

# Supporting Car-Parrinello Molecular Dynamics with UNICORE

Valentina Huber

Central Institute for Applied Mathematics, Research Centre Jülich,  
Leo-Brandt-Str, D-52428 Jülich,  
Germany,  
v.huber@fz-juelich.de

**Abstract.** This paper presents the integration of application specific interfaces in the UNICORE Grid infrastructure. UNICORE provides a seamless and secure mechanism to access distributed supercomputer resources. The widely used Car-Parrinello Molecular Dynamics (CPMD) application was selected as a first example to demonstrate the capabilities of UNICORE to scientists. Through the graphical interface, developed at Research Centre Jülich, the user can prepare a CPMD job and run it on a variety of systems at different locations. In addition, the "CPMD Wizard" makes it easy to configure the full set of the control parameters, cell properties, pseudopotentials and atom positions for the CPMD simulation.

## 1 Introduction

UNICORE (UNiform Interface to COmputer REsources) [4] provides a science and engineering Grid [10] combining resources of supercomputer centers and makes them available through the Internet. The UNICORE user benefits from the seamless access through the graphical *UNICORE Client* to the distributed resources to solve large problems in computational science without having to learn about the differences between execution platforms and environments.

One important success criterion for UNICORE is the integration of already existing applications. We selected the widely used Car-Parrinello Molecular Dynamics code [1] as a first application to be integrated in UNICORE. CPMD is an *ab initio* Electronic Structure and Molecular Dynamics program; since 1995 the development is continued at the Max-Planck Institut für Festkörperforschung in Stuttgart [3]. This application uses a large amount of CPU time and disk space and is the ideal candidate for a GRID application. Currently, multi processor versions for IBM Risc and Cray PVP systems and parallel versions for IBM SP2 and Cray T3E are available.

Presently a wide variety of groups and projects [11] are experimenting with Grid applications and middleware: Globus [12], Legion [13], WebFlow [14], Web-Submit [15], HotPage [16], Teraweb [17]. They provide simple interfaces that allow users to select an application, select a target machine, submit the job, and monitor the job's progress. Our approach goes far beyond the simplistic

functionality just mentioned. The new graphical CPMD user interface uses the standard functions of UNICORE for authentication, security and data transfer [9]. Furthermore the interface provides the users with a intuitive way to specify the full set of configuration parameters (specification of the input and output files, the library for pseudopotentials, etc.) for a CPMD simulation. In addition it allows to run the CPMD simulations, which comprise many steps in a pipeline. These steps, or "tasks", include importing, preprocessing and in many cases, several tasks in sequence (with data-flow type dependencies) must be executed to obtain the final results.

## 2 The CPMD Wizard

The input file required for CPMD is composed of different sections, contains over 150 different keywords, and has a complex format [2]. To prepare a correct CPMD job the user has to know the internal structure of the CPMD input in detail. A *CPMD Wizard* has been integrated to assist the user. It is started within the *UNICORE Client*.

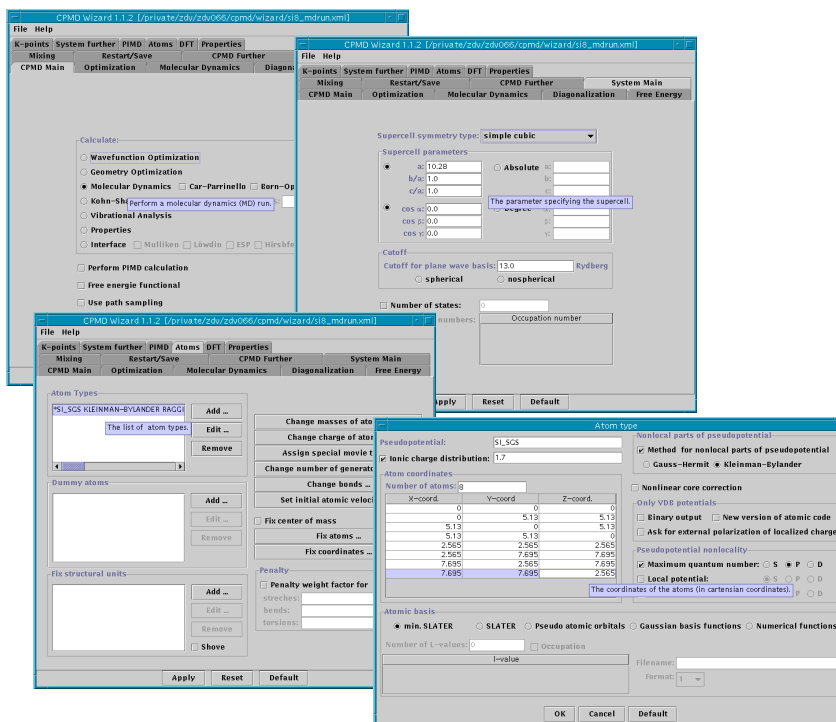


Fig. 1. CPMD Wizard generates the CPMD input automatically.

