A virtual reactor for simulation of plasma enhanced chemical vapor deposition

Krzhizhanovskaya, V.

Chapter 6. Problem Solving Environment *

6.1. Introduction

In the previous chapters we have introduced step by step, with increasing complexity, various models, implementations and numerical results of processes involved in Plasma Enhanced Chemical Vapor Deposition resulting in what we called a 'Virtual Reactor'. In this chapter we describe an *integrated, flexible* environment to support such virtual reactors over *distributed* computer systems.

There is a clear trend in modeling and simulation away from rigid simulation codes treating a specific fixed aspect, towards more sophisticated flexible problem-solving environments (PSEs) [1] because, as W.D. Hillis (MIT, 1982) puts it: '*Progress in natural science comes from taking things apart, progress in computer science comes from bringing things together*'. These environments are widely considered to be an essential emerging technology with high impact across all fields of science and engineering. PSEs are crucial to successfully address all kinds of multidisciplinary problems, where vast amounts of distributed data need to be managed and processed to discover patterns and knowledge contained within it [2]. Various research groups and scientific software companies subscribe to the relevance of these fully integrated simulators (like the Virtual Reactor) [3].

A PSE provides a complete integrated software environment for composing, compiling, running, controlling and visualizing applications in a specific area. It incorporates many features of an expert system through which it provides extensive assistance to users [4]. Processing, visualization and integration of information from various sources play an increasingly important role in science and engineering. These sources can be widely distributed, and the computational requirements can be highly variable, both in the type of resources required and the processing demands put upon these systems. Grid technology as integrating middleware is a major cornerstone of today's PSEs [5]. By offering a unified means of access to different and distant computational and instrumental resources, it brings unprecedented possibilities and benefits. Connectivity between distant locations, interoperability between different kinds of systems and resources, and high levels of computational performance are some of the most promising characteristics of the Grid. Grid technology provides dedicated support such as strong security, distributed storage capacity, and high throughput over long distance networks. Besides these immediate benefits, the computational resources of the Grid provide the required performance for large-scale simulations and complex visualization and collaborative environments, which are expected to become of major importance to many areas of computational physics. The Globus project [6,7] is perhaps the best-known example of the core software implementation of these Grid functionalities.

* Parts of this chapter were published in [24-27]

The importance of Grid-based PSE research and applications is recognized by international research foundations such as the European Science Foundation (Euresco) through the ESF-PSE initiative, and the NSF initiative on PSEs in the USA. PSEs are becoming *the* key technology to facilitate full exploitation of the unprecedented computational power offered by the Grid revolution [8]. Researchers are investigating many ideas to make this vision reality [9-12].

Recently, lots of efforts have been invested in developing Component Based Environments such as CCA [13], H2O [14], Mocca [15] and ProActive [16]. The concept of common component architectures is a promising approach to build PSE's due to features like light-weighted software components, support for dynamic behavior, scalability, etc. This approach can be beneficially applied for defining interoperable and reusable simulation models that can be automatically arranged into distributed interactive applications [17]. The advantages of these component technologies were used while building the Virtual Reactor PSE that is discussed in this thesis. Integration in to a Grid based PSE was done through the use of CrossGrid middleware [18] and previously developed interactive simulation and visualization tools developed for CrossGrid applications [19-21]. The PSE considered here is generic in that it can handle High Performance Computing (HPC) applications (one parameter setting calculated as fast as possible) as well as High Throughput Computing (HTC) (sweeping various parameter settings) applications and supports interactivity.

Within the CrossGrid team there is a focus on the development of Grid middleware components, tools and applications with a special focus on parallel and interactive computing. Interaction in this context, refers to the presence of a human in a processing loop, and a requirement for near real-time response from the computer system. The CrossGrid testbed largely benefits from the DataGrid [22] experience in testbed setup and Globus [7] middleware distributions.

As a design template, we used a generic architecture for research in e-Science, see [23], where 'information systems integrate available data with data from specialized instruments into distributed repositories and computational models are executed using the integrated data, providing large quantities of model output data, which is mined and processed in order to extract useful knowledge'.

The next sections describe the Virtual Reactor architecture, its most relevant components, the functionalities and the implementation into the Grid software.

6.2. Virtual Reactor Architecture

6.2.1. General description of the *application* components

The Virtual Reactor PSE includes the basic *application* components (modules) for the reactor geometry design; computational mesh generation; plasma, flow and chemistry simulation; editors of chemical processes and gas properties connected to the corresponding databases; pre- and postprocessors, visualization and data archiving modules. In addition to the *application* components we have the Grid *middleware* 'components' to support distributed simulations on the Grid. From the perspective of the Virtual Reactor PSE, the

Grid middleware can be considered as a set of 'components' that provide transparent access to the Grid resources and services.

