John von Neumann Institute for Computing

Scientific Computing with UNICORE

Dirk Breuer, Dietmar Erwin, Daniel Mallmann,
Roger Menday, Mathilde Romberg,
Volker Sander, Bernd Schuller, Philipp Wieder

published in

NIC Symposium 2004, Proceedings,
Dietrich Wolf, Gernot Münster, Manfred Kremer (Editors),
John von Neumann Institute for Computing, Jülich,
NIC Series, Vol. **20**, ISBN 3-00-012372-5, pp. 429-440, 2003.

http://www.fz-juelich.de/nic-series/volume20

Scientific Computing with UNICORE

**Dirk Breuer, Dietmar Erwin, Daniel Mallmann, Roger Menday,
Mathilde Romberg, Volker Sander, Bernd Schuller, and Philipp Wieder**

Central Institute for Applied Mathematics,
Research Centre Jülich, 52425 Jülich, Germany
E-mail: {*d.breuer, d.erwin, d.mallmann, r.menday*}*@fz-juelich.de*
{*m.romberg, v.sander, b.schuller, ph.wieder*}*@fz-juelich.de*

1 Introduction

Long before the term Grid Computing was coined by Foster and Kesselman[1] the development of UNICORE, a system offering a Uniform Interface to Computing Resources, was started. The goal was to provide users of the German supercomputer centers with a seamless, secure, and intuitive access to the heterogeneous computing resources at the centers consistent with the recommendations of the German Science Council[2]. A first prototype was developed in project UNICORE[3] to demonstrate the concept. The current production version was created in a follow-on project UNICORE Plus[4] which was completed in 2002. UNICORE's scope is expanded in projects like EUROGRID[5], GRIP[6] and OpenMolGRID[7] to provide support for additional application areas and functions like resource brokering.

Section 2 familiarises the term Grid and illustrates its major ideas and concepts. UNICORE as a realization of these ideas is introduced in Section 3 followed by a detailed description of the benefits UNICORE offers to users and application developers. Section 5 summarizes the paper and concludes with an outlook.

2 Grid Computing Concepts

Large-scale scientific research and engineering often relies on the collaborative use of distributed resources. An airplane designer, for example, might want to use a wind tunnel to evaluate the actual behavior of a particular part designed in a complex set of simulation runs on a supercomputer. Data gathered during the experiment is archived in data repositories which build the information clearinghouse for future post-processing and visualization tasks.

It is the fundamental idea of a computational Grid to facilitate the routine interaction of users with advanced problem-solving tools. A computational Grid uses high-speed networks to link people with computers, databases, and other devices.

According to Foster and Kesselman the definition of a computational Grid is as follows[1]:

"A computational Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high-end computational capabilities."

In a more recent article, Foster, Kesselman, and Tuecke emphasize the nature of the Grid with their description of the specific problem that underlies the Grid concept[8]:

"The real and specific problem that underlies the Grid concept is coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations".

A virtual organization is a resource sharing infrastructure built by a set of individuals and/or institutions who agreed on sharing rules regulating who is allowed to access which resource.

In this sense, the fundamental goal of building computational Grids is to enable the establishment of virtual organizations which facilitate the routine interaction of scarce high-performance devices without the need to replicate those expensive devices at each institution. Grids can be viewed as a middleware infrastructure serving scientists and engineers of a specific community. An important aspect here is that by referring to the construction of virtual organizations the underlying infrastructure does not necessarily consist of dedicated resources, but of protocols and services which can be used to access existing resources. Therefore, resources and individuals might be part of multiple Grids.

Authentication and authorization are essential functions in any distributed environment. For Grid environments, however, the complexity of the security infrastructure is increased by the fact that a Grid combines resources which are maintained by different independent organizations. The problem here is that each institution might have defined its own security policy. Even the existing local security infrastructure might be established based on different authentication systems. Hence, a virtualization of authentication and authorization services that complies with a well-defined security policy is essential for building a virtual organization and thus computational Grids.

The integration of resources and the exposure of the related services is another key function of computational Grids. Any service of the virtual organization is offered by some hosting environment, i.e. by some set of resources. A Grid resource management service has to adapt the high-level service request to an actual allocation using the hosting environment. The service request itself must contain the required information about 'what' is requested, and attributes describing 'where' it is requested. Again, virtualization becomes essential for virtual organizations since the exposed service should hide details of the underlying hosting environment. Therefore, the resource management service should embed a unique interface to different types of resources.

