

Application driven Grid developments in the OpenMolGRID project

Bernd Schuller, Mathilde Romberg, Lidia Kirtchakova

Research Center Jülich, Central Institute for Applied Mathematics, Jülich, Germany
{b.schuller,m.romberg,l.kirtchakova}@fz-juelich.de

Abstract. Within the frame of the OpenMolGRID project, several extensions to the underlying UNICORE Grid infrastructure have been developed. This article gives an overview of these developments, focussing on the support for complex scientific workflows and a newly developed command line client for UNICORE.

1 Introduction

The OpenMolGRID (Open Computing Grid for Molecular Science and Engineering) project¹ [1] deals with the integration of data, applications and the modelling of molecular design workflows in a Grid environment. The overall aim of the project is to provide an extensible, information rich environment for molecular engineering and molecular design, using resources available on the Grid in a seamless and secure fashion. The underlying scientific methodology, quantitative structure–activity relationship (QSAR), and other application-specific aspects of the project have been described in detail in [2] and [3]. From an information technology perspective, the unique challenges of the project, which make it an excellent application domain and testing ground for Grid techniques, can be summarised as follows:

- Molecular design and engineering is computationally very intensive, and can generate huge amounts of data;
- Data from diverse sources needs to be integrated using data warehousing techniques and made accessible seamlessly;
- The scientific processes, or workflows, typically involve diverse and heterogeneous data and compute resources;
- The workflows are fairly complex, involving multiple, dependent computational steps

As the Grid middleware underlying the OpenMolGRID system, UNICORE was chosen. The challenges summarised above resulted in the development of several new middleware components, that are generic enough to be used in other contexts as well.

The remainder of this paper is organised as follows. After introducing the UNICORE middleware we describe the components we have developed for workflow support and the new command line interface for UNICORE.

¹ OpenMolGRID is funded in part by the EC under contract IST-2001-37238

2 UNICORE

The OpenMolGRID system is based on UNICORE² (UNiform Interface to COmputer REsources) [4] as the underlying Grid middleware. UNICORE can be characterised as a vertically integrated Grid system, that comprises a graphical client and various server and target system components. UNICORE continues to be developed in the recently started EU FP6 project UniGrids [5], and is being used in various projects such as DEISA [6]. It is available fully open-source under a BSD-type license from the SourceForge repository [7].

For use within an application-centric project such as OpenMolGRID, the main asset of UNICORE is its excellent support for legacy software. UNICORE allows applications to be integrated into the Grid easily [8]. On the server side, the software can be installed "as-is". The client's graphical user interface is extensible by application specific user interface components (plugins).

Furthermore, UNICORE offers strong, end-to-end security through the use of an X.509 public key infrastructure. Security is an important requirement of the pharmaceutical industry, which is one of the key targeted user communities of the OpenMolGRID project.

UNICORE jobs are represented by a Java object, the Abstract Job Object (AJO). It is important to note that in order to be executable, such an AJO must contain all the information about Grid sites and resources. The user can create AJOs easily and comfortably using the graphical UNICORE client, but she needs to allocate resources, select the correct sites for the individual subtasks, and take care of data transfers between sites that may be needed. Obviously, in the case of a large, or rapidly changing Grid, this gets increasingly tedious and difficult.

Within OpenMolGRID, new components have been developed for making the resource selection and allocation procedures much easier, so the users can concentrate on their scientific applications instead of having to deal with the tedious details.

3 OpenMolGRID workflow support architecture

While OpenMolGRID adds significant functionality to the basic UNICORE software, it has been an important design principle to leave the basic UNICORE software untouched, if possible. Thus, all application plugins and server-side tools developed within OpenMolGRID are usable individually, and are fully functional when used outside the OpenMolGRID system, i.e. in a standard UNICORE context. Also, the system has been designed for maximum extensibility and flexibility, to be able to easily accommodate future changes and extensions in the OpenMolGRID system as well as changes in the underlying UNICORE software.

² UNICORE has been developed in several research projects partly funded by the German Ministry of Education and Research under Contracts 01-IR-703 (UNICORE) and 01-IR-001 (UNICORE Plus) and by EC under Contract IST-1999-20247 (EU-ROGRID) and IST-2001-32257 (GRIP).

OpenMolGRID workflow support consists of a metadata layer for applications, an extension of the client side plugin interface and several components that deal with resource management and workflow support.