In order to make the system user-friendly we need to hide the complexity of the underlying components. For that we have developed an advanced graphical user interface (GUI) that seamlessly integrates the disparate modules into *one* transparent user environment, which is presented to the user via a Web-interface and a Grid portal [24]. The architecture supports interaction on various levels: interaction with the workflow through the user interface, interactive visualization, control over the simulation processes and interactive job control on the middleware level.

The Virtual Reactor *application* components and their interdependencies are sketched in Fig. 6.1. It's important to note that there are in fact 5 solvers that were described in the previous chapters (1D and 2D plasma, 1D, 2D and 3D flow) that can be used in different combinations depending on the level of detail required.

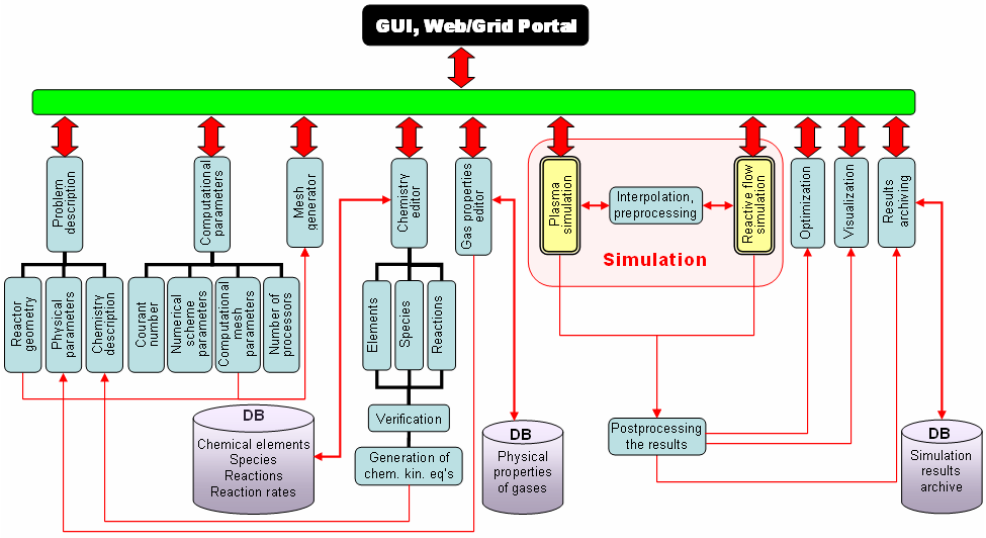

Fig. 6.1. Virtual Reactor *application* components and their interdependencies

The arrows in Fig. 6.1 indicate some of the interdependencies of the components. For instance, the solvers for plasma and reactive flow simulations use as input data:

- the description of the physical and chemical properties created by the editor components;
- parameters of the PECVD processes (like sticking probabilities of the species);
- the generated computational mesh;
- computational parameters (e.g. Courant number, numerical scheme parameters, number of processors for parallel computing, etc.).

The results of the simulations can be stored via an archiving component and retrieved later for comparison with new computational experiments. All the components are

stand-alone modules that can be used separately or in various arrangements in other studies. They are platform-independent and can be compiled and run on any operating system using public-domain compilers and libraries. Some of the most relevant components will be described below.

6.2.2. User interface components

The GUI allows setting up a workflow that describes the problem to be simulated, sets the reactor and computational parameters, controls the simulation process, visualizes and analyzes the simulation results, and gives access to the databases and result archives. An advanced visualization system provides a graphical representation of the results in real-time or postponed mode on different visualization systems. The GUI was implemented with the use of C/C++ and a platform independent GTK+ graphic library [28] as well as the Glade user interface builder [29].

To provide remote access to the Virtual Reactor, a Web interface was developed, using standard client-server technology. For this interface, we created HTML pages, Java applets, JavaScripts and CGI scripts. This system, like a local GUI, provides full control over the simulations. Fig. 6.2 shows some screenshots of the user interfaces described above.