Finally, users, applications, and higher-level services must be able to find appropriate resource candidates. Hence, access to an information repository, storing information about properties and facilities of each transient Grid resource, is required. A particular challenge Grid environments have to face with is their complexity. The routine use of a Grid as a problem solving environment should address this demand by appropriate user interfaces that actually use the virtualization functionalities of the Grid by providing an intuive and seamless access to the underlying resources.

The UNICORE software addresses the above listed issues and is discussed in the subsequent sections.

3 The UNICORE Architecture

As depicted in Figure 1 UNICORE defines a layered Grid architecture consisting of user, server and target system tier. An implementation of UNICORE is realized entirely in Java[a]

[a]Perl was the initial choice to implement the target system tier since many HPC systems did not offer a Java Virtual Machine. The Perl version is still operated at most sites although the majority of developments now rely on the Java version.

and is available as open source downloadable from the UNICORE Forum web site[9]. This paper introduces the basic concepts and components, for an extensive insight please refer to Ref. 4.

Although UNICORE evolves to become conform to the Open Grid Services Architecture[10] (OGSA) following the paradigm of 'everything being a service' an analysis has shown that the basic ideas behind UNICORE already honour this paradigm[11]. Therefore the architecture presented here introduces the fundamental models as they are and largely will remain without addressing the implemenation.

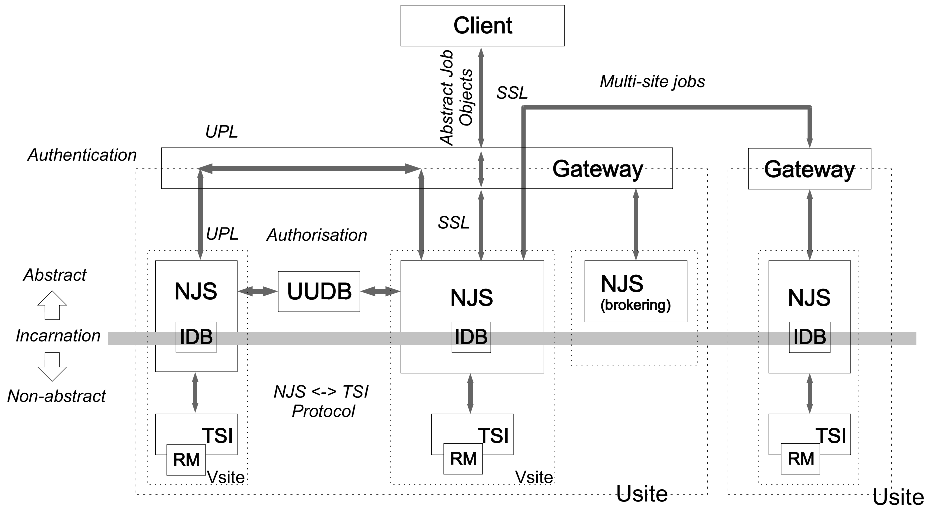

Figure 1. The UNICORE Architecture.

3.1 The Core Models and Modules

The UNICORE Client represents the user tier providing a graphical user interface to exploit the entire functionality offered by the server[b]:

- Creation, manipulation, and submission of jobs including workflow specification and resource assignment.
- Job monitoring and control.
- Management of the user's security means like keys and certificates.
- Provision of logging and debugging information.

The client communicates with the server by sending and recieving Abstract Job Objects (AJO) and file data via the UNICORE Protocol Layer (UPL) which is layered on top of the SSL protocol.

[b]It also exists a client API to integrate UNICORE client functionality into arbitrary applications.

The AJO is the realisation of the job model and central to UNICORE's philosophy of abstraction and seamlessness. It contains platform and site neutral descriptions of computational and data related tasks, resource information and workflow specifications along with user and security information. AJOs are sent as serialized and signed Java objects to the Gateway, followed by an optional stream of bytes if file data is to be transferred.