The *CPMD Wizard* is a graphical interface, implemented as a Java-2 application, which allows the user to generate the CPMD specific parameters automatically. It is composed of different panels matching the structure of different sections in the CPMD input file: *CPMD Main* specifies control parameters for calculation, *Optimization* - optimization parameters, *Diagonalization* - diagonalization schemes, *System Main* - information about the supercell, *Atoms* - atom positions and pseudopotentials, etc. (see Fig. 1).

The interface provides descriptions for each option, that pop up when the mouse lingers over a field. Based on the context, e.g. previously selected options, it allows or prevents input to depended fields. The inactive fields are indicated with a shadow color.

The Wizard prompts the user for missing mandatory data or to correct data that does not match the format specification. It uses XML as its internal data format to facilitate parsing and validation of CPMD input.

### 3 Preparation of CPMD Jobs

Creation of the input file is one of the tasks, which is greatly simplified by the *CPMD Wizard*. In addition, the CPMD job must contain resource specifications of input and output data sets that the CPMD application expects. A customized graphical interface, using standard UNICORE functions, guides the user.

Fig. 2 shows the input panel for one CPMD task, in this case a molecular dynamics run. It is divided into four areas: *Properties*, the configuration area for the CPMD calculation, data *Imports* and data *Exports*.

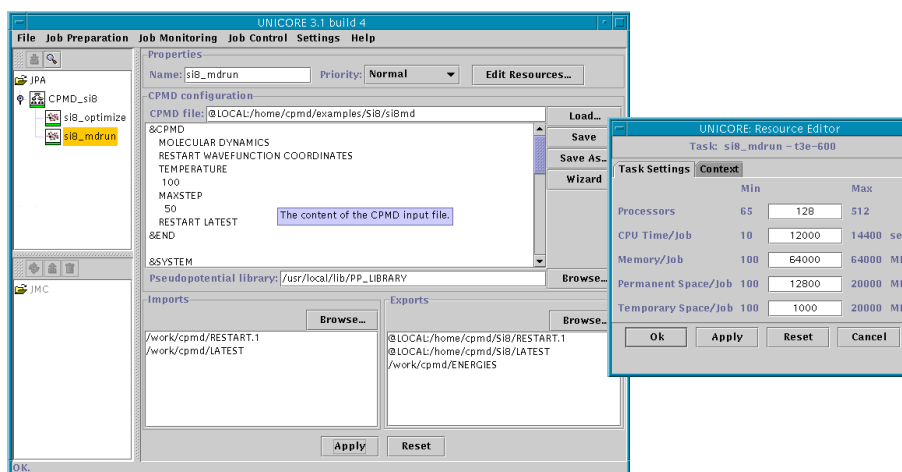


Fig. 2. GUI for the CPMD task.

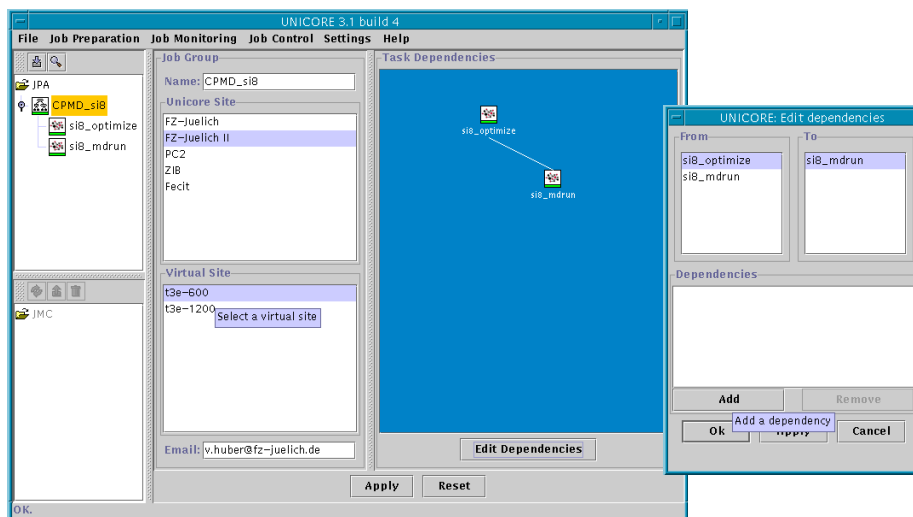
The *Properties* area contains global settings like the task name, the task's resource requirements and the task's priority. The resource description includes the number of processors, the maximum CPU time, the amount of memory, the required permanent and the temporary disk space. The Job Preparation Agent (*JPA*), part of *UNICORE Client*, knows about the minimum and the maximum values for all resources of the execution system, where the task is to be run, and incorrect values are shown in red.