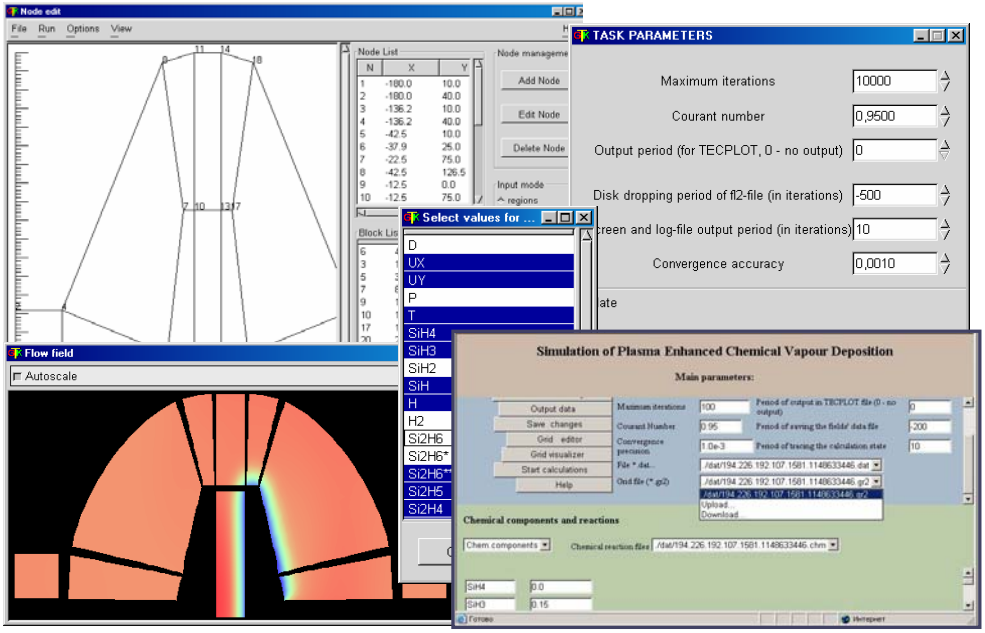

Fig. 6.2. Graphical user interface, stand-alone and Web-based versions.

6.2.3. Chemistry and gas editors components

The Chemistry Editor component provides access to the database of chemical elements, species and reactions, and allows selecting a set of chemical processes for building a customized chemistry model. The verification system incorporated into the module checks the balance of chemical elements in each new reaction introduced by the user, verifies the correctness of the reaction set and removes chemical species (and correspondent reactions) which have a zero net balance. Finally, the module generates a set of differential equations describing the chemical kinetics based on the system of reactions selected by the user from the data base.

The Gas Properties Editor provides access to the database of physical properties of gases and facilitates the introduction of new species into the user chemistry model. One of the important features of this Gas Editor is that it can suggest (based on the available kinetic data) approximate values of the specific heat of formation, the dependency of the specific heat capacity on temperature, and the Lennard–Jones potential parameters for unknown species of higher silanes (Si_nH_m, with n,m > 5). Fig. 6.3 shows some screenshots of these two editors.