The server tier is made up of the Gateway and the Network Job Supervisor (NJS). The Gateway acts as the secure entry point to a UNICORE site or Usite (the term Usite refers to an administrative domain), accepting and authenticating UPL requests and forwarding them to the NJS for further processing. The NJS represents resources with a uniform user mapping scheme and no boundaries like e.g firewalls betweem them. It realizes the virtualization of the underlying resources described in Section 2 by mapping the abstract job towards a target system specific one. This process is called Incarnation and makes use of the Incarnation Database (IDB). The IDB contains among others declarations of available resources like e.g. software, incarnations of abstract commands and site-specific administrative information. In addition to the Incarnation the NJS processes workflow descriptions included in an AJO, performs pre- and post-staging of files and authorises the user by requesting the UNICORE User Database (UUDB). The Gateway and NJS execute typically on dedicated secure systems behind a firewall, although the Gateway may also be placed outside a firewall or in a demilitarized zone.

The Target System Interface (TSI) implements the interface to the target host, the third tier of the UNICORE architecture. It is a stateless daemon executing on the target system and interfacing with the local resource manager represented either by a batch system like PBS[12], a batch system emulation on top of eg. Linux or a Grid resource manager like Globus' GRAM[13].

The UNICORE security model relies on the usage of permanent X.509 certificates issued by a trusted Certification Authority (CA) and SSL based communication across 'insecure' networks. Certificates are used to authenticate users and the different software components while authorization is subject to local sites to retain site-autonomy. To prevent AJOs from being compromised they are signed by the user's private key.

3.2 Additional Modules

Diverse projects are developing additional modules to enhance UNICORE's base functionality characterised in the section above. This includes e.g.:

- File transfer alternatives to the default byte streaming mechanism.
- A software layer to interoperate with Globus Grids[14].
- A resource broker designed to process information from heterogeneous sources.
- A scheduling component for a UNICORE virtual organization.
- Interactive access to a UNICORE Grid.

Exemplary for these modules the Alternative File Transfer (AFT) mechanism based on GridFTP[15] is described here.

Per default UNICORE transfers files from a client to a target system or between two target systems making use of Java's byte streaming technique. The necessary file headers are enocoded within an AJO which is sent previous to the byte stream via UPL. To transmit data more efficiently the EUROGRID project develops support for other (alternative) file transfer mechanisms. The requirement for such an mechanism is to improve achieved transmission rates e.g. by using multiple concurrent data streams, provide failsafe data transfer to resume interrupted transfers, offer encryption, and support Quality of Service. The evaluation of state-of-the-art file transfer protocols resulted in the integration of GridFTP into UNICORE, implementing a module called AFTGridFTP[16]. By extending the NJS with an AFT module UNICORE now offers the ability to transfer files between two target systems with increased performance. This activity demonstrates the importance of exploring synergies and moving towards interoperable Grids: it integrates Globus' implementation of GridFTP into UNICORE and makes use of the interoperability layer developed in GRIP to access this implementation.

4 UNICORE: The User's Perspective

UNICORE offers a seamless graphical interface to the user which allows for job preparation and job monitoring. The base Client includes the following elements for creating a job tree:

- Job and sub-job
- Tasks: script, command, transfer, file operation, import, export
- Flow control: repeat, doN, if-then-else, hold
- Dependencies for sequential relations

In addition, tasks for specific applications can be added to the Client by exploiting the plugin interface[17, 18]. Thereby existing software packages like for instance Gaussian, Car Parrinello Molecular Dynamics, or Fluent can be made accessible within UNICORE without any changes to the software package. A plugin developer who knows the application well may develop a specific GUI panel for input specification and for output handling of the application. The collection of plugins provides the user with a powerful Client to perform her tasks efficiently. Plugins for several applications from different areas are being developed in several EC-funded projects like EUROGRID, GRIP, and OpenMolGRID. The remainder of this section gives an overview of the application areas, developments, and use cases in the different projects.

4.1 Applications from EUROGRID

The EUROGRID project applies the concept of GRID computing to selected scientific and industrial application domains, addresses their specific requirements, and highlights the benefits of using GRIDs. The application domains address the following groups:

- Chemists and bio-molecular scientists
- Meteorological scientists and general users
- CAE engineers
- Scientific users of coupled HPC applications

In the area of chemistry and bio-molecular science standard simulation packages were integrated into the UNICORE environment.