The configuration area shows the data generated by the CPMD Wizard. Experienced users may use the data from existing jobs, stored on the local computer. The configuration data may be edited directly or through the Wizard. It is also possible to save data as a text file on the local disk.

For all atomic species, which will be used in the CPMD calculation, the path to the pseudopotential library has to be specified. The local pseudopotential files will be automatically transferred to the target system. Alternatively, the user can specify the remote directory for the pseudopotentials. If this field is empty, then the default library on the destination system will be used.

The *Imports* area lists the set of input files for the CPMD calculation, e.g. a restart file to reuse the simulation results from a previous step. The input files may reside on the local disk or on the target system. Local files are automatically transferred to the target system and remote files will be imported to the job directory.

The *Exports* area controls the disposition of the result files to be saved after the the job completion. In the example some of the output files will be stored on the target system and others, marked *@LOCAL*, will be transferred to the local system and can be visualized there.



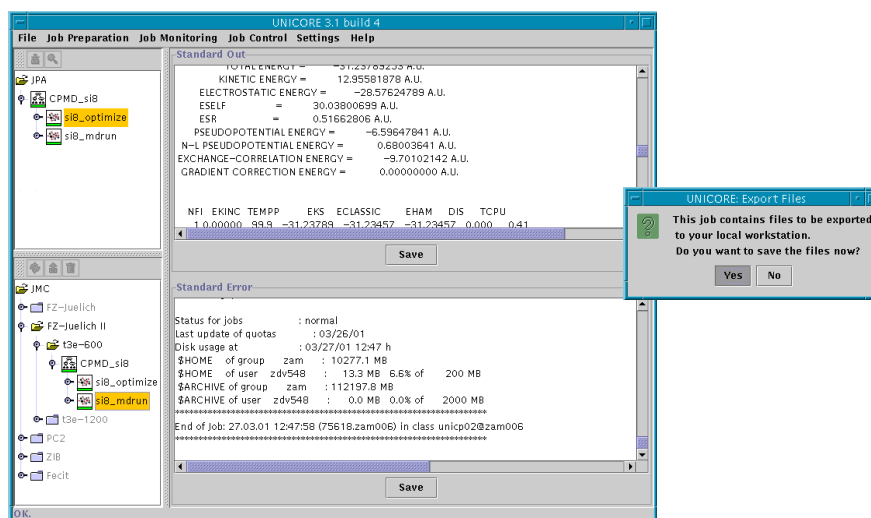
**Fig. 3.** CPMD job consisting of two tasks and dependency between them.

Fig. 3 represents an example of a CPMD job consisting of two steps: *si8\_optimize* task for the wavefunction optimization of a cluster of 8 Silicon atoms and *si8\_mdrun* task for molecular dynamics run. Both tasks will be executed on the same system, T3E in Jülich. The left hand side of the JPA represents the hierarchical job structure. The green color of the icons indicates the job as *Ready for submission*. The second task will be run only after the first one is completed. It uses the output files from the *si8\_optimize* task to reuse the results of the wavefunction optimization. This dependency is shown on the right hand side and represents a temporal relation between the tasks.

Before the CPMD job can be submitted to a particular target system, the interface automatically checks the correctness of the job. Prepared jobs can be stored to be reused in the future.

UNICORE has all the functions to group CPMD tasks and other tasks into jobs. Each task of a job may execute on a different target host of the UNICORE Grid. UNICORE controls the execution sequence, honoring dependencies and transfers data between hosts automatically.

## 4 Monitoring of CPMD Jobs



**Fig. 4.** The Job monitor displays the status of the jobs submitted to a particular system.

The user can monitor and control the submitted jobs using the job monitor part of the UNICORE Client. The job monitor displays the list of all jobs the user has submitted to a particular system. The job, initially represented by an

icon, that can be expanded to show the hierarchical structure. The status of jobs or parts of jobs are given by colors: green - completed successfully, blue - queued, yellow - running, red - completed not successfully, etc. It is possible to terminate running jobs or to delete the completed job from the list of jobs.