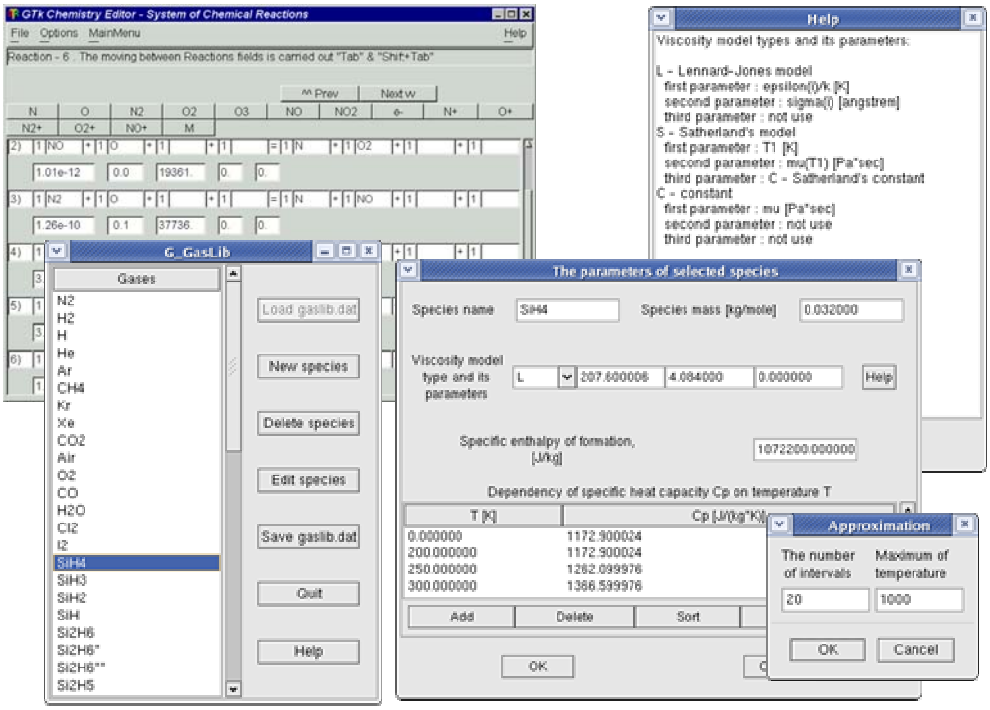

Fig. 6.3. Chemistry and Gas Properties Editors

6.2.4. Visualization components

Different visualization systems were studied and developed for visual analysis of the results. First a commercial package Amtec Tecplot [30] was used. Although it satisfied

most of the basic needs for results visualization, it lacks the flexibility to adjust and incorporate it into some other environment. To overcome this, a Java-based multi-purpose visualizer [24] was developed and incorporated into the Virtual Reactor. This component performs all essential functions of visualizing 2D and 3D objects and variable scalar and vector fields (e.g. density, pressure, temperature, species mass fractions, velocity vectors, etc.). A distinctive feature of the visualizer is that it is specially adapted to the multi-block computational meshes used for complex geometry simulations. For time-dependent simulations, the visualizer provides an option of creating animations. A screenshot of this visualizer is shown in Fig. 6.4.

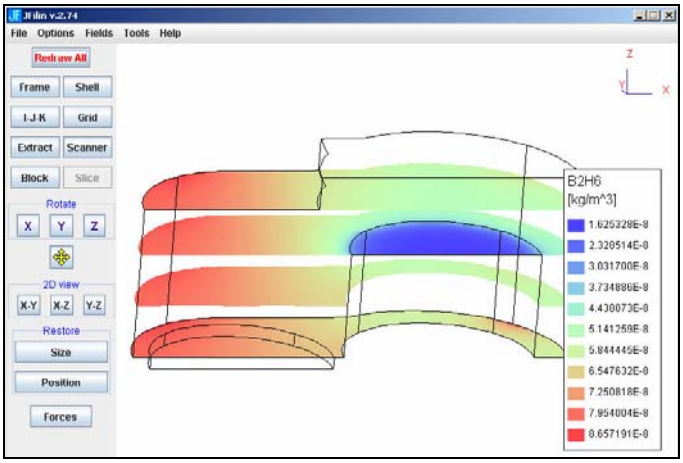

Fig. 6.4. Java visualizer

Another visualization component supports advanced 3D Virtual Reality for a wide range of visualization systems: a personal computer, the virtual reality DRIVE, the 3D immersive environment CAVE and the Personal Space Station, without the actual need to

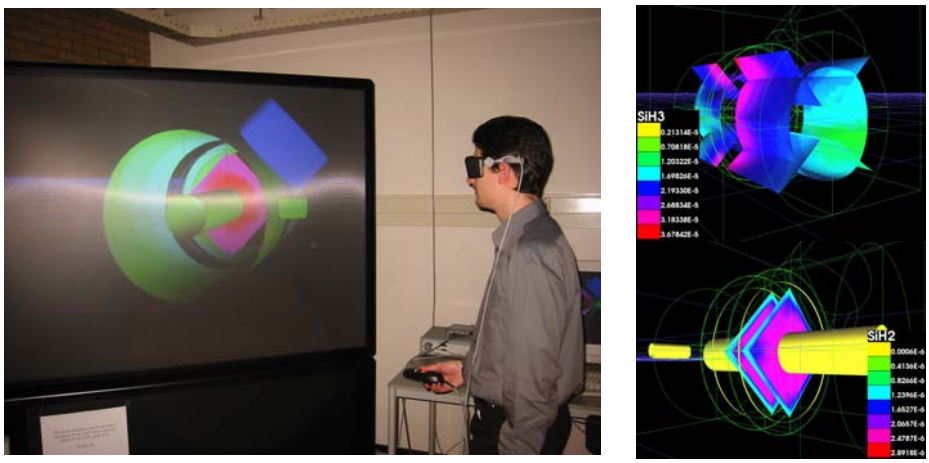

Fig. 6.5. Virtual reality interaction on the DRIVE visualization front-end of the Virtual Reactor

change the code [31,32]. The 3D immersive visualization system was developed based on the VTK and the CaveLib libraries. The DRIVE System uses Immersive Projection Technology with one large screen, active stereo with shutter glasses, electromagnetic tracking and multi-modal interaction devices. A snapshot of visualization on the DRIVE is shown in Fig. 6.5.