3.1 Application metadata

As is usual in UNICORE, integrating a software package into the Grid system is done by installing the software executable on a Grid node (a virtual site, or *Vsite* in UNICORE's terminology), and by providing a metadata file that is associated with the application. Additionally, an application-specific Client extension (a *plugin*) providing a graphical user interface should be provided.

In OpenMolGRID, the *application metadata* play an important role. They are used to define the services provided by the application, analogous to a WSDL document in the web services world. Each such service, called "task" within the OpenMolGRID system, is defined by the following elements:

- the task name
- input file(s) specification
- output file specification
- options

Input and output files are always associated with a high-level datatype, which can be compared to a MIME type. This allows matching input and output between subsequent steps in a complex workflow.

This metadata information is part of the server side resources (similar to, for example, the number of CPUs the site provides) and is sent to the Client. There, the metadata are used extensively by the workflow support components, as described in section 3.3 below.

3.2 Client-side plugin extension

On the Client side, plugins must implement a new interface, that allows the workflow support components to set up the job without user intervention. This interface has a function that returns the supported *tasks*, so that the system can match client side plugins and server side applications easily. Furthermore, the interface includes functions to set and query input and output filenames, and to set options.

3.3 Workflow support components

Using the application integration layer and the additional plugin interface described above, support for higher-level workflows could be introduced. Instead of preparing an executable UNICORE job with the usual methods, an XML workflow description is read by the workflow support components developed within OpenMolGRID. From this XML workflow, a UNICORE job is then generated automatically.

Compared to the usual UNICORE way of creating a job, this has a number of advantages. The workflows are fairly high-level, focussing on tasks and dependencies, and thus are very close to the actual scientific workflows the users are interested in. The user need not specify where a given task will be executed, nor need she worry about where certain software is available. The system will take care of mapping the workflow to an executable UNICORE job. The system will also take care of inserting data transfer tasks between Grid nodes, if needed.

The extensions to UNICORE which make up the OpenMolGRID workflow support are given schematically in Fig. 1. The main new component is the MetaPlugin, that deals with setting up a ready-to-submit UNICORE job from an XML workflow description. It is supported by a resource management plugin that reads the server-side application metadata and checks the availability of server-side applications and client-side plugins.

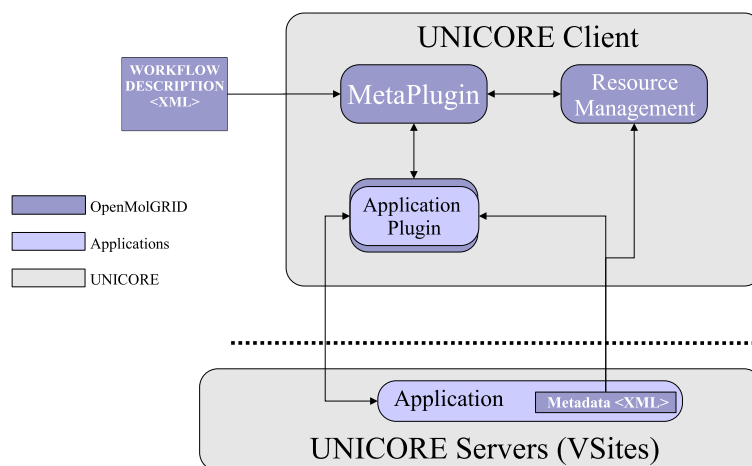


Fig. 1. Workflow support architecture

Apart from the basic functionality of setting up the job, the MetaPlugin has sophisticated functionality that allows "distributing" data-parallel tasks to run on multiple Grid nodes. Its design is highly modular and offers various interfaces for later enhancements. For example, currently resource allocation does not take dynamic parameters such as system load into account. However, more advanced brokering functions can be added easily.

3.4 XML workflow description

The workflows that play a role in OpenMolGRID are high-level workflows, focussing on *tasks* and their dependencies. Since none of the available workflow description schemas fully matched our needs, we developed a simple XML schema ourselves. As an example, a typical workflow to compute molecular descriptors

for a molecular structure starting from two-dimensional chemical structures can be expressed as:

- convert to the three-dimensional molecular structure
- optimize the molecular geometry
- compute molecular descriptors

In our workflow schema, this workflow would, in its simplest form, just state the tasks and dependencies and would look as follows

```
<?xml version="1.0"?>
<workflow>
<task name="2Dto3DConversion" identifier="convert" id="1"/>
<task name="SemiempiricalCalculation" identifier="optimize"
id="2"/>
<task name="DescriptorCalculation"
identifier="Calculate descriptors" id="3"/>
<dependency pred="1" succ="2"/>
<dependency pred="2" succ="3"/>
</workflow>
```

However, this workflow still needs some user input: the initial input data need to be setup, computational parameters must be chosen, the output data needs to be stored permanently. UNICORE might need some resource requests, for example CPU time allocation on resource management systems. The user might wish to parallelize a task on as many grid nodes as possible.