After a job or a part of a job is finished, the user can retrieve its output.

Fig. 4 presents the status of the jobs submitted to the *T3E* system in Jülich. The right hand side displays the summary standard output and standard error from two steps *si8\_optimize* and *si8\_mdrun* of *CPMD\_si8*.

## 5 CPMD Integration into UNICORE

Support for the application specific interfaces is based on the “plug-in concept” of the UNICORE Client. Fig. 5 presents the dialog for the setting of user defaults, where the user can specify the plug-in directory for the applications.

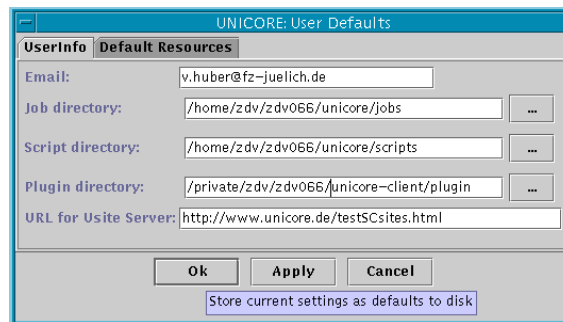


Fig. 5. User settings for the application plug-in directory.

The UNICORE Client scans this directory for the classes implementing the *IUnicorePluggable* interface. In the case of CPMD it is the class *CPMD\_Plugin*. The *CPMD\_Plugin* adds a new option “Add CPMD” to the menu of the JPA and provides methods to display the CPMD GUI in the UNICORE Client. Other classes required for the CPMD integration are: *CPMD\_JPAPanel* and *CPMD\_Container*.



Fig. 6. Basic classes for the CPMD integration.

The *CPMD\_JPAPanel* represents the GUI for the CPMD application and provides the methods to store the data in the *CPMD\_Container*.

The *CPMD\_Container* keeps the actual CPMD data, used for information exchange between the *CPMD\_JPA* and *Client*, which submits the jobs. It checks for correctness of the input data and builds the internal graph of the CPMD task including all required dependencies. In addition, the *CPMD\_Container* provides the application specific icon for the *Client*.

Fig. 6 presents the relationship between CPMD classes and *UNICORE Client*.

## 6 Outlook

The technique used for the CPMD integration is extensible to numerous other applications. We plan in such a way to develop the interfaces for MSC-NASTRAN, FLUENT and STAR-CD applications. These interfaces are going to be integrated into *UNICORE Client* for seamless submitting and controlling of jobs.

In the future it is planned to build a generic interface to allow easier integration of applications.

## References

1. Marx, D., Hutter, J.: *Ab Initio* Molecular Dynamics: Theory and Implementation Modern Methods and Algorithms of Quantum Chemistry (2000) 329–478
2. Hutter, J.: Car-Parrinello Molecular Dynamics - An Electronic Structure and Molecular Dynamics Program. CPMD Manual (2000)
3. Research Group of Michele Parrinello - <http://www.mpi-stuttgart.mpg.de/parrinello>
4. UNICORE Project - <http://www.fz-juelich.de/unicore>
5. UNICORE Forum e.V. - <http://www.unicore.org>
6. J. Almond, D.Snelling: UNICORE: uniform access to supercomputing as an element of electronic commerce. FGCS **15** (1999) 539-548
7. J. Almond, D.Snelling: UNICORE: Secure and Uniform access to distributed Resources via World Wide Web. A White Paper. <http://www.kfa-juelich.de/zam/RD/coop/unicore/whitepaper.ps>
8. Romberg, M.: UNICORE: Beyond Web-based Job-Submission. Cray User Group Conference (2000)
9. Romberg, M.: The UNICORE Grid Infrastructure. SGI'2000 Conference (2000)
10. Foster, I. and Kesselman, C. (editors), The Grid: Blueprint for a Future Computing Infrastructure, Morgan Kaufmann Publishers, USA, 1999
11. Global Grid Forum - <http://www.gridforum.org>
12. Globus: The Globus Grid Computing Toolkit - <http://www.globus.org>
13. Legion - <http://legion.verginia.edu>
14. Web Flow: Web Based Metacomputing - <http://www.npac.syr.edu/users/haupt/WebFlow>
15. WebSubmit: A Web Interface to Remote High-Performance Computing Resources - <https://www.itl.nist.gov/div895/sasg/websubmit/websubmit.html>
16. HotPage - <http://hotpage.npaci.edu>
17. Teraweb - <http://www.arc.umn.edu/structure/>