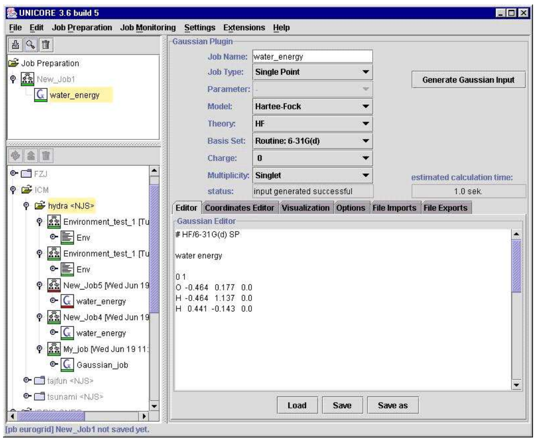

Figure 2. Gaussian plugin.

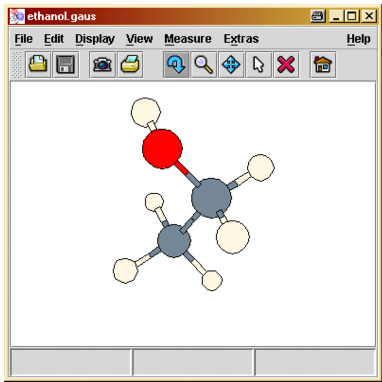

Figure 3. Output visualization.

The general purpose quantum mechanic package Gaussian98 is used to model a broad range of molecular systems under a variety of conditions. It performs its computations starting from the basic laws of quantum mechanics. The Gaussian98 plugin (see Figure 2) for the UNICORE client assists the user with the preparation of the input for the simulation. It reads existing input and recognizes keywords. The molecule coordinate editor accepts XYZ, Z-matrix and text format. The plugin offers a CPU time estimation based on known algorithm scaling, checks the availability of the program at a chosen Vsite and prepares and transfers files for the visualization with a standard visualization tool like Jmol (see Figure 3).

The package of molecular simulation programs Amber is used to simulate the molecular mechanical force field of biomolecules. The extended input preparation wizard (see Figure 4) covers all popular options for simulation ensembles like

- NVT (constant volume and temperature)
- NVE (constant energy)
- NPT (constant pressure and temperature)

The plugin offers extensive assistance to the user (see Figure 5) with a short help for each button, a keyword list and a keyword locator that navigates the user to the proper panel. The plugin checks the availability of Amber on the Vsite.

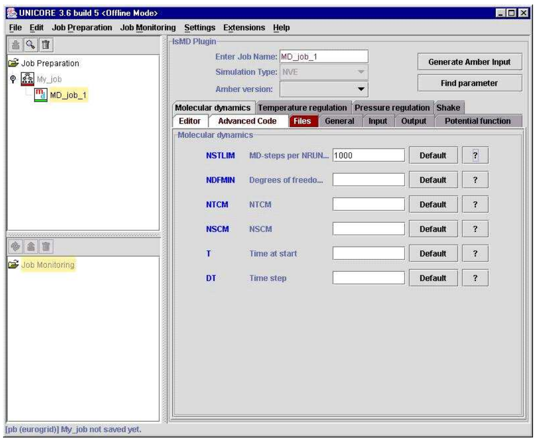

Figure 4. Amber plugin: Input preparation wizard.

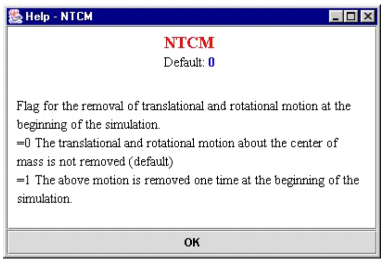

Figure 5. Amber plugin: Help.

Structural databases like the Protein DataBase (PDB) or sequence databases can be queried through the PDBsearch plugin. The query is prepared in the plugin (see Figure 6) and send to databases available through web interfaces. The PDB structure can be visualized in the plugin using external software (see Figure 7) like JMV (Java Molecular Viewer) from the K. Shulten group - Chicago or Jmol from sourceforge.