6.2.5. Grid middleware components

The Virtual Reactor PSE integrates the *application* components described above with the previously developed interactive simulation environment [32,33], using the common component technology [9] and Grid middleware. This is schematically shown in Fig. 6.6. The Grid middleware is based on the CrossGrid and the DataGrid testbeds [18,22] which use Globus distributions. The CrossGrid testbed shares resources across sixteen European sites [18]. The sites range from relatively small computing facilities in universities to large computing centers. National research networks and the multi-gigabit pan-European network Geant [34], assure interconnectivity between all sites. The actual network used for the Virtual Reactor, includes a local step inside the university via Fast or Gigabit Ethernet, a jump via a national network provider at speeds that range from 34 Mbit/s to 622 Mbit/s or even Gigabit to the national node, and a link to the Geant network at 155 Mbit/s to 2.5 Gbit/s.

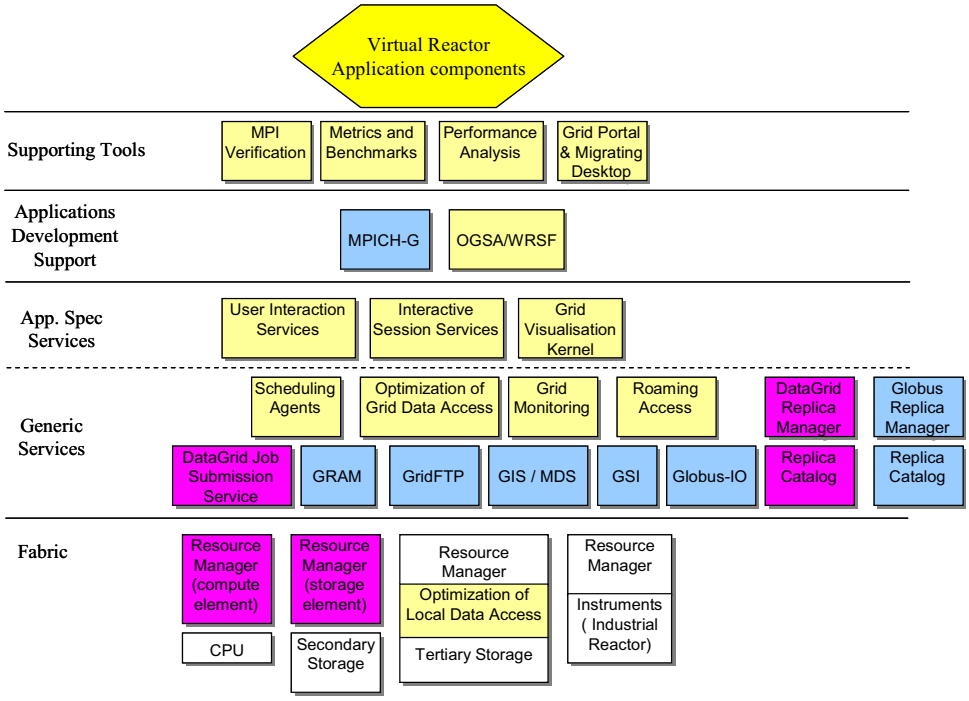

Fig. 6.6. The architecture of the Virtual Reactor *application* and Grid middleware components. The top hexagon indicates all the *application* components described in the subsections above. (1) Yellow: CrossGrid components, (2) Blue: Globus components (3) Red: DataGrid components. For more detailed description of this architecture see [23] and references in there.

A key component of the used Grid architecture is the Grid portal and the Migrating Desktop (MD) as its front-end [18]. The MD was implemented as a Web-based (programmed with Java and JavaScript) user interface for application management, grid and job monitoring, data and metadata management. It includes an authentication mechanism and advanced grid tools. It provides a transparent user work environment, independent of the operating system and hardware. It allows the user to access the Virtual Reactor *application* components, Grid and local resources from remote computers, to run applications, manage data files, and store personal settings, independent of the location or the terminal type. In addition, there is a set of performance analysis components such as OCM-G and G-PM [18] that can be used to monitor and fine-tune the performance of the parallel distributed simulations.

In this architecture, program executions are performed using the Globus job submission capabilities. The parallel solvers use MPICH-G2 as an implementation of the MPI for Grid [35], Globus I/O for inter-process communication and Globus DUROC for resource co-allocation. The Resource Manager in Fig. 6.6 is a component of the DataGrid that has been modified by the CrossGrid project.