All these things can be done automatically. The workflow schema includes support for

- grouping tasks;
- flow control, such as repeat loops and an if-then-else construct;
- using local data as input;
- requesting specific resources (for example, a certain data source);
- requesting for a task to be run on multiple sites (if the task is data-parallel and the input data can be split in multiple parts);
- setting options for a task;
- allocating resources (such as CPU time) to a task.

In effect, these workflows can be used very flexibly. They are high-level enough so the user need not specify every tiny detail. But, if needed or wanted by the user, they can include more detailed information about Grid nodes and resource requirements.

4 Command-line toolkit

UNICORE lacks a powerful command-line toolkit, or a simple API that could be used by programmers or application developers to make use of Grid resources

from within their applications. Furthermore, for convenient automatization of processes (for example, batch processing) the standard, graphical UNICORE client is not convenient. To address these issues, we have developed a powerful command-line interface to UNICORE.

This tool offers an AJO generation function that builds a UNICORE job dynamically from an XML workflow description. Furthermore, functions to run jobs, monitor them, and fetch the job results are provided. The job generation facility is based on the MetaPlugin, and uses the full OpenMolGRID metadata layer. Therefore, the same high-level workflows can be run in the GUI client and in the command-line client.

On top of the command-line client, a simple front-end has been developed that simplifies running workflows through the command-line client. A workflow can be submitted to the command-line client by placing it in the "request" queue. From this, a UNICORE job will be built and run, and ultimately the result files can be found in the result directory. This toolkit thus offers very powerful batch-processing capabilities.

The command-line toolkit is already used successfully in OpenMolGRID's data warehousing component[3]. When loading data into the data warehouse, Grid resources are used to perform conversion tasks and computation of supplementary data. In this fashion, the subsequent data mining steps are simplified, as standard computations have already been performed by the data warehousing components.

Specifically, for each chemical structure that is loaded into the warehouse, the following data transformations are performed. The structure is converted to a three-dimensional representation, which is then optimised. Finally, molecular descriptors are calculated, and all this data is stored in the warehouse. This process is fully automated by running the appropriate XML workflows through the command-line client. Because the data transformations are done on a per structure basis, the available Grid resources are used very efficiently, as the Grid sites are used in a round-robin fashion automatically by the MetaPlugin.

5 Summary

Within OpenMolGRID, we have developed a number of powerful tools for the UNICORE Grid infrastructure. While these tools were designed with a specific field of application in mind, they are completely generic, and can be used in a variety of contexts. We believe the workflow support tools can help speed up, automatise and standardise scientific processes by integrating data and applications on a UNICORE Grid and offering support for complex workflows. In addition, the command-line client toolkit offers new ways of accessing UNICORE resources that are highly suitable for batch-processing tasks, where no user intervention is needed or wanted.

These new tools are already used successfully in OpenMolGRID, and end users are starting to make use of the new possibilities for doing their science that this system offers.

References

1. The OpenMolGRID project: <http://www.openmolgrid.org>
2. Mazzatorta P., Benfenati, E., Schuller, B., Romberg, M., McCourt, D., Dubitzky W., Sild, S., Karelson, M., Papp, A., Bagyi, I., Darvas, F.: OpenMolGRID: Molecular Science and Engineering in a Grid Context; in: Proceedings of the PDPTA 2004 Conference, June 21-24, Las Vegas, Nevada, USA
3. Dubitzky, W., McCourt, D., Galushka, M., Romberg, M., Schuller, B. Grid-enabled data warehousing for molecular engineering; *Parallel Computing* **30** (2004), 1019–1035
4. Romberg, M.: The UNICORE Grid Infrastructure; *Scientific Programming Special Issue on Grid Computing* **10** (2002) 149–158
5. UniGrids homepage: <http://www.unigrids.org>
6. DEISA homepage: <http://www.deisa.org>
7. UNICORE SourceForge pages: <http://unicore.sourceforge.net>
8. Pytlinski, J., Skorwider, L., Huber, V., and Bala, P.: UNICORE - A uniform platform for chemistry on the Grid; *Journal of Computational Methods in Science and Engineering*, **2** (2002), 369–376