On the Structuring of a Molecular Simulator as a Grid Service

O. Gervasi\textsuperscript{1}\thanks{osvaldo@unipg.it}, C. Manuali\textsuperscript{1\dagger}\thanks{carlo@unipg.it}, A. Lagana\textsuperscript{2\ddagger}\thanks{lag@dyn.unipg.it} and A. Costantini\textsuperscript{2§}\thanks{alex@dyn.unipg.it}

\textsuperscript{1} Department of Mathematics and Informatics, University of Perugia, Perugia, Italy
\textsuperscript{2} Department of Chemistry, University of Perugia, Perugia, Italy

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

LNS0924005
Abstract

Computer design of processes and materials which have become nowadays indispensable for several technological and environmental advances are often based on the calculation of the efficiency of chemical reactions. The possibility of offering as a Service on the Grid the calculation of the reactive properties of the intervening elementary processes is of paramount importance. To cope with this demand we have established a Virtual Organization, called COMCHEM, committed to deploy on the EGEE production Grid the relevant applications. As a case study we consider here some atom diatom reactive scattering programs.
1 Introduction

The pervasive development of Grid infrastructures has stimulated the growth of specific scientific organizations interested in fostering the design and implementation of their applications on distributed platforms [1]. This is also the case of scientists operating in the field of Molecular and Matter Sciences and Technologies (MMST) who have recently established a Virtual Organization (VO) called COMPCHEM [2] relying on the Grid infrastructure of the EGEE European project [3]. The main task of COMPCHEM is at present the design and development of a sufficiently general Grid Empowered Molecular Simulator (GEMS) derived from a Simulator of Molecular Beam Experiments [4]. GEMS is, in fact, aimed at providing a computational engine devoted to the calculation of molecular structures and processes from first principles and designed to represent a building block of the realistic MMST simulations of some natural phenomena and modern technologies.

In this lecture we give a detailed account of the formation and evolution of the COMPCHEM VO and of the implementation on the Grid of the basic version of GEMS. On top of that we describe also the evolution of GEMS along two generalization lines. The first line is concerned with its extension into a more complex procedure in which some components can be either chained or alternated. Here, we consider, as an example, the case in which the original classical mechanics treatment (QCT) is alternated by the quantum mechanics (QM) one for dealing with the atom diatom reactive processes whose Potential Energy Surfaces (PESs) are available from the literature. The second line is concerned with the exploitation of the above mentioned articulation of GEMS to implement a Credit System (CRES) [5] rooted on procedures using Web Services technologies implemented on the Grid.

Accordingly, the paper is articulated as follows: in section 2 the characteristics of the COMPCHEM VO are presented; in section 3 the structure of the generalized version of GEMS is illustrated; in section 4 an extension of GEMS to quantum dynamics approaches is described; in section 5 an improved Grid distribution procedure is analyzed; in section 6 the structuring of GEMS in terms of Web Services is discussed and in section 7 an application of the resulting new prototype Framework is made.
2 The COMPCHEM Virtual Organization

As already mentioned, the COMPCHEM VO has been established by building on top of the Actions D23 (Metachem) [6] and D37 (GridChem) [7] of the European COST Chemistry domain and of some National Grid projects\(^1\) aimed at stimulating and coordinating the activities of the MMST community within the EGEE project. COMPCHEM is part of the NA4 generic application package of EGEE and plans to become a pillar of the MMST Specialized Support Centre (SSC) and a key component of some National Grid Initiatives (NGIs).

<table>
<thead>
<tr>
<th>Membership level</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. User (temporary pre-membership)</td>
<td><em>Passive</em>: Run a program implemented by other members of the VO.</td>
</tr>
<tr>
<td></td>
<td><em>Active</em>: Implement at least one program for personal use.</td>
</tr>
<tr>
<td>2. Software Provider</td>
<td><em>Passive</em>: Implement at least one program for use by other members.</td>
</tr>
<tr>
<td></td>
<td><em>Active</em>: Interactive management of the implemented program for collaborative usage.</td>
</tr>
<tr>
<td>3. Grid Deployer</td>
<td><em>Passive</em>: Confer to the Grid infrastructure at least a small cluster of nodes.</td>
</tr>
<tr>
<td></td>
<td><em>Active</em>: Operates above the minimal level as support for the Grid deployment and management.</td>
</tr>
<tr>
<td>4. Stakeholder</td>
<td>Take part to the development and the management of the VO</td>
</tr>
</tbody>
</table>