This Grid middleware supports different levels of interactivity [36]:

- Interactivity with the Client/Grid System: a user can continuously interact with a Grid client (MD, portal, application GUI, editors, other dedicated clients) without waiting for the conclusion of the jobs already submitted. This interactivity does not need any specialized infrastructure from a Grid testbed point of view.
- One-Way Interactivity with a running application: a user can see the output of the application running on the Grid via the MD client/GUI, synchronously with the application. This interactivity is assured through the infrastructure deployed on the Grid.
- Two-Way Interactivity with the running application: a user via the MD and application GUI can steer the running simulation, either providing some input data on-line as requested by the application or asynchronously, suspending the simulation, changing some input data and resuming it. At the same time, application output data is forwarded to the MD client. This interactivity was implemented using the Condor ByPass (Job Shadow) mechanism [37].

We have incorporated the Virtual Reactor into the Grid using the above described system. We achieved secure Grid access, resource discovery and registration, Grid data transfer, application initialization, editing physical and chemical properties DBs, parameter specification, job submission, distributed simulations and advanced 3D visualization.

We have tested the complete problem-solving environment on a number of tightly coupled clusters as well as on distributed computer systems (the CrossGrid and Russian-Dutch Grid testbeds), where separate modules of the PSE (databases, archives, computational core, visualization kernel and the user interface) were located and operating at different sites.

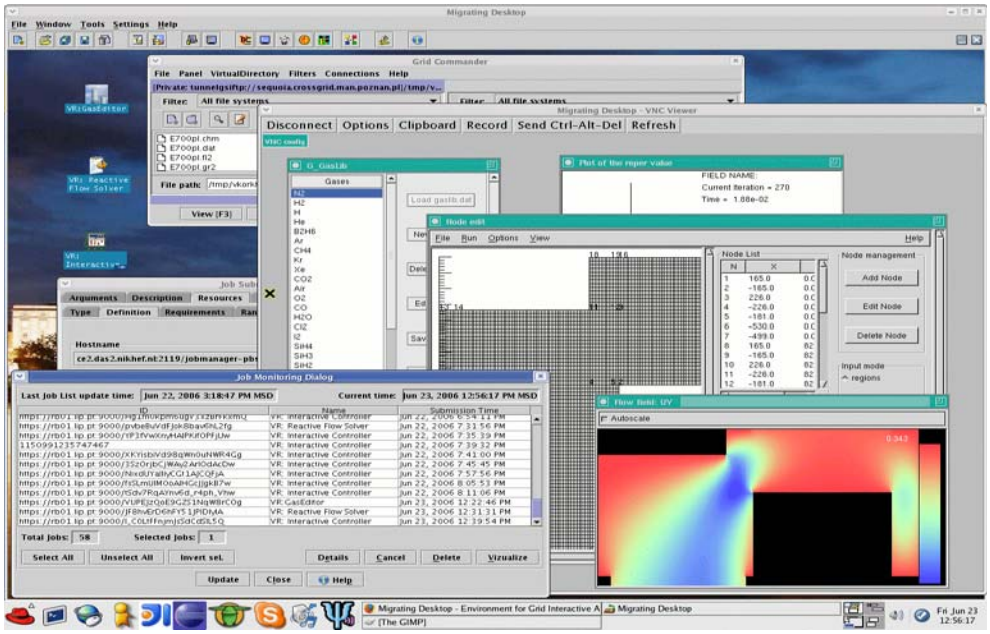

Fig. 6.7. Virtual Reactor on the Grid: Migrating Desktop Grid portal

6.3. Conclusions

The Virtual Reactor was implemented on the Grid and tested against different testbed configurations. We achieved secure Grid access, resource discovery and registration, Grid data transfer, application initialization, editing physical and chemical properties DBs, parameter specification, job submission, distributed simulations and advanced 3D visualization. The foremost conclusion of the presented work is that the core parallel solver can be efficiently exploited on clusters with high-speed interconnects within the RDG environment. The other components of the Virtual Reactor (loosely coupled or fully decoupled) can be distributed across the other Grid resources, thus maximizing the overall efficiency of the whole application.