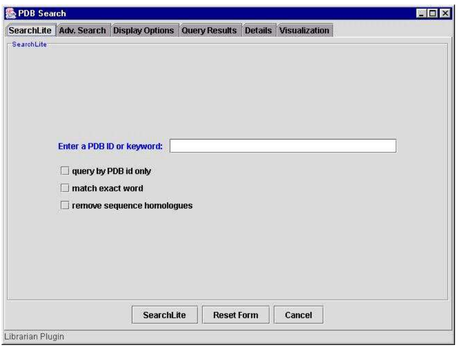

Figure 6. PDB Search plugin.

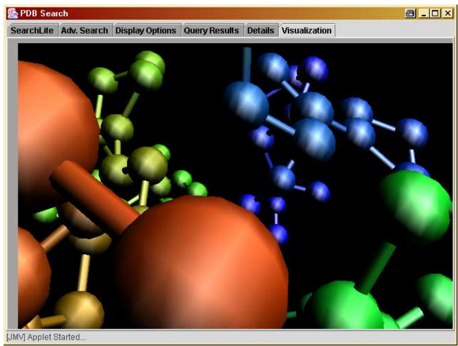

Figure 7. Output visualization.

The objective in the meteorological domain is to provide a high-resolution short range weather forecast for any desired region in the world on demand. The weather forecast is calculated with the relocatable version of the ”Local-Model” (LM) of the German Weather Service (DWD).

The front-end of this application is a UNICORE plugin (see Figure 8). The user chooses the model domain, grid resolution, forecast date and range and forecast products. After the submission of these information topographical data for the selected model domain are derived from high-resolution (1 km x 1 km) data sets at DWD. Furthermore, initial data and lateral boundary data sets for the LM are extracted from result data of the

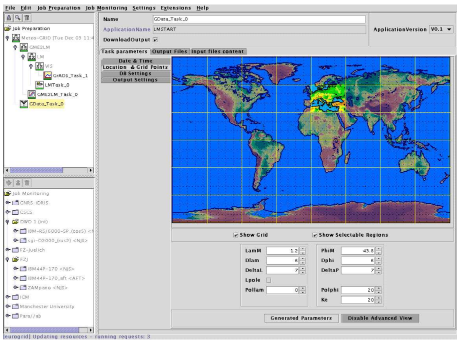

Figure 8. Meteo plugin.

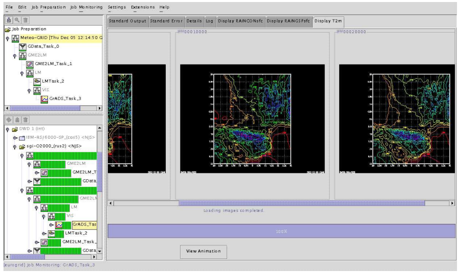

Figure 9. Forecast visualization.

global model (GME) of DWD. The GME results are Interpolated to the LM model grid and the LM forecast is calculated on any supercomputer available in the Grid. The forecast data (GRIB code) will be returned to the user and the LM forecast is visualized.

The Computer Aided Engineering is represented by an acoustic software chain made up of four applications with a complex dataflow (see Figure 10):

- ANADEL: produces an input file for the main solver from a mesh file coming out of a CAD tool.
- MODES: generates boundary conditions on part of the mesh structure.
- ACTI3S: is the main solver, usually required to run on a large parallel computer.
- COUCHA: provides post-treatment on the data computed by ACTI3S.

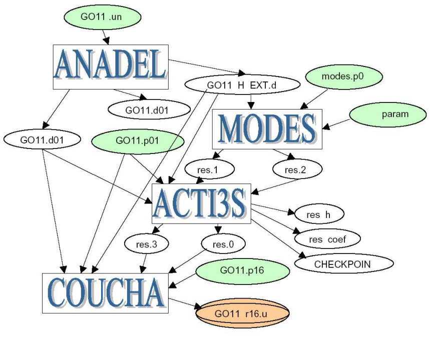

Figure 10. CAE Workflow.

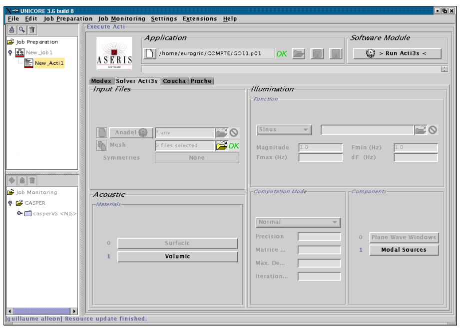

Figure 11. CAE plugin.