COMPCHEM is a VO articulated at various levels (see Table 1). The lowest or entry level is the user one (see item 1 of Table 1) bearing a temporary pre-membership. Scientists belonging to this level (*passive User*) get access to the software and hardware made available by the VO. This means that a COMPCHEM passive User can carry out massive computational campaigns using all the programs already implemented on the EGEE Grid for that purpose by relying on more than 25 sites and more than 8000 processors. Some scientists consider this already a good deal in return for the effort spent to get a Grid Certificate and switching into a Grid operational modality. As a matter of fact, this is indeed the case if one considers

\(^1\)Like the two Italian projects MIUR FIRB Grid.it project (RBNE01KNFP) on High performance Grid Platforms and Tools and MIUR CNR Strategic Project L 499/97-2000 on High performance Distributed Enabling Platforms.
that a computational scientist usually finds it difficult to access an equivalent amount of computing power without filing an application and trying to convince a committee of the scientific validity of the related computational project. The adoption of a Grid operational modality pays off even more when the users implement their own programs for personal use and step up, so far, to the level of active User.

The real step up, however, takes place when the users move to the status of permanent member at level 2 (see item 2 of Table 1). At this level they can act as passive Software Provider by committing themselves to implement and maintain some programs for usage by other members of the VO. This means that they need to provide a validated version of some codes and to implement them on the Grid for shared usage. For this purpose the VO offers both a certain level of internal support to develop the competences necessary to restructure the codes and the help of a central assistance service (Grid User Support Service) to facilitate the exploitation of the key features of the Grid platform.

The existence of a Software Provider level is an appealing feature of the COMPCHEM VO especially for the users exploiting the possibility of becoming an active Software Provider and enable their own software to work in a collaborative fashion (though, as we shall discuss in more detail later on, this feature is still in an experimental phase) with the programs of other members. This allows the members of COMPCHEM to keep pre-existing groups of interest gathered to take care of a given package and/or to form new ones (permanent or temporary) aimed at building larger and more complex simulations meant to be available for shared usage. As we shall discuss in more detail later on, this type of collaborative endeavour is one of the most strategic assets of COMPCHEM because it stimulates a great deal of research and a continuous improvement of the software patrimony. It prompts, in fact, not only the validation of stable and easy-to-use versions of the programs but also the development of the necessary support, documentation and maintenance procedures (including the proper handling of trademark and commercial obligations for commercial packages).

This has also shown to be a crucial cross point of the COMPCHEM VO mission. It has motivated, in fact, the members of the VO to structure the programs as services on the Web and has stimulated the introduction of some forms of Grid economy that has materialized in the already mentioned Credit System called CRES [5]. In CRES the COMPCHEM members are rewarded for the work done on behalf of the organization by being assigned a certain
quantity of (Virtual) Credits to be redeemed via a preferential utilization of the resources (including the financial ones) of the VO. Such development, in addition to leveraging on collaboration, stimulates also a certain extent of competition among the members to produce innovative services and improve the existing ones.

CRES is also a means to stimulate the COMPCHEM members to further step up their membership and to contribute to the infrastructure development (Grid Deployer) by conferring to the VO some of their computing resources (passive) and taking care of their deployment (active) on the Grid (see item 3 of Table 1). At later stages of the VO evolution a higher level of membership (Stakeholder) is also foreseen (see item 4 of Table 1) for members strongly committed to take care of its global management.

3 The Grid implementation of the original version of GEMS

GEMS is, as already mentioned, a generalization of SIMBEX [4] which is a package specialized in simulating the measurements of atom diatom single collision experiments (like those carried out on the crossed molecular beam machine of the Department of Chemistry of Perugia) [8] by performing three dimensional (3D) reactive scattering calculations for atom diatom systems using QCT techniques. The QCT techniques used by GEMS are concerned with the integration of the classical equations of motion relating the time derivates of positions and momenta of the various atoms of the system to the partial derivates of the Hamiltonian.