In addition to the High Performance Computing considerations discussed here, there are at least two more 'performance' aspects that play a role in such generic PSE's as the Virtual Reactor. One is that we might want to run a large set of simulations with slightly different parameters. Here systems like Nimrod and Condor can play an important role. Another 'performance' qualifier is given by the relative ease of setting up a computer simulation experiment in the first place as well as the repetition of such experiments. This requires intuitive methods to dynamically streamline the underpinning processes depending on their availability, their reliability, and the specific interests of chemists, engineers, researchers, and other end users. Scientific workflows, in which a workflow language expresses the flow of data and action from one step to another, provide one option for capturing such methods (see for instance [38,39]). Recently these types of experimental environments have drawn lots of attention. There are even efforts on the way to combine

parameter sweep systems (like Nimrod) with advanced workflow systems like Kepler or the VLe [Virtual Laboratory for e-Science Bsik project [40], see for instance [41]. Although these emerging technologies are very promising, further implementation of it in the PSE described here is out of scope of this thesis and can be considered as future work.

6.4. References

1. http://www.esf.org/esf_euresco_conference.php?language=0&domain=1&conference=139&meeting=3
2. P.M.A. Sloot; D. Frenkel; H.A. Van der Vorst; A. van Kampen; H.E. Bal; P. Klint; R.M.M. Mattheij; J. van Wijk; J. Schaye; H.-J. Langevelde; R.H. Bisseling; B. Smit; E. Valenteyn; H.J. Sips; J.B.T.M. Roerdink and K.G. Langedoen: "Computational e-Science: Studying complex systems in silico". A National Coordinated Initiative. White Paper. February 2007.
3. http://www.cfdrc.com/applications/semicon/plasma.html , http://www.fluent.com/about/news/pr/pr10.htm , http://www.semitech.us/consulting/pvt_growth/models/ , http://www.softimpact.ru/VIROOS-engl.html
4. Houstis, E., Gallopoulos, E., Bramley, R., Rice, J., 1997, "Problem-Solving Environments for Computational Science," IEEE Computers in Science and Engineering, 4, (3) pp. 18-21.
5. Fox, G., 2003, "Grid Computing environments," IEEE Computers in Science and Engineering, 10, pp. 68-72.
6. Globus: http://www.globus.org
7. Foster, I., Kesselman, C., 1997, "Globus: A Metacomputing Infrastructure Toolkit," Intl J. Supercomp Appl, 11(2), pp.115-128.
8. Houstis, E.N. and. Rice, J.R, 2000, "Future Problem-solving Environments for Computational Science," Math. Comp. Sim. 54, pp 243-257.
9. Marinescu, D.C. and Bölöni, L., 2000, "A component-based architecture for problem-solving environments," Math. Comp. Sim. 54, pp. 279-293.
10. Walker, D.W., Li, M., Rana, F., Shields, M.S., and Huang, Y., 2000. "The software architecture of a distributed problem-solving environment," Concurrency: Pract. Exp., 12, pp. 1455-1480.
11. Johnson, C., Parker, S.G., Hansen, C., Kindlmannn, G.L., Livnat, Y., 1999, "Interactive Simulation and Visualization," IEEE Computer, December, pp. 59-65.
12. http://www.teragrid.org/
13. The Common Component Architecture Forum, 2004. http://www.cca-forum.org/
14. D. Kurzyniec, T. Wrzosek, D. Drzewiecki, and V. S. Sunderam. Towards Self-Organizing Distributed Computing Frameworks: The H2O Approach. Parallel Processing Letters, 13(2):273-290, 2003.
15. ProActive project homepage. http://www-sop.inria.fr/oasis/ProActive

