CHARON System

A Framework for Applications and Jobs Management in Grid Environment

Jan KMUNÍČEK[1,2], Petr KULHÁNEK,[1,3] and Martin PETŘEK[1,3]

1) CESNET z. s. p. o., Zikova 4, CZ-16000 Praha, Czech Republic
2) Institute of Computer Science, Masaryk University in Brno
Botanická 68a, CZ-60200 Brno, Czech Republic
3) National Centre for Biomolecular Research, Masaryk University in Brno,
Kotlářská 2, CZ-61137 Brno, Czech Republic

Introduction

Job submission and its subsequent management that are crucial tasks for successful utilization of cluster and/or grid environments are controlled by various batch systems (PBS, Condor, LSF, scheduling components of grid middlewares such as Globus, gLite, and others). Each batch system has unique tools and different philosophy of its utilization. Moreover, the provided tools are quite raw and users have to perform many additional tasks to use computer resources properly.

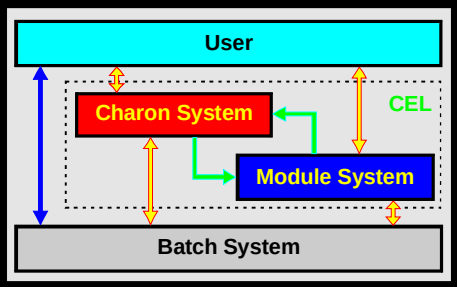


Figure 1. Charon Extension Layer built from Charon System and Module System.

Charon Extension Layer is currently implemented and accessible for utilization in **VOCE (Virtual Organization for Central Europe)** that represents a full set of grid services available for potential grid users within Central European region.

The problems associated with the utilization of low level batch system commands are solved in presented Charon System. The Charon System scheme is shown in **Figure 1**. Charon System works together with Module System, which is used for the management of installed application programs. These two parts then form ***Charon Extension Layer (CEL),*** which provides uniform and simple tools for job submission and management.

However, Charon Extension Layer does not limit original batch systems in any way. It just extends their functionality and simplifies their usage as much as possible for everyday submissions and monitoring of tens up to hundreds of complex computational jobs.

Module System

Charon Extension Layer uses special Module System that manages application software. Each software package is described by a specific *module*. The module contains all information, which is neccessary to work with a particular software (e.g. PATH setup, additional environment setup, etc.). This configuration information is internally stored in XML format that makes setup of Module System very straightforward.

The full module name consists from four parts:

name[:version[:architecture[:parallelmode]]]

The module name design allows high flexibility in software utilization. The user can specify only part of the full module name and Module System will use either default values or it will complete the name in such a way that the chosen module will best fit available computational resources.

Module system is able to:

- find the architecture and parallel mode that best fit available computational resources
- solve conflicts or dependencies between individual modules
- list available modules sorted into categories
- use pre-installed modules on WNs or install them on the fly if they are missing

Example of module name completion

amber (user specification) → amber:8.0:auto:auto (completion by default setup) → amber:8.0:pn3:single (final name resolved according to curent available computational resources)

Tested software

- amber - molecular dynamics suite
- amber-pmf - modified version of amber for free energy calculation
- cats - **c**onversion and **a**nalysis **t**oo**l**s
- turbomole - package for quantum chemical calculations
- qhull - compute convex hull, Delaunay triangulation, Voronoi diagram, etc.
- caver - tool for tunnel exploration in proteins
- octave - high-level language primarily intended for numerical computations
- povray - ray-tracing software
- raster3d - tools for generating high quality raster images of proteins or other molecules
- molscript - program for displaying molecular 3D structures

Conclusions / Future Plans

Charon Extension Layer presents a generic, uniform, and modular approach for job submission and management in wide range of grid environments. It has been successfully implemented and tested in **VOCE** (Virtual Organization for Central Europe) - see **http://egee.cesnet.cz/en/voce/Charon.html** using **LCG middleware**, in **METACentrum** (Czech national grid project) using **PBSPro batch system**, and in sets of local PC clusters using **OpenPBS batch system**.

Currently CEL provides following set of main featues:

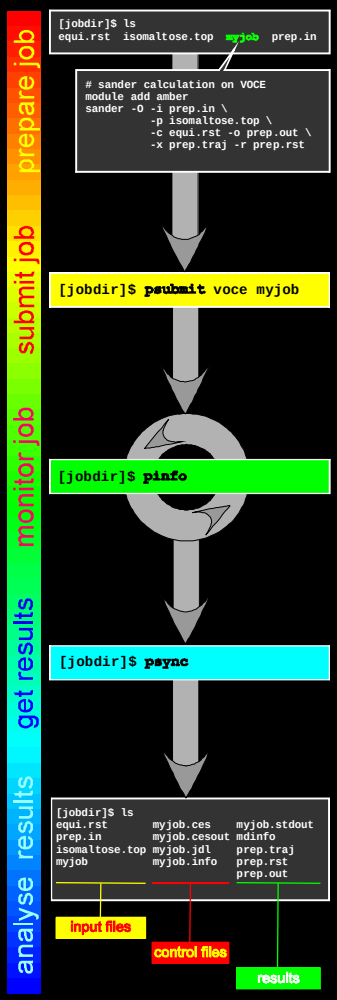
- encapsulation of a single computational job
- minimalization of overhead resulting from direct middleware usage (JDL file preparation, etc.)
- easy submission and navigation during job lifetime
- powerful software management and administration
- comfortable enlargement of available application portfolio
- enabling the freedom of choice between native grid environment and web based approaches
- enabling fast innovation in the development of new computational methods and techniques

Since **VOCE** currently supports **MPI jobs** we will plan to introduce this feature also in Charon. Support for multiple job submmisions (internally via **DAGs**) will be included for **gLite middleware**.