The up-to-date most significant utilization of the 3D reactive scattering computational engine of GEMS, has been concerned with the study of the N + N2 reaction to calculate the related thermal rate coefficients out of the corresponding state to state reactive probabilities to feed kinetic data banks for modeling the atmospheric re-entry of spacecrafts [9, 10].

The original version of the code is based on the execution of a large set of computational tasks ($n_{\text{event}}$) each consisting of the integration in time, step by step, of the classical trajectory associated with a given set of initial conditions. The step by step integration in time is carried out by an inner $\text{DO END DO}$ loop running on $k_{\text{step}}$ which is embodied inside the outer $\text{DO END DO}$ loop running on $k_{\text{event}}$ as sketched in Fig. 1 where the pseudo-code of the QCT reactive scattering computational engine of GEMS is shown.
The outer iteration is naturally distributable and is perfectly suited for a parameter study [11] while the inner one is strictly recursive. Because of this, the original QCT gridified version of GEMS iterates over the different sets of initial values of the positions and momenta of the three particles of the system considered and distributes the tasks of integrating the related 3D trajectories to their endpoint [12]. Depending on the complexity of the interaction and on the particular reaction path followed by a given trajectory, the integration time may vary significantly with no consequences on the memory request which is always small. However, since the sample of trajectories to be integrated (and therefore to be distributed over the Grid) is usually very large (it can easily sum up to the several millions especially when one looks for rare mechanisms [13]) it is apparent the advantage of implementing GEMS on the Grid.

```
Initialize the distribution environment
Read input data to define physical parameters
Create required virtual monitors
Do $k_{\text{event}} = 1, n_{\text{event}}$
    Perform preliminary calculations and generate initial conditions
    Do $k_{\text{step}} = 1, n_{\text{step}}$
        Perform the time integration step
        Carry out the step checks
        Check for the integration endpoint
    End Do $k_{\text{step}}$
    Work out the properties of the event
End Do $k_{\text{event}}$
Close the distribution environment
Perform the final statistical analysis and rendering on the virtual monitors
```

Figure 1: Pseudo code of the QCT 3D reactive scattering computational engine of GEMS.

As a matter of fact, in the Workflow Manager of the original version of GEMS some PHP scripts allow the user to select the simulation conditions and to launch the various computational tasks on the Grid. The evolution of the simulation is monitored through some JAVA applets which plot the intermediate results. The collection of the intermediate results is handled by invoking a service that takes care of their retrieval by polling the trajecto-
ries running on the Grid. The communication occurs via a TCP socket on a selected port numbered in the range 20000-24999 as specified in the JDL file during the submission phase. Elapsed time measurements confirmed that the QCT dynamical engine of GEMS is ideal for distributed runs on the heterogeneous networked environment of the computing Grid. As a matter of fact, bench runs integrating batches of 100,000 trajectories clearly indicated that routinary production runs on the Grid infrastructure scale perfectly well [14] despite the fact that individual (independent) computational tasks were all extremely variable in duration. The corresponding significant time saving convinced us to move forward and construct dynamic Workflows allowing the adoption of competitive paths using more demanding theoretical approaches and more Service-oriented procedures.

4 An extension of GEMS to quantum dynamics

A typical quite demanding alternative branching of GEMS is the one carrying out QM rather than QCT dynamics calculations. The method considered by us for implementing a QM treatment is the time dependent one adopted by the wave packet suite of programs called RWAVEPR [15, 16]. In spite of their deeply different nature, QM time dependent approaches first define, as is in the QCT ones, the system in its initial reactant arrangement inside a \( \texttt{DO END DO} \) outer loop running on \( k_{\text{event}} \) (like the one given in Fig. 1). This means that in RWAVEPR a wave packet, tailored to suit the initial conditions of the atom diatom system, is first defined in terms of the reactant \( R, r \) and \( \Theta \) Jacobi coordinates. Then a mapping from the reactant to the product \( (R', r' \text{ and } \Theta') \) Jacobi coordinates is performed to represent the wave packet in a form suited for an analysis in terms of product states. On the transformed wave packet a recursive iteration over time is performed, like in the QCT treatment, by the inner \( \texttt{DO END DO} \) loop running on \( k_{\text{step}} \).