This application chain is commonly used in the aeronautical industry. The complexity of the coupled application and its dataflow is covert by the UNICORE plugin (see Figure 11).

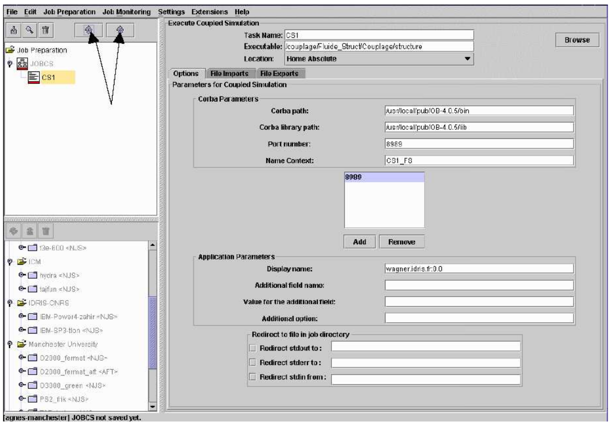

Figure 12. Coupled Simulation plugin.

HPC applications that are coupled using CORBA are facilitated by the Coupled Simulation Plugin (see Figure 12).

The plugin interacts with the Corba name server through the UNICORE Grid environment, and thus simplifies the starting phase of a coupled application.

4.2 OpenMolGRID: Support for Molecular Design and Engineering

The OpenMolGRID project addresses large-scale molecular design problems. It aims at the generation of a problem solving environment for molecular engineering by integrating applications and modeling molecular design workflows in a Grid environment based on UNICORE.

Workflow support is designed to be provided on the client side by an extension component to the base UNICORE Client which is called MetaPlugin, and several supporting components. The MetaPlugin will basically provide an item equivalent to a UNICORE sub-job, which can contain sub-jobs and sub-tasks of arbitrary complexity. However, in addition to the standard sub-job, the MetaPlugin contains extra functionality. This can be summarised in two main topics:

- Read predefined workflows and build UNICORE jobs from them,
- Find and allocate resources needed for the job.

The workflows are specified in XML format which is read by the MetaPlugin. From this workflow a UNICORE job is build within the job preparation area of the client. A component for resource management is used to find UNICORE sites and other resources that are needed for the job. Application specific interfaces (Plugins) supporting automated workflows communicate with the MetaPlugin via a special interface. On the server side, the software packages have to be wrapped by UNICORE applications. A metadata file describes the capabilities and specifics of the application to the client components.

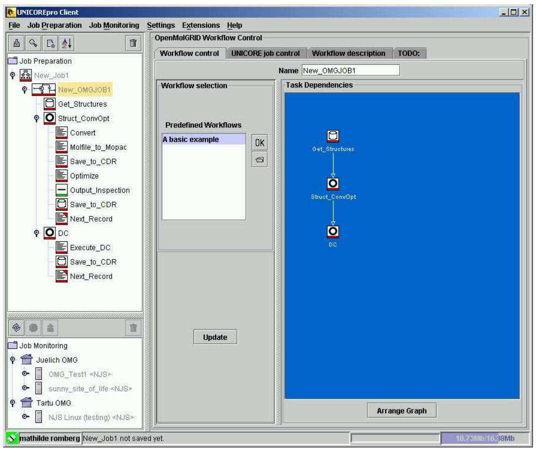

Figure 13. MetaPlugin main panel.

The described functionality will provide a way to model complex workflows and to remove UNICORE ”housekeeping” functions like adding dependencies, sub-groups, transfers etc. from the user. The user loads a workflow into the MetaPlugin of the Client. The plugin then takes care of

- Insertion of sub-groups.
- Distribution of tasks on several Vsites (split data).
- Insertion of auxiliary tasks (transfer, import, export, data conversion).
- Insertion of dependencies.
- Insertion of hold tasks to wait for manual user intervention before resuming processing the next UNICORE job step.
- Allocation of resources.

