should rely on a strong motivation. Learning new tools and mastering technical difficulties can be done only if a clear motivation is behind. We learned that the driving force to motivate a scientific group is the desperate need of computational resources. All the attempts to port scientific activities of groups with no real need of additional resources is destinated to fail in front of the first difficulties encountered. This is an important issue still present: too often a *build it and they will come approach* is used by Grid providers in Grid initiatives. This simply does not work. User communities should actually drive the building process of a Grid infrastructure by clearly defining their requirements in order to ensure the resulting infrastructure is of general interest.

- **Item 2: Involvement**

  The user's point-of-view is of paramount importance; a successful porting experience of an application should be driven by users. This aspect is not always taken into consideration from Grid providers or Grid developers, when they do not belong to the user community. So the role of human interaction is, in this case, fundamental and sometimes underestimated. We learned also to keep in mind the difference in background among different classes of people involved in the porting procedure. We realized that there is a communication problem among them: the problem could be alleviated if from the very beginning Grid providers establish a strong communication/participation channel towards the leaders of the user community to engage. The focus should be on understanding the user community and their needs and then

adapt as much as possible the infrastructure to such needs and not vice versa.

- **Item 3: Training**

  A crucial role is played by training. Specific training and tutorials are always needed to make users and application developers aware of the potentiality and the drawback of Grid infrastructure and services. The best approach we observed is when developers/users can be hosted in Grid providers teams for at least a couple of weeks. In such training activities, trainees can benefit of the learning-by-doing approach and of the close contact with the team, and discuss their progresses around the clock with them. Training will therefore be associated with the analysis of the computational requirements, which requires a strong cooperation among users and Grid providers. The learning process is on both sides: users should learn how to use the Grid while Grid providers should learn the details of the application and its usage in order to identify the right technical approach to enable the application and the computational experiments associated.

- **Item 4: Dissemination**

  We also learned the importance of dissemination about peers: "word of mouth" mechanism is of great importance and this is the typical way dissemination happens within scientific user community. Interested communities should be encouraged to interact with early Grid adopters which already successfully ported applications and learn directly from them and from their experience. This will help potential users in understanding whether the Grid infrastructure could be of some help for their computational needs and minimize the risk to misunderstand the difficulties of the porting procedure.

- **Item 5: Reliability**

  We experienced quite often that there is a lack of stability and reliability in the resources belonging to EGEE infrastructure. Several users were not able to start real production on the infrastructure due to this problem and were disappointed by the waste of time and efforts. Moreover, a lot of the Grid tools currently promoted are really intended for research and demonstrations and not for scientific production. Sometimes significant efforts should be done to make them

suitable for large-scale production usage by large and variegate scientific user communities. Again, this could be disappointing especially when a Grid tool, used in production just does not provide what it promised in the developing stages.

We finally observe that the successful porting of an application to a Grid environment highly depends on the smooth interaction of three key elements which define the environment, the infrastructure, the problem and the most important factor, people around them.

References

[1] Grid Initiatives: Lessons Learned and Recommendations Version 2.0

[2] International Journal of Grid and High Performance Computing, 1(1), 55-77, January-March 2009

[3] B. Jacob, L. Ferreira, N. Bieberstein, C. Gilzean, J.-Y. Girard, R. Strachowski and S. Yu, (2003). Enabling applications for Grid computing with Globus. IBM Redbook, retrieved from www.redbooks.ibm.com/abstracts/sg246936.html?Open

[4] T. Goodale, S. Jha, H. Kaiser, T. Kielmann, P. Kleijer, A. Merzky, J. Shalf, and C. Smith, (2008). A Simple API for Grid Applications (SAGA). Grid Forum Document GFD.90, Open Grid Forum

[5] See for instance the EU-IndiaGrid implementation of secure storage

[6] The EU-IndiaGrid project

[7] https://secure.um.edu.mt/eumedgrid/questionnaire/wp4/

[8] M. Riedel, A. Streit, F. Wolf, T. Lippert and D. Kranzlmüller, (2008). Classification of Different Approaches for e-Science Applications in Next Generation Computing Infrastructures IEEE Fourth International Conference on e-Science, eScience '08, Indianapolis, Indiana, USA. - IEEE, 2008. - 978-1-4244-3380-3. - S. 198 - 205 DOI: 10.1109/eScience.2008.56

[9] Cristian Mateos, Alejandro Zunino and Marcelo Campo, (2008). A survey on approaches to gridification, SOFTWARE PRACTICE AND EXPERIENCE Softw. Pract. Exper.; 38:523-556

[10] http://euindia.ictp.it/grid-tools-and-utilities/reserve-smp-nodes-1

[11] The Triana Project. Retrieved from www.trianacode.org/

[12] The Taverna Workbench 1.7. Retrieved from http://taverna.sourceforge.net/

[13] Retrieved from www.mygrid.org.uk

[14] Retrieved http://portal.p-grade.hu/

[15] https://portale.egrid.it:8443/gridsphere/gridsphere

[16] Pag. 52 and the following in the EU-IndiaGrid Deliverable 4.3 available at www.euindiagrid.eu