The key difference between the time dependent QM and the QCT treatment is, however, the operation performed during the individual time step that in RWAVEPR is concerned with the application of the Hamiltonian operator \( \hat{H} \) to the wave packet. The action of the Hamiltonian on the partial wave function \( \Psi^J \) in the body fixed representation reads in atomic units [19]:

\[
\hat{H}\Psi^{JKp}(R, r, \Theta, t) = \left[ \hat{T}_{Rr\Theta} + V(R, r, \Theta) \right] \Psi^{JKp}(R, r, \Theta, t) + C_{K,K\pm1}^{Jp} \Psi^{JK\pm1p}(R, r, \Theta, t) \tag{1}
\]
which has to be calculated for all values of \( J \). In eq. 1:

\[
\hat{T}_{Rr\Theta} = -\frac{1}{2\mu_R} \frac{\partial^2}{\partial R^2} - \frac{1}{2\mu_r} \frac{\partial^2}{\partial r^2} - \left( \frac{1}{2\mu_R R^2} + \frac{1}{2\mu_r r^2} \right) \\
\left( \frac{1}{\sin \Theta} \frac{\partial}{\partial \Theta} \sin \Theta \frac{\partial}{\partial \Theta} - \frac{K^2}{\sin^2 \Theta} \right)
\]

with \( \mu_R \) being the atom diatom reduced mass, \( \mu_r \) the diatomic reduced mass, \( J \) the total angular momentum quantum number, \( K \) its projection on the body fixed quantization axis, \( p \) the parity of the system, \( V(R, r, \Theta) \) the potential term and \( C_{JpK, K\pm 1} \) the centrifugal (Coriolis) coupling terms

\[
C_{JpK, K\pm 1} = -\frac{[J(J+1) - K(K \pm 1)]^{1/2} [j(j+1) - K(K \pm 1)]^{1/2}}{R^2}
\]

where \( j \) is the rotational quantum number.

This means that the same distribution scheme of Fig. 1 already implemented for the QCT treatment perfectly applies also to the QM time dependent one. There is, however, a significant difference: the size of the matrices used (and therefore the related memory request) increases dramatically with \( J \) because the number of discrete values of \( K \) grows as \( 2J + 1 \). Accordingly, when seeking convergence with \( J \) the needed quantity of memory for fixed \( J \) calculations increases with \( J^2 \).

5 The improved Grid distribution

To run RWAVEPR a more general distribution procedure was adopted. To this end a file (grid_server) containing a list of machines having the same architecture as the one for which the executable has been produced and which are tentatively assumed to be all available (active) on the Grid is created. The procedure creates also the file job_mask that contains all the necessary information about the jobs to be submitted and controls both their submission and retrieval. The job_mask file is created using a simple script that specifies the related ranges (and steps) of the involved parameters (like the constraint \( K_{max} = \min(J, j) \) setting the maximum value of \( K \)).

To make the calculations more suited for distribution on the Grid, the Centrifugal Sudden (CS) [17, 18] approximation was also adopted. The CS computational scheme assumes that \( K \) is a good quantum number (and
therefore $K' = K$). As a result, calculations dealing with matrices of dimension $D(2J + 1)$ (with $D$ being the dimension of the matrices for the $J = 0$ calculation) are split into $2J + 1$ independent computational tasks of dimension $D$ (or lower) for each $J$ value deflating the memory request.

After the job_mask file is created the job submission procedure is started. In the first iteration the file containing the input data is generated from a template by filling the fields containing the relevant parameters. Next, the procedure selects randomly the machine on which running the jobs and the jobs are submitted to the Grid. Subsequently, the related lines in the job_mask file are updated by specifying the identifiers of the jobs and the machines on which they have been submitted while their status on the Grid is converted into ready. Immediately afterwards a check of the status of the jobs, a new jobs launch and a run start are performed. As a matter of fact, the job status is periodically monitored by the job check procedure. This procedure, like the submission one, iterates over the whole number of jobs and reads the content of the line associated with the job in the job_mask file. From it the actual status of the jobs is returned and, when appropriate, new jobs are launched and begin to run. The subsequent submission and checks are managed using the “crontab” command in order to carry out the calculations during the night (Central European Time) when the European segment of the network is less busy. The number of submitted jobs is kept small (much smaller than the total number of the original request) in order to avoid saturation of the Grid Resource Broker.