The user has to tune the input parameters of the major tasks to his needs before submitting the job. Figure 13 shows an early version of the MetaPlugin after loading an example workflow.

5 Concluding Remarks

The successful work of the UNICORE projects and the developments in related EU projects created a production-ready Grid solution. The concepts developed by UNICORE also influenced the standards work of the Global Grid Forum especially through active participation by members of project GRIP. The present direction in Grid computing as promoted by the Global Grid Forum with strong support from industry aims towards integrating Grid technology and Web Services in an Open Grid Service Architecture

(OGSA). Project GRIP demonstrated interoperability with Globus and showed that UNICORE is largely compatible with OGSA. The first Grid Services have been implemented in project GRIP. A proposal for an EU Framework 6 project called UniGridS (Uniform Access to Grid Services) has been submitted to create an OGSA compliant, service based implementation of UNICORE.

Acknowledgments

The authors wish to acknowledge the work of the project partners in project UNICORE, UNICORE Plus, EUROGRID, GRIP, and OpenMolGRID. Full details on the projects can be found at the respective web sites. Projects UNICORE and UNICORE Plus were funded in Part by BMBF unders grant 01 IR 703 and 01 IR 001. Projects EUROGRID, GRIP, and OpenMolGRID were funded in Part by the European Commission under grants IST-1999-20247, IST-2001-32257, and IST-2001-37238.

References

1. I. Foster and C. Kesseman, eds., *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufman Publishers, 1998.
2. Wissenschaftsrat, "Stellungnahme zur Versorgung von Wissenschaft und Forschung mit Höchstleistungsrechenkapazität", Drs. 2104/95, Kiel, 1995.
3. D. Erwin, ed., "UNICORE – Uniformes Interface für Computing Ressourcen", final project report (in German), 2000, `http://www.unicore.org/`.
4. D. Erwin, ed., "UNICORE plus final report – uniform interface to computing resources" Forschungszentrum Jülich, 2003, ISBN 3-00-011592-7.
5. "The EUROGRID Project", Web Site, `http://www.eurogrid.org/`.
6. "The Project GRIP", Web Site, `http://www.grid-interoperability.org/`.
7. "The OpenMolGRID Project", Web Site, `http://www.openmolgrid.org/`.
8. I. Foster, C. Kesselman, and S. Tuecke "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *Proc. of the 7th Int'l Euro-Par Conf. (Euro-Par 2001)*, LNCS 2150, R. Sakellariou, J. Keane, J. Gurd, and L. Freeman, eds., Springer-Verlag, 2001.
9. "The UNICORE Forum", Web Site, `http://www.unicore.org/`.
10. "The Open Grid Service Architecture Working Group", Web Site, `https://forge.gridforum.org/projects/ogsa-wg/`.
11. D. Snelling, "UNICORE and the Open Grid Services Architecture" *Grid Computing: Making The Global Infrastructure a Reality*, F. Berman, G. Fox, and T. Hey, eds., 2003, John Wiley & Sons, pp. 701-712.
12. "The Portable Batch System", Web Site, `http://www.openpbs.org/`.
13. "Globus Resource Management", Web Site, `http://www-unix.globus.org/developer/resource-management.html`.
14. D. Snelling, S. van den Berghe, G. von Laszweski, Ph. Wieder, D. Breuer, J. MacLaren, D. Nicole, and H.-Ch. Hoppe, "A UNICORE Globus Interoperability Layer", *Computing and Informatics*, vol. 21, 2002, pp. 399-411.

15. B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal, and S. Tuecke, “Data Management and Transfer in High Performance Computational Grid Environments”, *Parallel Computing Journal*, vol. 28 (5), May 2002, pp. 749-771.
16. D. Breuer, “Alternative File Transfer Mechanism for UNICORE: GridFTP”, *Proc. of the 15th IASTED International Conference Parallel And Distributed Computing And Systems (PDCS 2003)* , 2003.
17. V. Huber, “Supporting Car-Parrinello Molecular Dynamics Application with UNICORE”, *Proc. of the Computational Science - ICCS 2001 International Conference, San Francisco*, part I, May 2001, pp. 560-566.
18. R. Ratering, “UNICORE Pro: Plugin Programmer’s Guide”, `http://www.unicore.org/downloads.htm`.