Acknowledgements

EGEE project is funded by European Commission (contract number IST-2003-508833). Financial support from the Ministry of Education, Youth, and Physical Training of the Czech Republic (contract number MSM0021622413) is gratefully acknowledged.

Charon System - Job Flow

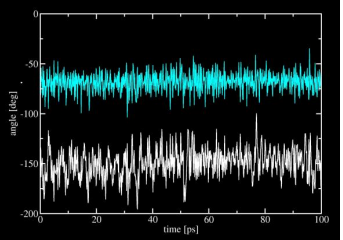


Molecular Dynamics of Isomaltose

```
Job name        : myjob
Grid job name   : myjob (Job type: generic)
Job directory   : skurut4.cesnet.cz:/home/kulhanek/jobdir
Job project     : -none-
==========================================================
Alias           : -none-
Organization    : voce
Profile         : default
----------------------------------------
NCPU            : 1
Resources       : -job match-
Properties      : -none-
Sync mode       : gridcopy
----------------------------------------
Start after     : -not defined-
==========================================================
Do you want to submit job to GRID environment (YES/NO) ?
YES
Please wait packing data ...
Submiting job ...
Job was successfully submitted to GRID environment.
```

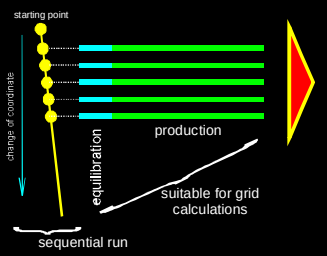
```
Job name    : myjob
JOB ID      : https://skurut3.cesnet.cz:9000/bx06C-
R9mB5uquZarwpCPQ
Grid job name : myjob (Job type: generic)
Job directory : skurut4.cesnet.cz:/home/kulhanek/jobdir
Job project   : -none-
==========================================================
Alias           : -none-
Organization    : voce
Profile         : default
----------------------------------------
NCPU            : 1
Resources       : -job match-
Properties      : -none-
Sync mode       : gridcopy
----------------------------------------
Start after     : -not defined-
==========================================================
Job was submited at     : 2005-10-12 14:16:28
   and was queued for   :  0d 00:04:28
Job was started at      : 2005-10-12 14:20:56
   and was running for  :  0d 00:02:12
Job was finished at     : 2005-10-12 14:23:08
```

```
Starting synchronization procedure.
   downloading sandbox ...
   completing data ...
   dowloading data from SE ...
   unpacking result archive ...
   cleaning ...
Synchronization was successfully finished !
```



Chemical Applications

! Free Energy Calculations by Molecular Dynamics Simulations



Grid environment is suitable for free energy calculations by *umbrella sampling*, especially for its 2D version (**Figure 2**).

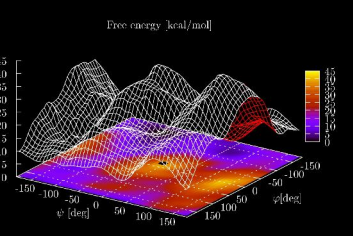


Figure 2. Free energy surface for isomaltose molecule calculated by *umbrella sampling* method.

! Numerical Calculations of Hessian Matrixes

Hessian matrix contains the second derivatives of energy with respect to coordinates. This matrix is widely used in computational chemistry: e.g. for geometry optimizations (especially for transition structures), for thermochemistry analysis, etc.

In the case that analytical implementation of its calculation is not available, the numerical approach is often used. Numerical approach (central diferences) requires **3xNx2** of **independent** gradient evaluations.

$$\begin{pmatrix} \frac{\partial^2 E}{\partial x_1 \partial x_1} & \frac{\partial^2 E}{\partial x_1 \partial y_1} & \frac{\partial^2 E}{\partial x_1 \partial z_1} & \cdots & \frac{\partial^2 E}{\partial x_1 \partial x_n} & \frac{\partial^2 E}{\partial x_1 \partial y_n} & \frac{\partial^2 E}{\partial x_1 \partial z_n} \\ \frac{\partial^2 E}{\partial y_1 \partial x_1} & \frac{\partial^2 E}{\partial y_1 \partial y_1} & & \cdots & & \frac{\partial^2 E}{\partial y_1 \partial y_n} & \frac{\partial^2 E}{\partial y_1 \partial z_n} \\ \frac{\partial^2 E}{\partial z_1 \partial x_1} & & & & & & \frac{\partial^2 E}{\partial z_1 \partial z_n} \\ \vdots & & & \ddots & & & \vdots \\ \frac{\partial^2 E}{\partial x_n \partial x_1} & & & & & & \frac{\partial^2 E}{\partial x_n \partial z_n} \\ \frac{\partial^2 E}{\partial y_n \partial x_1} & \frac{\partial^2 E}{\partial y_n \partial y_1} & & \cdots & & \frac{\partial^2 E}{\partial y_n \partial y_n} & \frac{\partial^2 E}{\partial y_n \partial z_n} \\ \frac{\partial^2 E}{\partial z_n \partial x_1} & \frac{\partial^2 E}{\partial z_n \partial y_1} & \frac{\partial^2 E}{\partial z_n \partial z_1} & \cdots & \frac{\partial^2 E}{\partial z_n \partial x_n} & \frac{\partial^2 E}{\partial z_n \partial y_n} & \frac{\partial^2 E}{\partial z_n \partial z_n} \end{pmatrix}$$