After all jobs are completed, a retrieve command is launched. The retrieve command collects all the files defined as output in the wave.jdl script corresponding to the jobs associated with the job_ids listed in it and puts them in a specific directory. The procedure ends with an updating of the corresponding lines of the job_mask file performing also an analysis of the information retrieved to check the correct end of the RWAVEPR program (for example, the successful completion of the time propagation) before including related results.

6 The new service-oriented COMPCHEM VO architecture

To exploit the above mentioned articulation of GEMS in alternative paths and its suitability to provide Grid Services we have developed a JAVA proto-
type Framework called YATTEC (Yet Another Tool To Enhance Chemistry). YATTEC supports the use of collaborative and competitive procedures based on a Web Service grounded Credit System, by adopting a Service Oriented Architecture (SOA) [20].

The developed SOA organization consists essentially of two JAVA servers and one JAVA client. The two JAVA servers are the YATTEC Provider, or YP (used to hold the COMPCHEM Web Services) and the YATTEC Registry, or YR (used to drive the initial discovery of the Web Services offered by the VO). The the JAVA client is, instead, the YATTEC Client, or YC, in which neither executables need to run nor Grid Certificates need to be issued.

The COMPCHEM users wishing to run some programs on the Grid can use YC to perform the following actions:

1. Manage the basic Grid operations (for example creating and destroying the Grid Proxy).

2. Search the programs on YR (in other words, search for a YP exposing a Web Service offering the execution of the requested programs).

3. Optionally introduce changes in the requested programs and compile new executables on the selected YP to replace the ones provided as default.

4. Use the result of action 2 (or 3) to start the execution of the Grid job after passing (when the default one is not appropriate) a user-specific input file.

5. Monitor the jobs status.

6. Retrieve the results.

To implement the above procedures, some Web Services had to be developed to wrap the programs and manage their execution on the Grid. In particular, the Web Services needed to manage the Grid Proxies, to handle the YATTEC Framework, to compile different executables (based on the available PESs proposed by the YP or on the new PESs proposed by the user), to monitor the jobs status and to retrieve the results were also developed. Moreover, YC was also transformed into a JAVA applet (called YATTEC Applet, or YA) in order to allow the use of the YATTEC Framework also in a clientless scenario.
This solution enables the active Software Providers of Table 1, hereinafter identified as those members of COMPCHEM, able to setup and manage YPs, to expose their services in an open, standard and secure way suited to serve all kinds of users including those having little familiarity with the considered program.

The implemented prototype Framework, in fact, brings the users to a new level of friendliness and the system to a new level of portability and expertise. As a matter of fact, using our framework there is no need to bear Grid Certificates, to handle binary programs, to choose among the EGEE Grid resources the suitable Computing Elements (CEs), Storage Elements (SEs) and User Interfaces (UIs), and to know as well specific commands of the operating system, as is usually required in the EGEE Grid environment. The operations to be carried out by the user, in fact, are mainly based on natural-like language (for example, when searching for a program of interest, the VO name, the program name, as well as some keywords matching the desired program description and functions are a sufficient means). Moreover, the same procedures are used to carry out various Framework-side operations like those related to the Grid resources match-making for the specific programs offered.

7 An application to RWAVEPR

The above described Web Service approach to GEMS has been implemented in the first application described here not only to choose between QM and QCT approaches but also to apply them to different chemical systems. Offering the possibility of choosing a different system, in fact, does not mean only to allow the user to change, in the input file, the values of some parameters (like the masses and the energy considered). It means instead also the possibility of adopting a proper PES formulated as a suitable functional form. For this reason the application is designed to search on the Web for the availability of good estimates of the parameters of some types of PESs before starting the integration of the scattering equations. If no suitable PESs are available a search is made on the Web for the availability of a sufficiently large set of ab-initio potential energy values for the system considered. If such values are available they are fitted to a proper functional form using an appropriate fitting procedure (like the ones we have recently developed [21]). If they are not available, one has to produce them by running a set of high level ab-initio electronic structure calculations using again
an ad hoc procedure (like the one we have recently developed [22]).