16. Maciej Malawski, Dawid Kurzyniec, Vaidy S. Sunderam: MOCCA - Towards a Distributed CCA Framework for Metacomputing. IPDPS 2005
17. Bubak, M., Funika, W., Balis, B., Gubala, T., Malawski, M., Radecki, M., Rycerz, K., Smetek, M., Szepieniec, T., CYFRONET Contribution to CoreGRID: Problem Solving Environments, Tools and GRID Systems, in: Bubak, M., Turala, M., Wiatr, K. (Eds.), Proceedings of Cracow Grid Workshop - CGW'04, December 13-15 2004, ACC-Cyfronet UST, 2005, Krakow, pp. 45-57.
18. CrossGrid EU Science project: http://www.eu-CrossGrid.org/
19. P.M.A. Sloot; A. Tirado-Ramos; A.G. Hoekstra and M. Bubak: *An Interactive Grid Environment for Non-Invasive Vascular Reconstruction*, in 2nd International Workshop on Biomedical Computations on the Grid (BioGrid'04), in conjunction with Fourth IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid2004), (CD-ROM IEEE Catalog # 04EX836C) IEEE, Chicago, Illinois, USA, April 2004. ISBN 0-7803-8431-8.
20. A. Tirado-Ramos; P.M.A. Sloot; A.G. Hoekstra and M. Bubak: *An Integrative Approach to High-Performance Biomedical Problem Solving Environments on the Grid*, Parallel Computing, (special issue on High-Performance Parallel Bio-computing) vol. 30, nr 9-10 pp. 1037-1055. (Chun-Hsi Huang and Sanguthevar Rajasekaran, editors), 2004.
21. Iskra K.A., Belleman R.G., Van Albada G.D., Santoso J., Sloot P.M.A., Bal H.E., Spoelder H.J.W. and Bubak M., 2002, "The Polder Computing Environment, a system for interactive distributed simulation," Concurrency and Computation: Practice and Experience (Special Issue on Grid Computing Environments), 14, pp. 1313-1335.
22. Hoschek, W., Jaen-Martinez, J., Samar, A., Stockinger, H., Stockinger, K., 2000, "Data Management in an International DataGrid Project," Proc. of the IEEE/ACM International Workshop on Grid Computing, Dec. 17-20, 2000, Bangalore, India.
23. P.M.A. Sloot; A. Tirado-Ramos; I. Altintas; M.T. Bubak and C.A. Boucher: *From Molecule to Man: Decision Support in Individualized E-Health*, IEEE Computer, (Cover feature) V. 39, N 11 pp. 40-46. November 2006.
24. V. V. Krzhizhanovskaya, P.M.A. Sloot and Yu. E. Gorbachev. *Grid-based Simulation of Industrial Thin-Film Production*. Simulation: Transactions of the Society for Modeling and Simulation International, Special Issue on Applications of Parallel and Distributed Simulation in Industry, January 2005, V. 81, No. 1, pp. 77 – 85
25. Krzhizhanovskaya V. V., Gorbachev Yu. E. and Sloot P.M.A. A Grid-Based Problem-Solving Environment for Simulation of Plasma Enhanced Chemical Vapor Deposition. Proceedings of the International Conference "Distributed Computing and Grid Technologies in Science and Education". Publ: JINR, Dubna, 2004, pp. 262-271
26. Gorbachev Yu.E., Zatevakhin M.A., Ignatiev A.A., Krzhizhanovskaya. Virtual Laboratory for Research and Education. XI International Conference on High Technologies and Quality of Education and Science, St. Petersburg Polytechnic University. 27-28 February 2004. Publ.: SPbSPU, St. Petersburg, 2004, pp. 59-60

27. V.V. Korkhov; V.V. Krzhizhanovskaya and P.M.A. Sloot: A Grid Based Virtual Reactor: A case study of parallel performance and adaptive load balancing. Journal of Parallel and Distributed Computing, V. 68/5, May 2008, pp 596-608. http://dx.doi.org/10.1016/j.jpdc.2007.08.010
28. http://www.gtk.org/
29. http://glade.gnome.org/
30. http://www.tecplot.com/
31. http://www.science.uva.nl/research/scs/projects/visualisation/index.php
32. R.G. Belleman: *Interactive Exploration in Virtual Environments*, PhD thesis, University of Amsterdam, Amsterdam, The Netherlands, April 2003.
33. Belleman, R.G. and Sloot, P.M.A., 2000, "The Design of Dynamic Exploration Environments for Computational Steering Simulations," Proc. of the SGI Users' Conference, pp. 57-74.
34. Geant: http://www.dante.net/server.php?show=nav.007
35. Karonis, N., Toonen, B., and Foster, I., 2003, "MPICH-G2: A Grid-Enabled Implementation` of the Message Passing Interface," Journal of Parallel and Distributed Computing.
36. Personal communication with Stefano Beco – Marco Sottilaro, Datamat S.p.A.
37. Douglas Thain and Miron Livny, "Multiple Bypass: Interposition Agents for Distributed Computing", Journal of Cluster Computing, volume 4, pages 39-47, 2001.
38. L. Altintas et al., “A Framework for the Design and Reuse of Grid Workflows,” *Proc. Scientific Applications of Grid Computing*, (SAG 04), LNCS 3458, Springer, 2005, pp. 119-132.
39. F. Neubauer, A. Hoheisel, and J. Geiler, “Workflow-Based Grid Applications,” *Future Generation Computer Systems*, Jan. 2006, pp. 6-15.
40. The Virtual Laboratory for e-Science project: http://www.vl-e.nl
41. AppLeS Parameter Sweep Template (APST) Project: http://grail.sdsc.edu/projects/apst/