In this lecture we confine our attention to the case in which a suitable PES is available and therefore it needs only to be structured for the Web Service RWAVEPR program execution. To illustrate the case considered a screenshot of the execution of YC is shown in Fig. 2.

As apparent from the figure, the users have first to specify the username by filling the related field called “Username”. Then, after creating the Grid Proxy (this is optional if an already created Grid Proxy is still valid) the users can start the execution of the RWAVEPR program. If the run is performed for training purposes on the default system it is sufficient press the button “Start”. In this case, a default PES (called `pes.f`) and a default input file are used. The users can, instead, choose to prepare their own input file still for the default PES by pressing the button “Choose Input File”. Alternatively, the users can also choose to adopt a different PES by selecting it among those available in the ad hoc panel or to adopt a new one using the “Change Surface” button. In both the latter cases, before getting started, the users have to produce the new executable by pressing the “Compile” button.

After the job starts, some useful information are returned on the Operations Log panel. In particular, information on the URL where to check the
job’s status and on the URI where to fetch the output $S$ matrix elements when the job ends are returned. Both information are automatically provided and written into the proper fields. When the job’s checking phase will return the typical Grid message of job successful completion “Done (Success)”, the users can access the final results by clicking the “Get Results from the Grid” button which triggers the Web Service responsible for the acquisition of the results via the Simple Object Access Protocol (SOAP) over HTTP or HTTPS and then store them in the local (current) directory of YC.

In order to complete the work, additional features had to be implemented. In particular, we found it useful to add the “Reset” button (to delete the previous compiled version of the program and reset to use the default PES and the default input file), the “Status” button (to check the status of YC or, in other words, to check the Web Services availability on YP and the network connection between YP and the EGEE Grid platform) and the “Save” button (to allow the local storing of the Operation Log for future reference).

Specific validation tests of the implemented Framework have been carried out for the RWAVEPR program with three different PESs: the system-default pes.f, the lagrobo3.f and the lag4.f which are, respectively, the LEPS [23], the LAGROBO3 [24] and the LAGROBO4 [25] PESs used in our laboratory for the $N + N_2$ system.

Particular attention has been paid to use an univocal nomenclature (mainly based on the program’s name, on the user’s name and on the starting operations date) to identify all the files involved in the whole process (including the .txt results file) in order to avoid the typical problems of multi-user environments (in which different users can execute the same program at the same time) and of job-intensive environments (in which the same user can execute the same program several times).

8 Conclusions

We have discussed in this paper the efforts undertaken to further articulate and develop the activities of COMPCHEM a Virtual Organization established in EGEE to assemble a Grid enabled molecular simulator GEMS as a computational engine of more complex computational procedures aimed at tackling most of the modern technological and environmental challenges. Our goal here has been the discussion of the efforts made to structure GEMS
in a way that facilitates a user friendly adoption of collaborative and alternative computational schemes. This has led to a structuring of the VO into different levels of collaboration and has also led to the exploitation of Web Service technologies aimed at supporting the design of a Credit System. As a particular case study we have considered the articulation of the GEMS Workflow into a QM alternative to the original QCT treatment to calculate the reactive probabilities for atom diatom reactive processes. To this end, we have implemented a JAVA prototype Framework structuring the reactive scattering engine of GEMS as a Web Service transparent to the Grid (black-box like) application. Such effort has been particularly targeted to make the Grid application truly user friendly and oriented towards a Web Service-based Credit System aimed at rewarding the contributions of the COMPCHEM VO members to the common endeavours of the VO.

9 Acknowledgments

This work has been financially supported by the European Space Agency (ESA Contract 21780/08/NL/ME), the European COST CMST initiative (Action D37). Specific mention needs to be made to EGEE III, a project funded by the European Union under contract IST-2003-508833, the Italian MIUR FIRB Grid.it project (RBNE01KNFP) on High performance Grid Platforms and Tools, and to the Italian MIUR CNR Strategic Project L 499/97-2000 on High performance Distributed Enabling Platforms.
References


